

**Comprehensive Dynamics, Pervasive Services & Geo-Clusters:**  
**a general Architecture towards a GeoChannel Web**

**Xuelin He**

*Thesis submitted for the Degree of*


*Doctor of Philosophy (PhD)*

*University College London*

*May, 2017*

## **Declaration**

I, Xuelin He hereby confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Signature: Xuelin He 

Date: 26/05/2017

## **Abstract**

The Web (World Wide Web) generally reflects the relations between things. The current Web largely works on relatively static or stable relations for an information world. However, a massive kinds of dynamic and random relations populate the physical world, which expects an extension to the techniques of the current Web to facilitate applying the dynamic contexts of things.

This research proposes a new Web type, the GeoChannel Web, for mapping and applying real-world dynamic and random relations to systematically support various context-based applications and services.

A set of new concepts and technologies are proposed and designed for the GeoChannel Web. As new technical components, GeoPolicy Language can define context conditions; Geo-off-on communication model works for adaptively throttling communications based on dynamic contexts of things; AccessFeed can control dynamic correlation and interaction between things; Geoww-URI scheme as a novel URI model introduces context into traditional URIs for identifying GeoChannels, i.e. dynamically contextualized resources with condition-matched users.

GeoChannel Discovery supports discovering GeoChannels from a virtual or physical world. GeoChannel Catalogue Services harvest and search real-time metadata that characterizes contexts of GeoChannels; while the physical-world approach supports acquiring Geoww-URIs of pervasive services from our physical environment. GeoChannel Web Engine is a service for mapping out context-based relations between things into a global-scale dynamic topology graph network named a GeoChannel Web. Technical mechanisms are explored for forming GeoChannel networks. Geonoon platform as a prototype implementation of these new techniques has been developed for supporting the use case projects that demonstrate creative and representative applications.

The GeoChannel Web provides new techniques for contextualizing, identifying, discovering, clustering, and bridging things. All these new technical facilities constitute a Context-Organize layer that can fit into the general Web fabric, to provide new infrastructure for organizing real-world interaction and activities based on latest context. It can especially improve the current GeoWeb from a data-snapshot-browse-query platform into a live environment (e.g. a live Digital Earth) populated by crowdsourced dynamics, pervasive services and geo-clustered interaction.

## **Acknowledgements**

It was a long journey for my PhD research work that has experienced much or episodic frustration, anxiety, inspiration, excitement, harvest and happiness. Now when it comes to the time point I have eventually written and summed up this work and contributions, I would like to express my profound gratitude to people who have helped and accompanied me on this journey.

First and foremost, special gratitude goes to my family for the persistent support and great patience at all times. During this long period of my PhD study, my wife took the full responsibility for my family. I really enjoyed the time when my lovely daughter stayed with me in the UK. I am grateful to my grandparents, parents and sisters. My family have given me their unequivocal support throughout, as always, for which my mere expression of thanks likewise does not suffice.

Thanks to the tolerance of my nominal supervisor, Mr Jeremy Morley, I had the chance and environment to think over and do what I am interested in, trying all my efforts to explore something new and novel targeting original contributions.

Thanks also go to the Geospatial Science community in the University of Nottingham. For a main part of time in my PhD phase I have been staying here and benefited a lot from this environment, using research facilities and enjoying friendships. I especially express my thanks to Professor Mike Jackson for his kind help.

I am delivering my thanks to the academics in University College London. Thanks to Dr Claire Ellul and Dr Jonathan Iliffe for their beneficial advices in this period.

I was very impressed by the careful thesis inspection by the examiners, Mr Philip James and Dr Maurizio Gibin. I am grateful for their appropriate comments and guidance, and for having given me an enlightening process of my thesis defence.

I acknowledge the financial support from the UK/China Scholarship Program for Excellence.



## Table of Contents

<b>Abstract.....</b>	<b>2</b>
<b>Acknowledgements.....</b>	<b>4</b>
<b>Table of Contents .....</b>	<b>5</b>
<b>List of Figures.....</b>	<b>10</b>
<b>List of Tables .....</b>	<b>13</b>
<b>List of Listings .....</b>	<b>14</b>
<b>Glossary.....</b>	<b>15</b>
<b>1 Introduction .....</b>	<b>20</b>
1.1 Motivation.....	21
1.1.1 The current Web: some types of usage .....	21
1.1.2 The physical world: dynamic, random and context-based relations .....	29
1.1.3 Research Objectives and Tasks.....	34
1.2 Research Questions .....	35
1.3 Research Outcome .....	37
1.4 Project Scope.....	39
1.5 End-Users of this Research .....	40
1.6 The Structure of this Thesis .....	41
<b>2 The GeoChannel Web: the Concept &amp; Component Framework.....</b>	<b>43</b>
2.1 Introduction.....	43
2.2 The concept of GeoChannel.....	43
2.3 The GeoChannel Web.....	45
2.3.1 Mapping and applying dynamic and random relations .....	46
2.3.2 Bridging Things in Context: application scenarios .....	47
2.4 The Framework of Functionality Components .....	52
2.5 Review on related Techniques .....	54
2.5.1 Basic techniques: HTML, HTTP, URL .....	54
2.5.2 “geo:” URI and GeoRSS.....	55
2.5.3 Geo-tag, Geo-filter and spatial-extended SQL.....	57
2.5.4 XACML & GeoXACML .....	59
2.5.5 Challenges to existing Techniques.....	63
2.6 Summary .....	68
<b>3 GeoPolicy Language.....</b>	<b>70</b>
3.1 Introduction.....	70
3.2 GeoPolicy Language: a new Language for the GeoChannel Web.....	70

3.3	Design of Language Elements for GeoPolicy .....	71
3.3.1	The core part of language elements .....	71
3.3.2	The extension mechanism for domain-specific elements .....	76
3.4	Extended-KML Encoding Model for the GeoPolicy Language.....	76
3.4.1	The Strategy of extending KML for the GeoChannel Web .....	77
3.4.2	Extended KML Class Diagram for GeoChannel Technology.....	78
3.4.3	Examples of GeoPolicy Conditions .....	81
3.5	Summary .....	85
<b>4</b>	<b>Geo-off-on Communication Model .....</b>	<b>87</b>
4.1	Introduction.....	87
4.2	The principle: discriminating GeoContext relations for communication.....	88
4.2.1	Partitioning GeoContext Zones by GeoPolicy Conditions .....	88
4.2.2	Grading GeoContext Zones with discriminative Communication .....	91
4.3	Design for the Geo-off-on Communication Model.....	94
4.3.1	Geo-off-on: a Cooperative & Adaptive Communication Model.....	94
4.3.2	Pull/Push Elements Information Model .....	96
4.4	Application Scenarios .....	96
4.5	Summary .....	98
<b>5</b>	<b>GeoAccessFeed.....</b>	<b>99</b>
5.1	Introduction.....	99
5.2	GeoAccessFeed -- Framework & Principle .....	100
5.2.1	Components Relational Model .....	100
5.2.2	The Principle of GeoChannel Access-Organize .....	102
5.3	AccessFeed: Correlating Things in Context.....	105
5.3.1	Desired Features of AccessFeed .....	106
5.3.2	AccessFeed Functional Components .....	107
5.4	AccessFeed Information Model: the Framework.....	108
5.4.1	AccessFeed structure Model with Extended KML Encoding .....	108
5.4.2	Accessibility module Information Model .....	111
5.4.3	TopologyLink module Information Model .....	114
5.5	AccessFeed Workflow Process Model.....	117
5.6	Updating AccessFeeds .....	119
5.6.1	Cooperative Update by Multi-level Mechanisms .....	120
5.6.2	GeoBlindZone & GeoLagZone: Wake-up & Catch-up .....	122
5.6.3	AccessFeed Update Service .....	126
5.6.4	Geo-Off-On Communication Model for Updating AccessFeeds.....	128
5.7	Application Scenarios .....	137
5.8	Summary .....	140

<b>6</b>	<b>Geoww-URI Scheme.....</b>	<b>141</b>
6.1	Introduction.....	141
6.2	A New URI Scheme for the Web.....	141
6.2.1	Demands and expected Capability of a new URI Scheme.....	141
6.2.2	Geoww-URI: “geoww://” URI Scheme.....	143
6.3	“geoww://” URIs for Identifying & Selecting Resources .....	145
6.4	“user@geoww://” URIs for Identifying & Targeting Users .....	148
6.5	The Usage of the “geoww://” URIs: GeoChannel Web Addresses.....	150
6.6	Summary .....	152
<b>7</b>	<b>GeoChannel Discovery .....</b>	<b>154</b>
7.1	Introduction.....	154
7.2	The Framework & Approaches.....	154
7.2.1	The Component & Process Framework .....	155
7.2.2	Physical-world Approach & Virtual-world Approach.....	156
7.3	Physical-world Approach.....	158
7.3.1	QR-code for Geoww-URI.....	158
7.3.2	RFID for Geoww-URI .....	159
7.3.3	Physical Web with Geoww-URI.....	159
7.4	GeoChannel Catalogue Service.....	163
7.4.1	The Metadata Model of GeoChannel Catalogues .....	164
7.4.2	Catalogue Service Interface Model.....	167
7.5	Real-time Harvesting of GeoChannel Metadata .....	168
7.5.1	PubSubHubBub: a Real-time PubSub Protocol .....	168
7.5.2	The Information Model for GeoChannel Metadata Feeds .....	170
7.5.3	The Component Roles Model for Harvesting GeoChannel Metadata.....	172
7.6	OpenSearch for GeoChannels: GeoContext and KML Enablement .....	174
7.6.1	OpenSearch with Geospatial & Temporal Extension .....	174
7.6.2	Description of GeoChannel Search Service .....	176
7.6.3	KML-extension for Search Responses.....	180
7.7	Context-topology Query Service.....	184
7.7.1	The concept of Context Topology .....	185
7.7.2	Querying Context Topology Network .....	186
7.8	Summary .....	187
<b>8</b>	<b>The GeoChannel Web &amp; a Prototype Platform .....</b>	<b>189</b>
8.1	Introduction.....	189
8.2	Dynamic GeoChannel Networks of Resources & Users.....	189
8.3	GeoCluster & GeoBridge.....	192
8.3.1	GeoClustering Users for Different GeoServeZones.....	192

8.3.2	GeoBridge Mechanism for GeoClusters Union .....	197
8.4	Federation of Geo-Compatible GeoChannel Resources .....	201
8.5	Discovery & Navigation of Resources & Users.....	206
8.6	The GeoChannel Web .....	209
8.6.1	GeoChannel Web: a Global Dynamic GeoChannel Network .....	209
8.6.2	Features of the GeoChannel Web .....	211
8.7	The Full Architecture Model.....	217
8.7.1	Bring All Together: the Complete Architecture.....	217
8.7.2	Context-Organize Layer: an Abstraction & Infrastructure .....	222
8.8	A Prototype Implementation: the Geonoon Platform .....	225
8.8.1	Components Workflow Framework.....	225
8.8.2	Front-end Facility .....	227
8.8.3	GeoChannel Discovery Services.....	234
8.8.4	AccessFeed Update Service .....	236
8.8.5	GeoChannel Web Engine.....	237
8.9	Summary .....	244
<b>9</b>	<b>Use Cases .....</b>	<b>246</b>
9.1	Introduction .....	246
9.2	Mass Vehicles Motion with inter-vehicle Interaction in New York City .....	247
9.2.1	Background & Objectives .....	247
9.2.2	Functionality Components Architecture .....	251
9.2.3	Simulation: Reproducing Mass Vehicles motion process.....	252
9.2.4	Real Vehicle Motion Process with GeoAccessFeed Technology .....	255
9.2.5	Mapping Dynamic & Random Relations to GeoChannel Networks .....	259
9.2.6	Application: Inter-vehicle Networking Interaction .....	263
9.3	Virtual-Real-World Services & Interaction in British Museum .....	265
9.3.1	Background & Objectives .....	265
9.3.2	Pervasive Services in Virtual & Physical World .....	268
9.3.3	Clustered Interaction: Bridging Things in Context .....	273
9.3.4	Joining Virtual & Physical World: new patterns for Clustering & Connecting.....	276
9.3.5	Software Implementation of this use case.....	277
9.3.1	Online Demonstration.....	281
9.4	Assessment of Use Cases and Technology .....	283
9.4.1	Proven Technology of the GeoChannel Web.....	283
9.4.2	Limitations and Drawbacks .....	286
9.5	Summary .....	288
<b>10</b>	<b>Research Summary and Future Work .....</b>	<b>289</b>
10.1	Contributions: new Techniques & Facilities .....	289

10.1.1 Framework of new Techniques and Facilities .....	289
10.1.2 GeoPolicy Language.....	290
10.1.3 Geo-Off-On Communication Model.....	291
10.1.4 GeoAccessFeed.....	293
10.1.5 Geoww-URI Scheme .....	293
10.1.6 GeoChannel Discovery technology .....	294
10.1.7 The architecture of the GeoChannel Web .....	295
10.1.8 Geonoon: a prototype platform.....	295
10.2 Addressing Research Questions.....	296
10.2.1 Answering the research sub-question 1 .....	296
10.2.2 Answering the research sub-question 2 .....	297
10.2.3 Answering the research sub-question 3 .....	297
10.2.4 Answering the research sub-question 4 .....	298
10.2.5 Answering the research sub-question 5 .....	298
10.2.6 Answering the research sub-question 6 .....	298
10.2.7 Answer to the general question.....	299
10.3 Evaluation & Discussion.....	301
10.3.1 The GeoChannel Web and other types of Web.....	301
10.3.2 Review on the Research Output.....	304
10.3.3 Potentials of Development.....	304
10.4 Recommendation for Future Research.....	305
10.4.1 Further implementing GeoChannel facilities .....	306
10.4.2 Theory & Practice for the GeoChannel Web .....	308
10.4.3 Normalization & Standardization .....	311
10.4.4 Commercialization: Potentials & Opportunities .....	312
<b>References .....</b>	<b>316</b>
<b>Appendix 1 – GeoPolicy Language: extended-KML Schema .....</b>	<b>332</b>
<b>Appendix 2 – Geo-off-on Communication Model: extended-KML Schema .....</b>	<b>351</b>
<b>Appendix 3 – AccessFeed: extended-KML Schema.....</b>	<b>354</b>
<b>Appendix 4 – GeoChannel &amp; Discovery: extended-KML &amp; -XML Schema.....</b>	<b>360</b>
<b>Appendix 5 – GeoChannel Markup Language.....</b>	<b>367</b>
<b>Appendix 6 – Metadata Information Model for GeoChannel Catalogue Services .</b>	<b>369</b>
<b>Appendix 7 – Demo Project: Mass Dynamics of real-time Maps &amp; Context Relations for Taxi Trips in New York City .....</b>	<b>372</b>
<b>Appendix 8 – Demo Project: Virtual-Real-World Services &amp; Interaction in British Museum.....</b>	<b>373</b>

## List of Figures

Figure 1-1: The size of the indexed World Wide Web.....	23
Figure 1-2: Graphic representation of a minute fraction of the WWW, demonstrating hyperlinks related Wikipedia.....	23
Figure 1-3: Smart eyeglasses observe physical-world resources from changing positions and view angles dynamically and randomly .....	31
Figure 1-4: Dynamic and random relations between roads and vehicles .....	31
Figure 1-5: Dynamic maps: real-time maps delivering on-demand dynamics .....	31
Figure 1-6: The increasing availability of contexts captured by ubiquitous sensors .....	33
Figure 1-7: The peculiarity and diversity of some types of Web .....	34
Figure 2-1: The concept of GeoChannel ---- contextualized resources, users and conditions .....	44
Figure 2-2: The concept of GeoChannel----an organizing unit for a set of context-based relations .....	45
Figure 2-3: Example: the GeoChannel Web for mapping and applying dynamic, random & context-based relations between real-world things .....	46
Figure 2-4: The GeoChannel Web for bridging things in context.....	47
Figure 2-5: Example: GeoChannel technology for achieving large-scale dynamic maps.....	48
Figure 2-6: Example: GeoChannel technology for implementing pervasive services .....	50
Figure 2-7: Context-based clustering and connecting: GeoChannel technology for new-pattern social networking .....	51
Figure 2-8: Components framework: new techniques and facilities expected for the GeoChannel Web ..	53
Figure 2-9: XACML Policy Language Model .....	60
Figure 2-10: XACML Architecture.....	61
Figure 2-11: XACML Data Flow Model .....	61
Figure 2-12: Architecture of a (Geo)XACML based access control system .....	63
Figure 3-1: Extended-KML class diagram with the newly-proposed GeoChannel elements.....	78
Figure 3-2: KML GeoPolicy: Geometry elements and Condition elements .....	79
Figure 3-3: An example about GeoPolicy conditions for controlling access to GeoChannel resources .....	83
Figure 4-1: The concept of GeoContext Zones (GeoBlindZone, GeoObserveZone, GeoListenZone, GeoServeZone) .....	88
Figure 4-2: Partition and discriminate GeoContext Zones with different GeoPolicy conditions for resource access control or for scalable on-demand communication .....	91
Figure 4-3: The Geo-off-on communication model with adaptive pull (request/respond) & push (publish/subscribe) patterns.....	95
Figure 5-1: GeoAccessFeed system components relational model .....	101
Figure 5-2: GeoAccessFeed working principle & components relational model .....	103
Figure 5-3: Functional components of a GeoChannel AccessFeed.....	107
Figure 5-4: An example of road network with adjacent segments illustrating TopologyLink concept and function .....	115
Figure 5-5: All cooperative approaches work jointly for getting the latest AccessFeed (update).....	120
Figure 5-6: Example about the phenomena of GeoBlindZone and GeoLagZone .....	123
Figure 6-1: The Geoww-URI schema model “geoww://”: syntax & semantics for component parts .....	144
Figure 7-1: The components framework for GeoChannel discovery and subsequent access to resources	

.....	155
Figure 7-2: The analogy of discovery mechanisms for the Web and for GeoChannel resources.....	157
Figure 7-3: An example of encoding a Geoww-URI using QR Code for identifying and discovering a GeoChannel resource .....	158
Figure 7-4: The Eddystone beacon protocol for a Physical Web .....	160
Figure 7-5: Eddystone beacons facilitate discovering GeoChannels in the physical world .....	162
Figure 7-6: The roles and workflow in the PubSubHubBub protocol.....	169
Figure 7-7: The component roles and workflow for harvesting GeoChannel catalogue metadata.....	172
Figure 7-8: Extended-KML encoding model for OpenSearch responses & GeoChannel Objects.....	182
Figure 7-9: Context topology network – an indoor street view example .....	185
Figure 8-1: An example depiction of GeoChannel resources with users matching access-organize conditions .....	190
Figure 8-2: : An example of GeoChannel network with connected nodes of users and resources.....	190
Figure 8-3: An example of GeoChannel network bridged by resource and users .....	190
Figure 8-4: A same web resource acts as different GeoChannel resources for grouping and working for clients in different GeoClusters.....	195
Figure 8-5: GeoBridge for union of multiple GeoClusters on a same target resource .....	200
Figure 8-6: The example of client networks within GeoClusters.....	200
Figure 8-7: The example of cross-GeoCluster client networks enabled by the GeoBridge mechanism ..	200
Figure 8-8: An example of federating Geo-Compatible resources on different clients at different time points/periods and forming a network of resources.....	203
Figure 8-9: An example of GeoChannel network for discovering and navigating resources.....	207
Figure 8-10: An example of GeoChannel network for discovering and navigating clients .....	208
Figure 8-11: A conceptual depiction of the GeoChannel Web networks .....	209
Figure 8-12: A full view for the GeoChannel Architecture Model .....	218
Figure 8-13: The components workflow framework for Geonoon platform.....	226
Figure 8-14: Component modules of GeoAccessFeed technology .....	227
Figure 8-15: GeoPolicy processor: component modules and workflow.....	229
Figure 8-16: Controller for Geo-Off-On communication model with adaptive pull & push patterns .....	230
Figure 8-17: AccessFeed processor components relational model.....	232
Figure 8-18: Geoww-URI parser for processing Geoww-URIs .....	233
Figure 8-19: The components framework for GeoChannel Catalogue Service.....	235
Figure 8-20: The software stack for GeoChannel Web Engine.....	238
Figure 8-21: Geoww-URLs for identifying graph nodes (of resources and clients) and AccessFeeds for controlling their connection relations in GeoChannel Web (networks) dynamically.....	241
Figure 8-22: GeoBridge of GeoClusters and GeoFederation of resources .....	242
Figure 8-23: The example of GeoChannel networks enabled by GeoCluster and GeoBridge mechanism .....	242
Figure 9-1: Mapping pickup and dropoff spots of taxi trips in New York City from 2009–2015 (Schneider 2015) .....	248
Figure 9-2: Demo project: Large-scale real-time map and dynamic network of vehicle & roads for New York City.....	250
Figure 9-3: Functionality components architecture for Use Case 1 .....	251
Figure 9-4: Software components framework for reproducing mass vehicle motion processes .....	252

Figure 9-5: Large-scale dynamic and real-time maps of taxi trips in New York City .....	255
Figure 9-6: A travel route with segments represented by a set of related AccessFeeds with TopologyLink modules for controlling vehicle-segment correlation and segment-segment transition .....	256
Figure 9-7: Software components framework for mapping vehicle-road relations along connected road segments.....	258
Figure 9-8: An overview of the road networks in New York City .....	260
Figure 9-9: Road networks need to be transformed into GeoChannel context topology networks .....	260
Figure 9-10: Sample 1: GeoChannel context topology network for the taxi-trip roads in New York City.....	261
Figure 9-11: Sample 2: GeoChannel context topology network for the taxi-trip roads in New York City.....	261
Figure 9-12: Example 1: Mapping dynamic & random relations into GeoChannel networks for roads and vehicles.....	262
Figure 9-13: Example 2: Mapping dynamic & random relations into GeoChannel networks for roads and vehicles.....	262
Figure 9-14: Inter-vehicle networking interaction: application of dynamic relations on GeoChannel networks .....	263
Figure 9-15: The audio guide system in the British Museum .....	265
Figure 9-16: British Museum collection viewable on Google Street View Maps.....	266
Figure 9-17: ‘The Museum of the World’ microsite for exploring connections between the world’s cultures.....	266
Figure 9-18: Spherical coordinate system and Equirectangular projection of panorama .....	268
Figure 9-19: Different contexts (e.g. location, heading, pitch) of users viewing panoramas .....	269
Figure 9-20: Different context (e.g. location, heading, pitch) conditions for audio resources .....	269
Figure 9-21: Eddystone beacons deployed in the museum for broadcasting Geoww-URIs .....	271
Figure 9-22: Vagile Eddystone beacons with reconfigurable Geoww-URIs for testing pervasive services in physical world .....	271
Figure 9-23: Coordinate systems for smart devices to get context attributes for on-site tour in British Museum.....	272
Figure 9-24: Eddystone beacons identifying objects can cluster visitors nearby .....	274
Figure 9-25: The GeoChannel network with resources and clients in physical world .....	274
Figure 9-26: The GeoChannel context topology network for online viewers .....	275
Figure 9-27: GeoBridging nearby GeoClusters on the context topology network for cross-GeoCluster interaction.....	275
Figure 9-28: Joining the physical and virtual world ---- GeoChannel network for clustering and connecting resources & clients for pervasive services and participative interaction .....	277
Figure 9-29: The software components framework for the Use Case 2 .....	278
Figure 9-30: Demo project: Virtual-Real-world services and interaction in the British Museum.....	282
Figure 10-1: Research contributions: a set of new techniques and facilities for the GeoChannel Web ...	290
Figure 10-2: Answer to the research question: an GeoChannel architecture for correlating things dynamically .....	300
Figure 10-3: The GeoChannel Web differs from other types of Web .....	302
Figure 10-4: The GeoChannel Web adds a Context-Organize Layer for the general Web .....	303



## List of Tables

Table 2-1: The deficiencies of current techniques for the GeoChannel Web functionality .....	64
Table 3-1: GeoPolicy Language – functional elements for representing conditions of context .....	72
Table 5-1: GeoChannel AccessFeeds workflow process model: events-actions mapping .....	119
Table 7-1: Interface model mapping between OGC CSW and GeoChannel Catalogue Service .....	167
Table 7-2: OGC proposal of geospatial & temporal extension to OpenSearch query parameters in a search request .....	175
Table 8-1: Geo-Clustering clients and federating resources on a GeoChannel network .....	191
Table 8-2: The concepts comparison between the general Web, Social Web and GeoChannel Web .....	211
Table 8-3: The software components of GeoAccessFeed processor (at front end) .....	228
Table 8-4: The software implementation of GeoChannel Discovery Services .....	234
Table 8-5: The software implementation of updating AccessFeeds .....	237
Table 8-6: The software implementation of GeoChannel Networks add-ons .....	241
Table 9-1: The use cases demonstrate the functionality, capability and usage of the GeoChannel Web technology .....	284
Table 10-1: Answering the research sub-question 1 .....	296
Table 10-2: Answering the research sub-question 2 .....	297
Table 10-3: Answering the research sub-question 3 .....	297
Table 10-4: Answering the research sub-question 4 .....	298
Table 10-5: Answering the research sub-question 5 .....	298
Table 10-6: Answering the research sub-question 6 .....	298
Table 10-7: Answering the general research question .....	299
Table 10-8: Analogy on related technical components between the GeoChannel Web and the general Web .....	303

## List of Listings

Listing 3-1: an example of GeoPolicy conditions demonstrating the usage of the KML GeoPolicy language. ....	82
Listing 3-2: an example of GeoPolicy demonstrating the GeoPolicy language capability of expressing complex spatial conditions .....	84
Listing 5-1: The outline structure of an AccessFeed element within a GeoChannel element .....	109
Listing 5-2: The <Accessibility> element syntax in extended-KML encoding for AccessFeed .....	111
Listing 5-3: Geometries as geo-tags for space-based search to identify AccessFeeds of interest .....	114
Listing 5-4: The <TopologyLink> element syntax in extended-KML encoding for AccessFeed .....	116
Listing 5-5: The <UpdateAccessFeed> element in extended-KML for updating an AccessFeed using two patterns .....	127
Listing 5-6: The <Pull> and <Push> elements syntax for the Geo-off-on communication model .....	129
Listing 5-7: GeoChannel AccessFeed <Pull> element: syntax and content exemplification .....	136
Listing 6-1: Examples of AccessFeeds used for exemplifying the “geoww://” URI scheme .....	145
Listing 6-2: An example of supporting Geoww-URIs in KML Placemark balloons .....	151
Listing 7-1: GeoChannel metadata encoding model extending Atom/RSS for harvesting GeoChannel catalogue metadata .....	170
Listing 7-2: An example of GeoChannel metadata feed for harvesting GeoChannel catalogue metadata via PubSubHubBub protocol .....	171
Listing 7-3: The syntax for the information structure of GeoChannel OpenSearch service coverage included in OpenSearchDescription document .....	178
Listing 7-4: an example of GeoChannel OpenSearch description document .....	178
Listing 7-5: The syntax for the information structure of GeoChannel OpenSearch responses (results) in extended-KML encoding .....	183
Listing 8-1: The AccessFeeds example of a same target resource for different GeoChannel resources and user GeoClusters .....	194
Listing 8-2: An example of some AccessFeeds that mutually declare the GeoBridging of GeoServeZones/GeoClusters on a same target resource .....	198
Listing 8-3: An AccessFeed example for illuminating Geo-Compatibility and federation of GeoChannel resources .....	202
Listing 8-4: The controlling mechanisms encoded in <Pull> and <Push> elements for the Geo-off-on communication model .....	231
Listing 9-1: Example: AccessFeed_1 with TopologyLink mechanism for controlling vehicle-segment correlation and segment-segment transition .....	257
Listing 9-2: Example: AccessFeed_3 with TopologyLink mechanism for controlling vehicle-segment correlation and segment-segment transition .....	257
Listing 9-3: The samples of some AccessFeeds and Geoww-URIs for Use Case 2 on GeoBridging and identifying nearby GeoClusters for cross-GeoCluster interaction .....	280

## Glossary

This glossary only lists the new terms coined and new concepts defined in this research.

<b>Term</b> <i>(defined in Section)</i>	<b>Definition</b>
<b>GeoChannel</b>  <i>(Chapter 2)</i>	<p>GeoChannel is an architectural model or refers to a specific working unit with specific context conditions, for organizing, discovering, access-controlling, clustering and federating resources and clients.</p> <p>From a working unit perspective, a GeoChannel involves its target resource, access condition and current users that meet this condition, which can dynamically correlate the target resource with a latest cluster of users. In terms of the architectural perspective, the GeoChannel architecture involves a set of new component facilities that can fit into and enhance the current Web fabric to work for the GeoChannel Web populated by crowdsourced dynamics, pervasive services and clustered interaction.</p>
<b>Context,</b>  <b>GeoContext</b>  <i>(Chapter 2)</i>	<p>Context (or GeoContext) refers to certain attributes of resources or clients.</p> <p>Note: there is not essential difference of usage between the two words when used in this thesis document. Sometimes GeoContext explicitly refers to context attribute in the physical world.</p> <p>Context attributes usually act as the parameters of a defined condition for bridging things, e.g. for controlling access to resources, clustering clients, throttling communication between resources and clients, etc.</p>
<b>GeoAccessFeed</b>  <i>(Chapter 5)</i>	<p>GeoAccessFeed is a term for grouping s set of techniques, involving GeoPolicy language, Geo-Off-On communication model, AccessFeed and Geoww-URI. The core part of GeoAccessFeed is AccessFeed technique.</p>
<b>AccessFeed</b>  <i>(Chapter 5)</i>	<p>GeoChannel AccessFeed is the core component of the GeoAccessFeed technology.</p> <p>An AccessFeed is a normative-format file (or message) that contains access-control information involving the context relation conditions between clients and resources. At runtime an AccessFeed virtually functions as a client-side autonomous program unit module, which, on one hand, can locally executes access-control logic for dynamically correlating resources with clients based on context conditions, and on the other hand can work with AccessFeed update Service to get latest updates to this AccessFeed about changes on access-organizing logic for</p>

	<p>adjusting control behaviours.</p> <p>AccessFeed has normative information and working models. This research has extended KML language for encoding AccessFeeds.</p>
<p><b>Geoww-URI</b></p> <p><i>(Chapter 6)</i></p>	<p>By analogy to Http-URL for identifying general web resources for the general Web, Geoww-URI is a new URI encoding scheme and technique for the GeoChannel Web. Geoww-URIs can be used for identifying and discovering GeoChannels (e.g. pervasive services), resources, clients, clusters and their relations.</p>
<p><b>Contextualize, Geo-Contextualize</b></p> <p><i>(Chapter 3)</i></p>	<p>(Geo-)Contextualize refers to defining GeoPolicy conditions for resources, clients or their relations. AccessFeeds can Geo-Contextualize general web resources that may not have their GeoContext originally.</p> <p>Note: there is not essential difference of usage between the two words when used in this thesis document. Sometimes Geo-Contextualize explicitly refers to defining context attribute for physical-world things.</p>
<p><b>GeoPolicy Language</b></p> <p><i>(Chapter 3)</i></p>	<p>A new condition-expressing language for defining context conditions (named GeoPolicy). An extended-KML encoding model has been codified for this GeoPolicy Language.</p>
<p><b>GeoPolicy</b></p> <p><i>(Chapter 3)</i></p>	<p>A GeoPolicy is a context condition expressed by using the GeoPolicy Language. GeoPolicy conditions are used in AccessFeeds for controlling the Geo-off-on communication model, for controlling accessibility of target resources, and for controlling the clustering of users.</p>
<p><b>Geo-off-on communication model</b></p> <p><i>(Chapter 4)</i></p>	<p>The word Geo-off-on implies context-based switch (on/off) condition for control purpose. The Geo-off-on communication model employs GeoPolicy conditions for controlling and throttling client-resource communication in an adaptive and cooperative manner that can dynamically and automatically adjust communication behaviours based on latest context relations between resources and clients, for improving the efficiency and scalability of computing and networking.</p>
<p><b>GeoClientZone</b></p> <p><i>(Chapter 4)</i></p>	<p>This is a concept and technique for discriminating and classifying clients into different zones (groups) based on their contexts for improving computing/networking efficiency and scalability.</p> <p>A certain GeoClientZone refers to a context (physical or virtual) range/scope that meets a specific GeoPolicy.</p> <p>By applying different GeoPolicy conditions, generally GeoClientZones can involve specific zones e.g. GeoServeZone, GeoListenZone, GeoObserveZone, GeoBlindZone, GeoLagZone, etc., as explained below respectively.</p>
<p><b>GeoServeZone</b></p>	<p>A GeoServeZone refers to a context range that confines a GeoChannel</p>

<i>(Chapter 4)</i>	<p>resource' accessibility to a specific community of users.</p> <p>A GeoChannel resource is accessible to the clients whose contexts fall within its GeoServeZone if they meet the GeoPolicy condition of an &lt;Accessor&gt; element within the AccessFeed of this resource.</p>
<b>GeoListenZone</b>  <i>(Chapter 4)</i>	<p>A context range that is defined (resulted in) by the GeoPolicy enforced on the &lt;Push&gt; element. Client falling within the GeoListenZone can receive real-time updates of the AccessFeed via push pattern of communication.</p>
<b>GeoObserveZone</b>  <i>(Chapter 4)</i>	<p>A context range that is defined (resulted in) by the GeoPolicy enforced on the &lt;Pull&gt; element. Client falling within the GeoObserveZone can get updates of the AccessFeed via pull pattern of communication.</p>
<b>GeoBlindZone</b>  <i>(Chapter 4)</i>	<p>The context range/scope that is excluded from GeoServeZone, GeoListenZone and GeoObserveZone. Clients fall within the GeoBlindZone if they cannot meet all the GeoPolicy conditions for the GeoServeZone, GeoListenZone and GeoObserveZone. Clients in GeoBlindZone need to use other accessorial mechanism to get AccessFeed update rather than using the pull and push communication that defined in the AccessFeed.</p>
<b>GeoLagZone</b>  <i>(Chapter 4)</i>	<p>GeoLagZone generally refers to the context range/scope of GeoObserveZone and GeoBlindZone. Usually there is time latency for GeoLagZone's clients to get latest AccessFeed updates.</p>
<b>GeoChannel Accessor</b>  <i>(Chapter 8)</i>	<p>An Accessor refers to the client-side-running part (component) of a GeoChannel resource (application). So an Accessor is virtually a web-client program, providing individual and custom function (e.g. user interface for users to interact with its corresponding GeoChannel (node) resources. Differing from general web-client programs, Accessors can be equipped with GeoChannel technical mechanism to be able to plug/mashup on the Accessor Bus and can interoperate between Accessors of multiple GeoChannel.</p>
<b>GeoCluster</b>  (GeoClustering)  <i>(Chapter 8)</i>	<p>A GeoCluster (or GeoCommunity) refers to a group of users that can currently meet the GeoPolicy condition for accessing a GeoChannel resource. These users fall within this resource's GeoServeZone. A GeoCluster can be identified by its Geoww-URI.</p> <p>GeoClustering is the process of clustering/grouping a resource's clients into a GeoCluster based on their specific GeoContexts. This is virtually the access-control process of this target resource for filtering clients. The GeoChannel architecture provides a common GeoCluster Service facility and the Geoww-URI technique for the GeoClustering process.</p> <p>GeoCluster can function as a GeoWeb-native mechanism for correlating</p>

	clients for their interaction, to facilitate building and running participative and cooperative applications/services such as social networking.
<b>GeoBridge</b> (GeoBridging) <i>(Chapter 8)</i>	<p>GeoBridge is a mechanism for connecting/bridging multiple GeoClusters to form a bigger GeoCluster (i.e. GeoClusters union) with extended space/time coverage (involvement).</p> <p>The GeoBridge mechanism can work for propagating dynamics, relaying services and connecting users (for interaction) across individual GeoClusters.</p> <p>The techniques of AccessFeed, Geoww-URI and GeoCluster Service can work jointly to support the GeoBridge mechanism.</p>
<b>Geo-Compatibility</b> (Geo-Compatible) <i>(Chapter 8)</i>	<p>Geo-Compatibility (short for GeoContext-Compatibility) is a new concept for characterizing certain similarity or proximity on the GeoContexts of different GeoChannel resources. If there is overlap between the GeoServeZones of two resources, that is, if these resources' access-control conditions can be met by a same client concurrently, then these resources are considered as being Geo-Compatible. They have the potential to interoperate with each other via the Accessor Bus of this client.</p>
<b>GeoFederate</b> (GeoFederating) (GeoFederation) <i>(Chapter 8)</i>	<p>GeoFederating refers to the federating (e.g. integration) of Geo-Compatible resources to achieve compound functions/services of these individual GeoChannel resources.</p> <p>GeoFederating needs to simultaneously meet several conditions: these resources are mutually Geo-Compatible; they have common client(s); and their Accessors are programmed for interoperation.</p> <p>GeoFederation is organized by an AccessFeed that orchestrates resources, bridged by the Accessor Bus of a common client, and conducted via the Accessors of these resources.</p> <p>GeoFederating works for resource-resource dynamic correlation (for compound functions/services), which can be in contrast to access control for client-resource correlation (for pervasive services), and in contrast to GeoClustering for client-client correlation (for social networking or cooperative applications).</p>
<b>GeoChannel Network</b> <i>(Chapter 2, Chapter 8)</i>	<p>GeoChannel resources, clients and access relationships can make up dynamic networks named GeoChannel network whose topology is sensitive to the changes of clients interest, resources/clients' GeoContexts and access-control conditions. GeoChannel network is a dynamic resource-user relationships network on which resource nodes bridge user nodes and user nodes bridge resource nodes.</p>

<p><b>the GeoChannel Web</b></p> <p><i>(Chapter 2, Chapter 8)</i></p>	<p>The GeoChannel Web is a concept that refers to a globe-scale GeoChannel network for mapping and applying dynamic, random and context-based relations between real-world things. The GeoChannel Web features highly dynamic network topology.</p> <p>As a new type of Web, the GeoChannel Web is a new architectural system with a set of enabling technical components designed in this research.</p>
<p><b>Context-Organize Layer</b></p> <p><i>(Chapter 8)</i></p>	<p>The GeoChannel technology architecture is virtually an organizing system based on contexts of things. From a systematic perspective and high-level abstraction, essentially all the GeoChannel architectural components (designed in this research) can constitute a Context-Organize Layer, for organizing, correlating and bridging things dynamically and conditionally based on their latest contexts.</p> <p>The Context-Organize Layer is a new architectural facility that can fit into the general Web fabric for extending the functionality of the general Web to support various context-based applications and services.</p>

# 1 Introduction

The Web (World Wide Web) correlates things in an extensive world. Theoretically and technically, the Web can bridge and connect anything; however, due to the consideration or concern on security, privacy, necessity of business, throttling resources of networking and computing, etc., control mechanisms are often applied for organizing conditional connection and interaction.

The current Web correlates things in some ways, for example, based on IP domains, organization or business boundary, social or interest relations, relevance of content, etc. Usually the correlation is organized in a pre-configured way. For example, hyperlink relations between webpages, friend or interest relation on social media/platforms, inner- or inter-connection within or cross companies, etc. In these scenarios, these relations reflected by the current web are relatively static or stable. Increasingly, there are also dynamic relations organized by the Web. For example, the paradigms of discovering and connecting nearby people based on their real-time positions, and searching or recommending content or goods according to clients' interest or demands, etc. These paradigms demonstrate rudimental usage of the current Web for correlating things in somewhat dynamical ways, especially for an information world.

The physical world is populated by massive dynamic, random and context-based relations, which may changes over various context attributes such as space, time, interest or other sensor conditions. For example, the relations between a bus and passengers, between a road and vehicles or pedestrians, between a shopping shelf and customers, between an exhibition booth and visitors, between a real place and online map viewers who may pan and look maps to browser this venue, etc. Essentially the dynamic and random relations between real-world things are also a kind of resources that can be mapped and applied for supporting extensive novel applications and services. The growing availability of mobile sensors, which can capture dynamic context of various attributes, is enabling a rising field of context-based applications, for example, pervasive services (Strimpakou et al. 2006) and smart city (Neirotti et al. 2014), etc.



This research proposes the GeoChannel Web, as a new type of Web with a set of enabling techniques and facilities, for mapping and applying dynamic and random relations in the real world, to systematically support various novel applications and services. Essentially the GeoChannel Web adds a Context-Organize layer into the architectural stack of the general Web, for bridging real-world things with their latest context conditions.

This introductory chapter of the thesis firstly presents the research motivation by identifying the gap between the usage of the current Web and the need of mapping and applying real-world dynamic relations. The research questions are then formulated. And then some notes are given about this research's outcome, project scope and targeted end users. This chapter concludes by briefing the structure of the thesis.

## **1.1 Motivation**

This section firstly reviews the usage of the current Web that can be categorized into some types, which have some common features and also have their own peculiarity. Then an observation is made on the physical world featured by massive dynamic and random relation between things. A new type of Web with corresponding capabilities is then proposed to extend the current Web to cater for networking real-world things with dynamic context conditions.

### **1.1.1 The current Web: some types of usage**

The Web (World Wide Web) has been evolving in terms of its techniques and usages, from originally hyperlinking documents to currently connecting varieties of things. Progressively new ideas with enabling techniques are incorporated into its architectural stack to make the general Web more useful and competent for increasing fields. There is only one general Web, but with various focuses and usages.

For investigating the state of art of the Web, this section presents a comprehensive review which categorizes the development and usages of the Web into some phases or types, by employing some terms (or jargons) for characterizing their peculiarities. Usually a later-emerging type does not supersede existing types, but adds new facility and functionality into the architecture of the general Web for meeting new kinds of applications and services.

### **1.1.1.1 The Resource Web**

#### ***1.1.1.1.1 The origin of the Web***

The Web (World Wide Web) originated for organizing the relation of digital documents to facilitate navigating users between web pages. So it was initially a Document Web as basically the hyperlinked entities were documents. The first web site (Berners-Lee 1989) of the WorldWideWeb (W3) project (Berners-Lee & Cailliau 1990) is an information retrieval initiative aiming to give universal access to a large universe of documents. Progressively the connected entities have become generalized into general resources (such as multimedia content, web services, software components or code, etc.), so here the term ---- the Resource Web is used for this type of Web for characterizing its usage, which is a global collection of documents and other resources served on the Internet that is a global system of interconnected computer networks.

The Resource Web is the basic and most frequently-used type of the general Web, and functions as the foundation stone for some other Web types (that will be reviewed in subsequent sections). Its basic functionality and usage is for retrieving resources and providing services. Actually we are largely using and working on the Resource Web nowadays. All the web sites, applications and services can fall into this category. Among these, it is most mentionable that web search engines (services) work with the Resource Web, for discovering and retrieving various web resources.

#### ***1.1.1.1.2 The network & size of the Resource Web***

A big graph network can map the structure of the Resources Web, with nodes representing individual web resources and edges mapping hyperlink relations. From an outlined view, the Resource Web comprises hyperlinked resources, which make up a structure of networks. This logical model can be represented by a massive graph structure with nodes and directed edges for mapping the hyperlink relations between web resources.

Virtually a search engine can build a massive network graph reflecting the hyperlink relation among resource nodes. How big is the Resource Web? As of 2016 June, it was estimated



### **1.1.1.2 The Geospatial Web**

The Geospatial Web (GeoWeb for short) has been evolving by progressively being equipped with geospatially-enabled facility and functionality since the convergence of the technologies about the Web (World Wide Web) and GIS (Geographic Information System).

#### ***1.1.1.2.1 The Web with geo-context***

The concept, facility and practice of the GeoWeb is defined and characterized in literature from different visions and perspectives with specific focuses respectively. Lake (Lake & Farley 2007), the inventor of GML, observes the GeoWeb is an integrated, discoverable collection of global general services and data that support the use of geographic data in a range of domain applications. This opinion that emphasizes data services/applications thinks the global, national, or local Spatial Data Infrastructure (SDI) (Nebert 2004, GSDI 2013, EuropeanCommission 2007) are each instances of the GeoWeb. Reed (2007) as the CTO in OGC (Open Geospatial Consortium) believes the GeoWeb should go further beyond data services and SDIs, as it is not just a large number of mash-ups or even the hundreds of SDI's that have been successfully deployed, but is about the complete integration and use of location at all levels of the internet and the web. Scharl & Tochtermann (2007) envisions that the GeoWeb will have a profound impact on managing knowledge, structuring workflows within and across organizations, communicating with like-minded individuals in virtual communities; and they will not only reveal the geographic distribution of Web resources and services, but also bring together people of similar interests, browsing behaviour, or geographic location.

#### ***1.1.1.2.2 Pervasive Location-based Services***

Weiser (Weiser 1993) introduces the area of ubiquitous computing, which is sometimes also described as pervasive computing (Satyanarayanan 2001, Saha & Mukherjee 2003) that provides pervasive services. This pervasive (ubiquitous) computing paradigm reflects a vision of people and environments augmented with computational resources that provide information and services when and where desired (Abowd & Mynatt 2000). Context (Henricksen et al. 2002) is the main condition element for invoking pervasive services.

Location-Based Services (LBSs) belong to the spectrum of pervasive services, where the context mainly involves space/time attributes about users and/or service resources. Location-based services can be defined as services that depend on and are enhanced by positional information of mobile device (Hirsch et al. 2006). LBSs are mobile services for providing information or functional/application services that are selected or filtered under consideration of the users' contexts such as current locations or those of other persons or mobile devices (Kupper 2005).

Pervasive Location-Based Services termed in this research refers to LBSs populating the virtual and physical worlds pervasively. That is, when users/people interface with web maps or stay/move in the physical world, context-sensitive services (applications) are available anywhere and anytime. Geospatial location information (e.g. physical positions or map-view status) can facilitate the capability to discover, filter, customize, personalize, contextualize and even automatically trigger/drive on-demand and relevant services.

#### **1.1.1.3 The Social Web**

The Social Web (W3C 2010) is a set of relationships that link together people over the Web.

##### ***1.1.1.3.1 Social Media & Social Networking***

Social media (Ahlqvist et al. 2008) refers to the means of interactions among people in which they create, share, and/or exchange information and ideas in virtual communities and networks. Online Social Networks (OSNs) for social-networking services that connect people for interaction, are a form of social media, which gain in popularity increasingly, for example currently some representative products/platforms such as Facebook, Google plus +, and Twitter, etc.

The functions and advantages of social media involves many aspects, for example, quick and immediate communication; ability to reach large audience; easy publishing and distribution of content; minimal or no cost; interactive capabilities; community, friendship, and relationship building opportunities; repository for audience feedback; etc.

#### ***1.1.1.3.2 The GeoSocial Web***

Social Network Sites with location-based features, e.g. including location information into shared contents, are referred as Location-based Social Networks (LBSNs) (Roick & Heuser 2013), which are boosted by growing penetration of GPS equipped devices such as smartphones allowing users to constantly share geographic location information on their current whereabouts. This has led the way to an augmentation of existing Social Network Sites with location-based features.

The Location-based social networks, namely GeoSocial networks, comprise a GeoSocial Web, which takes location information as a new dimension to Social Web, with geo-tagged user-generated media such as texts, photos, and videos, etc., and for recording location history of users. Essentially, location as a GeoContext in the social networks bridges the gap between the virtual and physical world. And location context can functions as bridges/links that may form the new social structure made up of individuals connected by the interdependency derived from their locations in the physical world as well as their location-tagged media content (Zheng & Zhou 2011). GeoSocial networks make use of users' location information to facilitate or enhance social networking functionality or for added value of commercial activities such as marketing for merchants.

Some methods are employed for associating locations with users on GeoSocial networks. One method is automatically collecting users' positions status to the central server(s) and may make them visible to all or part of other users. An example is the functionality of "finding people nearby" provided by some social media that allow a user to initiate an interaction (such as chat) to other users identified by their locations.

Another method, check-in, enables a user to "check in" to the venues listed in GeoSocial application software such as Foursquare, from which this user can choose places based on individual preferences, specific incentives, etc. Currently check-in(s) on stationary POI (places of interest) mainly work for sharing location information with friends, and for proximity-based information filter or merchant places recommendation such as products/services promotion.

Check-in functions as a mechanism for correlating users with location context, and for grouping people on a same POI (Point of Interest) virtually. In effect the check-in mechanism can form dynamic groups of participants, which differ from relatively-static social groups based on memberships. This dynamic clusters/communities effect makes analysing check-ins on GeoSocial Networks a meaningful methodology for research (e.g. Cramer et al. 2011, Scellato et al. 2011b, Melia 2012, Malmi et al. 2012, and Brown et al. 2012) on the real-world status about events and interactions.

As reviewed briefly above, social media (networks) have strength on the capability of connecting people and organizing interactive activities for collective intelligence and joint work. Currently many application practices leverage the social networking capability to research on geospatial status, for example, for understanding the dynamics (Quercia et al. 2010, Cranshaw et al. 2012, Kling & Pozdnoukhov 2012) or mobility (Cho et al. 2011, Gao et al. 2011, Noulas et al. 2012) of cities or communities; or to support collaborative GeoWeb applications, for example, Vervoort et al. 2010, Sani & Rinner 2011, etc.

#### **1.1.1.4 The Sensor Web**

The term Sensor Web is being used to describe a middleware layer between sensor networks and applications: "Web accessible sensor networks and archived sensor data that can be discovered and accessed using standard protocols and application programming interfaces (Botts et al. 2007, Broring et al. 2011) ".

##### ***1.1.1.4.1 Sensor Web Enablement***

The notion of "Sensor Web" has been largely influenced by the Open Geospatial Consortium (OGC) Sensor Web Enablement (SWE) initiative (OGC ), which is develop "an infrastructure that enables an interoperable usage of sensor resources by enabling their discovery, access, tasking, as well as eventing and alerting within the Sensor Web in a standardized way". Thus, the Sensor Web is to sensor resources what the WWW is to general information sources—an infrastructure allowing users to easily share their sensor resources in a well-defined way (Nittel 2009).

The SWE initiative standardizes a suite of web services by defining their information and

interface models. The SOS (Sensor Observation Service) (OGC 2012) enables querying and inserting measured sensor data and metadata; the SAS (Sensor Alert Service) (Simonis 2006) and its successor the SES (Sensor Event Service) (Echterhoff & Everding 2008) push sensor data to subscribed clients; The SPS (Sensor Planning Service) (Simonis & Echterhoff 2011) enables tasking of sensors (e.g., setting the sampling rate of a sensor). Discovery of sensors is supported by implementations of Sensor Instance Registry (SIR) (Simon Jirka 2010) and Sensor Observable Registry (SOR) (Jirka et al. 2010). This research mainly involves the technique about Sensor Event Service.

#### ***1.1.1.4.2 Sensor Event Service***

The functional objective of “comprehensive dynamics” in this research calls for architectural facility for sourcing and disseminating real-time (low-latency) geospatial events. This research explores a Sensor-Event-Service equivalent component to be natively built into the GeoWeb fabric. This facility component is expected to comply with and even generalize technical and functional requirements for standard Sensor Event Services. So this section reviews related specifications in this domain.

There have been some OGC's efforts on standardizing sensor (or general) event services to achieve push-based and asynchronous communication (Broring et al. 2011). This enables eventing, i.e., the automatic publication of data that is of interest for the user (without him having to repeatedly pull for that data), and alerting, which incorporates the publication of more significant data to the user who is supposed to react in a domain or application specific way upon receipt of this kind of data.

#### ***1.1.1.4.3 Citizens as Sensors & VGD***

Geospatial dynamics can be sourced from physical sensor devices deployed in our environment, or from human sensors populating this planet. Citizens-as-voluntary-sensors (Goodchild 2007a) supplies a paradigm to enable a citizen sensor network (Sheth 2009), which refers to an interconnected network of people who actively observe, report, collect, analyse, and disseminate information via text, audio, or video messages. Crowdsourcing, citizen sensing and sensor web technologies contribute to live geography (Resch et al. 2011) and active environment (Boulos et al. 2011). In complement to in-situ sensor devices and remote sensing platforms, an approach, i.e. SWE of VGI, is identified as an essential step towards the implementation of a Digital



Earth's Nervous System (Longueville et al. 2010).

The notion of citizens as sensors is essentially the similar to the VGI paradigm. To highlight the dynamic nature of citizen-contributed geospatial events, here an alternative term, VGD (Volunteered Dynamics), is proposed here, which, to some extent, differs from VGI that also involves static/archived data sets; while VGD basically involves dynamical updates about our physical or social environment.

A typical example of VGD is crowdsourcing traffic information on Waze (Google 2013c). Waze as previously an Israeli start-up company has been acquired by Google and now its real-time traffic information integrated into Google's platform systems such as Google Maps. Waze just leverages the power of crowdsourcing that travellers submit and use real-time traffic information at a global range.

Currently user-generated content, including VGI or VGD, is largely communicated via social media, or constrained at specific applications and operated on specialized systems/platforms (e.g. Waze for traffic). This research will explore GeoWeb-native and general mechanism and facility to facilitate publishing and utilizing comprehensive geospatial dynamics contributed and consumed by general public users.

### **1.1.2 The physical world: dynamic, random and context-based relations**

The subsections below firstly observes the physical world with massive dynamics, and then proposes a new type of Web for mapping and applying dynamic, random and context-based relations.

#### **1.1.2.1 Real-world relations with context attributes**

The physical world is populated by massively dynamic and random relations between things. Here the meaning of some words are clarified as following.

$$\textbf{Relation} = f(\text{context}_a, \text{context}_b)$$

**Relation:** the word “relation” in question here is based on some context attributes of things involved, for example, space, time, speed, orientation, scope of a map view or any others

detectable attributes by various sensors (or by other ways). A specific kind of relation usually involve its concerned context attributes conditions. Giving some real-world scenarios: in a museum, the visitors related to items that are exhibited; in a shop, the customers related to goods on shelves; in a city area, the passengers related to buses, and the people (passengers or pedestrians) and vehicles related to roads; etc.

**Dynamic:** means that the relations changes over things' context. A dynamic relation is often caused by dynamically-changing context attributes that this relation is based on. "Dynamic" does not only mean physical motion or movement, but may also mean a change, activity or process that involves its changing context attributes other than space/time context. For example, in a botanical garden, even though it seems there is not so many obviously perceivable motions or movements like those on a highway or sport fields, there is still massive dynamic processes, such as the germination of seeds, growth of trees, evolvement of soil elements, colour change of leaves, etc. And in human society, the changes on relationship of friends, colleagues, business partner, shared interests, etc.

**Random:** means that the relations between things may happen or change irregularly, or without planning, predicting or in-advance arranging. A random relation is often caused by randomly-changing context attributes on which this relation is based.

Some real-world scenarios about dynamic and random relations based on context attributes are exemplified. Figure 1-3 depicts the relation between observers with smart eyeglasses and their ambient environment. Smart eyeglasses have built-in sensors that can capture some context attributes for the observers, such as space position, visual angle (heading, pitch), etc. An observer's viewing behaviour is usually highly random in terms of place locations and view scopes, which result in random and dynamic relations between an observer (namely client) and the sights (namely resources) that he/she can see. In this case, the context attributes (e.g. position, heading and pitch of views) can determine or control the correlation between the resources and clients.

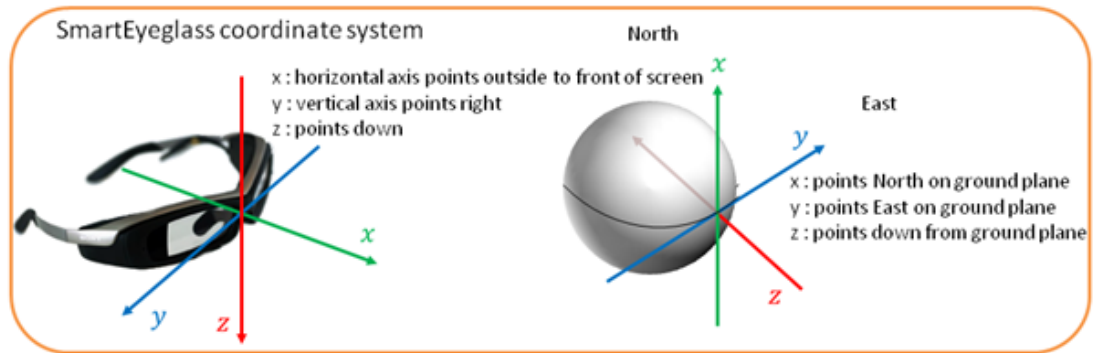


Figure 1-3: Smart eyeglasses observe physical-world resources from changing positions and view angles dynamically and randomly

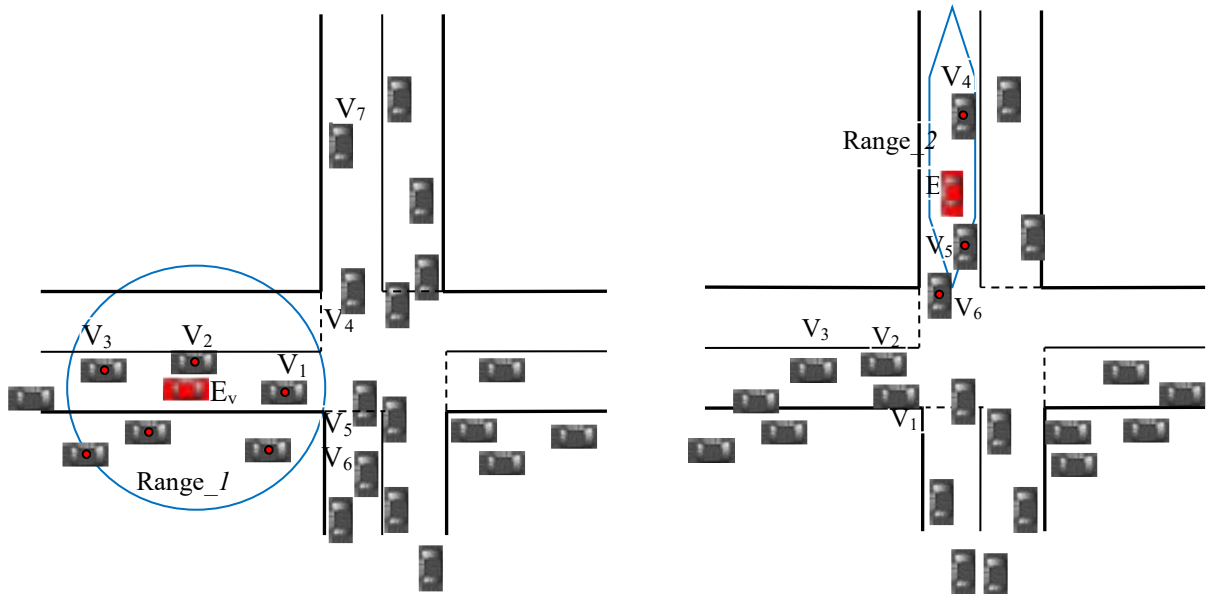


Figure 1-4 (a): Members:  $V_1, V_2, V_3$

Figure 1-4 (b): Members:  $V_4, V_5, V_6$

Figure 1-4: Dynamic and random relations between roads and vehicles

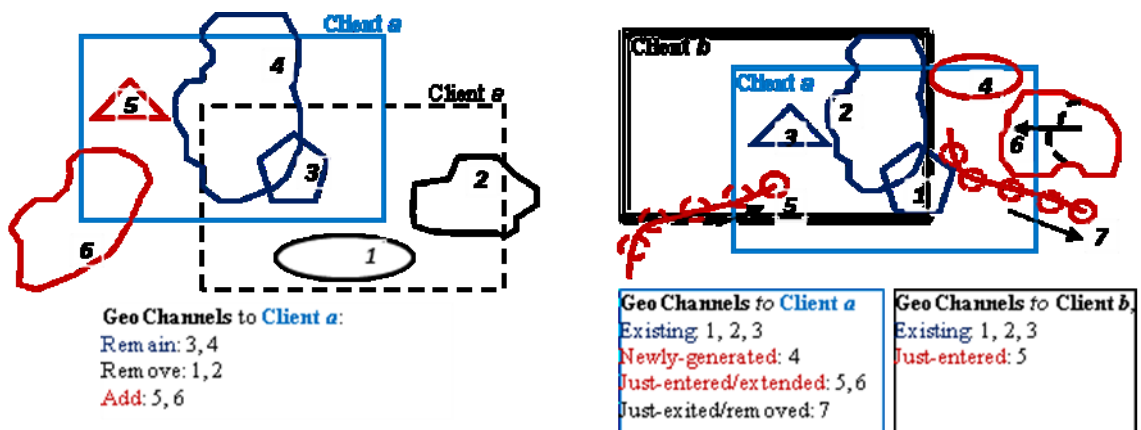


Figure 1-5 (a): Moving viewport (client)

Figure 1-5 (b): Moving objects (resources)

Figure 1-5: Dynamic maps: real-time maps delivering on-demand dynamics

Figure 1-4 exemplifies a traffic scenario where dynamic and random relations take place between roads and vehicles, and between vehicles and vehicles. Assuming the red car is a resource (e.g. acting as a roaming ad hoc network platform), and nearby cars (acting as clients) can be clustered and connected when they meet some specific context conditions. At a specific time moment, those that have been clustered are depicted in the red-dot vehicles. When the red-colour vehicle moves, or when it changes the clustering condition, this cluster may eliminate some previously-clustered vehicles, which now cannot access this roaming ad hoc network, and new members of vehicles may get involved for participating in this cluster. In this case, the contexts (i.e. position, speed, direction, etc.) of the vehicles affect the cluster relations.

Another example is shown in Figure 1-5, which depicts dynamic maps for viewers (namely clients) to observe moving objects (namely resources). The viewing behaviour (e.g. panning, zooming) of viewers (namely clients) and the movement of objects are both random, which result in dynamic access relations (i.e. visible or invisible) between resources and clients. A client's context can be represented by the viewport parameters, and a resource's context is its latest position. Their contexts relation determines their dynamic access relation.

More examples about dynamic and random relations based on context attributes can be given. For example, the relations between a bus and passengers, between a shopping shelf and customers, between an exhibition booth and visitors, between a real place and online map viewers who may pan and look maps to browser this venue, etc.

#### **1.1.2.2 A new type of Web for dynamic & context-based relations**

As exemplified in last section, various dynamic, random and context-based relations populate the physical world, which may changes over various context attributes such as space, time, interest or other sensor conditions. Essentially the dynamic and random relations between real-world things are also a kind of resources that can be mapped and applied for supporting extensive novel applications and services. The growing availability (as shown in Figure 1-6) of mobile sensors, which can capture dynamic context of various attributes, is enabling a rising field of context-based applications, for example, pervasive services (Strimpakou et al. 2006) and

Page 33 of 373

types of Web.



Figure 1-7: The peculiarity and diversity of some types of Web

The GeoChannel Web will feature highly dynamic network topology, which reflects the real status of the physical world populated and affected by various dynamic and random relations based on various contexts. The GeoChannel Web can extend the current Web to cater for networking real-world things with dynamic context conditions.

### 1.1.3 Research Objectives and Tasks

The **functional objective** of this research is to experiment towards a new type of Web, the GeoChannel Web, for mapping and applying dynamic, random and context-based relations between real-world things to support various context-based applications and services, for example, comprehensively real-time dynamics, pervasively location-based services and GeoClustered users interaction.

For fulfil this functional objective, the **technological aim** of this research is to explore a new architectural system model with enabling technical components for achieving context-based dynamic correlation between real-world things in a context-aware, on-demand, automatic and scalable way.

The **research tasks** will involve:

- 1) to frame a set of new concepts with new terminologies for describing and characterizing new ideas, models and systems;
- 2) to conceive a general architectural system model with components and workflow models;
- 3) to design a series of new techniques and facilities for these architectural components;
- 4) to implement a prototype system for this new architectural model;
- 5) to design and develop use case projects for demonstrating the technology and functionality of this new system architecture;
- 6) to conclude this research work about contributions, assessment and suggestions.

## 1.2 Research Questions

The general aim of this research is to develop an architecture model with enabling techniques and facilities towards implementing a new type of Web featuring its functionality objective (described in section 1.1.3). The analysis on its functional components and technical challenges raises research questions to be addressed in this project.

The main research question can be formulated as below:

**What architectural system can conduct dynamic correlation of real-world things based on their context conditions**, so as to improve the current Web for mapping and applying real-world dynamic and random relations to support various context-based application and services?

This general question implies the need to design a new architectural infrastructure that may constitute a new type of Web, which can facilitate bridging real-world things by mapping and

applying their dynamic and random relations.

To answer this general research question above, some technical constituents have been identified, which make up the following sub-questions correspondingly.

**1) What concept model with enabling components framework can work for mapping and applying dynamic and random relations between real-world things?**

Answering this question is to frame a common concept that can generalize and characterize real-world things with dynamic relations in terms of their representation and identification, and to architect a component framework model for their discovery, connection and interaction, based on and controlled by latest context conditions.

**2) What language can be generically competent for representing context conditions between real-world things?**

This question is to design a general-purpose language with normative syntax and semantics for defining context relation conditions. This language should be expressive enough for expressing various context attributes that can be captured currently, and should be extensible to support future sensors that can detect more kinds of context attributes.

**3) What model can achieve adaptive communication between mutually interested things based on their latest context relations?**

This is to design an adaptive communication model for throttling communication traffic between things based on their latest context relation, which can automatically adjust communication patterns or frequency for improving networking and computing scalability.

**4) How can real-world things be dynamically correlated based on their latest context?**

This question is to design a mechanism for dynamically connecting or disconnecting things with context conditions. This mechanism is critical for access-controlling resources to enable pervasive services and for improving scalability of dynamic and on-demand systems.

**5) What URL model can work for identifying things and relations with their latest context**



conditions?

This question raises a task to design a novel URL model for encoding specific format URLs for contextualizing and identifying resources, clients or their relations. Different from traditional Http-URL model for the general Web, this new URL model will work for a new type of Web explored in this research.

**6) How can contextualized things be discovered in a context-aware way?**

Discovering things (for example, resources or clients with their context conditions) is literally to get their URLs encoded by employing a certain URL model that Question 5 explores above.

Multiple means are to be investigated, which may involve virtual-world and physical-world approaches, for example, searching catalogue service or detecting sensors of signals. So the work needs to design information models for catalogue metadata and for update/query messages, as well as interface model for search services, and the techniques for encoding specially-modelled URLs into visual tags or wireless signals for detecting.

### **1.3 Research Outcome**

This research contributes a new architectural infrastructure, termed the GeoChannel Web, with a set of enabling techniques and facilities. The research outcome mainly comprises the following components:

1) The concept and model of the GeoChannel Web, for mapping and applying real-world dynamic and random relations between things, to broadly support various novel applications and services.

The GeoChannel Web can weave real-world things (e.g. resources, clients and access relationships) into a global-range dynamic network that can facilitate discovering, federating resources and conducting dynamically clustering and connecting for interaction.

This new type of Web comprises a set of practically technical components invented as bellow.

2) GeoPolicy Language, an expressive language for defining context conditions to bridging real-world things populating the GeoChannel Web.

In some sense, the GeoPolicy Language for the GeoChannel Web can be analogized to HTML for the general Web, but the two languages work for different purposes.

3) Geo-off-on communication model, an adaptive communication model for throttling communication traffic between related nodes of things.

By analogy to Http protocol for the general Web, the Geo-off-on Communication Model works for the GeoChannel Web.

4) GeoAccessFeed, a mechanism for controlling dynamic connection relations between things.

5) Geoww-URI, a novel URI model for identifying and contextualizing things and relations with latest context and conditions.

The Geoww-URI model functioning as a native URI model for the GeoChannel Web is a counterpart of Http URI schema for the general Web.

6) GeoChannel Discovery technology, the means for discovering GeoChannels, i.e. dynamically contextualized resources with condition-matched users, in either physical-world or virtual-world ways.

7) Geonoon, a prototype platform, materializing and implementing these new concepts, models and techniques invented for the GeoChannel Web.

The Geonoon platform supplies common infrastructure to support building, running and using various GeoChannel-enabled Web applications and services, and its technique or facility components can also be fused into the existing Web fabric such as Web mapping systems, geo-browsers, Digital Earth (Virtual Globe) platforms, etc., to improve and enhance their functionality for comprehensive dynamics, pervasive services and context-clustered-users

interaction.

8) Use case projects: two integrative use case projects have been developed and tested for demonstrating the functionality and capability of all the new techniques and facilities devised in this research. These use case projects are representative and relevant, novel and creative, practical and useful. They can inspire more innovative applications and service that can be supported by the GeoChannel Web platform.

## **1.4 Project Scope**

According to the functional and technical objectives proposed (in section 1.1.3) and research questions identified (in section 1.2), the boundary about work involvement can be defined in this research project.

- This research mainly involves the following technical/task scope

The GeoChannel Web architecture with enabling technical components. Overall, this research project mainly focuses on the design of models and mechanisms, e.g. system structures, information schemas, service interface, process procedures or working flows, of this GeoChannel Web architecture. It also involves a prototype implementation of some facility components of this architectural technology, with a basic construction of a prototype platform named Geonoon.

- Out of scope

General theory or specific content about pervasive services. This research does not involve developing general theory or specific content about pervasive services. This research involves designing a mechanism and facility for contextualizing and access-controlling resources to enable context-aware services.

Complete software implementation or practical production platform. This project does not cover the complete software development for implementing all the technical models and for supplying a whole production/commercial platform, which is left for future work.

## 1.5 End-Users of this Research

The GeoChannel Web technology is expected to be widely-adopted and to become a native architectural facility in the Web infrastructure. The end-users of this research fall into two groups.

The first group covers academic researchers and software developers (or vendors) for the Geonoon platform, the general Web, GeoWeb, Sensor Web, Social Web (media), LBS services, and other application/service platforms (such as web mapping systems, geo-browsers, Digital Earths, etc.). The outcomes of this research are the blue print for further developing and implementing a complete, practical and even commercial Geonoon platform. The models, mechanisms and technical facilities can also be fused into, or can inspire the research or development of, other systems or platforms, e.g. Digital Earth. Developers, architects and vendors can employ this new architectural paradigm to easily re-architect existing or develop new applications or services for the advantage of supporting real-time geospatial dynamics, location-based services and geo-clustered social-networking collaboration.

The second group involves suppliers and users of the GeoChannel-enabled services and applications. For example, publishers/subscribers of geospatial (sensor-event) dynamics, providers/consumers of pervasive services, participants of geo-clustered social networking, etc. GeoChannels acting as local resources and communities provide condition-explicit media for supplying, delivering, discovering, and utilizing resources (e.g. information and services), and for engaging in local activities. The general public will benefit from this kind of new mass media facilities that have respectively local focuses and are distributed anywhere, with stationary or mobile coverage of geographic ranges. It is easy to establish, use and participate in a GeoChannel. Anybody can arbitrarily create their own GeoChannels, or randomly join in (or disjoin from) others' GeoChannels. As this GeoChannel architecture expects to democratize the Sensor Web (i.e. sensor event services), to provide the new geo-cluster social media paradigm, and to populate the planet with GeoContext-aware services, ordinary people can publish or exploit globally-available comprehensive dynamics, supply or access pervasively location-based services, collaborate with people within and cross GeoClusters, and can take "the Earth as

Universal Desktop" to interact with the real world. All of these capabilities are delivered by the GeoChannel Web.

## 1.6 The Structure of this Thesis

This thesis document contains *ten* chapters, which involve this research work in several parts and stages. An introductory part (chapter 1) is followed by the main stages (chapter 2 to chapter 8) which design and implement this GeoChannel Web architecture. Subsequently Chapter 9 demonstrates uses cases and Chapter 10 concludes this research.

Each chapter usually begins with an introductory section prefacing its work, and ends with a summary concluding the addressed aspects.

*Chapter 1* sets the general context and introduces the content of this research. By categorizing the current Web types and identifying the gaps of their usage for the physical world, it proposes a new type of Web for mapping and applying dynamic and random relation in the real world. Research questions are then formulated with their objectives, and the research outcomes are outlined with their original contributions.

*Chapter 2* presents an overview on the architectural model of the proposed GeoChannel Web, to set a holistic view facilitating sectional descriptions in later chapters that design and implement its architectural components. It firstly devises new concepts of GeoChannel and the GeoChannel Web, and then give a general view on the components framework with brief analysis about the features or capabilities of new technical facilities to be invented in this research.

*Chapter 3* designs GeoPolicy Language, a new condition-expressing language for defining context conditions (named GeoPolicy) of things on the GeoChannel Web. A core set of the language elements are designed which are extensible for further supporting more kinds of context attributes that may be acquired by future sensors. An extended-KML encoding model has been codified for the GeoPolicy Language.

*Chapter 4* designs Geo-off-on Communication Model, which is a context-aware mechanism for adaptively controlling and throttling communication by dynamically and automatically adjusting communication patterns and frequency based on latest context relation between things, for improving the efficiency and scalability of computing and networking.

*Chapter 5* designs GeoAccessFeed. This generally refers to the GeoChannel Access-Organize

technology, which is a technical system for dynamically bridging things based on their mutual interests and contexts. AccessFeed as the functional entry of the GeoAccessFeed technology has been codified in terms of its information model, workflow model and updating mechanism.

**Chapter 6** designs Geoww-URI schema, a novel URI model for identifying things with their latest context and relation. Geoww URIs introduce context into traditional URIs for identifying GeoChannels (i.e. dynamically contextualized resources with condition-matched users).

**Chapter 7** explores the technical system for GeoChannel Discovery functionality. Two categories of approaches are elaborated, both of which are for eventually getting the Geoww URIs for GeoChannels of interest. The virtual-world approach is to search GeoChannel Catalogue Service, and the physical-world approach is to acquire encoded Geoww URIs from various media (such as QR code labels, RFID tags, WiFi / beacon signals, etc.) from our physical environment. A series of normative models for information, interfaces and workflow processes are designed for GeoChannel Discovery.

**Chapter 8** describes further architectural components with a full model of the GeoChannel Web, and presents Geonoon as a prototype implementation of this architectural model. The GeoChannel Web can constitute a Context-Organize layer fitting into the current Web stack, becoming a new type of infrastructure for novel applications and services with ambient context.

**Chapter 9** demonstrates typical usage, extensive capabilities and promising potentials of the GeoChannel Web technology by exemplifying use cases, which in turn serve as a test bed for validating and evaluating the design of this GeoChannel Web architectural model.

**Chapter 10** concludes this research project by reviewing the work conducted, checking the fulfilment of research objectives and summarizing the research contributions. Future work is also suggested to further improve this GeoChannel Web architectural infrastructure.

## **2 The GeoChannel Web: the Concept & Component Framework**

### **2.1 Introduction**

The research motivation presented in Chapter 1 has identified the gap between the usage of the current Web and the need of the physical world which is populated by massive context-based dynamic relations. So this research proposes a new type of Web, termed the GeoChannel Web, for mapping and applying dynamic, random and context-based relations between real-world things to support various innovative applications and services. The GeoChannel Web is expected to become a general platform with enabling techniques for identifying, discovering, correlating, clustering, bridging and connecting things in context.

This chapter sets a holistic view facilitating sectional descriptions in later chapters that design and implement architectural components for the GeoChannel Web. It firstly interprets new concepts of GeoChannel and the GeoChannel Web, and then give a general view on the components framework with brief analysis about their features or capabilities expected. A review on some existing techniques identifies their deficiency for the GeoChannel architecture, which calls for a set of new techniques and facilities to be invented in this research.

### **2.2 The concept of GeoChannel**

A new concept of GeoChannel is introduced in this section. This GeoChannel concept is for abstracting, generalizing and characterizing real-world things with their context-based relations.

The word “channel” is not a fresh concept, as it is frequently used in many fields with long-established usage. For example, radio or TV channels, web (page, site or service) channel, social-networking channel, map-layer channel, etc. When talked in these scenarios, this word channel usually refers to a resource, especially in the field of the general Web.

GeoChannel, however, is a new term and concept proposed in this research. Generally a GeoChannel involves three constituents, i.e. resource, clients and condition. Firstly, a GeoChannel has one or some target resources, just like the general concept of channel that traditionally refers to a resource. Secondly, clients (or namely users) are part of a GeoChannel. A client might be a person, a device or another resource that uses or interact with the target

resource. Thirdly and most peculiarly, a context condition is an integral part of a GeoChannel. Context refers to certain attributes of a resource or/and a client. A context condition refers to a certain requirement (or qualification) that the context of the target resource or/and its clients must meet or satisfy.

For real-world things, this research usually uses the term GeoContext for referring to the general word context. GeoContext refers to physical-world properties of resources or clients. For example, space, time, speed, direction of movement, heading or pitch of sight line, a map view status (e.g. position of the viewport, detail of level, etc.), and any other detectable or sensible properties by sensors. This research also uses the term GeoPolicy for referring to the general phrase context condition.

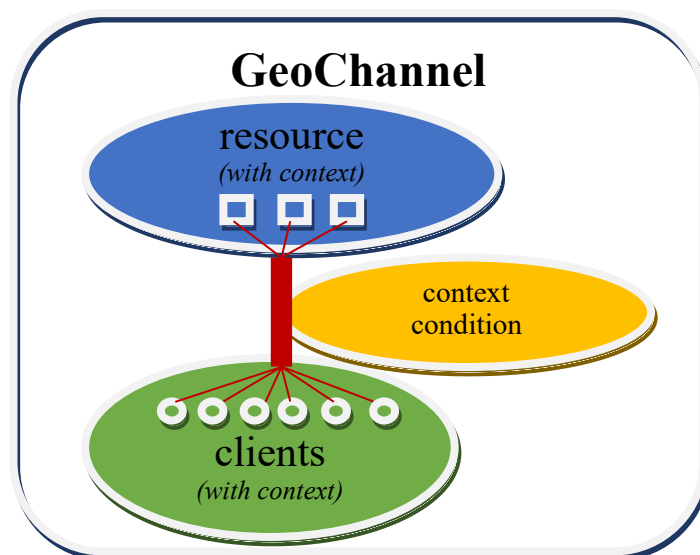


Figure 2-1: The concept of GeoChannel ---- contextualized resources, users and conditions

In this setting, the concept of GeoChannel can be depicted in Figure 2-1. The context condition can determine the resource-client relation. Clients are those that are interested in the target resources and matching the GeoContext condition.

In some sense, a GeoChannel is essentially an organizing unit for grouping a set of relations based on context. Figure 2-2 describes this concept. That is, if viewed from a perspective of



connections, a GeoChannel is virtually a collection of relations that may change dynamically and randomly, subject to the changes of contexts of the resource or/and clients.

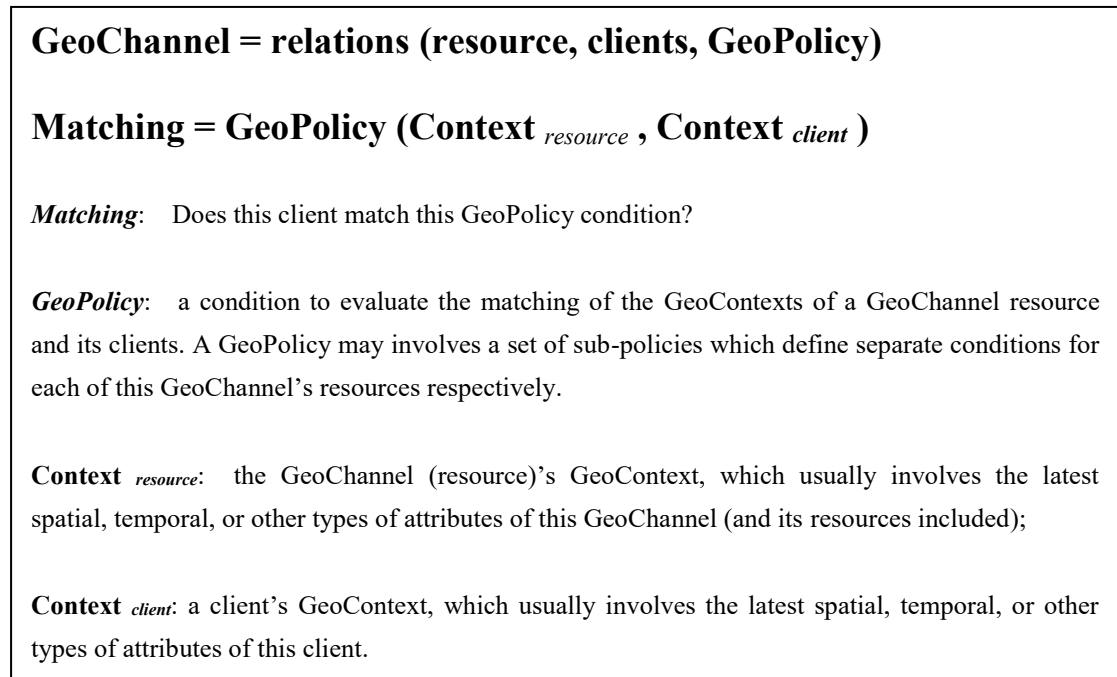


Figure 2-2: The concept of GeoChannel----an organizing unit for a set of context-based relations

So the GeoChannel concept is for abstracting, generalizing and characterizing real-world things with their context-based relations. GeoChannel is the working unit for GeoChannel networks and for the GeoChannel Web.

## 2.3 The GeoChannel Web

As illuminated in Section 2.2, GeoChannel is essentially an organizing unit for grouping a set of relations based on context. In a GeoChannel, the resource(s), clients and their relation can make up a dynamic networks named GeoChannel network. And further, multiple GeoChannels can form a bigger GeoChannel network, on which the constituent GeoChannels usually have some similarity or proximity in terms of the contexts of resources and/or clients. The GeoChannel Web is expected to be a globe-scale GeoChannel network which involve a massive number of constituent GeoChannel networks.

The GeoChannel Web features highly dynamic network topology, which is sensitive to the changes of clients interest, resources/clients' contexts and correlation (e.g. access-control)

conditions. Any changes of these elements will affect the topology of the GeoChannel Web.

The subsections below will generally illustrate the functionality expected for the GeoChannel Web, so as to identify its technical components to be designed in this research.

### 2.3.1 Mapping and applying dynamic and random relations

Differing from the current types of Web which largely reflect relatively static or stable relations, the GeoChannel Web is expected to be used for mapping and applying dynamic and random relation between real-world things based on their context.

As an example, Figure 2-3 illustrates mapping dynamic relations between things into a GeoChannel network. Figure 2-3(a) shows some GeoChannel resources and clients at a specific time point. The boundary shapes just visually represent the context conditions that interested clients need to match for accessing corresponding GeoChannel resources, although a context condition does not necessarily involve space attributes. The contexts of resources and clients or context conditions themselves may change randomly. At this moment, Figure 2-3(b) reflects the accessing relation status corresponding to Figure 2-3(a). Once changes happen to Figure 2-3(a) at another time moment, the network topology in Figure 2-3(b) will change correspondingly.

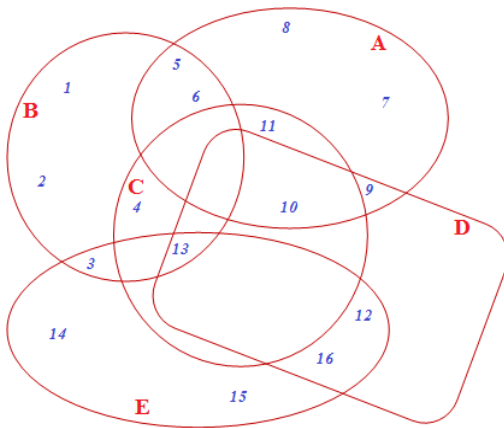


Figure 2-3 (a): An example depiction of GeoChannel resources with users matching access-organize conditions

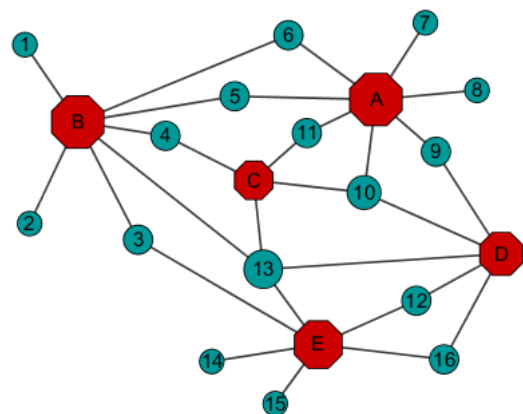


Figure 2-3 (b): An example of GeoChannel network with connected nodes of users and resources

Figure 2-3: Example: the GeoChannel Web for mapping and applying dynamic, random & context-based relations between real-world things

So the GeoChannel Web can reflect this kind of context-based dynamic relations, which can be

applied for various applications and services. Next section will exemplify their application.

### 2.3.2 Bridging Things in Context: application scenarios

The GeoChannel technology can bridge things based on their context attributes. Figure 2-4 illustrates this scenario. The GeoWeb works for mapping real-world things with their space context attributes, usually in a snapshot way such as a map. The GeoChannel Web can map the dynamic relations between things based on their contexts. The relations and/or contexts might be but are not limited to space attributes. They might be based on any kinds of properties that can be detected or sensed by physical or human sensors. These properties can function as the parameters of contexts that affect the relations between things.

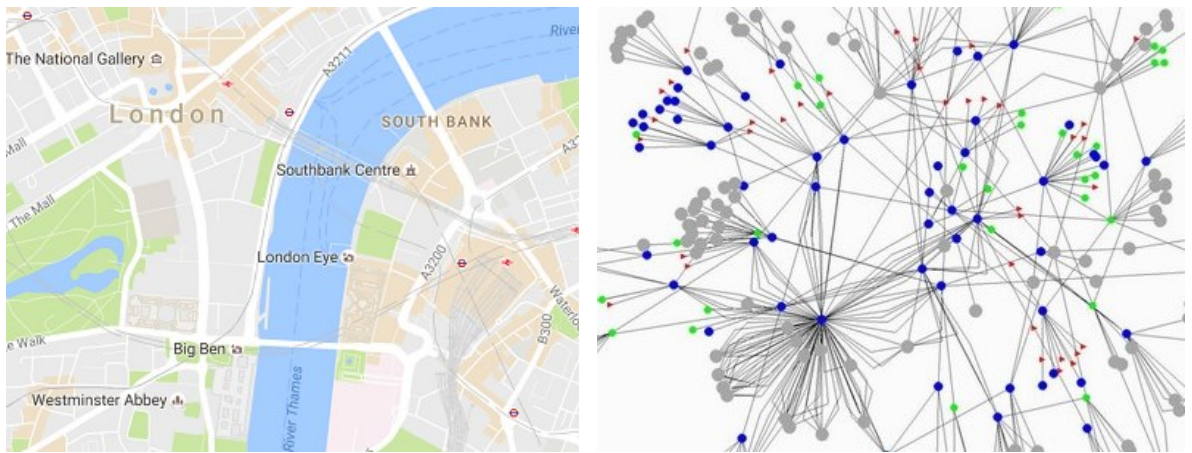


Figure 2-4: The GeoChannel Web for bridging things in context

These dynamic, random and context-based relations are valuable for practical applications, as exemplified in the following subsections.

#### 2.3.2.1 Dynamic & Real-time Maps

A challenge to a dynamic system such as a dynamic map is the scalability problem when it involves in handling a growing amount of work which expects its configuration to be adapted and optimized to accommodate that growth. This mainly involves server-side, client-side and network scalability. It is not an easy work to achieve large-scale dynamic and real-time maps currently, due to the challenges to the capacity of server/client-side computation, bandwidth and traffic cost of networking, and fast consumption of mobile device battery. A map server needs

to serve more clients when clients are increasingly connect to this server, and a client (e.g. a mobile device) needs to process (e.g. mash up) more resources if it is accessing more dynamic resources concurrently.

Here a practical scenario is given to exemplify this kind challenge. As shown in Figure 2-5, in New York City area there are lots of running taxis. Now a web map application for public use is desired to be able to browse the real-time movement of any taxis by any web users in a computing/networking/ battery resource -efficient way. For this objective, technically there are some specialness:



Figure 2-5: Example: GeoChannel technology for achieving large-scale dynamic maps

**a)** Dynamic resource objects. Distinct from current static-snapshot web maps which usually deal with stationary geographic features (e.g. imagery, placemarks, and geometries) that are geo-referenced via their fixed coordinates, the taxi-trip map is meant to present real-time movement of running taxis which change spatial positions randomly (as opposed to shuttle buses running on scheduled routes). Current static map techniques (such as WMS, WCS, WFS) can filter target objects based on map viewports and fixed coordinates of objects when a map viewport is changed. However, the dynamic taxi map application desired here needs to deliver real-time dynamics via real-time communication connections between server nodes and map users.

**b)** Numerous resource objects and clients. Many running taxis and browsing clients are

involved in this web map application system. Every taxi is an individual source of dynamics. To browse the real-time movement dynamics of any taxis, a client may connect (subscribe) to all (or many) taxi resources, each of which may be connected to and accessed by many clients of web maps. This tends to result in the problem of scalability on computing and communicating capability of servers, clients and networks.

So an appropriate solution is needed for dynamically and context-sensitively organizing access to these individual objects for optimizing the use of computing and networking resources to improve scalability.

For this application scenario, some technical components to be designed for the GeoChannel Web are identified as following:

- 1) A new language facility for expressing context conditions to filter resources in fine granularity;
- 2) An adaptive communication model for throttling communication and interaction based on latest context attributes;
- 3) An automatic mechanism for dynamically correlating resources and clients based on their latest context attributes;
- 4) A graph engine for mapping network topology of correlated resources and clients.

Actually, this dynamic maps example shown in Figure 2-5 is a use case project to be developed in this research. Chapter 9 will demonstrate the details.

#### **2.3.2.2 Generic Platform for Pervasive Services**

Pervasive services (as reviewed in Section 1.1.1.2) are context-based services. The GeoChannel Web can bridge resources with clients in a context-aware way, so it also expects to become a generic platform for pervasive services.

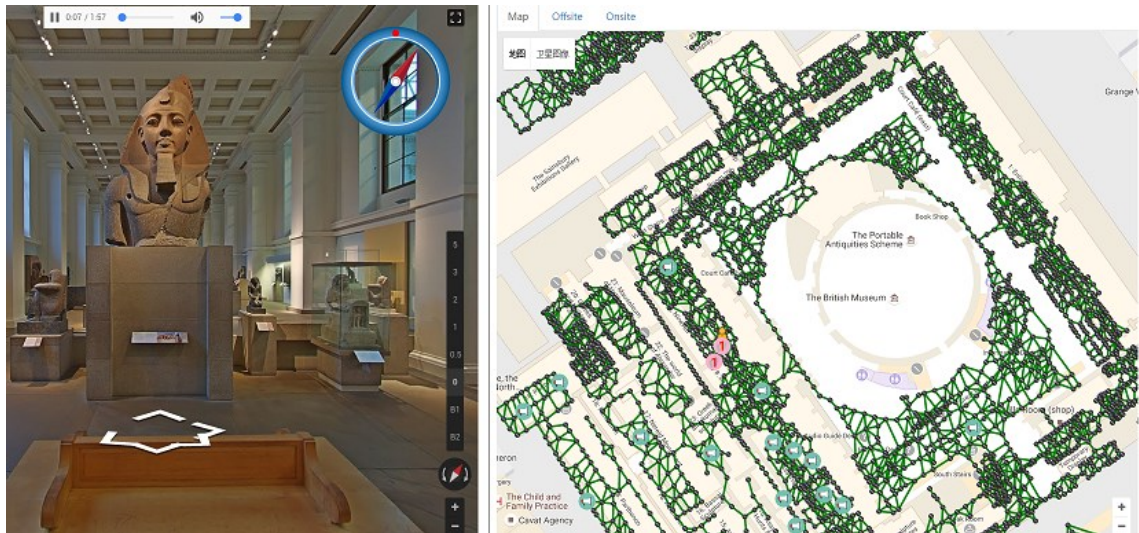


Figure 2-6: Example: GeoChannel technology for implementing pervasive services

An application example is shown in Figure 2-6, which is a use case project to be developed in Chapter 9. This project is about to achieve pervasive services in virtual world, physical world and even crossing virtual and physical world. The virtual-world pervasive services refer to online services for off-site users. For example, apart from panorama imagery resources, the viewers of indoor street view maps for the British Museum can access other types of resources (e.g. audio guide) based on their context (e.g. view position, heading, pitch, etc.). Physical-world pervasive services refer to context-aware services for on-site visitors. Cross-virtual-physical-world pervasive services are for both on-site and off-site clients. Chapter 9 will demonstrate this project in detail. More application scenarios can be exemplified in the fields such as smart home, smart city, intelligent transport, etc.

For pervasive services, some technical components to be designed for the GeoChannel Web are identified as following:

- 1) A new language facility for representing context condition of pervasive services;
- 2) A new URI technique for identifying contextualized resources;
- 3) A mechanism for dynamically correlating resources with clients, e.g. organizing access to pervasive service resources;



#### 4) Discovery technology for discovering pervasive services.

##### 2.3.2.3 New-pattern Social Networking

Current social networking is mostly based on proprietary platforms (service providers). Clients with their accounts on a same social media can connect each other for interaction based on their relationships of friends, followers or group members, etc. This kind of social networking is bound by specific service platforms and based on relatively fixed, static and explicit social relationships.

The real-world social interaction usually happens in an ad-lib way for/on an ad-hoc event, activity or opportunity in a specific context. For example, people (probably un-acquaintances) meet or gather on a venue or via an activity, such as a residential community, a conference, an event, a journey or a cooperative task, etc. They just make temporary interaction or cooperation in a certain context, without having to know mutual identities or to keep permanent (long-term) social relationships. This kind of real-world scenarios call for a new social-networking service paradigm for dynamic gathering participants on an activity/event that can act as an ad-hoc (temporary) medium with context for bridging people's interaction.

The GeoChannel Web proposed has native functionality for clustering and bridging users within and across GeoChannels based on context conditions. So it is expected to have native capability to support new pattern social-networking, i.e. context-based social-networking pattern, which can become a new social-networking paradigm supplementary to existing social media practice.

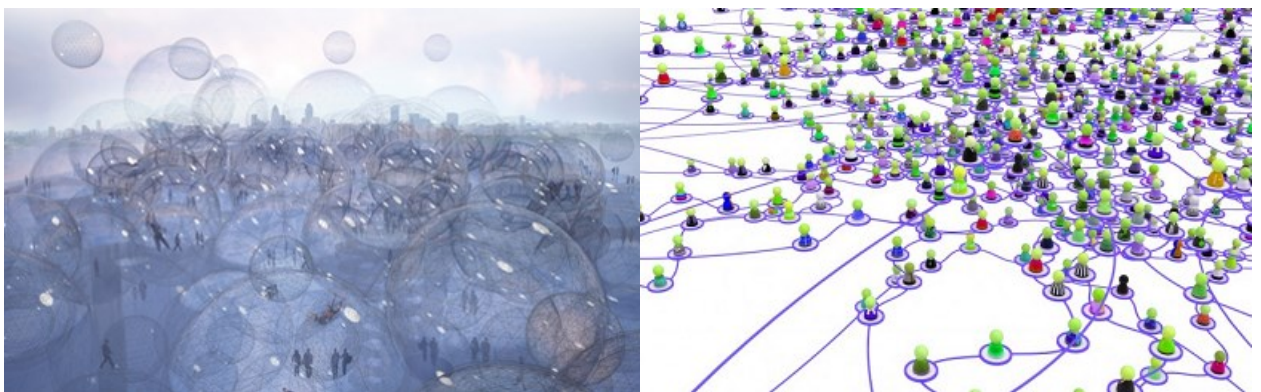


Figure 2-7: Context-based clustering and connecting: GeoChannel technology for new-pattern social networking

Context-based social networking pattern provided by the GeoChannel Web will feature new means for clustering, identifying, discovering and bridging people for participative interaction and activities. Conceptually Figure 2-7 illustrates some application scenarios. For example, clustering and connecting people in an environment with arbitrarily-defined context condition, such as on a certain route of travel, in a running train, around the goods shelves across several shops, etc. This capability is far various or advanced as opposed to the rudimental approach of GeoSocial Web (reviewed in Section 1.1.3) which can only make use of location points or check-in(s) for discovering and connecting people.

Chapter 9 will demonstrate a use case where on-site or online museum visitors can get dynamically clustered, discovered and connected with each other for interaction based on their context.

To achieve context-based social networking for the GeoChannel Web, some technical components to be designed:

- 1) A new expressive language for defining context conditions;
- 2) An automatic mechanism for clustering and bridging people based on latest context;
- 3) A new URI model for identifying participants in context;
- 4) Discovery technology for discovering people in specific context for interaction;
- 5) A network engine for mapping the dynamic relations between people.

Chapter 9 will demonstrate a novel use case for crossing virtual-physical world social networking supported by the context-based social pattern provided by the GeoChannel Web.

## **2.4 The Framework of Functionality Components**

As desired to be a new type of Web, the GeoChannel Web needs to be equipped with some native technical components to fulfil its functionality objective for mapping and applying dynamic, random and context-based relations between real-world things to support various



context-based applications and services.

The application scenarios exemplified in last section have identified some technical components expected for the GeoChannel Web. This section conceives the full framework of functional components to be designed in later chapters.

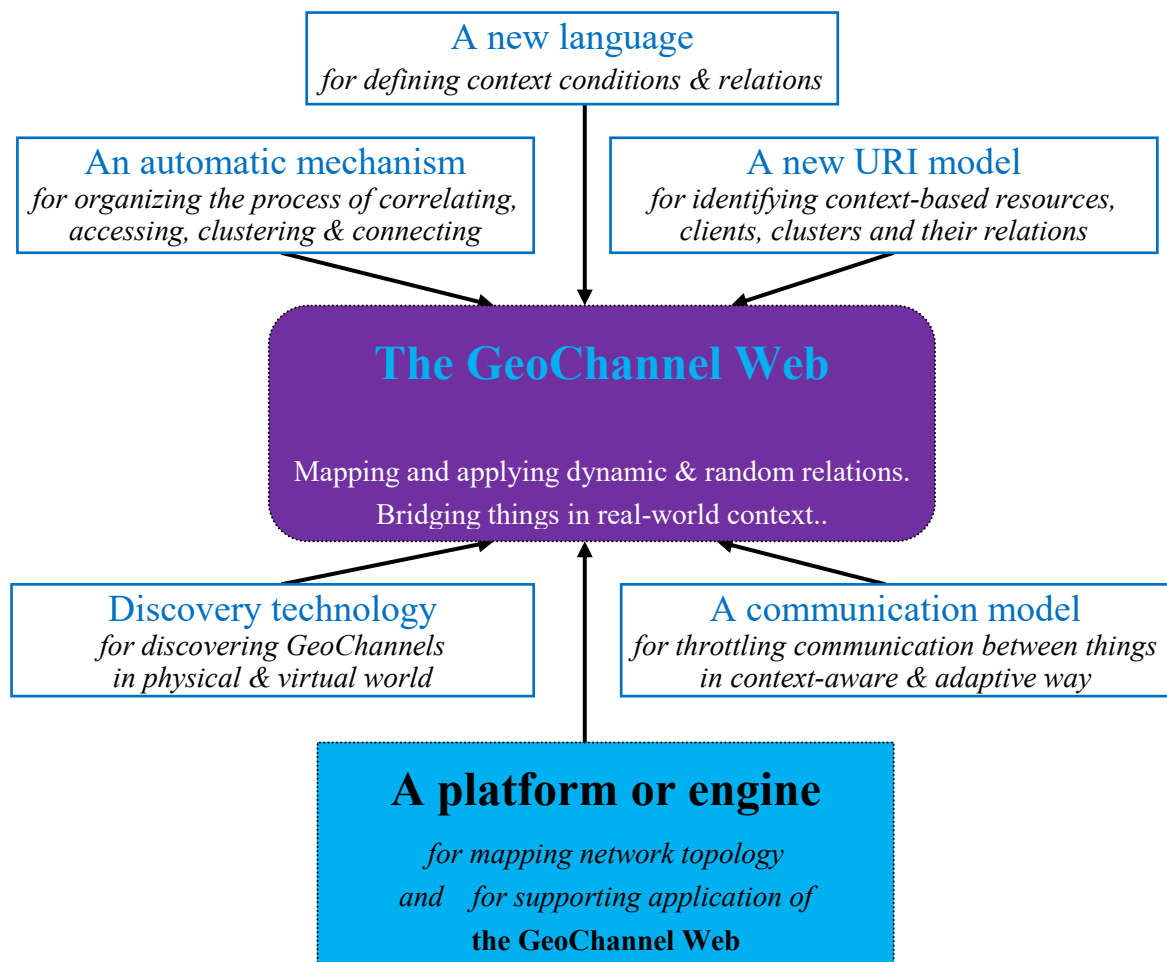


Figure 2-8: Components framework: new techniques and facilities expected for the GeoChannel Web

- 1) A new language for expressing context conditions to control correlating things and to control adjusting communication.
- 2) An adaptive communication model for throttling communication between things in context-aware and adaptive way, on the support of the new context-expressing language.

- 3) An automatic mechanism for controlling dynamic correlating, clustering and connecting things based on their latest context, on the support of the new context-expressing language and the adaptive communication model.
- 4) A new URI model for identify contextualized resources, clients, clusters and their relations.
- 5) Discovery technology for discover contextualized resources, clients, clusters and their relations from virtual and physical world
- 6) Networking mechanisms for connecting things into GeoChannel networks or the GeoChannel Web
- 7) A prototype platform with graph network engine for the GeoChannel Web to map topology relations, facilitate discovering and connecting things, and to support demonstration applications.

The expected components framework for the GeoChannel Web architecture is illustrated in Figure 2-8. More detailed requirements or specifications for these expected functionality components will be further identified in later chapters which design corresponding technique or facility components.

## **2.5 Review on related Techniques**

For the expected technical components identified in last section, the subsections below review some related techniques to examine their competence or deficiency for the GeoChannel Web, so as to identify a list of new techniques and facilities to be invented in this research.

### **2.5.1 Basic techniques: HTML, HTTP, URL**

Some fundamental techniques enable the general Web. Among them, the basic and key components include HTML, URL, HTTP, etc. The web resources are usually described by HTML, identified and located by URLs, linked by hyperlinks and accessed via HTTP.

HTML (Hypertext Markup Language) (Berners-Lee 1991a) (Faulkner et al. 2016) is used to

represent web pages, as well as to create user interfaces for web and mobile applications. HTML elements form the building blocks of HTML pages. HTML can embed scripts written in languages such as JavaScript which affect the behaviour of HTML web pages. HTML markup can also refer to Cascading Style Sheets (CSS) to define the look and layout of text and other material.

HTTP (Hypertext Transfer Protocol) (Berners-Lee 1991b) (Fielding et al. 1999b) (Belshe 2015) is an application layer protocol for distributed, collaborative, hypermedia information systems. It is the foundation of data communication for the general Web. HTTP functions as a request–response protocol in the client–server computing model.

The general Web uses the “http://” (Fielding et al. 1999) or “https” URI scheme (RFC3986 2005) for identifying and locating web resources. URL (Uniform Resource Locator) (Berners-Lee 1994) (WHATWG 2016), informally termed a web address (Connolly & Sperberg-McQueen, eds. 2009), is a reference to a web resource that specifies its location on a computer network and a mechanism for retrieving it. Every HTTP URL contains a schema prefix “http” or “https” that indicates using HTTP protocol to access the targeted web resource.

From a view on the web graph structure, these basic technical components (HTML, HTTP, URL) play fundamental roles on the nodes and edges of hyperlink relation of the Resource Web, actually they also function as the cornerstone technology for other types of usages of the general Web (as reviewed in Chapter 1), although many other technical components have increasingly joined the web stack.

## **2.5.2 “geo:” URI and GeoRSS**

### **2.5.2.1 “geo:” URI scheme**

The “geo:” URI scheme is defined by the Internet Engineering Task Force's RFC 5870 (IETF 2010) as a Uniform Resource Identifier (URI) for geographic locations using the “geo” scheme name. A “geo” URI identifies a physical location in a two- or three-dimensional coordinate reference system in a compact, simple, human-readable, and protocol-independent way.

The “geo:” URI scheme can add geography coordinates into information entities such as GeoSMS (OGC 2011c), vCards (IETF 2011), etc. A simple geo URI might look like: geo:37.786971,-122.399677,500;u=36, where the four numerical values represent latitude longitude and altitude and an optional "uncertainty" value respectively. For example, a geo URI may be included on a web page, as HTML: <a href= "geo:37.786971, -122.399677; u=35"> A Headquarters </a>. It could be used in an Atom feed or other XML file.

So the “geo:” URI is a paradigm for adding place geographic coordinates information (as a kind of context attribute) into information entities. To some extent, there is some similarity to the idea of involving context conditions into a URI for the GeoChannel Web, as described in Section 2.4.

#### **2.5.2.2 GeoRSS**

GeoRSS (GeoRSS 2013), which stands for Geographically Encoded Objects for RSS Feeds, is a simple proposal (Reed et al. 2006) for geo-enabling, or tagging, RSS (Really Simple Syndication (RSS 2009)) feeds with location information.

GeoRSS is basically an extension to RSS, with added capability of geo-annotating data items in RSS feeds with space/time attributes. Actually, the GeoRSS extension can apply to currently the two main web feed formats, i.e. both RSS and ATOM (Atom Syndication Format (Nottingham & Sayre 2005, Gregorio 2007)). GeoRSS enables geographically tagged feeds, to extend existing web feeds with geographic information. A web feed is a document (often XML-based) which contains content items, usually summaries or structured metadata about the content source of a resource (such as an information, service, application, etc.). A RSS/ATOM usually is a wrapper for pieces of regularly and sequentially-updated content about a resource.

There are currently two primary geographic encoding formats for GeoRSS, i.e. GeoRSS-Simple and GeoRSS-GML, for adding geographic location (e.g. point/line/boundary) to web Feeds. GeoRSS-Simple is a lightweight format supporting basic geometries (point, line, box, polygon) and covers the typical use cases when encoding locations, while is a formal GML (Portele 2012) Application Profile for supporting a greater range of features, notably coordinate reference

systems other than WGS-84 latitude/longitude.

Update-publishing of resource content is one of the main purposes of using web feeds. GeoRSS as an extension of general web feeds naturally inherits this capability. Generally a web feeds processor (e.g. feed reader, aggregator, etc.) takes a simple mechanism for checking and finding resource updates. It periodically polls the source of this feed to get the latest feed version to identify possible updates of the feed-associated resources. This is typically a pull-pattern communication refreshed in a certain time interval.

The PubSubHubBub (Fitzpatrick et al. 2010) is a protocol that adds real-time functionality of update notification for web feeds. This is server-to-server protocol, which will be reviewed in details in Section 7.5 where it will be employed for harvesting real-time updates of GeoChannel metadata for GeoChannel Catalogue Service. GeoRSS can also employ this PubSubHubBub protocol for delivering real-time updates of resource information in scenarios of server-to-server communication.

### **2.5.3 Geo-tag, Geo-filter and spatial-extended SQL**

The current GeoWeb mostly employs geo-tag and geo-filter paradigm for organizing resources' uses. For example, existing GeoWeb data-supply services such as WMS, WCS, WFS, etc., are all based on the geo-filter mechanism for retrieving data subsets according to spatial query conditions. LBSs (location based services) are mostly service-content-filter systems which select service information content with spatial relevance to a client's location context.

Geo-tagging is this process of annotating objects and online resources with geospatial context information, ranging from specific point locations to arbitrarily shaped regions (Inversini et al. 2012). By adding geographical identification metadata to various media, geo-tags become a form of geospatial metadata (Luo et al. 2008). Generally the GeoWeb deals with two categories of information, one of which is spatial data that has geographic dimension, and another is non-spatial information that does not have location reference initially. The geo-tagging process can just assign or attach spatial attributes, such as geographic coordinates or geometry features, etc., to an information object such as a place name, a photo/image, an activity event, etc.

Geo-tagging can take embedding and attaching approaches. The former is to modify an information object itself to embed a space-attribute field, e.g. geo-tagging a photo; and the later attaches (or points to) additional message/file, which contains the space attribute, to the original information object without modification internally. Geo-tags can facilitate identifying, discovering and organizing information resources by exploiting their spatial attributes/tags.

Geo-filtering refers to the process of selecting/filtering information subset of interest by space-based query. Geo-filtering just searches against spatial features of spatial data or geo-tags to find out target information objects. So Geo-tagging and Geo-filtering are a pair of related processes for organizing uses of resources (e.g. information/services).

There are encoding languages available for expressing filtering (i.e. selecting) conditions, e.g. the xml-based Filter Encoding standard (OGC 2010b), and the plain text-based CQL (common Query Language) defined in the Catalogue Service standard (OGC 2007b), etc. Custom search conditions can be expressed by using these filter encoding languages. For example, clients can search OGC's WFS, WCS and WMS data-supply services with specific query conditions. And sometimes a web mapping service may support fixed (or implicit) condition query. For example, a Web Map Tile Service (WMTS) (OGC 2010a) serving pre-rendered geo-referenced map tiles for base maps can employ a unified query condition, which uses a client's map viewport as a query parameter to request those map tiles that can fall into this geographic range. This explicit-condition paradigm generally exists in current web mapping services such as Google/Bing/Yahoo/OpenStreetMaps.

Spatial-extended SQL (Egenhofer 1992, 1994) language has some capability for expressing spatial conditions. Spatial SQL is applicable for spatial data query and retrieval in a programming manner basically used by software developers or database users. Geo-filtering usually employs spatial-extended SQL for defining filtering conditions.

Geo-filtering can work in two types of interaction patterns between clients and services. In the request/respond pattern, a geo-filtering service responds to its clients by returning the search result that are usually data/content snapshots frozen at the time point when this result was

filtered. The publish/subscribe pattern is stateful, as the server keeps a client's subscription that contains spatial filter condition, and pushes condition-matched content to the client once this new content is available. Currently the request/respond pattern dominates the Web mapping practices due to the stateless nature for server implementation and performance featuring simplicity and scalability, but this paradigm lacks timeliness as it cannot deliver really real-time updates to clients, even if the clients frequently poll the server. In this case, the over-frequent pollings instead tend to impair the scalability of networking and computing resources. By contrast, the publish/subscribe pattern can push real-time updates; however, the stateful server tends to suffer from scalability problem of performance. The server needs to re-evaluate all the subscriptions with geo-filter conditions once any new or updated content happens to the content repository. To include, generally the request/respond pattern is better scalable but less real-time, while the publish/subscribe pattern is better real-time but less scalable.

#### **2.5.4 XACML & GeoXACML**

Access control is a well-established technological domain, with many theoretical models and practical solutions. Generally, these access control models basically fall into one (or sometimes a combination) of two forms, i.e. discretionary or mandatory, with categorized as Discretionary Access Control (DAC) and Mandatory Access Control (MAC) respectively. There are many access control models that implement DAC and/or MAC. Some most widely recognized models include Identity Based Access Control (IBAC), Role Based Access Control (RBAC) (Sandhu et al. 1996), Lattice Based Access Control (LBAC) (Sandhu 1993), and Attribute Based Access Control (ABAC), etc. IBAC is primarily DAC mechanisms, as it is not best suited to address MAC requirements (Yuan & Tong 2005). RBAC can be configured to do DAC or MAC. LBAC falls into the MAC camp. The ABAC (Attribute Based Access Control) is a late-emerged model, which overcomes the limitations of some dominant access control models (e.g. discretionary-DAC, mandatory-MAC and role based-RBAC) while unifying their advantages (Jin et al. 2012).

Attribute Based Access Control (ABAC) is based on subject, object, and environment attributes and supports both mandatory and discretionary access control needs (Yuan & Tong 2005).

Currently a predominant implementation to ABAC is XACML (eXtensible Access Control Markup Language), which is a standard (OASIS 2012) access control language system with its working model, allowing establishing an access control system which can be used to manage access for Service Oriented Architectures (SOA).

XACML standardizes three essential aspects of access-control work (Axiomatics 2012) (Sun 2006):

- XACML policy language – used to express access control policies (rules and conditions). Many rules can be combined into one policy. Many policies and policy sets can be combined into larger policy sets. Figure 2-9 illustrates the XACML Policy Language Model (OASIS 2012).
- XACML request/response protocol – the request/response language expresses queries about whether a particular access should be allowed (requests) and describes answers to those queries (responses). This is to query a decisioning engine that evaluates real-world access requests against existing XACML policies. The result, either Permit or Deny, is returned as an XACML response.

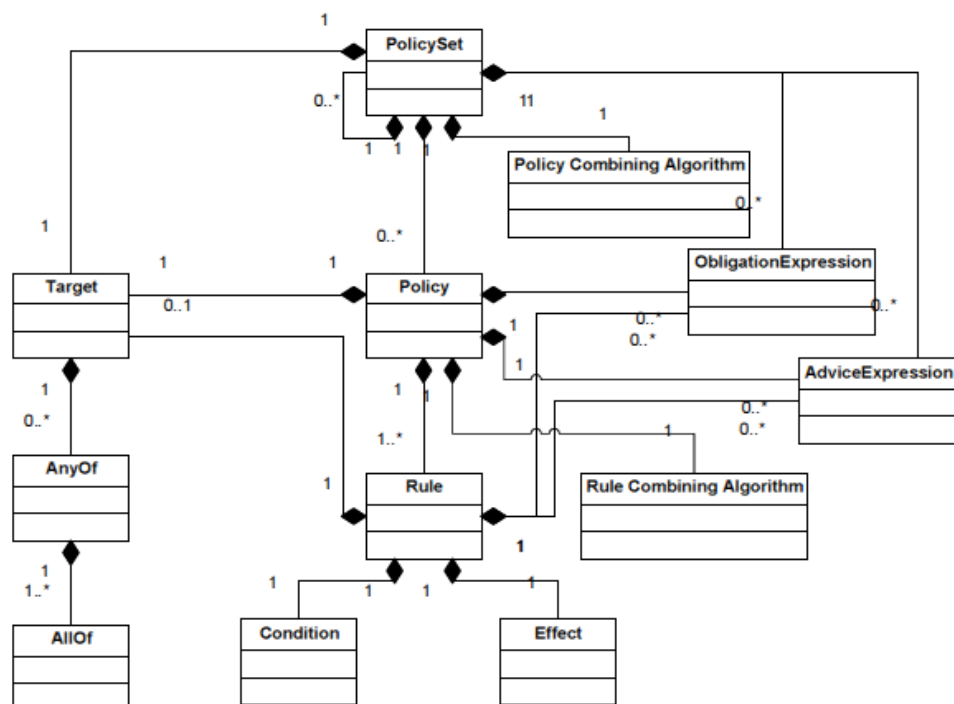


Figure 2-9: XACML Policy Language Model  
(OASIS 2012)



## XACML Architecture

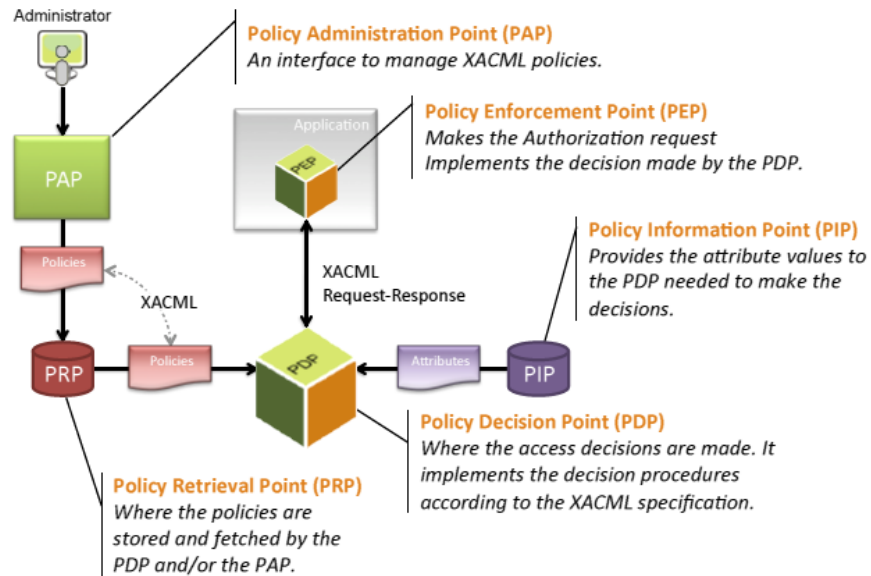


Figure 2-10: XACML Architecture  
(Axiomatics 2012c)

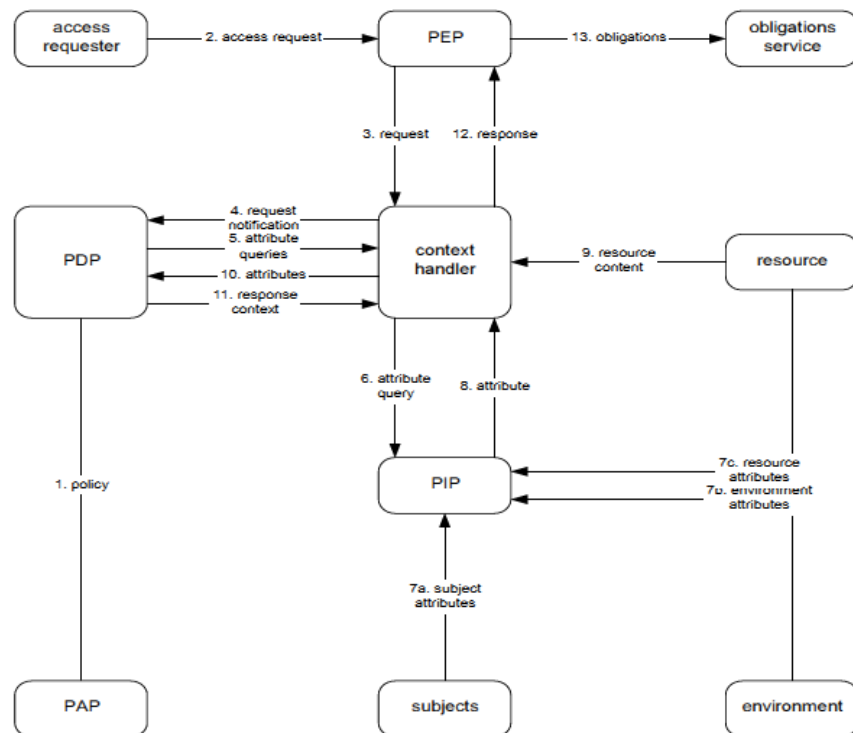


Figure 2-11: XACML Data Flow Model  
(OASIS 2012)

- XACML reference architecture – provides a standard for the deployment of necessary software modules to achieve efficient enforcement of XACML policies. As shown in the XACML Architecture (Figure 2-10), at the core, a Policy Decision Point (PDP) evaluates policies against access requests provided by Policy Enforcement Points (PEP). The PDP or PEP may also need to query a Policy Information Point (PIP) to gather descriptive attributes about the user or the information asset to which access is requested. Policies are maintained via a Policy Administration Point (PAP). Figure 2-11 illustrates XACML Data Flow Model (OASIS 2012).

In a typical XACML usage scenario (Sun 2006): a subject (e.g. human user, workstation) wants to take some action on a particular resource. The subject submits its query to the entity protecting the resource (e.g. file system, web server). This entity is called a Policy Enforcement Point (PEP). The PEP forms a request (using the XACML request language) based on the attributes of the subject, action, resource, and other relevant information. The PEP then sends this request to a Policy Decision Point (PDP), which examines the request, retrieves policies (written in the XACML policy language) that are applicable to this request, and determines whether access should be granted according to the XACML rules for evaluating policies. That answer (expressed in the XACML response language) is returned to the PEP, which can then allow or deny access to the requester.

XACML is a generic access control policy language, which provides standard extension points for defining new functions, data types, combining logic, etc. GeoXACML (Geospatial eXtensible Access Control Markup Language), which is an OGC (Open Geospatial Consortium) standard (OGC 2011a), just makes use of XACML's extension mechanism to supply some geospatially-specific components to meet the need for access control of geospatial resources. This extension provides support for spatial data types and spatial authorization decision functions. Those data types and functions can be used to define additional spatial constraints for XACML based policies. GeoXACML can be used to establish an Access Control Mechanism to protect the access to OpenGIS Web Services (OWS).

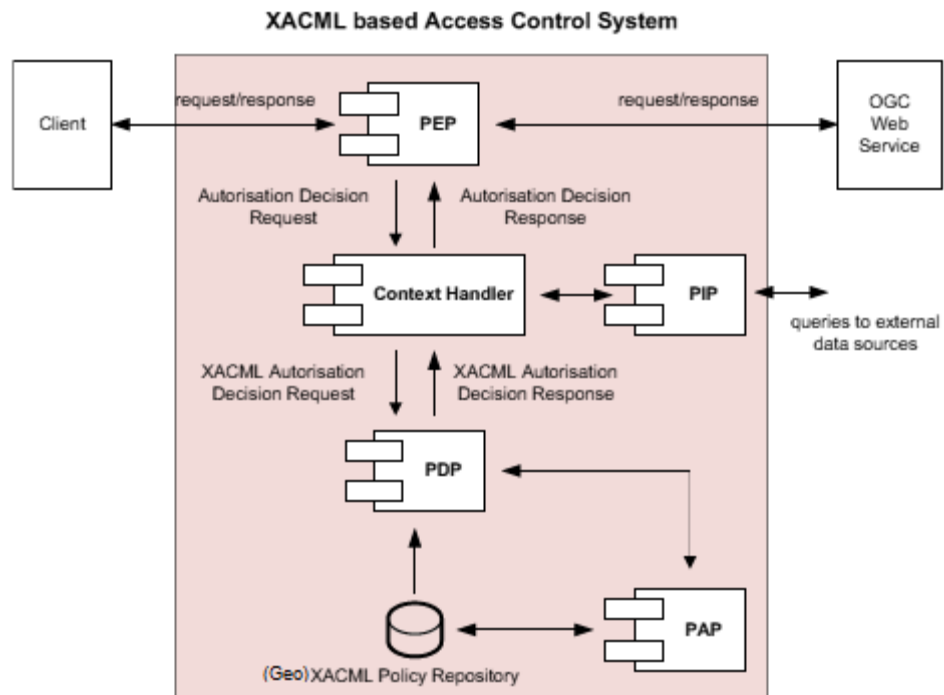


Figure 2-12: Architecture of a (Geo)XACML based access control system  
(Herrmann 2011)

Figure 2-12 (Herrmann 2011) shows the common architecture of (Geo)XACML based access control systems with showing the internal information flow. The access control system serves as a proxy component that intercepts messages exchanged between the clients and the OGC service instances. After intercepting a request or response message the access control process is initialized through an authorization decision request send by a Policy Enforcement Point (PEP) to a Policy Decision Point (PDP). The PDP will evaluate matching policy elements and return an authorization decision response to the PEP that in turn will act accordingly.

### 2.5.5 Challenges to existing Techniques

For the related techniques reviewed heretofore, this sections examines their deficiency for the GeoChannel Web mission

Firstly, a general gathering of the deficiencies are presented in Table 2-1 about current related techniques' applicability for the GeoChannel Web functionality that had been identified in Section 2.4, and then some details will be explained subsequently.

### 2.5.5.1 HTTP, Http-URL, geo:URI

HTTP protocol for the general Web does not involve a mechanism for adaptive communication and interaction between things considering their context attributes.

The general Web uses the “http://” URI scheme for identifying and locating web resources. This general URI scheme does not have a native mechanism for encoding or associating resources’ context information into their http URIs.

The “geo:” URI scheme can add geography coordinates into information entities such as GeoSMS, vCards, etc., nevertheless it is too simple to incorporate complex context attributes other than position information.

Table 2-1: The deficiencies of current techniques for the GeoChannel Web functionality

<b>Current techniques</b> <b>Access-Organize tasks</b>	<b>Web URI schema (http-URI)</b>	<b>geo: URI scheme</b>	<b>HTTP</b>	<b>Geo-tag &amp; Geo-filter</b>	<b>GeoRSS</b>	<b>(Geo)XACML</b>
Identify resources/clients/clusters with contexts	no	partly	no	no	partly	no
context & policy condition-expressing approach (language)	no	no	no	partly	partly	yes
contextualization: dynamic/random context (conditions)	no	no	no	challenging	partly	yes
Dynamically perform correlation/access-control for client-resource, resource-resource, and client-client	no	no	no	partly	no	partly
Adaptive, timely & scalable networking and computation, considering context relationships between things	no	no	no	no	no	no
GeoCluster	no	no	no	no	no	no

### 2.5.5.2 Geo-tag, geo-filter and spatial-extended SQL

Although geo-tag and geo-filter is a current mostly-used paradigm for organizing resources’ use, e.g. discovering or retrieving information, such as for Web maps or current LBSs, it is still

incapable for the GeoChannel Web tasks.

- 1) Current geo-tagging and geo-filtering techniques mostly work for data snapshots or service information content that keep their geo-context stable relatively; otherwise it is challenging for performing frequent and random changes on their space/time attributes due to lack of efficient mechanism for communicating and reflecting updates.
- 2) Geo-filter can change spatial query conditions when searching resources to get client interested information resource, but has no functionality to perform resource-resource or client-client correlation such as resource federation or users geo-clustering.
- 3) It basically works for geo-referencing and searching/filtering data or service content information by using their spatial features, but does not have access-control or correlation functionality.
- 4) It does not have a context-sensitive and adaptive communication/interaction mechanism for throttling networking and computing. Their client-server interactions do not exploit geo-context relations between clients and resources for adapting and optimizing computation and networking behaviours.
- 5) Centralized service architecture requires a server to have sufficient throughput capability for service performance scalability.
- 6) Geo-tag and geo-filter are too simplex to meet complex representation of comprehensive context conditions. Especially, they do not deal with other varieties of context attributes beyond space/time features, for example, those context attributes captured by various sensors increasingly used.
- 7) Spatial-extended SQL language has some capability for expressing spatial conditions; however it does not meet the demand in the GeoChannel correlation mission. Spatial SQL is applicable for spatial data query and retrieval in a programming manner basically used by software developers or database users. In contrast, automatically correlating resources with clients and clustering clients in similar contexts a mission of the GeoChannel Web which needs

an easy means for defining context policy conditions in a non-programming way.

### **2.5.5.3 GeoRSS**

GeoRSS has the following deficiencies:

- 1) Although GeoRSS can encode geospatial content (e.g. geographical points, lines, and polygons of interest and related feature descriptions.) as part of a Web feed, it is still a onefold solution limited to geospatial features, lacking adequate mechanisms for supporting contextualization with other types of context attributes.
- 2) Only resources' GeoContext but no resource-client accessing condition information can be present in GeoRSS. It does not perform access control, and not perform dynamic correlation between resources or clients.
- 3) It can neither cluster users nor identify user groups.
- 4) Lack of real-time resource-client communication. Web feeds' inbuilt polling-based mechanism cannot achieve instant updates of resource status, while delivering resources' latest GeoContext in real time is a compulsory requirement for GeoChannel correlation mission. Although the PubSubHubBub (Fitzpatrick et al. 2010) may add real-time capability to Web feeds, it is a server-to-server protocol for federated delivery of feed updates, so it cannot improve GeoRSS for real-time server-client communication.
- 5) Lack of efficiency and scalability on communication and computation for resource-client interaction. There is no inbuilt mechanism for context-aware and resource-throttled interaction. In a GeoRSS application, all clients poll a server to get feed updates about the resource status in a unified pattern (e.g. frequency), without consideration of different GeoContext relations between different clients and the target resource. Actually, much communication/computation for GeoRSS feed updates may be pointless operations that waste networking or computing resources.

### **2.5.5.4 (Geo)XACML**

(Geo)XACML has more seemingly-applicable functions for the GeoChannel Web tasks. (Geo)XACML features a complete language tool for expressing access-control policy conditions that can be updated dynamically, so it can dynamically correlate clients with resources. However, (Geo)XACML only focuses on access-control functionality. It is still incapable for the GeoChannel Web mission, due to its deficiencies as identified below:

- 1) No technical facility can work for identifying resources/clients/clusters with context.
- 2) Its existing policy language (with various extensions available currently) is still short of elements for expressing the context and correlation conditions about GeoChannel Web resources and clients, e.g. the context attributes captured by various sensors.
- 3) It does not function for performing resource-resource or client-client correlation, or for clustering clients.
- 4) Issue about continuous access to resources. (Geo)XACML request/response interaction model for access control and resource retrieval is suitable to acquire static snapshots of a resource, but is incapable for keeping continuous access to a resource with a streaming nature or requirement. For example, sensor dynamics stream, continuous social interaction services, etc.
- 5) Scalability problem on computing. It is a centralized system architecture which processes all policy-evaluation tasks centrally, tending to suffer from performance bottleneck problem when serving massive clients concurrently.
- 6) Inefficiency and scalability problem on communication. It does not have a mechanism for discriminating and throttling communication interaction initiated from clients. Any clients, regardless of their different geo-context status, can launch communication to the server anytime, even in the cases where and when their geo-context status apparently cannot meet the access-control policy. Thus these pointless communications waste much networking resources and aggravate the computing scalability problem on the server for processing these pointless evaluation of access requests. Desirably, an efficient system should consider to discriminate in favour of pressing communication for those clients that are more approaching the matching of

the access condition, but to discriminate against non-pressing communication for those clients that are currently far from matching of this condition. However, the (Geo)XACML does not have this kind of technical strategy or mechanism.

7) (Geo)XACML services for access control are basically proprietary services that exclusively serve specific organizations or customer communities for specific business. So they usually cannot cater for free uses for the general public, while open availability and accessibility is just a compelling demand for the GeoChannel Web mission.

**To conclude**, the existing techniques reviewed in Section 2.5 cannot fully meet the demands and requirements for the functionality components as identified in Section 2.4. This research needs to design and develop these new technical components for the proposed GeoChannel Web.

## **2.6 Summary**

This chapter has set a holistic view facilitating sectional descriptions in later chapters that design and implement architectural components for the GeoChannel Web.

The new concepts of GeoChannel and the GeoChannel Web have been framed and interpreted in this chapter. The GeoChannel concept is for abstracting, generalizing and characterizing real-world things with their context-based relations. A GeoChannel is essentially an organizing unit for a collection of relations that may change dynamically and randomly, subject to the changes of contexts of the resource or/and clients.

In a GeoChannel, the resources, clients and their relations can make up a dynamic GeoChannel network and multiple GeoChannels can form a bigger GeoChannel network. The GeoChannel Web is a globe-scale GeoChannel network which involve a massive number of constituent GeoChannels. The GeoChannel Web features highly dynamic network topology.

The GeoChannel Web is for mapping and applying dynamic, random and context-based relations between real-world things. It can bridge things in context, so it can be used in



extensive fields, for example, dynamic and real-time maps, pervasive services and context-based social networking, etc.

The analysis on some application scenarios has helped identified the components framework expected for the GeoChannel Web. A review on some related techniques has revealed their deficiencies for the functionality components of the GeoChannel Web.

With the research tasks identified, the subsequent chapters in this thesis will design all the new technical components, implement a prototype of the new architectural model, and develop use case projects for validating the GeoChannel Web technology.

## 3 GeoPolicy Language

### 3.1 Introduction

The general Web has been progressively equipped with various information models, such as the HTML for representing web pages and creating user interfaces. Taking the Geospatial Web for example, many encoding languages (formats) exist for representing data with space/time context, e.g. GML, KML, ARML (OGC 2014), etc. However, there is a lack of a general language for representing context conditions for controlling the networking relations between things weaved on the Web. This Chapter designs the GeoPolicy Language to fill this gap. In some sense, the GeoPolicy Language for the GeoChannel Web can be analogized to HTML for the general Web, while the two languages work for different purposes.

This chapter firstly identifies some desired capabilities of the GeoPolicy language. Then this language is designed with a core set of its language elements which are extensible for further supporting more kinds of context attributes that may be acquired by future sensors. An extended-KML encoding model has been codified for the GeoPolicy Language. Some examples are given for using this language.

### 3.2 GeoPolicy Language: a new Language for the GeoChannel Web

A new general-purpose language is desired for defining context relation conditions for the GeoChannel Web and for the general Web. This research designs this language, termed GeoPolicy Language. The prefix “Geo” of this name might indicate a main focus on the geospatial field initially in this research process; however, progressively this GeoPolicy Language has extended to support expressing various context conditions in broad domains in addition to the geospatial domain. So the GeoPolicy Language is a general-purpose context-condition language that can be used for the general Web.

The GeoPolicy Language is expected to feature some capabilities:

**Expressive enough:** It should be able to express various context attributes that can be captured

currently, such as space, time or other detectable attributes by current sensors;

**Extensible:** It should be extensible to support future sensors that can detect more kinds of context attributes, and to support defining domain-specific context conditions.

**Embeddable:** It can seamlessly fuse into other languages (i.e. host languages), for example other data-representing language such as GML, KML, ARML (OGC 2014), etc., to make them become conditional resources or services. Host languages and the GeoPolicy language can mix up according to both grammar. This feature can make the GeoPolicy Language widely applicable with various application scenarios where corresponding host languages exist.

**Encoding-neutral:** This language itself is format/encoding-independent. It can be represented by different encoding formats that host environments/languages are using.

**Normative:** It has normative and unambiguous model for its syntax and semantics.

### 3.3 Design of Language Elements for GeoPolicy

A core set of language elements has been designed for this GeoPolicy Language. More on-demand elements can be incorporating into this core part. In addition, an extension mechanism is provided for adding domain-specific components with capability of expressing specific context attributes in various application domains.

#### 3.3.1 The core part of language elements

The core-part language elements (as listed in Table 3-1) prefixed with a “geoww” namespace can currently support defining some context conditions for real-world things. For example, the components provide a set of elements (or operators) about space, time, mathematics, logic and measurements of some sensors.

The physical-world context largely involves space/time attributes, so a fine-compiled set of elements are supplied for constructing space/time or related context conditions. In Table 3-1, the sections about Topological and Geometric functions with Numeric, Arithmetic and Logical

operators just work for this purpose.

The virtual-world context is also a scope that the GeoPolicy Language works for. The current and future scope may involve the fields of online maps, games and VR (Virtual Reality) or AR (Augmented Reality), etc. Taking web maps for example, a set of language elements are currently provided for contextualizing map resources and clients by parameterizing the viewport status of map viewers, so as to describe resource-client relations during the map-browsing interaction.

Sensor measurement is a big involvement of the GeoChannel Language employing sensors to capture and describe the context of real-world things. Currently some popular sensors built-in mobile devices are included in the language elements. More kinds of sensors, for example those that will appear in wearable devices (such as smart glasses and smart band, VR or AR devices) will be incorporated into this language.

Table 3-1: GeoPolicy Language – functional elements for representing conditions of context

<b>GeoPolicy Language</b> core elements <i>(geoww as the namespace prefix of the following elements)</i>	<b>Description</b>  Function_name (arguments: Value_Type): Result_Value_Type	
	<b>Topological Functions</b>	
< Contains>	Contains(g1:Geometry, g2:Geometry) : boolean	
<Crosses>	Crosses(g1:Geometry, g2:Geometry) : boolean	
<Disjoint>	Disjoint(g1:Geometry, g2:Geometry ) : boolean	
<Equals>	Equals(g1:Geometry, g2:Geometry) : boolean	
<Intersects>	Intersects(g1:Geometry, g2:Geometry) : boolean	
<Overlaps>	Overlaps(g1:Geometry, g2:Geometry) : boolean	
<Touches>	Touches(g1:Geometry, g2:Geometry) : boolean	
<Within>	Within(g1:Geometry, g2:Geometry) : boolean	
<IsWithinDistance>	IsWithinDistance(g1:Geometry, g2:Geometry, d:double) : boolean	
	<b>Geometric Functions: Constructive Geometric Functions</b>	
<Buffer>	Buffer(g:Geometry, d:double) : Geometry	Polygon, MultiPolygon
<Boundary>	Boundary(g:Geometry) : Geometry	Point, LineString,

		Polygon
<ConvexHull>	ConvexHull(g:Geometry) : Geometry	Polygon, Point
<Centroid>	Centroid(g:Geometry) : Geometry	Point
<Difference>	Difference(g1:Geometry, g2:Geometry) : Geometry	
<SymDifference>	SymDifference(g1:Geometry, g2:Geometry) : Geometry	
<Intersection>	Intersection(g1:Geometry, g2:Geometry) :	Point, LineString, Polygon
<Union>	Union(g1:Geometry, g2:Geometry) : Geometry	
<Interpolate>	Interpolate (from:Point, to:Point, fraction:double):	Point
<Offset>	Offset (from:Point, distance:double, heading:double):	Point
<OffsetOrigin>	OffsetOrigin (to:Point, distance:double, heading:double):	Point
<EncodePoint>	EncodePoint (latitude:double, longitude: double):	Point
<EncodePath>	EncodePath (g1:Point, g2: Point, ..., gn:Point):	LineString
<EncodePolygon>	EncodePolygon (g1: LineString, g2: LineString, ...):	Polygon
<GetPaths>	GetPath (g1: Polygon):	an array of LineStrings
<DecodePath>	DecodePath (g1: LineString):	an array of Points
<LatLngBounds>	LatLngBounds (SouthWest: Point, NorthEast: Point):	Polygon
<BoundingBox>	BoundingBox (g1: geometry):	Polygon
<GetNorthEast>	GetNorthEast (g1: geometry):	Point
<GetSouthWest>	GetSouthWest (g1: geometry):	Point
	<b>Geometric Functions: Scalar Geometric Functions</b>	
<Area>	Area(g:Geometry) :	double
<Distance>	Distance(g1:Geometry, g2:Geometry) :	double
<Length>	Length(g:Geometry) :	double
<Heading>	Heading(from:Point, to:Point): double	Headings are expressed in degrees clockwise from North within the range (-180,180).
<Latitude >	Latitude (g:Point):	double
<Longitude>	Longitude (g:Point):	double
<Altitude>	Altitude (g:Point, <i>altitudeMode</i> : string): double <i>altitudeMode</i> : clampToGround, relativeToGround, absolute	
	<b>Numeric Comparison Functions</b>	
< GreaterThan>	GreatThan(a1: decimal, a2: decimal):	boolean
<GreaterThanOrEqual>	GreatThanOrEqual(a1: decimal, a2: decimal):	boolean
< LessThan>	LessThan(a1: decimal, a2: decimal):	boolean

<LessThanOrEqual>	LessThanOrEqual(a1: decimal, a2: decimal): boolean	
	Arithmetic Functions	
<Add>	Add (arguments: decimal):decimal	
<Subtract>	Subtract (a1: decimal, a2: decimal): decimal	
<Multiply>	Multiply (arguments: decimal): decimal	
<Divide>	Divide (a1: decimal, a2: decimal): decimal	
<Mod>	Mod (a1: integer, a2: integer): integer	
<Abs>	Abs (a: decimal): decimal	
<Round>	Round (a: double): double	
<Floor>	Floor (a: double): double	
	Logical Functions	
<AND>	AND (condition List: boolean): boolean	
<OR>	AND (condition List: boolean): boolean	
<NOT>	NOT (condition: boolean): boolean	
	Map Context Functions	
<viewZoom>	ViewZoom () : number	to get the map view zoom level number
<viewCenter>	ViewCentre (): Point	
<viewHeading>	ViewHeading (): double	range from 0 to 360 degrees
<viewTilt>	ViewTilt (): double	range from 0 to 360 degrees
<viewRoll>	ViewRoll (): double	range from −180 to +180 degrees
<viewPoint>	ViewPoint(): Geometry	Point
<viewportGlobeBounds>	ViewportGlobeBounds():Geometry	Polygon
<viewportCornerBounds>	ViewportCornerBounds (HitTestMode: string): Geometry HitTestMode: globe, terrain, buildings	Polygon, null
<GroundAltitude>	GroundAltitude (latitude:double, longitude:double): double	
<streetViewPosition>	The placement of the camera focus for a panorama image.	
<streetViewHeading>	The rotation angle around the camera locus in degrees relative from true north.	
<streetViewPitch>	The angle variance "up" or "down" from the camera's initial default pitch, which is often (but not always) flat horizontal.	
<TimePrimitive>	including <TimeStamp> and <TimeSpan> elements, for representing time conditions.	
<Region>	for representing spatial condition about a client’s viewport status. A Region is said to be "active" when the bounding box is within the	

	client's view and the LOD requirements are met.
	<b>User GeoLocation Functions</b> which may employ some sensors or software approaches to get the following measurements.
<clientLocation>	clientLocation (): Point
<clientLatitude>	clientLatitude (): double
<clientLongitude>	clientLongitude (): double
<clientAltitude>	clientAltitude (): double
<clientSpeed>	clientSpeed (): double
<clientHeading>	clientHeading (): double
<accuracy>	accuracy (): double
<altitudeAccuracy>	altitudeAccuracy (): double
	<b>Some sensor functionality</b> (extensible for various sensors) These sensors can capture various measurements of the context of things.
<inSignalRange>	inSignalRange (signalType:string, signalId:string, lowLimit: decimal, higherLimit: decimal) : Boolean
<accelerometer>	The acceleration force in m/s <sup>2</sup> that is applied to a device on all three physical axes (x, y, and z), including the force of gravity.
<linearAcceleration>	The acceleration force in m/s <sup>2</sup> that is applied to a device on all three physical axes (x, y, and z), excluding the force of gravity.
<gyroscope>	A device's rate of rotation in rad/s around each of the three physical axes (x, y, and z).
<rotationVector>	The orientation of a device by providing the three elements of the device's rotation vector.
<orientation>	Degrees of rotation that a device makes around all three physical axes (x, y, z).
<ambientTemperature>	The ambient temperature in degrees Celsius (°C).
<magneticField>	The ambient geomagnetic field for all three physical axes (x, y, z) in $\mu$ T.
<gravity>	The force of gravity in m/s <sup>2</sup> that is applied to a device on all three physical axes (x, y, z).
<proximitySensor>	The proximity of an object in cm relative to the view screen of a device.
<lightSensor>	The ambient light level (illumination) in lx.
<pressureSensor>	The ambient air pressure in hPa or mbar.
<relativeHumiditySensor>	The relative ambient humidity in percent (%).
	<b>Programming logic</b>
<variable>	to incorporate a parameter that may change dynamically
<value>	to give a known value (e.g. a constant) for evaluating with another parameters (e.g. a variable)

<b>&lt;event&gt;</b>	to indicate a event name/type that to be listened/subscribed to
<b>&lt;extensionGeoww&gt;</b>	<b>Extension to Geoww namespace</b> for extending the core-part language elements to support other domain-specific GeoPolicy definition.

The GeoPolicy Language can express context conditions with scalable complexity and comprehensiveness combining the physical-world with virtual-world context. Section 3.3.3 will demonstrate some examples on the usages of this language.

### 3.3.2 The extension mechanism for domain-specific elements

The term “context” is a comprehensive concept that itself may involve countless kinds of attributes characterizing the properties of things. It is impossible to compile a fully comprehensive context attributes for all things in the world. The GeoPolicy language just progressively involves in various fields, by providing an extension mechanism for incorporating domain-specific context attributes.

In addition to the “geoww” namespace, third parties can define domain-specific namespaces to add new language elements for those context attributes that cannot be represented by existing language elements. This extension mechanism using specific prefixes for corresponding elements can play as context ontology (Strimpakou et al. 2006) and taxonomy for expanding and grouping the expression of various context attributes for things.

As listed in Table 3-1, the <extensionGeoww> element works for adding new elements that have not been defined in the current core-part GeoPolicy language. Extended elements with their corresponding namespace prefixes can be nested in the <extensionGeoww> element.

## 3.4 Extended-KML Encoding Model for the GeoPolicy Language

As the expected capability listed in the starting part of Section 3.3, the GeoPolicy Language itself is encoding-neutral, and can be integrated into other languages to extending their functionality of expressing context conditions. This means the GeoPolicy Language can be encoded into different formats. For example, if used in the field of augmented reality, it can be



embedded into ARML (Augmented Reality Markup Language) (OGC 2014) by using an extended-ARML encoding format, to work with the host language of ARML.

GeoChannels are a new kind of resources to be introduced into the GeoWeb, which is currently populated with data snapshots of geospatial information. Now the attempt and effort of this research is to make GeoChannels become GeoWeb-native resources. To achieve this aim, an idea is to exploit the GeoWeb's resource representing language KML.

### **3.4.1 The Strategy of extending KML for the GeoChannel Web**

By analogy to HTML that has enabled a hyperlinked Document Web, KML (Keyhole Markup Language) (OGC 2008) is a widely-supported language for encoding, carrying and presenting geospatial information and thus has become a popular GeoWeb language, especially being widely used on geo-browsers and web-browser based mapping and applications. It will make significant and representative sense to enable KML to natively support GeoChannel technology. Firstly, this effort will make GeoChannels fall into the same family of geospatial resources that currently include spatial data prevalent on the GeoWeb. Secondly, by incorporating GeoChannel accessing functionality into KML, real-time updates can be natively achieved, while current KML standard employs pull-based mechanism to get updates by polling the sources of dynamics, without having achieved truly real-time updates. In addition, KML with GeoChannel-accessing capability can further help step towards the vision of "the Earth as universal desktop" for interacting with the physical world, as leveraging the popularity of KML can help populate the virtual world (such as a Digital Earth or map) with comprehensive dynamics, pervasive services and participatory collaboration with additional capability, such as location-based services and geo-clustered interactions, being brought to end users via GeoChannels.

The OGC KML standard provides a mechanism for extensions ---- to add additional elements that contain information beyond what is available in the current KML standard, by using a new namespace prefix. In this thesis document, a namespace prefix, i.e. **geoww** is used, which points to the URI reference of this namespace <http://www.geoww.net> as below:

```
xmlns:geoww="http://www.geoww.net"
```

Note: the namespace (web site) <http://www.geoww.net> is dedicated for this PhD project.

### 3.4.2 Extended KML Class Diagram for GeoChannel Technology

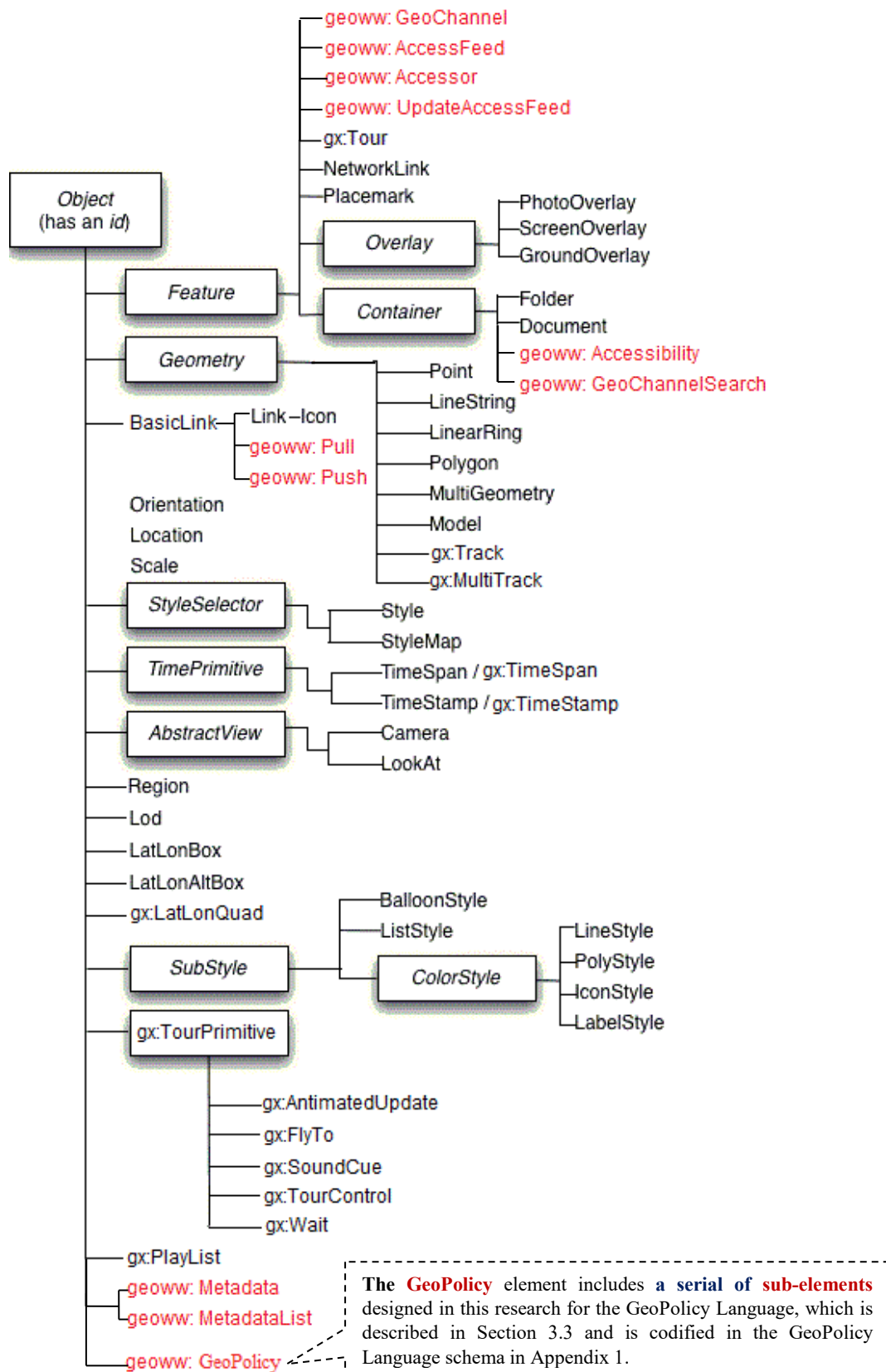


Figure 3-1: Extended-KML class diagram with the newly-proposed GeoChannel elements

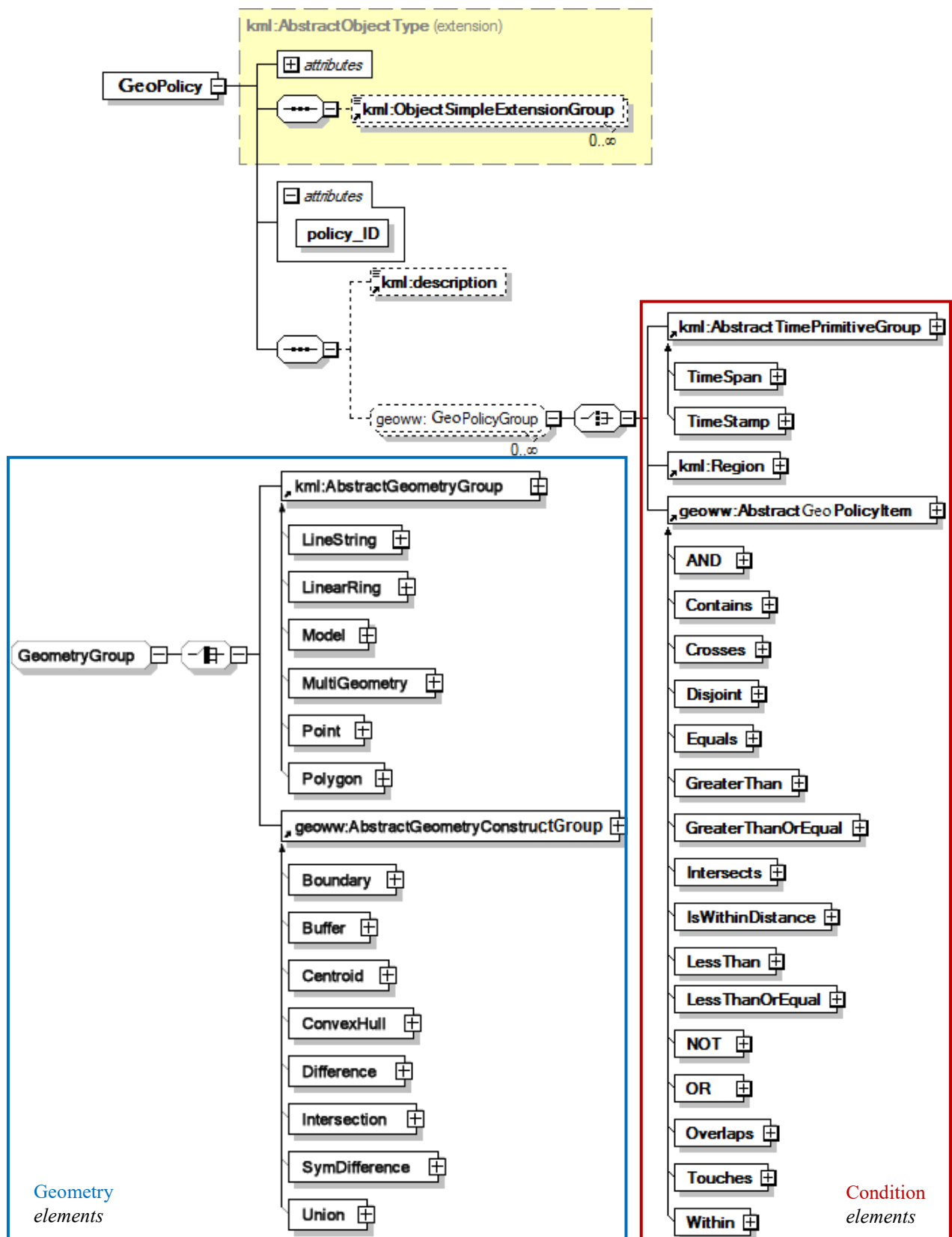


Figure 3-2: KML GeoPolicy: Geometry elements and Condition elements

By using this namespace-prefixing way, those geo-browsers or map viewers that do not currently support this proposed GeoChannel-encoded KML can just silently ignore this snippet, and the rest of the KML file should load and work without errors. In the future, if this proposed mechanism of KML-carried GeoChannel information is widely adopted, then this namespace-prefix will not be needed when this GeoChannel technique is normalized into future KML standard.

Actually, this research largely extends the current standard KML language for encoding multiple information models designed in this research. Figure 3-1 shows a high-level framework of KML with these extensions, which will be described in onward chapters

This section just designs a normative extended-KML model for encoding the GeoPolicy Language. Although only one entry for the GeoPolicy Language is shown in Figure 3-1 from a high-level view, there is actually a very rich model with many detailed extensions to KML. Appendix 1 in this thesis codifies the extended-KML model for the GeoPolicy Language, which currently supports all the elements and functions as listed in Table 3-1 for representing various context conditions.

These added elements are legal extension conforming to the extending rules codified by current KML standard (OGC 2008). Figure 3-2 shows some of the deriving/extending relations between standard KML elements and GeoPolicy condition elements. These relations are legally designed to make sure that these extended elements and existing standard elements of KML can seamlessly and compatibly integrate with each other. For example, these new elements can be included within current standard KML elements/files at right positions, and can also include specific standard KML elements inside.

Here this section does not elaborate all the extended elements codified in Appendix 1 and listed in Table 3-1, but just describes some considerations on designing a KML-compatible model for the GeoPolicy Language.

Compatibility and seamlessness is featured when designing this KML-extended encoding model, which ensures the joint and cooperative work between standard KML language and the extended elements for the GeoPolicy language. In this aspect, an outstanding capability is that existing and extended KML elements can combine and interchange with each other. This

capability makes it flexible to express GeoPolicy with scalable complexity. For example, KML standard `<Region>`, `<TimeStamp>` and `<TimeSpan>` elements can arbitrarily mix with newly-proposed relation analysing elements (as listed in Table 3-1). Another more advanced example is the interchangeability between standard KML geometry elements and the newly extended geometry-constructing elements, as shown in the left part (within the blue boundary) of Figure 3-2. This substitutable mechanism enables the capability for constructing very complex geometry objects/shapes for defining delicate spatial conditions, and enables harmonious and seamless integratability between the new GeoPolicy language components with existing KML standard language.

Now the GeoPolicy language can enhance the capability of KML on working for dynamic maps, pervasive services and other context-based applications, as the GeoPolicy language can turn the KML-encoded geospatial data into context-aware and condition-controlled resources for information retrieval or service provision. Chapter 5 that designs the GeoAccessFeed technology (which employs the GeoPolicy Language) will further illuminate technical details.

### **3.4.3 Examples of GeoPolicy Conditions**

Here are examples of GeoPolicy segments for demonstrating the usage of this KML-encoded GeoPolicy language.

The GeoPolicy in Listing 3-1 defines spatial and temporal conditions for people to access a university resources (e.g. online library services, or campus-based social networking) that are limited to on-site clients. The time for accessing must fall into the period of a specified academic year. Two spatial conditions are alternatives, i.e. a client's map viewport status or his/her physical location. A KML standard `<Region>` element is employed for testing the client's viewport. This `<Region>` is said to be "active" when the bounding box (`<LatLonAltBox>`, as shown by the red boundary in Figure 3-3) is within the client's map viewport and the LOD (Level Of Detail, `<Lod>`) requirements are met. Another spatial alternative spatial condition is to check whether a client is physically located in this campus which is illustrated using the cyan colour in Figure 3-3. This condition incorporates the element

of <clientLocation> which, when this condition is evaluated, will be substituted using the actual value of this client geographic location.

Location information can be sourced from GPS and location inferred from network signals such as IP address, RFID, WiFi and Bluetooth MAC addresses, and GSM/CDMA cell IDs, etc. Most modern Web browsers now provide the HTML 5 interface to get a client's current physical location under the permission/approval of this client. The processor program of GeoPolicy conditions can just make use of this HTML 5 interface to acquire this location information and use it to substitute the <clientLocation> variable in the GeoPolicy condition.

Listing 3-1: an example of GeoPolicy conditions demonstrating the usage of the KML GeoPolicy language.

```
<geoww:GeoPolicy policy_ID="Policy123">
  <description>a local access-control policy for the resources within the scope of an university
campus</description>
  <TimeSpan>
    <begin>2012-09-20</begin>
    <end>2013-06-15</end>
  </TimeSpan>
  <geoww:OR>
    <Region>
      <LatLonAltBox>
        <north>52.9554038322</north>
        <south>52.9474011396</south>
        <east>-1.1815575782</east>
        <west>-1.1930893052</west>
      </LatLonAltBox>
      <Lod>
        <minLodPixels>256</minLodPixels>
      </Lod>
    </Region>
    <geoww:Within>
      <clientLocation/>
      <Polygon>
        <outerBoundaryIs>
          <LinearRing>
            <coordinates>-1.1875143196,52.9481628944,0
-1.185128888,52.9485015048,0 -1.1845997498,52.9474011396,0
-1.183689884,52.9480625709,0 -1.1826182073,52.9488003139,0
-1.1815575782,52.950324499,0 -1.1816736884,52.9515971945,0
```

```

-1.1829881523,52.9528300584,0 -1.1845327222,52.9524072707,0
-1.1861528352,52.9554038322,0 -1.1916927038,52.9550818466,0
-1.1926260152,52.9550245184,0 -1.1930893052,52.9545030323,0
-1.1875143196,52.9481628944,0</coordinates>
</LinearRing>
</outerBoundaryIs>
</Polygon>
</geoww:Within>
</geoww:OR>
</geoww:GeoPolicy>

```

The Geometry encoding format makes use of existing KML syntax and semantics for representing geometry objects (e.g. the campus scope in cyan colour in Figure 3-3), which are actually the parameters (i.e. children elements) of these topological or geometric functions (elements) as shown in Table 3-1 and in Figure 3-2.

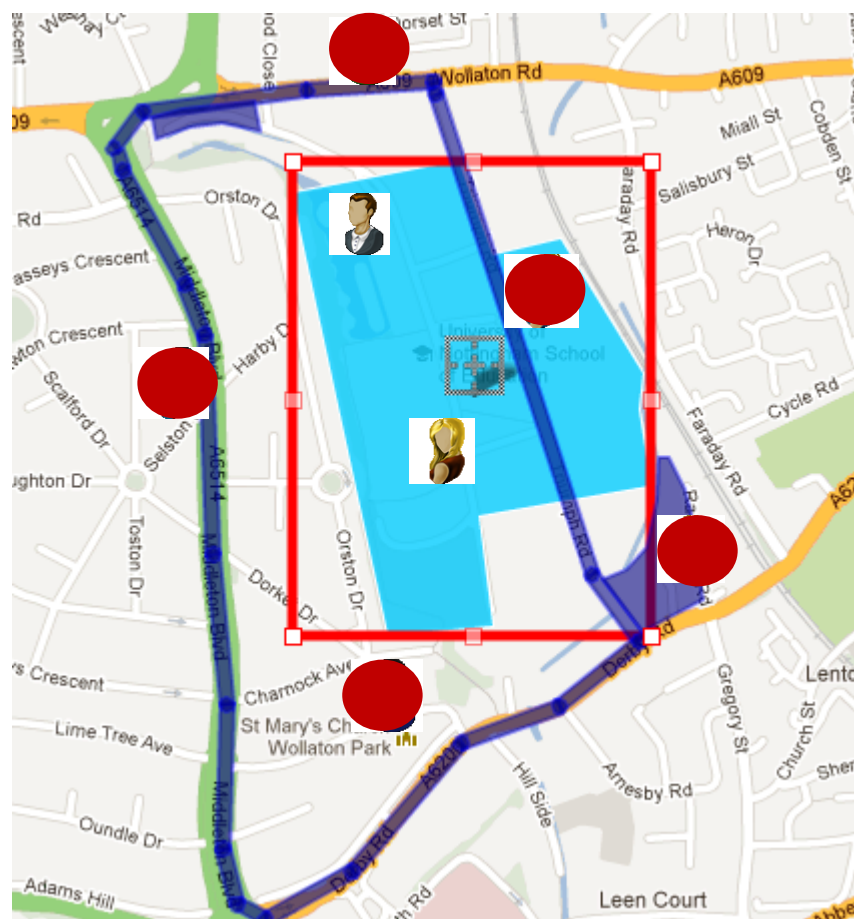


Figure 3-3: An example about GeoPoicy conditions for controlling access to GeoChannel resources

Multiple-level nesting and interchangeable relations are applicable for flexibly and scalably authoring complex spatial conditions. These Geometric constructing elements (functions) taking

existing geometry object as parameters can produce (i.e. result in) new geometry objects, which again can act as parameters to be used by other geometric constructing/measuring elements (functions). In addition, Standard KML-encoded geometry objects and these extended geometric elements (functions) produced geometries are inter-substitutable. So this nesting and interchangeable mechanism supplies an expressive capability for this KML GeoPolicy language to represent spatial conditions with scalable complexity.

Another segment of a GeoPolicy in Listing 3-2 exemplifies this complex application. This example describes a scenario where clients within a certain proximity of the motor roads or parking areas, as illustrated in blue colour in Figure 3-3, can receive the online traffic broadcast in certain periods of daily time. This policy segment in Listing 3-2, for clarity and brevity, only outlines some major elements without detailed content. Several geometry-constructing elements and geometry-encoded elements can be present in this policy. The <Buffer> element is used for representing the road area, and for expressing a certain proximity around a client's current location. Primitive geometries such as <Point>, <LineString> and <Polygon> elements are also present in this policy. The <Union> merges the traffic areas (road and parking). The <Overlaps> element takes these standard and extended KML elements as its parameters for testing spatial relation, along with the time condition (using the <Timespan> element) to form this complex GeoPolicy condition. A client's position [clientLocation] as a variable will be automatically substituted using a real-time location value by the GeoChannel AccessFeed processor.

Listing 3-2: an example of GeoPolicy demonstrating the GeoPolicy language capability of expressing complex spatial conditions

```
<geoww:GeoPolicy policy_ID="Policy12">
  <description>this local access-control policy tests whether a client is located in a certain proximity of a
  specific road or a parking area within two specific time scope. If so, then this client can access a GeoChannel which
  delivers space-time-specific traffic information</description>
  <geoww:OR>                                <!-- to test the time condition whether the client's arrival falls into any of time scopes -->
    <Timespan>                                </Timespan>                                <!-- a time scope -->
    <Timespan>                                </Timespan>                                <!-- another time scope -->
  </geoww:OR>
  <geoww:Overlaps>                            <!-- to test the spatial proximity between a client with the road and the parking area -->
    <geoww:Buffer>                            <!-- a certain vicinity around this client -->
      <clientLocation/>                        <!-- the client's real-time location -->
    </geoww:Buffer>
    <geoww:Union>                            <!-- the merged spatial scope of this road and the parking area -->
      <Polygon>                                </Polygon>                                <!-- a parking area -->
      <Polygon>                                </Polygon>                                <!-- another parking area -->
      <geoww:Buffer>                            <!-- a certain scope of a road route -->
        <LineString>                          </LineString>                          <!-- a road route -->
      </geoww:Buffer>
    </geoww:Union>
  </geoww:Overlaps>
</geoww:GeoPolicy>
```



```
</geoww:Buffer>  
</geoww:Union>  
</geoww:Overlaps>  
</geoww:GeoPolicy>
```

So just as exemplified, this KML GeoPolicy language used within a GeoChannel AccessFeed can express rich spatio-temporal conditions for controlling the expected behaviours and actions of this AccessFeed.

### 3.5 Summary

This chapter has designed GeoPolicy Language for expressing context attribute conditions of real-world things weaved in the GeoChannel Web, which maps and applies dynamic and random relations between things based on their context attributes. In some sense, the GeoPolicy Language for the GeoChannel Web can be analogized to HTML for the general Web, while the two languages work for different purposes.

A core set of language elements with extensible mechanism have been designed, and a normative schema model by extending KML has been codified. Some examples have exemplified the functionality and usage of this language.

The GeoPolicy Language features the following capabilities:

- 1) A general-purpose language for representing context conditions

It is expressive enough for expressing various context attributes that can be captured currently, such as space, time or other detectable attributes by current sensors; and it is extensible to support future sensors that can detect more kinds of context attributes, and to support defining domain-specific context conditions.

- 2) Multi-purpose functionality

The context conditions defined by the GeoPolicy Language can be used for various purposes, for example, for correlating resources with clients, for clustering clients in similar contexts, for defining and controlling access to pervasive services, or for controlling networking relations

between things, etc.

### 3) Scalable, encoding-neutral and embeddable (integratable)

It can define any context conditions with scalable complexity. This language itself is format/encoding-independent. So it can be represented by different encoding formats that host environments/languages are using. It can seamlessly fuse into other languages (i.e. host languages), for example other data-representing language such as GML, KML, ARML (OGC 2014), etc., to make them become conditional resources or services. Host languages and the GeoPolicy language can mix up according to both grammar. This feature can make the GeoPolicy Language widely applicable with various application scenarios where corresponding host languages exist.

### 4) Normative model

It has normative and unambiguous model for its syntax and semantics. For example, the extended\_KML schema codified in Appendix 1.

### 5) Easy-to-use

Context conditions are defined in a non-programming way. There is no need to re-programming applications or services, but only need to re-define context conditions by using this language

The GeoPolicy Language is a key technical component in the GeoAccessFeed technology system that involves some other techniques such as Geo-Off-On communication model, AccessFeed, Geoww-URI scheme, etc. All these techniques, which will be designed in coming chapters, work on the base of the GeoPolicy language. In the GeoAccessFeed technology, this GeoPolicy language is used for defining GeoServeZone, GeoListenZone, GeoObserveZone and GeoBlindZone, for controlling-access to GeoChannel resources, for dynamically correlating a resource with clients, for throttling communication for the Geo-Off-On communication model, or for clustering clients or federating resources.

## 4 Geo-off-on Communication Model

### 4.1 Introduction

Dynamic and random context relations usually imply that interaction between things is also changing in terms of the necessity for timeliness and frequency of communication. As identified in Chapter 2, there is a demand on an adaptive communication model, which can automatically adjust communication behaviours based on dynamic context relations, for throttling communication so as to save network bandwidth, reduce traffic cost, slow battery consumption of mobile devices and mitigate the overload of computation on server and client side. So resource-saving and scalability-improving calls for automatically adjustable communication patterns between things with dynamic context relations.

This chapter designs an adaptive communication mechanism, termed Geo-Off-On Communication Model, to cater for the GeoChannel Web which weaves real-world things into a network based on their dynamic context relations, for saving resources and improving scalability on network, computation, battery, etc.

Similar to GeoPolicy (designed in Chapter 3) prefixed with “Geo” which may imply space/time involvement, this prefix actually indicates a general involvement of various context conditions apart from space/time attributes. The name Geo-Off-On means that context conditions may function as a switch (i.e. off/on) for controlling and adjusting communication patterns between things involved.

The Geo-Off-On Communication Model is basically built on HTTP and WebSocket (or some other protocols) as constituent pull/push communication patterns, with an additional context-aware mechanism layer that employs GeoPolicy Language (designed in Chapter 3) for controlling communication patterns and frequencies. From this perspective, the Geo-Off-On model can be considered as a new or higher-level communication protocol taking into account context attributes of participants.

By analogy to HTTP and WebSocket protocols working for the general Web, the Geo-Off-On Communication Model can largely work for the GeoChannel Web populated by

dynamically-bridged things for their interaction.

## 4.2 The principle: discriminating GeoContext relations for communication

The basic strategy for the Geo-Off-On communication model is to give different priorities to clients that have different GeoContext relations with a target resource for their communication. This mechanism of discriminative communication can generally save networking and computing resource and can improve the scalability of a system.

### 4.2.1 Partitioning GeoContext Zones by GeoPolicy Conditions

As the GeoChannel concept interpreted in Chapter 2, a GeoChannel generally involves a contextualized resource and clients with respective context attributes named GeoContexts. The GeoContext relations between a GeoChannel resource and its different clients vary according to the difference of clients' GeoContexts. In this sense, the clients can be considered to be located in different context scopes, named GeoContext zones, which can be represented by different GeoPolicy conditions that reflect corresponding GeoContext relations between a GeoChannel and its clients.

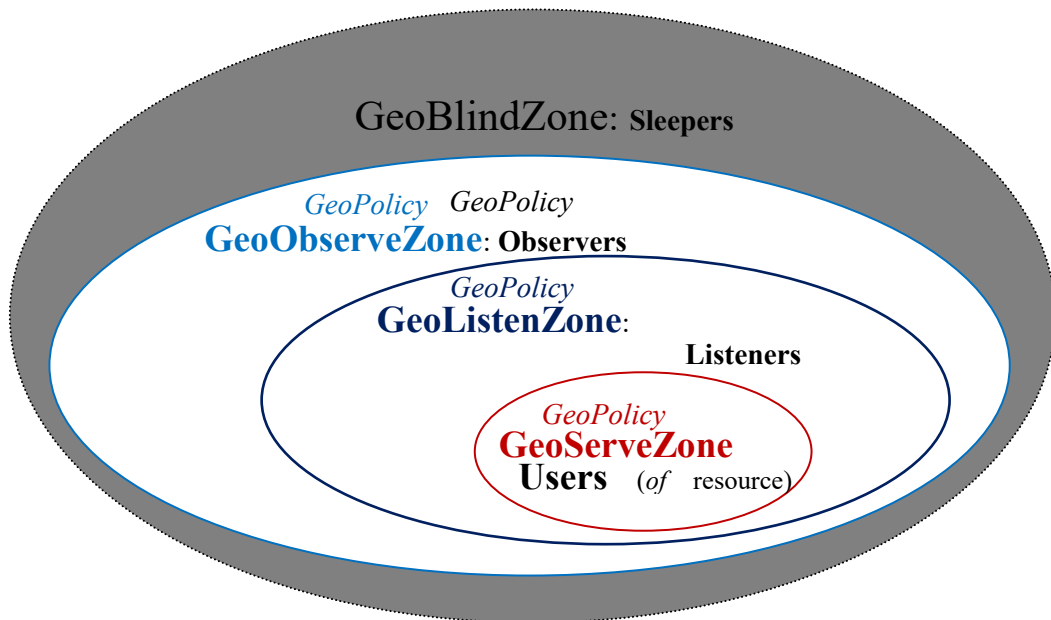


Figure 4-1: The concept of GeoContext Zones (GeoBlindZone, GeoObserveZone, GeoListenZone, GeoServeZone)

Intuitively Figure 4-1 depicts the concept of GeoContext zones by using a space partition

diagram, although GeoContext does not necessarily involve geographic or spatial scope. Sometimes a GeoContext zone may be a logically (mathematically) confined scope that may not necessarily or not be able to be portrayed as a physical range intuitively. For example, if a GeoPolicy condition takes clients' viewport (2D/3D) of a map (or earth) or other non-spatial attributes as parameters to match a GeoChannel's context, all possible matched users form a mathematical collection for a GeoContext zone that may not be like the geo-fencing scenario as depicted above.

This research proposes a category model for partitioning GeoContext zones, as illustrated in Figure 4-1. In this schema, all clients that are interested in a resource are categorized into four different GeoContext zones, termed GeoServeZone, GeoListenZone, GeoObserveZone and GeoBlindZone, and their clients are named Users, Listeners, Observers and Sleepers correspondingly.

Each GeoContext zone is defined by using a specific GeoPolicy condition, which can be generally expressed as below:

$$\textit{Matching} = \textit{GeoPolicy}_{\textit{Zone}_i}(\textit{Context}_{\textit{resource}}, \textit{Context}_{\textit{client}})$$

A client, whose GeoContext (i.e. space/time or other types of attribute) matches the GeoPolicy  $\textit{Zone}_i$  (i.e. making the *matching* a value of true), falls into the GeoContext zone  $\textit{Zone}_i$ . If a client meets multiple GeoContext zone conditions, then usually it is considered to belong to a GeoContext zone that has the highest priority among these matched GeoContext zones. For example, a client that concurrently meets both GeoServeZone and GeoListenZone is a GeoServeZone client.

In a specific GeoPolicy condition, the context parameters may involve various kinds of attributes of a resource and its clients. For example, a client's context may involve its physical location, its map viewport parameters, and time information. So a GeoContext zone defined by using a GeoPolicy condition may not be necessarily a physically geographic area (e.g. a range constrained by geo-fencing (Reclus & Drouard 2009, Schneider et al. 2008) that uses space topological predicate "within"), but it may be a mathematically logical scope resulted by this GeoPolicy condition. The illustration (in Figure 4-1) about GeoContext zones just conceptually depicts their scopes, which may not really be a geographically-bounded range; and these GeoContext zones are not necessarily adjacent geographically.

Consequently, based on the principle of partitioning GeoContext Zones as illuminated above, for a GeoChannel, a normal setting is, usually the range of the GeoListenZone is a superset of

the GeoServeZone. That is, at a specific moment, the GeoListenZone usually contains the current GeoServeZone, and involves an additional scope that may cover immediate-near-future GeoServeZone(s) that this GeoChannel is or will be proceeding to target. Normally, the GeoObserveZone usually contains (is broader than) the GeoListenZone, which in turn contains (is broader than) the GeoServeZone. Figure 4-1 just conceptually illustrates this interrelation between these different GeoContext zones.

GeoContext zones can be dynamically adjusted by updating their definition of GeoPolicy conditions. Partitioning GeoContext zones is a dynamic task which may be performed in a manual or automatic way. For a specific resource, a certain algorithm code can be programmed for automatically updating partition of its GeoContext zones. For example, in scenarios of a moving vehicle acts a mediator platform for geo-clustering clients (i.e. grouping other vehicles within a certain proximity) for their interaction. This mediator vehicle as a resource can use a sub-program to automatically modify its GeoContext zones (e.g. GeoServeZone, GeoListenZone, GeoObserveZone and GeoBlindZone) by updating corresponding GeoPolicy conditions that define these GeoContext zones respectively.

GeoServeZone, in the category model shown in Figure 4-1, is a most centric GeoContext zone. A **GeoServeZone** is a physical or logical zone defined by a specific GeoPolicy condition named access-control policy to confine a targeted range, fallen in which the users can be served by this GeoChannel's resources (services). This access-control policy may involve the augments of spatial, temporal or other types of context attributes of this GeoChannel and its clients for evaluating the matching/entitlement of GeoChannel clients to this policy. It is said that matched users are within this GeoServeZone, otherwise are out. A GeoServeZone may be a single zone, or may consist of multiple parts forming a logical zone.

GeoServeZone is the context scope in which access-policy-matched clients are located. These clients (named users), which are now accessing (connecting with) a resource, should have the highest priority for interacting with this resource. Those clients that are in non-GeoServeZone (i.e. outside the GeoServeZone) should have differentiated priority levels for client-resource communication. The non-GeoServeZone is meant to be divided into some parts, i.e. some other GeoContext zones, each of which is for identifying and accommodating different group of differentiated priority clients. Clients falling into a same GeoContext zone have the same priority level to communicate with the resource.

Usually a GeoServeZone is a context scope that confines a GeoChannel resource to a specific community of users. A simple example is geo-fencing (Reclus & Drouard 2009, Schneider et al. 2008) that may be considered as a use case of this GeoServeZone concept. In the scenario of

geo-fencing, the access-control policy usually specifies a user must locate within a specific area that is a physical zone geographically, for example, inside a circled range, or outside this circle but involving other area on the earth.

GeoContext zones are defined by different GeoPolicy conditions for controlling resource access and for throttling communication in a context-aware way. This clients--discriminating mechanism dynamically categorizes clients into different groups which can then have different patterns and priority levels for interact with a GeoChannel resource, to achieve on-demand, context-aware and scalable communication. This technology will be significant for improving the scalability of networking and computing for dynamic web maps, social-networking interaction and other types of real-time Web services and applications.

Section 4.2.2 will expound different GeoContext zones with their corresponding priorities and behaviours for communicating with a GeoChannel resource.

#### 4.2.2 Grading GeoContext Zones with discriminative Communication

Different GeoContext zones imply different context relations that in turn imply different priorities of communication between a target resource and its clients. So defining and grading GeoContext zones is to discriminate clients with different eligibilities and behaviours for communicating with a GeoChannel resource.

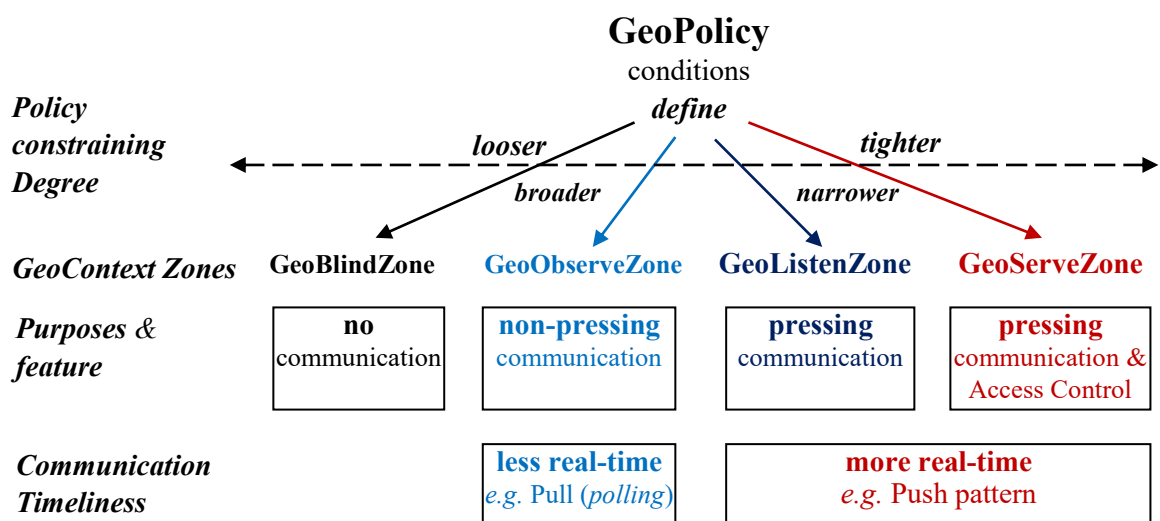


Figure 4-2: Partition and discriminate GeoContext Zones with different GeoPolicy conditions for resource access control or for scalable on-demand communication

Client-resource interaction is achieved via the communication between them. This technical strategy of GeoContext zones for improving system scalability, on one hand is to suppress excessive communication, on the other hand must ensure the timeliness of delivering latest

status information (such as GeoContext and access conditions). This strategy is fulfilled by discriminating in favour of those pressing communication but against those non-pressing (or unnecessary) communication. Now the question rises, what is, or how to differentiate or identify, pressing and non-pressing communication?

Generally there is no quantitative metrics for identifying pressing or non-pressing communication between a resource and its clients. Here some general (qualitative) principles or guidelines are given for defining different GeoContext zones.

Firstly, GeoServeZone is directly defined by a specific GeoPolicy condition according to the service aims of this resource at a certain time. Clients in GeoServeZone (i.e. Users) have access to and have high-level priority for communicating with this resource. They can immediately know their entitlement of access once this resource's access policy changes. In the case of not satisfying the latest access policy, a current GeoServeZone client will immediately become a non-GeoServeZone client.

Secondly, GeoListenZone usually needs to involve such a cluster of clients which, although not meeting the accessing condition of GeoServeZone at this moment (time), are context-nearly satisfying it, or can hopefully/possibly meet it if some little changes happening to the GeoContext of this resource or these clients or happening to current GeoPolicy of GeoServeZone. In this sense, so the scope of the GeoListenZone is usually adjacent to the GeoServeZone in context. In addition, two other types of areas may also be considered to be involved into GeoListenZone. If this resource, in a near or immediate future, plans to update its GeoServeZone's GeoPolicy to incorporate a new context area into this GeoServeZone, or it (i.e. this resource) can predict that some clients which are currently located in a non-GeoServeZone area will proceed to current GeoServeZone by changing their GeoContexts themselves, then these two kinds of areas can be considered as constituent parts of the GeoListenZone currently. So the GeoPolicy for defining GeoListenZone can take into account of these context areas. The clients in GeoListenZone, although do not have access to this resource itself, have high-priority of contacting this resource (or its agent) for exchanging status information. GeoListenZone clients can get real-time updates of this resource's status such as latest access policy, with which these clients can immediately get access to consume this resource once they meet the latest access policy. In this case of satisfying the latest access policy, they are actually being involved into the latest GeoServeZone.

Thirdly, GeoObserveZone involves the GeoContext scope in which the clients' communication behaviour with this resource is restricted to some extent. Compared with the GeoListenZone clients, the GeoObserveZone clients' GeoContexts are usually less satisfactory to this resource's



access policy. So generally GeoObserveZone is contextually further from GeoServeZone than GeoListenZone is. The GeoPolicy for defining GeoObserveZone is usually looser than the GeoPolicy that defines GeoListenZone. The GeoObserveZone policy just defines a wider range in which those clients are being considered to have non-pressing demand on client-resource communication. They are usually equipped with less real-time communication pattern, in contrast to the clients of GeoServeZone and GeoListenZone. The purpose to equip GeoObserveZone clients with certain communication capability is to give them some degree of opportunity to exchange status information with the resource. When getting the latest GeoPolicy conditions defined by this resource, a GeoObserveZone client checks its matching, and may then become a client member of other GeoContext zones such as GeoListenZone or GeoServeZone.

GeoBlindZone involves the clients that cannot meet any GeoPolicy conditions for defining GeoServeZone, GeoListenZone and GeoObserveZone. GeoBlindZone clients are termed Sleepers. Just as the name implies, they themselves do not have any communication opportunity for contacting the resource. On one hand, this completely suppressed communication can improve the system scalability; however, on the other hand, this technical strategy also brings potential risk that may cause disfunction of applications. Section 5.6 will illustrate this issue and will address this kind of problem by using supplementary technical mechanism to “wake up” the Sleepers in GeoBlindZone.

As illustrated in Figure 4-2, clients belonging to different GeoContext zones have different entitlements or features for their behaviours. Users (i.e. clients in GeoServeZone) are entitled to access this resource. For communicating status information such as GeoContexts or GeoPolicy conditions with the resource, Users and Listeners can perform pressing communicate in a more real-time way, while Observers take a less real-time manner. Sleepers (i.e. clients in GeoBlindZone) are constrained from communicating with this resource. The specific configuration of communication patterns and their functioning parameters can be dynamically set and adjusted on the fly by this resource. Consequently this discriminating roles model for clients can throttle client-resource interaction by differentiating their communication behaviours, in order to improve networking and computing scalability.

**Note:** In this chapter, the words: client and user, are usually different terms referring to different group of resource consumers. As illustrated in Figure 4-1, the word clients has a broad scope, not only including users that are eligible for or are using a resource, but also involving those (e.g. Sleepers, Observers, Listeners) who have interest/desire to access but may be not qualified for this resource at a certain time. Once a client gets the access, it can be grouped into the users. A consumer’s status is changing (from a general client to a user or vice versa) as the

accessibility for this consumer changes over time.

### **4.3 Design for the Geo-off-on Communication Model**

To improve networking and computing scalability and to avoid unnecessary interaction, Section 4.2 has proposed a novel strategy to partition and grade GeoContext zones for throttling client-resource communication and interaction. This section below materializes the technical implementation that mainly involves the controlling process model (i.e. working mechanism) and the information model, which are described in the sub-sections.

#### **4.3.1 Geo-off-on: a Cooperative & Adaptive Communication Model**

The current Web has provided some basic communication protocols, such as HTTP, WebSocket, etc., for exchanging or transferring data in general ways, without consideration on the context of senders/receivers. The pull pattern of request/response interaction and the push pattern of publish/subscribe interaction paradigms make up the basic communication patterns in the general Web.

Two basic principles or namely objectives have been taken into account when designing the Geo-Off-On Communication model: to maximize the possibility of getting latest updates (from counterparties), and to minimize the consumption of the resource of network and computation.

For delivering timely updates, ideally a simple and direct solution is to establish and keep a persistent Push connection (using WebSockets or Server-Sent-Events communication pattern) between things (e.g. a resource to each client), and then the latest updates, once happened, can be pushed to all receivers instantly. However, this solution might be not practical sometimes, or might be not necessarily expensive in terms of network/computing resource occupation. Persistent connections are usually resource-consuming that tend to impair performance scalability of a system when keeping massive number of persistent connections simultaneously. Another simple solution may resort to traditional Pull approach by polling frequently in a certain time interval to query possible updates. This solution can neither achieve really timely update nor use resources efficiently. Usually updates happening irregularly on one object cannot be exactly matched by the polling from another object, and over-frequent polling communication can also waste much networking and computing resource.

Now a cooperative solution, which fuses the Pull and Push patterns adaptively, is employed in the Geo-Off-On model. To achieve the two objectives (i.e. timely updates-getting and optimal resource-using), This Geo-off-on communication model can automatically switch

communication methods between Pull and Push patterns, and can dynamically adapt its communication behaviour such as adjusting polling frequency or turning on/off communication, in a context-aware way according to latest clients/resources' GeoContexts and GeoPolicy conditions. Figure 4-3 shows the cooperative (hybrid) pull (request/respond) with push (publish/subscribe) communication pattern framework.

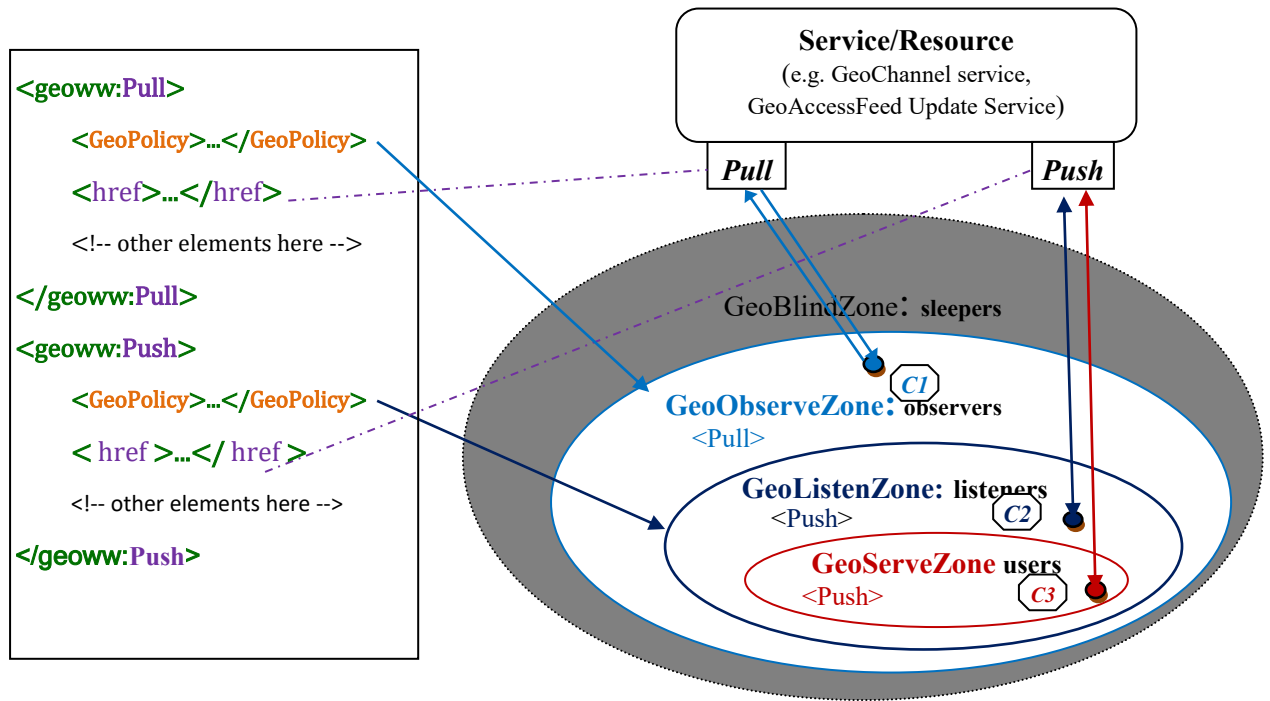


Figure 4-3: The Geo-off-on communication model with adaptive pull (request/respond) & push (publish/subscribe) patterns

This Geo-off-on communication model is enabled by Pull/Push elements information model (as designed in next section). Firstly, the current GeoPolicy conditions discriminate the clients into different GeoContext zones. The GeoPolicy conditions for the <Pull> and for the <Push> elements virtually define the GeoObserveZone and GeoListenZone correspondingly. GeoListenZone clients (namely listeners) and GeoServeZone clients (namely users) can use Push communication pattern, while GeoObserveZone clients (namely observers) uses Pull pattern. GeoBlindZone clients (namely sleepers) are suppressed for communication. So clients belonging to different GeoContext zones have different priorities/entitlements to communicate with the target resource by using different patterns or configuration of communication.

Secondly, when a client changes its GeoContext, for example, transiting between different GeoContext zones, it may take different configuration of communication for interacting with a resource. The GeoPolicy conditions can be on-the-fly adjusted via an updating mechanism

(described in Section 5.6), which results in re-partitioning GeoContext zones that assign clients with new entitlements or configuration on communication behaviours. So changes happening to either clients' GeoContext or resources' GeoContext can lead to automatically shift of communication patterns between <pull> and <push>, and dynamically adjustment of communication frequency, in a GeoContext-aware and on-demand way.

This Geo-off-on model can function as a general communication protocol for between-objects (e.g. resources, clients) interaction with GeoContext-aware, on-demand and adaptive communication for improving resource efficiency and system scalability. For example, when a GeoChannel service works for disseminating geospatial dynamics/events, providing location-based services or intermediating users' interaction, this Geo-off-on model can smartly throttle and adjust service-client and client-client communication adaptive to the dynamic context relationships between resources and users, and between users. Chapter 5 will exemplify the Geo-Off-On communication model for delivering the updates of GeoAccessFeeds.

#### **4.3.2 Pull/Push Elements Information Model**

The Geo-Off-On communication model is materialized by its information model. As shown in the left part of Figure 4-3, which mainly outlines the content of a <Pull> and a <Push> elements, the <Pull> and the <Push> elements may be both present in the case that both pull and push communication patterns are needed; or either of them is present in the case that only pull or push pattern is needed.

The normative information schema for the Geo-Off-On communication model is codified in Appendix 2.

A typical example application of the Geo-Off-On communication model is for delivering dynamic updates of AccessFeed (described in Chapter 5). The detailed syntax for this use is exemplified in Listing 5-6. Section 5.6 will elaborate the technical details.

### **4.4 Application Scenarios**

The Geo-Off-On Communication Model can be widely used in many application domains. It can especially play an active role in dynamic environments where participants of communication are subject to context changes and expect adaptive communication behaviour for saving resources and for improving scalability

The following exemplifies some application scenarios.

### **1) A native communication model for the GeoChannel Web**

The Geo-Off-On Communication Model is basically built on HTTP and WebSocket (or some other protocols) as constituent pull/push communication patterns, with an additional context-aware mechanism layer that employs GeoPolicy Language (designed in Chapter 3) for controlling communication patterns and frequencies. From this perspective, the Geo-Off-On model can be considered as a new or higher-level communication protocol taking into account context attributes of participants.

By analogy to HTTP (or other protocols e.g. WebSocket) working for the general Web, the Geo-Off-On communication model can act as a native communication model largely working for the GeoChannel Web populated by dynamically-bridged things for their interaction; while it can also be used for the general Web.

### **2) Updating AccessFeeds discriminately**

The Geo-Off-On communication model was originally motivated by and then designed for the task of updating AccessFeeds in an on-demand and adaptive way. AccessFeed is a technical component of GeoAccessFeed which is an umbrella term for a set of techniques involving GeoPolicy language, Geo-Off-On communication model, AccessFeed and Geoww-URI scheme, etc. Section 5.6.4 will elaborate the Pull/Push information structure with delicate mechanisms for the Geo-Off-On communication model to conduct adaptive updates of AccessFeeds. Clients accessing (or interested in) a same resource with its AccessFeed may be discriminated to cater for timeliness and necessity on getting AccessFeed updates. Section 4.2 illuminated the principle (see Figure 4-1, Figure 4-2) and Section 5.6 will describe the implementation of updating AccessFeeds by using the adaptive Geo-Off-On communication model.

### **3) Inter-vehicle interaction on demand**

Things with changing context attributes usually expect adjustable communication behaviours between these things for saving resources and for improving scalability of a system. Use Case 2 in Chapter 9 presents massive vehicles moving on the road network of a city. The inter-vehicle interaction can make use of the Geo-Off-On communication model, which can prioritize the communication between vehicles running on adjacent segments on the road topology networks, but throttle communication between vehicles away from each other and without direct context topology relations. The prioritized interaction works for communicating relevant traffic information between near-distance participants on a route, while usually less relevance is in demand between faraway participants.

## 4.5 Summary

This chapter has designed the Geo-off-on communication model for the GeoChannel Web, on which the dynamic relations between real-world things call for an adaptive communication model for automatically adjusting communication behaviours based on dynamic context relations, for saving resources and for improving scalability.

The Geo-off-on communication model features timeliness, efficiency, adaptability and scalability for between-objects (e.g. resources, clients) communication and interaction. It can throttle communication based on latest context, so as to save network bandwidth, reduce traffic cost, slow battery consumption of mobile devices and mitigate the overload of computation on server and client side.

The term, Geo-off-on, implies Geo (e.g. space/time)-switch (on/off) for GeoContext-aware controlling and throttling of communication. The smart mechanism of the Geo-off-on communication model is to discriminate clients that have different contexts for diversified client-resource interaction. That is, a resource can dynamically categorize its clients into different groups, namely partitioning GeoContext zones, by defining different context conditions that involve evaluating clients' contexts. Partitioning a full GeoContext domain into multiple zones is not necessarily a physical division (such as space/time partitions), but may be a logical division when a context attribute is parameterized by some non-physical factors. These GeoContext conditions can automatically determine the belongingness of a client with its specific GeoContext to a specific GeoContext zone. This resource can specify different communication patterns and configuration for different GeoContext zones. Thus this client-discriminating strategy can smartly suppress those unnecessary communications, and can favour those pressing client-resource interactions.

By analogy to the HTTP communication protocol for the general Web, the Geo-off-on is a native communication model for the GeoChannel Web, while it can also be used for the general Web.

## 5 GeoAccessFeed

### 5.1 Introduction

The GeoChannel concept (proposed in Chapter 2) reflects a group of dynamic relations between a resource and clients with specific context attributes. The GeoPolicy language has been designed in Chapter 3 for defining context conditions, and the Geo-Off-On communication model designed in Chapter 4 can adaptively update GeoPolicy conditions. Now it is a task to design a mechanism for dynamically correlating things in contexts, i.e. for operating dynamic relations between things based on their context conditions in a GeoChannel.

A context condition for correlating a resource with its clients is namely a GeoChannel access-control policy, which makes a GeoChannel resource conditionally available and accessible. Only contextually matched clients can connect or keep connected to a GeoChannel for using its resources. This mechanism can makes general Web services become context-based GeoChannel services. That is, a GeoChannel's services target a specific community of users which meet the GeoPolicy of this GeoChannel. This access control process for dynamically correlating a resource with clients is also termed GeoChannel Access-Organize in this research.

This Chapter elaborates the GeoAccessFeed technology that centres on AccessFeed technique. Firstly, Section 5.2 describes the framework and principle of the GeoAccessFeed technology. Section 5.3 identifies the functional features and components expected for AccessFeed technique. The detailed information model and workflow process model of AccessFeed are designed in Section 5.4 and Section 5.5. And then Section 5.6 designed cooperative mechanisms for updating AccessFeeds.

The GeoAccessFeed technology system is the core component in the full GeoChannel technology architecture. Actually GeoAccessFeed is an umbrella term used for grouping a set of techniques, involving GeoPolicy language, Geo-Off-On communication model, AccessFeed and Geoww-URI scheme, etc., all of which share this joint name GeoAccessFeed. Prior chapters and this chapter design some of these techniques, and next chapter will illuminate Geoww-URI technique.

## 5.2 GeoAccessFeed -- Framework & Principle

This section here gives a general overview of the GeoAccessFeed system, describing its technological principle and components relational model, which can facilitate elaborating detailed techniques in subsequent sections of this chapter.

GeoAccessFeed is the name for terming this GeoChannel Access-Organize technology. The term GeoAccessFeed refers to a context-based Access-Organize technological framework that is centred on the GeoChannel AccessFeed facility, which is the core component of this GeoAccessFeed technology.

It is useful to examine the term GeoAccessFeed in more detail. This is a compound name, with three parts basically. The “Geo” prefix indicates the context (e.g. space/time) nature that this technology involves for GeoContexts, control conditions, etc. The part “Feed” implies somewhat similar feature to web feeds in terms of the usage for delivering metadata and its updates about a resource. However, apart from this similarity, GeoAccessFeed differs from web feed substantially, whether on their information model, working model or on their functionality. And the constituent part “Access” indicates client-resource relation which may involve controlling resource access, correlating clients with a resource, GeoClustering clients, or organizing resources, etc. So the term GeoAccessFeed can basically reflect functional involvement of this GeoChannel Access-Organize system. Actually, this GeoAccessFeed technology has a wide range of capability and usage, among which some main functionality include geo-contextualizing and identifying resources, delivering updates, access control, and organizing resources and clients, etc.

### 5.2.1 Components Relational Model

The GeoAccessFeed technology for a GeoChannel Access-Organize system involves several functional components. Figure 5-1 illustrates the components framework.

The GeoAccessFeed system mainly includes three components (as illustrated in the left part of Figure 5-1), i.e. Geoww-URI, AccessFeed (required) and AccessFeed Update Service, among which a Geoww-URI and an AccessFeed are specific to a resource, and the AccessFeed Update Service can be deployed as a common web service to be used by any resources.



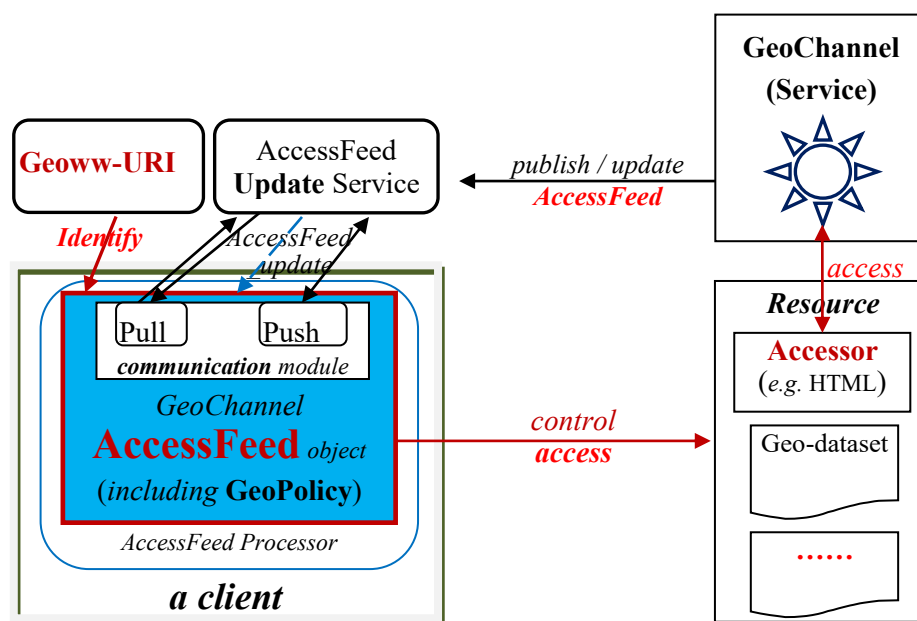


Figure 5-1: GeoAccessFeed system components relational model

AccessFeed is the core component of the GeoAccessFeed system. A GeoChannel's AccessFeed is a specific-structured information file/message which, at runtime supported by a processor of AccessFeeds, functions as a client-side autonomous program unit that contains its internal components (modules), i.e. communication module, access-control module and control-condition module. The main functions of an AccessFeed involve controlling access to resources, and acquiring updates of this AccessFeed via the communication with the AccessFeed Update Service. Both of the access control and communication actions are controlled by the condition module named GeoPolicy. In a rough sense, an AccessFeed possesses integrated function of feed (for delivering updates) and of access control.

The AccessFeed Update Service is for supplying dynamic updates of AccessFeeds to resources' clients. If a resource does not change its GeoContext and access-control policy, then there is no need to employ this update service. In this case, an AccessFeed can do the access-control work using static conditions. The update service is used to get AccessFeed updates published by a GeoChannel service, and then to update clients with a latest version of AccessFeed that will function as a new version of the AccessFeed program unit for controlling access to this resource and for controlling communication with this update service by using the updated controlling condition. For communication with the AccessFeed Update Service, an AccessFeed can use the Pull and/or Push patterns, both of which may be controlled by a GeoPolicy condition respectively.

At a certain moment when the access-control condition is satisfied, this client can access the

target resource identified by this AccessFeed. As shown on the right part of Figure 5-1, for example, to get access to a geospatial dataset, or to fetch and then run an Accessor (such as a HTML program) that will connect to the target GeoChannel for accessing its resources.

### **5.2.2 The Principle of GeoChannel Access-Organize**

With the pre-knowledge of geo-contextualizing resources, partitioning GeoContext Zones and GeoAccessFeed components model, now this section here is to introduce the working principle of this GeoAccessFeed technology for Access-Organize of resources.

Its working principle mainly involves some points as listed below, and is illustrated in Figure 5-2:

- 1) a generic and expressive policy language for expressing controlling conditions. See details in Chapter 3.
- 2) a cooperative and adaptive communication model for on-demand, timely and scalable communication. See details in Chapter 4.
- 3) partitioning GeoContext Zones and discriminating priority levels by using corresponding controlling conditions, for improving scalability (of communication and computing). See details in Chapter 4.
- 4) encoding access and communication control logic into an AccessFeed in a non-programming manner. See details in Section 5.4.
- 5) dynamically updating AccessFeeds for altering access conditions, communication configuration, etc. See details in Section 5.6.

Figure 5-2 illustrates an overall schematic diagram which brings related components together and depicts various relations about reference (identifying/pointing), control, connection or information flow, etc. To make it more understandable of this GeoAccessFeed technology, the following is to illuminate its complex process from different perspectives and in a brief way.

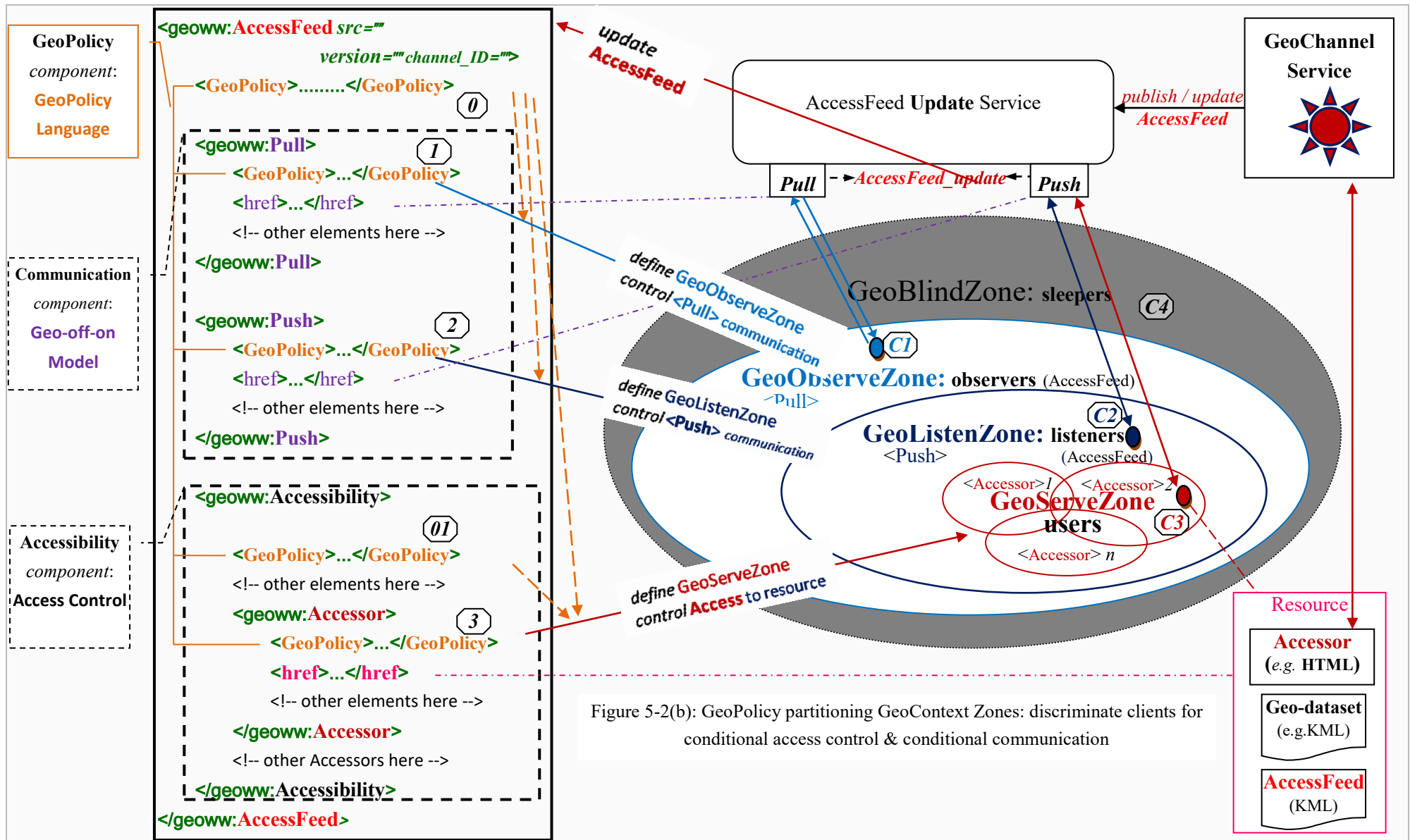


Figure 5-2(a): AccessFeed Components with GeoPolicy respectively

Figure 5-2: GeoAccessFeed working principle & components relational model

A GeoChannel resource publishes and updates its AccessFeed which can be discovered, bookmarked and run by its clients of interest. An AccessFeed is a structured information file/message (as exemplified in Figure 5-2 (a)) which usually contains multiple components. The <Accessibility> component is for identifying and controlling the access to target resources. The Communication component can usually equip itself with Pull-pattern and/or Push-pattern communication capability. This is a cooperative and adaptive communication model termed Geo-Off-On that is elaborated in Chapter 4. The <GeoPolicy> component can define multiple separate conditions for controlling the <Accessibility> and the Communication components. These GeoPolicy conditions are represented by using GeoPolicy language.

It is noted that, as the content of an AccessFeed can be updated dynamically, at a specific moment an AccessFeed may contain parts of these components or is even fully empty.

Clients interested in and meant to access a resource load (i.e. run in the processor of AccessFeeds) this resource's AccessFeed which is then instantiated into an autonomous program unit for controlling access to this resource and for communicating with an AccessFeed-Update Service, given this AccessFeed has these constituent components. At a certain time moment/phase, a client usually belongs to a specific GeoContext Zone, which is defined by a GeoPolicy condition embedded in this AccessFeed

An AccessFeed may contain several GeoPolicy conditions. For example, in Figure 5-2 (a) the condition (1) with (0) defines the GeoObserveZone controlling the Pull-pattern communication; the condition (2) with (0) defines the GeoListenZone controlling the Push-pattern communication; and the condition (3) with (0) and (01) defines the GeoServeZone controlling resource access and having the Push-pattern communication capability.

As illustrated in Figure 5-2 (b), which marks four clients (denoted by **C1**, **C2**, **C3** and **C4**) that are located in different GeoContext Zones. At this moment only the client **C3** located in the GeoServeZone can access this GeoChannel resource; non-GeoBlindZone clients (**C1**, **C2** and **C3**) can get updates of this AccessFeed., while client **C4** cannot as it is located in the GeoBlindZone. For communicating with the AccessFeed Update Service, the client **C1** uses Pull pattern to polling updates, while client **C2** and **C3** can use Push pattern to get instant (real-time) updates. A received AccessFeed update will substitute/update the existing AccessFeed program unit on a client. When a client's GeoContext (i.e. space/time attribute) changes, or it gets a new AccessFeed update, these (existing or new) GeoPolicy conditions will be re-evaluated and then this client may change to another different GeoContext Zone within which clients have another type of attribute/entitlement for resource access and for

communication. For example, the client **C4** that is currently within the GeoBlindZone may change to a non-GeoBlindZone zone to be able to get AccessFeed updates. At a specific time moment, different clients may have different versions of the same resource's AccessFeed, as these clients have different opportunities (priorities) for getting a latest AccessFeed update. However, Figure 5-2 here simply shows a same AccessFeed for explaining the working principle. Actually, each client uses/runs its own version of AccessFeed for determining its own belongingness to what GeoContext Zone which affects (controls) the entitlement or behaviour for resource access or communication.

The basic behaviours of an AccessFeed are to organize conditional access to resources and conditional communication for getting AccessFeed updates. The conditional access conducts policy-based continuous control of access to resources; and the conditional communication performs on-demand, scalable, adaptive and cooperative client-server communication, which may report a client's GeoContext to server, and may get back (from server) AccessFeeds that carry updates/changes of a resource's GeoContext, policy and adjusted client-side communication configuration, etc.

The GeoChannel Access-Organize process for controlling access to resources and for controlling communication is GeoContext and policy-aware and driven, as any changes happened to the GeoContexts of a GeoChannel and its clients, or to the policy condition, will trigger a working cycle of GeoChannel Access-Organize. Dynamic and context-aware control on the communication with the server (e.g. AccessFeed Update Service), and dynamic and context-aware control on the access to resources (e.g. GeoChannels), can be achieved by dynamically adjusting GeoPolicy conditions imposed on the communication (i.e. the Pull/Push elements) and on the Accessibility/Accessor elements, within an AccessFeed. For example, nomadic GeoChannels can dynamically push or return latest GeoPolicy (i.e. spatio-temporal condition) to its clients that can then dynamically throttle communication in a context-aware way with remote services (e.g. AccessFeed-Update Service or Policy evaluation Service).

The subsequent sections in this chapter will elaborate the constituent components of this GeoAccessFeed technology.

### **5.3 AccessFeed: Correlating Things in Context**

AccessFeed is the core component in the GeoAccessFeed technology system, as depicted in Figure 5-1 and in Figure 5-2. This section onwards illuminates its technical details involving

expected features and functional components, information model and workflow model, as well as updating mechanism.

In a generalized sense, there are two basic functions for AccessFeed: organizing access and bridging things. Conditionally organizing dynamic access for a client/resource to a resource can imply controlling the dynamic relation between the involved parties. Thus generally AccessFeed can correlate things on organizing access and bridging things. So AccessFeed just meets the demand of the proposed GeoChannel Web for mapping and applying dynamic and random relations between things in context.

### **5.3.1 Desired Features of AccessFeed**

The AccessFeed technique is expected to satisfy the desires on:

- automatically detecting and communicating (in pull or/and push patterns) a client's GeoContext;
- capability of context awareness and sensitiveness for access and communication control;
- jointly working with server-side facility to achieve scalable complexity on access-control policy/condition;
- GeoWeb flavour, that is, making GeoChannels and their access-control work become GeoWeb-native resources and functionality, with the capability and convenience for mashing up, bookmarking and sharing GeoChannels, and for easily geo-contextualizing general (i.e. originally non-geospatial) resources to embedding their accessing points into the GeoWeb space;
- normative and easy-to-use model for authoring client-side logic for GeoChannel access control, including representing access-control mechanism in a non-programming way;
- seamlessly following and assorting with GeoChannel Discover Service in terms of information and work flow.

These desires above have motivated the design of AccessFeed technique as a key facility component for the GeoChannel architecture.

### 5.3.2 AccessFeed Functional Components

A GeoChannel AccessFeed is a text-based file (or message) that contains access-control information involving the GeoContext relation between clients and resources. At runtime an AccessFeed virtually functions as a client-side autonomous program unit module, which, on one hand, can locally execute access-control logic for dynamically correlating resources with clients based on existing context conditions, and on the other hand can work with remote services to get latest updates to this AccessFeed about changes on access-organizing logic for adjusting control behaviours.

So some main functional components are usually included within an AccessFeed as shown in Figure 5-3. These functional modules work jointly to achieve the functional objectives of a GeoChannel AccessFeed.

The GeoPolicy module defines context conditions that may evaluate the relation between resources' and clients' GeoContexts, for controlling the work of the Accessibility and the Communication modules. The Communication module, employing adaptive communication patterns and configuration based on GeoPolicy conditions, works for interaction between this runtime AccessFeed and remote services for communicating latest updates on client-resource GeoContexts and on this AccessFeed. The Accessibility module dynamically controls the access to target resources based on GeoPolicy conditions. And the TopologyLink module represents the relation between relevant AccessFeeds in terms of their GeoContexts on similarity or proximity for navigating between these AccessFeeds, by analogy with the hyperlink relation between HTML pages for the general Web.

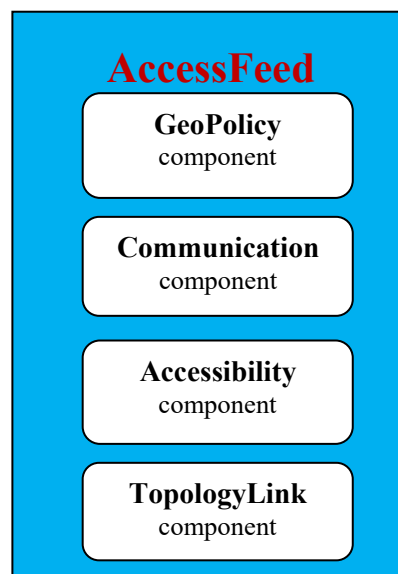


Figure 5-3: Functional components of a GeoChannel AccessFeed

The subsequent sections in this chapter will design the information model and workflow model for these functional component modules.

## **5.4 AccessFeed Information Model: the Framework**

This section proposes an extension to current standard KML for supporting the representation of AccessFeeds. An AccessFeed may have functional components as illustrated in Figure 5-3. Since previous Chapters have codified the GeoPolicy Language and the Geo-Off-On communication model, this section does not need to describe the GeoPolicy and the communication components for AccessFeed, but just designs the information models for the whole structure and its Accessibility and TopologyLink modules in the sub-sections.

### **5.4.1 AccessFeed structure Model with Extended KML Encoding**

The principle and mechanism of AccessFeed is basically encoding-neutral. Some data/information-encoding standards such as CityGML, ARML (OGC 2014), etc., can be extended for representing AccessFeeds in their domain-specific applications. This research opts for KML language. The strategy of extending KML language to support the GeoChannel Web has been illuminated in Section 3.4.1. KML (Keyhole Markup Language) (OGC 2015) is a language for representing and presenting geospatial information for web maps or earth browsers. It is a popular data format for the GeoWeb. Just due to the popularity of KML, an encoding model that extending KML for AccessFeeds can redound to demonstrating the functionality and popularizing the adoption of the AccessFeed technique.

The current standard KML is basically a geospatial data-encoding format, lacking the capability of representing access-control logic. So this research attempts to extend the current KML encoding standard to support GeoChannel AccessFeed that works as a client-side mechanism as opposed to (Geo)XACML which is basically a server-side solution for access control.

Actually, Chapter 3 and Chapter 4 have extended KML standard for encoding the GeoPolicy Language and the Geo-Off-On Communication Model, as codified in Appendix 1 and Appendix 2. Now this chapter further extends KML for representing AccessFeed information model, which is codified in Appendix 3.

To give a full context of the proposed KML extension in this research, Figure 3-1 in Section



3.4.2 illustrates the whole class tree diagram for KML elements with newly-added elements that are designed from Chapter 3 to Chapter 7. Among these added elements, some are involved in AccessFeed information model.

The following describes the syntax, semantics and working mechanism for extended-KML-encoded GeoChannel AccessFeeds. The formal information model of GeoChannel AccessFeeds is codified in Appendix 3, which uses the W3C XML Schema language (W3C 2012) to describe the extended-KML grammar for conformant GeoChannel information instances such as AccessFeeds.

For the clarity and brevity of illumination, the following describes the full AccessFeed syntax in several parts (Listing 5-1, Listing 5-2, Listing 5-4, Listing 5-5, and Listing 5-6) , and mainly highlights some major content that is more involved in the access-control mechanism, with purposely omitting some other content. The full syntax model for AccessFeed is codified in Appendix 3.

Listing 5-1 briefly outlines the structure of an <AccessFeed> element which is contained within a <GeoChannel> element that extends the KML <Feature> abstract element. The <GeoChannel> element can occur at any positions in a KML file where standard KML feature elements (such as Placemark, NetworkLink, Document, Folder, PhotoOverlay, etc.) can occur. This mechanism makes GeoChannels (as a new kind of geospatial resources) able to be treated as normal (i.e. standard) KML features to be encoded in KML files, that is, to make GeoChannels become normal resources to the GeoWeb. GeoChannel elements and standard KML elements can coexist/mix within a KML file.

Listing 5-1: The outline structure of an AccessFeed element within a GeoChannel element

**Syntax:** (only an outline is listed here, the Schemas are codified in *Appendix 3* and *Appendix 4*)

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2"
      xmlns:geoww="http://www.geoww.net/geochannel">
  <!-- there may be some other standard KML data snippets here -->

  <geoww:GeoChannel channel_ID="id123">    <!-- the id attribute is an Unique Identifier (UID)-->
    <!-- the GeoChannel element may only contain
         Metadata element, or ChannelFeed element, or may contain both of them -->

    <geoww:Metadata>...</geoww: Metadata>    <!--Its data model is depicted in Listing 7-5,
                                                including the returned data from GeoChannel Discovery service -->
    <geoww:AccessorSelector>...</geoww:AccessorSelector>

    <geoww:AccessFeed src="http://..kml" version="2012-10-08" feed_ID="123">

      <GeoPolicy>...</ GeoPolicy>

      <geoww:Pull>...</geoww:Pull>
```

```

<geoww:Push>...</geoww:Push>

<geoww:Accessibility>...</geoww:Accessibility>    <!-- Its data model is depicted
<geoww:TopologyLink>...</geoww: TopologyLink>                                     in Listing 5-2-->

</geoww:AccessFeed>

<!-- there may be some other standard KML data snippets here -->

<geoww:UpdateAccessFeed>... </geoww:UpdateAccessFeed>

</geoww:GeoChannel>

<geoww:UpdateAccessFeed>... </geoww:UpdateAccessFeed>

<!-- there may be some other standard KML data snippets here -->

</kml>

```

The <GeoChannel> element mainly contains three parts of sub-elements: <Metadata>, <AccessorSelector> and <AccessFeed>. The <Metadata> is for carrying the returned catalogue metadata from a GeoChannel Catalogue Service which will be designed in Chapter 7. The detailed structure and content of the <Metadata> element is described in Listing 7-5 and in Appendix 6.

Syntactically, a <AccessFeed> element can also lie outside a <GeoChannel> element, that is, it can sit at any places where a KML <AbstractFeatureGroup> element (such as <Placemark>, <GroundOverlay>, <Document>, etc.) can appear. For example:

```

<kml xmlns="http://www.opengis.net/kml/2.2" xmlns:geoww="http://www.geoww.net/geochannel">
  <geoww:AccessFeed src="http://example.com/feed028.kml" version="2012-10-08" feed_ID="123">
</kml>

```

The <AccessFeed> element works for GeoChannel Access-Organize, which contains a set of sub-elements, among which the <Pull> and <Push> elements for the Geo-Off-On Communication Model work for dynamically delivering updates of an <AccessFeed> itself; the <GeoPolicy> element expressed by the GeoPolicy Language works for controlling context conditions; the <Accessibility> element functions for controlling access to target resources of a GeoChannel; the <TopologyLink> element involves the context topology relations to other AccessFeeds.; and the <UpdateAccessFeed> element works for conveying the updates of AccessFeeds.

## 5.4.2 Accessibility module Information Model

This section presents the information model for the <Accessibility> component in AccessFeed.

Listing 5-2: The <Accessibility> element syntax in extended-KML encoding for AccessFeed

**Syntax:** (only an outline is listed here, the Schemas are codified in *Appendix 3* and *Appendix 5*)

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2"
  xmlns:geoww="http://www.geoww.net/geochannel">
  <geoww:GeoChannel channel_ID="" <!-- this channel_ID attribute is required (a GUID)-->
    <geoww:Metadata>...</geoww:Metadata>
    <!--to choose which Accessor(s) to be run when matching GeoPolicy conditions. -->
    <geoww:AccessorSelector>...</geoww:AccessorSelector>

    <geoww:AccessFeed src="http://...kml" version="2012-10-08" feed_ID="123">

      ① <GeoPolicy>...</GeoPolicy> <!-- a common space-time-aware condition for controlling the
        Pull/Push/Accessibility elements-->

      <geoww:Pull>...</geoww:Pull> <!--This Pull element may be absent. -->
      <geoww:Push>...</geoww:Push> <!--This Push element may be absent. -->

      <geoww:Accessibility client_specific="true">
        <description>...</description>
        ② <GeoPolicy>...</GeoPolicy> <!-- a common space-time-aware condition for controlling
          all the Accessors elements at the channel level-->

        <!-- there may be 0,1 or more Accessors listed below -->
        <geoww:Accessor accessor_ID="ID123456" access="true">
          <name>...</name>
          <description>...</description>
          ③ <GeoPolicy>...</GeoPolicy> <!-- the space-time-aware condition for controlling
            this Accessor element at the Accessor level-->

          <href>...</href> <!-- URL for this Accessor. This element is required if
            the access attribute of this Accessor is true -->

          <bridgeWith>...</bridgeWith> <!-- for bridging this GeoCluster/GeoServeZone to others-->
          <Geometry>...</Geometry> <!-- for indicating the spatial scope (coverage)-->
          <Overlay>...</Overlay> <!-- for indicating the spatial scope (coverage)-->
          <geoww:ContentType>...</geoww:ContentType>
        </geoww:Accessor>

        <!-- The Accessors (like this one below) may not be present if their access attribute is false-->
        <geoww:Accessor accessor_ID="ID123457" access="false">
          <name>...</name>
          <description>...</description>
          <!--here the <href> element is absent when the access attribute of this Accessor element is false-->
          <Geometry>...</Geometry>
          <Overlay>...</Overlay>
        </geoww:Accessor>

      </geoww:Accessibility>
    </geoww:AccessFeed>
  </geoww:GeoChannel>
</kml>
```

The content of the <Accessibility> element may directly come from a GeoChannel AccessFeed KML file (fetched after the process of a Discover Service) that had statically pre-encoded full content of the <Accessibility> inside a <AccessFeed> element, or may come from dynamical update returned from a remote GeoChannel Update Service. Mostly the latter source is the case, which is to dynamically get/update or conceal the <Accessibility> content.

The <Accessibility> element has an attribute field “client\_specific” which indicates whether or not this <Accessibility> content is specific to a certain client. As the AccessFeed can report (via the Pull/Push elements) client-specific context to a remote access-control service, which may return an updated AccessFeed with this client-specific <Accessibility> part. This “client\_specific” attribute as a mark field will be checked when updating the whole AccessFeed.

An <Accessibility> element usually contains some sub-elements of <Accessor>, each of which can have the <href> sub-element that is the URL address about the target resource (Accessor) of a GeoChannel. However, there may be no <Accessor> element inside the <Accessibility> element, or no URL information inside a certain <Accessor> element, if no corresponding resource is accessible to this client at a certain time. The present <Accessor> elements and their URL information may change dynamically over time, subject to the returned value from an AccessFeed Update Service. An <Accessor> element may have one or multiple sub-element <bridgeWith>, which is used for bridging (merging) GeoClusters/GeoServeZones by GeoBridge mechanism.

GeoChannel services (through which the target resources are provided by a GeoChannel) are exposed and accessed through GeoChannel Accessors. A GeoChannel may have one or multiple Accessors, which are virtually a kind of Web client application modules that are distributed on the Web and fetched to the client side for running. When running, a GeoChannel Accessor can connect to a GeoChannel to use corresponding GeoChannel services. A user must get the URL of a GeoChannel Accessor to fetch and run this Accessor. In general sense, an Accessor corresponds to a resource delivered via a GeoChannel. The <AccessorSelector> element prior to the <AccessFeed> element is used to choose which Accessor(s) contained in the <AccessFeed> element to be run for accessing corresponding GeoChannel resource(s).

This AccessFeed technique here is designed as a general access-control facility which is not limited to controlling access to GeoChannel resources, but can also be able to function on other types of resources. An <Accessor> element can have a sub-element <ContentType> that indicates the type information about the <href>’s URL-pointed resource. By knowing the resource type, the AccessFeed processor can use corresponding function to deal with this

resource. For annotating content type information of resources embedded or referenced within a KML file, N. Freed, 1996 ref\_num3357 type information. A similar solution is employed within the AccessFeed information model. This research here proposes the Media Type information about a GeoChannel Accessor is annotated as below:

```
<geoww:ContentType>
  <geoww:MediaType>application</geoww:MediaType>
  <geoww:MediaSubType>GeoChannel+Accessor</geoww:MediaSubType>
  <geoww:MediaFileExtension>html</geoww:MediaFileExtension>
</geoww:ContentType>
```

This AccessFeed technique can be used to control access to other types of resources other than GeoChannels. The official list of some common MIME Media types assigned by the IANA (Internet Assigned Number Authority) is published on (IANA 2013), and new media types, e.g. this *Application/GeoChannel+Accessor* information, can be registered according to certain procedures as indicated in (IETF 2013).

#### ● Nested AccessFeeds

It is noted that the target resources identified by the <href> element within an <Accessor> can be another AccessFeed. This can form a mechanism of nesting/chaining AccessFeeds in the access-organize system of GeoAccessFeed. This nested/chained-AccessFeeds mechanism has very significant application for organizing and accessing resources (e.g. dynamics, services, interaction, etc.) hierarchically. For example, by analogy to current Web mapping capability of browsing map data snapshots in different LoD (Level of Detail), the nested-AccessFeeds technique can work for organizing and accessing a city's event news, location-based and social-networking services at various space/time hierarchies. The Geoww-URI model technique, Accessor Bus facility and GeoChannel Network technique (in Chapter 8) will take into account this nested/chained-AccessFeeds mechanism.

#### ● Geo-tags for resource discovery

The AccessFeed information model has inbuilt resource-discovery functionality by space-based searching. The information model of AccessFeeds supports incorporating geometric objects within a <AccessFeed> elements. These geometries can function as spatial tags, which can be searched/filtered by spatial topology query to find out GeoChannel resources of interest with specific spatial involvement such as geographic coverage, range/scope, etc.

Listing 5-3: Geometries as geo-tags for space-based search to identify AccessFeeds of interest

```

<geoww:AccessFeed src="http://...km/" version="2012-10-08" channel_ID="id123">

  <MultiGeometry>
    <Point> ... </Point>
    <LineString> ... </LineString>
    <Polygon> ... </Polygon>
  </MultiGeometry>
} geo-tags

<geoww:Pull>...</geoww:Pull>
<geoww:Push>...</geoww:Push>

<geoww:Accessibility>

  <geoww:Accessor accessor_ID="ID123457" access="true">
    <name>...</name>
    <href>...</href>
    <Geometry>...</Geometry>
    <Overlay>...</Overlay>
  </geoww:Accessor>
} geo-tags

  <geoww:Accessor accessor_ID="ID45678" access="false">
    <name>...</name>
    <Geometry>...</Geometry>
    <Overlay>...</Overlay>
  </geoww:Accessor>
} geo-tags

</geoww:Accessibility>

</geoww:AccessFeed>

```

The access to target resources is controlled by GeoPolicy conditions, as the <AccessFeed> (<Accessibility>) syntax described in Listing 5-2 (e.g. the condition statements marked as ①, ②, ③). These <GeoPolicy> elements are about context conditions that are expressed by using the GeoPolicy language.

### 5.4.3 TopologyLink module Information Model

The TopologyLink module in an AccessFeed represents the relations between this AccessFeed in question and some other AccessFeeds with their corresponding resources. The relevance between these AccessFeeds is about the GeoContexts of their resources in terms of the similarity or proximity on their GeoContext conditions (i.e. GeoPolicies) that correlate clients with resources.

Figure 5-4 illustrates a scenario where the TopologyLink mechanism can play a role. In a road network, the immediately connected road segments function as service resources, their

corresponding AccessFeeds have certain GeoContext relations. The TopologyLink module in an AccessFeed for a road segment can involve some other relevant AccessFeeds corresponding respective GeoChannel resources. In Figure 5-4, assuming it is an undirected road network, the AccessFeed of No\_3 can link to AccessFeeds of No\_1, No\_4, No\_5, No\_6 and No\_7, as their adjacency and connectivity or reachability. This is just the topology relation of the GeoContexts of these AccessFeeds corresponding to their target resources/services.

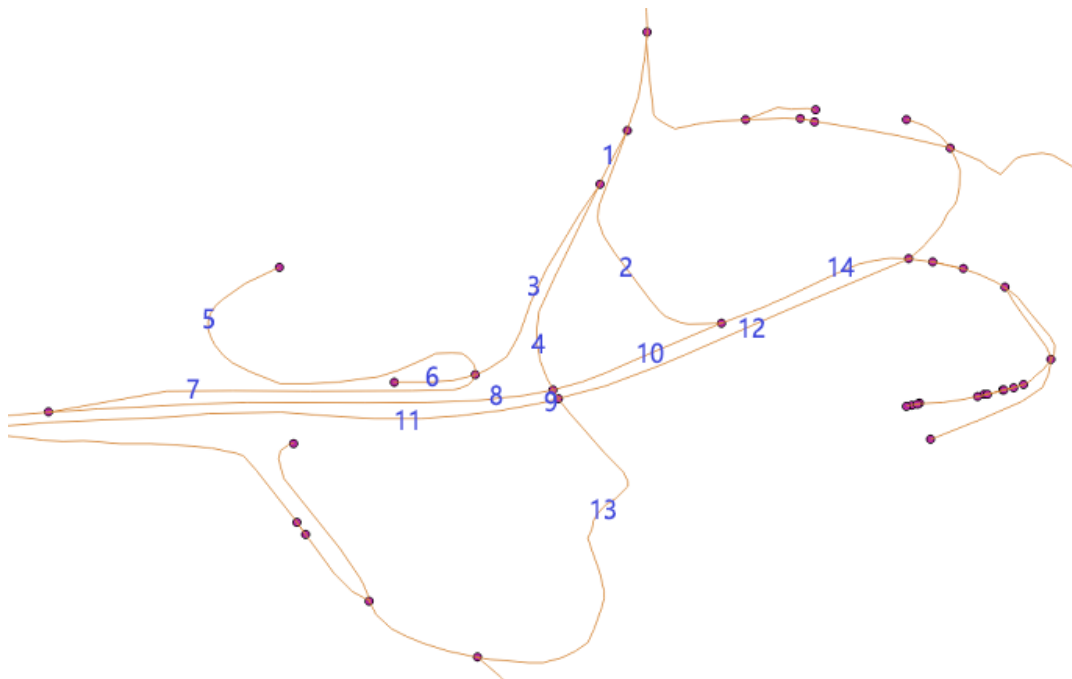


Figure 5-4: An example of road network with adjacent segments illustrating TopologyLink concept and function

In a generalized sense and in other application scenarios, TopologyLink relations are not necessarily limited to space/time topology, but can refer to various context attribute relations other than space/time context. The GeoPolicy Language designed in Chapter 3 can work for various context attribute scenarios.

The information model of the TopologyLink element is presented in Listing 5-4. A <TopologyLink> can include multiple <Link> elements with respective GeoPolicy conditions for controlling the switching to their target AccessFeeds, which are URLs denoted by ①, ② and ③ in Listing 5-4. In this snippet example, it is specially noted that, ① has a URL schema with a prefix “geoww://”, differing from the <http://> URL schema in ② and ③. The geoww-URL is a new URI schema technique designed in this research and will be illustrated in Chapter 6.

Listing 5-4: The <TopologyLink> element syntax in extended-KML encoding for AccessFeed

```

<geoww: AccessFeed>

    < geoww: GeoPolicy>...</geoww: GeoPolicy>    <!-- a common space-time-aware condition for
controlling the                                     Pull/Push/Accessibility elements-->

    <geoww: Pull>...</geoww: Pull>
    <geoww: Push>... </geoww: Push>
    <geoww: Accessibility>... </geoww: Accessibility>

    <geoww: TopologyLink>

        < geoww: GeoPolicy>...</ geoww: GeoPolicy>

        <geoww: Link>
            < geoww: GeoPolicy>...</ geoww: GeoPolicy>
            ① <href> geoww://... </href>    <!--geoww-URL for a target AccessFeed -->
        </geoww: Link>


        <geoww: Link>
            < geoww: GeoPolicy>...</ geoww: GeoPolicy>
            ② <href> http://... </href>    <!-- URL for a target AccessFeed -->
        </geoww: Link>

        <geoww: Link>
            < geoww: GeoPolicy>...</ geoww: GeoPolicy>
            ③ <href> http://... </href>    <!-- URL for a target AccessFeed -->
        </geoww: Link>

    </geoww: TopologyLink>

</geoww: AccessFeed>

```



The function of the Topology Links within an AccessFeed is mainly for navigating between these relevant AccessFeeds. When a client's GeoContext changes into a certain status that can satisfy the GeoPolicy condition of a Topology Link-pointed AccessFeed, this client will be getting and interacting with this new AccessFeed with its corresponding resource. The effect is to conditionally switch/navigate to another GeoChannel. In this sense, the AccessFeed TopologyLink mechanism for the GeoChannel Web can be analogized to the hyperlink relations between HTML pages for the general Web that navigates clients between web resources. The difference is, the hyperlinks on a web page usually navigates users by click actions, while the TopologyLink mechanism switches GeoChannels in an automatically way when a client's GeoContext matches the GeoPolicy of a target GeoChannel. So the AccessFeed technique with TopologyLink mechanism can cater for a pervasive service environment where various



ubiquitous sensors can detect the context attributes of clients and resources.

Actually, the content included in the <TopologyLink> of a certain AccessFeed can be adjusted dynamically based on GeoContext of a client and/or resource. For example, in the road network shown in Figure 5-4, when a vehicle travels on a certain segment such as segment No\_3, its <TopologyLink> may include Links to No\_1 and No\_4 if the vehicle is towards upward direction, and may include Links to No\_5, No\_6 and No\_7 if the vehicle is towards downward direction in this figure. The AccessFeed Update Service that will be designed in Section 5.6 works for updating AccessFeed content dynamically. This adaptive mechanism for adjusting Topology Links to meet clients' changing GeoContexts can achieve context-aware services which are brought or triggered by adaptive Topology Links. So AccessFeed technique can achieve smart pervasive services.

The use cases in Chapter 9 will demonstrate a street view example, which exemplifies the application of the TopologyLink function for navigating between multiple AccessFeeds of resources that have adjacent GeoContexts.

## **5.5 AccessFeed Workflow Process Model**

The normative information model for AccessFeed files/messages has been designed in previous sections (and codified in Appendix 3, Appendix 4 and Appendix 5), and now it is necessary to design an unified working process model for runtime AccessFeeds to behave prescriptively for fulfilling access-organize functionality on GeoChannel resources. A normative working process model can guide and facilitate standardizing the implementation of AccessFeed processors that may be designed by different vendors.

This section is to prescribe the normal procedures of runtime AccessFeeds of GeoChannel. Here it is not meant to present a concrete implementation of a GeoChannel AccessFeed processor, but is to outline some common tasks (i.e. event-driven actions) and processing workflow as guidelines that actual AccessFeed processors should take into account.

A GeoChannel AccessFeed represented in an extended-KML-encoded file, once having been instantiated into a runtime AccessFeed object on a client, virtually becomes a client-side autonomous program unit with all working logic as encoded in its information model with

sub-model components, e.g. Accessibility model (i.e. the <Accessibility> element), communication model ((i.e. the <Pull> and <Push> elements), and GeoPolicy model (i.e. <GeoPolicy> element). The specific functions and behaviour logic is to be interpreted and executed by an AccessFeeds processor that virtually turns an extended-KML-encoded AccessFeed into a program unit at runtime.

There are basically three phases to deal with an AccessFeed program:


1. Start. When a client starts to run an AccessFeed, chosen from the search result list of a GeoChannel Catalogue service, or from the bookmarks of GeoChannels (AccessFeeds), the client-side processor program of AccessFeeds will firstly try to fetch this latest version of this target AccessFeed. Once received, and then proceeds to the phrase 2;
2. Runtime: the event of receiving this fetched AccessFeed file triggers the action to instantiate this AccessFeed object, then it enters into the runtime phrase, becoming an event-driven program unit, which is listening to some events and then taking corresponding actions accordingly, as described in Table 5-1.
3. End. When the client cancels the running of this AccessFeed, if it is currently connected with the GeoChannel service, then this connection will be terminated to stop using this GeoChannel service, and then the AccessFeeds processor will cease this AccessFeed program unit and release relevant resources.

A runtime AccessFeed object as an event-driven working unit responds to various events by taking corresponding actions. The Table 5-1 below just shows some main events-actions mapping.

It is noted that the procedure of processing steps described here is just a brief workflow framework for reference implementation. It is not necessary for an actual AccessFeed processor implementation to completely follow these identical steps. It is only necessary to fulfil the mechanism of these events-actions mapping as exemplified here for achieving autonomous program units of runtime AccessFeeds. An actual AccessFeed processor implementation may

consider more technical details for optimizing processing efficiency and responsive timeliness to the changes of clients' context and resources' access-control policy conditions.

Table 5-1: GeoChannel AccessFeeds workflow process model: events-actions mapping

<div> <div>Events</div> <div>(external) Actions</div> </div>		AccessFeed file/message received via Pull/Push	Time elapsed/ changed	Map (2D/3D) View changed	geoLocation changed
	Update/ instantiate AccessFeed itself	update/instantiate AccessFeed object by new version or <Accessibility> with "client-specific=true"			
	Evaluate <GeoPolicy> elements	evaluate all <GeoPolicy> elements	evaluate those <GeoPolicy> elements involved in Time context, e.g. KML <TimePrimitive > involved.	evaluate those <GeoPolicy> elements involved in map view context, e.g. KML <Region> involved.	evaluate those <GeoPolicy> elements involved in client's location context, e.g. <clientLocation> involved.
	Control Accessors (start/stop/keep running)	Check all-level (i.e. its own level and all ancestor levels) GeoPolicy conditions for each <Accessor> elements: 1) If all-level policies are true, then keep this Accessor running, or fetch its Accessor program and start running it; 2) Otherwise, terminate those running Accessors.			
	Pull/Push:  contact server (report client's context to the server to get AccessFeed update)	If the Pull/Push-level and ancestor-level policies are all true, and			
		if <refreshMode> is "onChange",	if this timing event matches the <refreshMode> "onInterval" or "onExpire",	if this view-changed event matches <viewRefreshMode> "onstop" or "onPolicyToTrue",	if <locationRefreshMode> is "onChange",
		then contact the server by sending message with defined format and content.			

## 5.6 Updating AccessFeeds

Dynamically updating AccessFeeds to reflect latest GeoPolicy conditions and communication configuration for controlling access to target resources, for changing methods of clustering clients and for adjusting communication configurations is a critical work for dynamically



- 1) fetching the latest version of AccessFeed file, when starting to run an AccessFeed chosen from the GeoChannel Discovery results or a bookmark list;
- 2) the AccessFeed Processor refreshes an AccessFeed at a regular time interval;
- 3) the client can manually refresh this AccessFeed;
- 4) <adhocRefreshInterval> for clients in GeoBlindZone to get AccessFeed updates;
- 5) the <Pull> element requests the AccessFeed update;
- 6) the <Push> element receives the AccessFeed update pushed;
- 7) when an Accessor starts to connect to a GeoChannel service (node), the later will checks the client-side version of in-use AccessFeed, if it is stale, then the latest version will be returned to this client;
- 8) when the client is connected to and using a GeoChannel service, once the GeoPolicy change happens, this GeoChannel service will instantly return this new AccessFeed version to this client.

Some main components and workflows are involved in these multi-level mechanisms:

***AccessFeed (program/object) unit:***

An AccessFeed object unit usually has the inbuilt Geo-off-on communication component. Both or either of the <Pull>/<Push> modules can be present/absent, and can dynamically become present or absent after updating of this AccessFeed. The <Pull>/<Push> communication parameters model supplies a set of sophisticated mechanisms for adaptive and automatic communication behaviours, for example: the setting and functioning principles of the refresh actions triggered by the events of changes on time, map viewport or physical location, etc.; the on-the-fly adjustable time interval for refresh, or time-interval-triggered refresh constrained in a specific period; dynamically re-configuring Pull/Push communication model by on-the-fly updating of an AccessFeed; mutually cooperation or switch/substitution between <Pull> and <Push> communication patterns; wake-up mechanism by <Pull> to <Push>; etc.

***Processor (client-side) for AccessFeeds:***

It Supports system-configured interval refresh, or manual refresh for updating the current

version of AccessFeed on a client. When starting to run an AccessFeed (chosen from the results of GeoChannel Discovery service, or from bookmark list), it firstly fetches the online latest version of this AccessFeed, but not uses the local bookmarked version that might have got stale.

#### ***AccessFeed Update Service:***

When having received a Pull request, check the client-side version number information about a GeoPolicy in the pull request message, only respond with new version or custom update (i.e. client-specific <Accessibility> part) of GeoPolicy if any. When having received a Push connection request and then established this connection, check the client-side version number information about a GeoPolicy in the push connection request message, immediately respond with a new version (if any) of the GeoPolicy.

#### ***GeoChannel Service:***

When starting to run an Accessor as its control condition (GeoPolicy) is true, the GeoChannel service to be connected will firstly check the version information of the current GeoPolicy used by this client. If it is stale (i.e. not the latest version), then this GeoChannel service will not be formally connected, but will instead return the latest version of its GeoPolicy to this client, which can then re-evaluate this latest GeoPolicy. If the evaluation result is true, then this client will again start to run this Accessor to formally connect to this GeoChannel service. During the period of connected state when a client is formally using a GeoChannel service, the latter, on one hand, is providing service (e.g. delivering geospatial dynamics, or location based services, etc.), on the other hand, will also notify the client of the latest version of the GeoPolicy when a new update happen. On receiving this update, this client will re-evaluate this GeoPolicy, to determine whether or not to disconnect (when evaluation is false) or keep connected (when evaluation is true) to this GeoChannel service.

### **5.6.2 GeoBlindZone & GeoLagZone: Wake-up & Catch-up**

The strategy taken by the Geo-off-on communication model (designed in Chapter 4) for the GeoAccessFeed access-organize technology partitions the GeoContext scope into different GeoContext zones in order to throttle the communication between a resource and its clients. A resource discriminates different GeoContext zones by setting different GeoPolicy conditions, which result in different priority levels (i.e. communication patterns or configurations) for various clients to get a latest update of this resource (e.g. an AccessFeed). This GeoAccessFeed technology results in the phenomena of GeoBlindZone and GeoLagZone.

The concept of GeoBlindZone, as illuminated in Section 4.2, refers to the context scope in which the clients themselves cannot communicate with the target resource. GeoBlindZone clients (named Sleepers) don't meet the GeoPolicy conditions for the Pull/Push communication patterns. So they themselves cannot get the updates of the target resource (e.g. an AccessFeed).

The concept of GeoLagZone, to some extent, is an alias of GeoObserveZone (as illustrated in Section 4.2), as the GeoObserveZone clients (named Observers) usually undergo some latency time (i.e. a lag) to get real-time updates of the target resource by using the Pull communication pattern.

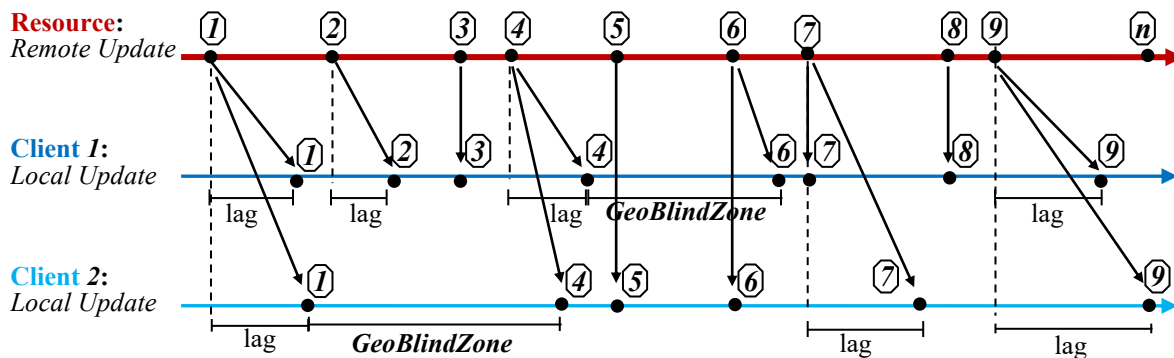


Figure 5-6: Example about the phenomena of GeoBlindZone and GeoLagZone

GeoBlindZone and GeoLagZone are resource-clients relative, and may change over time. Figure 5-6 exemplifies the phenomena of GeoBlindZone and GeoLagZone. The remote resource (such as an AccessFeed) is updated over time, indicated by using different version numbers. The clients (e.g. Client\_1 and Client\_2) with different Geo-contexts (i.e. located in different GeoContext zones) get these updates via the Geo-off-on communication model respectively. Client\_1 got the updates of ①, ②, ④ and ⑨ by using the Pull communication pattern resulting in time lags more or less, and got the updates of ③, ⑦ and ⑧ in real time via Push communication. However, it missed the update of ⑤ as it was within the GeoBlindZone once. With regard to Client\_2, it got the real-time updates of ⑤ and ⑥ and the lag updates of ①, ⑦ and ⑨, but missed the updates of ② and ③ for the GeoBlindZone reason and missed the update of ⑧ for the reason of less frequency to poll the server. The manners for Client\_1 to get the update of ⑥ and for Client\_2 to the updates of ④ will be explained in Section 5.6.2.2.

#### 5.6.2.1 Advantage & Disadvantage: Avail & Avoid

The Geo-off-on communication model with the phenomena of GeoBlindZone and GeoLagZone

can improve networking and computing efficiency and system scalability, as it can allocate more network bandwidth and processing capability to those more pressing resource-client interactions. GeoBlindZone clients (i.e. Sleepers) are completely suppressed to communication with the target resource. GeoLagZone clients (i.e. Observers) periodically polls the resource using Pull pattern which is a stateless protocol that does not require the server to retain session information or clients' status; and the system does not need to guarantee the Observers to get every updates of the target resource, for example, Client\_2 in Figure 5-6 missed the update of ⑧ for the reason of less-frequently polling the server. So the GeoBlindZone and GeoLagZone techniques can save much networking and computing capability that can then cater for other clients such as Users in GeoServeZone or Listeners in GeoListenZone for pressing interaction with the target resource, and the scalability of resource-accessing systems are improved when there are increasing various clients.

Accordingly, it is usually a good practice to intentionally and appropriately produce GeoBlindZone and/or GeoLagZone to avail efficiency and scalability of networking and computing. By pre-setting and dynamically-adjusting GeoPolicy conditions in AccessFeeds to partition Geo-Context zones that can on-the-fly discriminate pressing and non-pressing interaction tasks between various clients and the target resource.

However, the GeoBlindZone and GeoLagZone method may also introduce potential risk of malfunctions that unmatch the intention of an application. For example, given a scenario when a target resource suddenly changes its GeoServeZone to have current GeoBlindZone included, all these current GeoBlindZone clients (i.e. Sleepers) won't know they have been authorized with the access to this resource as they themselves have no opportunity to contact (or be contacted by) the server for getting informed in advance of the latest GeoPolicy conditions of this resource. So they will keep the status of acting as "Sleepers" but can never become "Users" of this resource. The time lag (latency) of GeoLagZone on receiving resource updates may also cause dissatisfaction sometimes. For example, given a scenario when the target resource suddenly changes its GeoServeZone to have current GeoLagZone (GeoObserveZone) included, a time lag usually occurs before the Observers can turn into the role of resource "Users" until the time reaches for next polling to the server.

So further technical mechanisms are expected for waking up the Sleepers in GeoBlindZone, and for the Observers in GeoLagZone to catch up the latest resource updates. Section 5.6.2.2 will address these issues.

#### **5.6.2.2 Mechanisms for Wake-up & Catch-up**



As the GeoBlindZone clients (i.e. Sleepers) themselves do not have any communication opportunity for contacting the server (resource), it is needed to resort to external approach to wake up the Sleepers, i.e. to inform the GeoBlindZone clients of latest resource update such as an updated AccessFeed.

As illustrated in Figure 5-5 in Section 5.6.1, a multi-level communication framework has been designed for delivering the updates of an AccessFeed, involving the AccessFeed level, Pull/Push model level, AccessFeed processor level, etc. Besides the Pull/Push approaches (as denoted as ⑤, ⑥ in Figure 5-5) internally embedded within the Pull/Push communication model, an AccessFeed may contain an “adhocRefreshInterval” element (as denoted as ④ in Figure 5-5) for updating this AccessFeed itself. On the AccessFeed processor level, there are also external approaches (as denoted as ②, ③ in Figure 5-5) for updating an AccessFeed. A certain time interval can be set or adjusted for the processor to automatically retrieve the latest updates to all AccessFeeds it currently deals with, or manually refreshing can be conducted any time by the users. In Figure 5-5, Client\_1 that got the update of ⑥ and Client\_2 that got the updates of ④ just benefit from these external approaches rather than the internal Pull/Push communication model. These external forces are supplementary technical mechanism to for the Sleepers in GeoBlindZone to get AccessFeed updates. Once the GeoPolicy conditions in a latest AccessFeed indicates this Sleeper client can now meet the Pull/Push communication conditions, this Sleeper has been waken up and become a non-Sleeper client (e.g. Observer, Listener or User) that can now update this AccessFeed by itself via the internal Pull/Push communication model. This is the mechanism to wake up deactivated (hibernated) Pull/Push communicate of those clients from GeoBlindZone.

For reducing the time lag of getting resource (e.g. AccessFeed) updates, some technical measures can be employed. One action is to reduce parameter of the <refreshInterval> within the <Pull> element of the AccessFeed. Another action is to bring current GeoLagZone (GeoObserveZone) clients, if they need more timely resource updates, into the GeoListenZone by dynamically adjusting the GeoPolicy condition for the <Push> element to include these current GeoObserveZone clients (i.e. Observers). Once these Observers get this newly-adjusted AccessFeed, then they can immediately become Listeners that can receive real-time AccessFeed updates onwards. These measures comprise the technical mechanism for catch-up with latest resource updates.

### 5.6.3 AccessFeed Update Service

In this GeoAccessFeed access-control framework, an AccessFeed functions as a client-side component. When the <Pull> and/or <Push> are present within an AccessFeed, it will communicate with a server-side component, named AccessFeed Update Service, for getting the updates of this AccessFeed itself. The section here designs the model of AccessFeed Update Service.

#### 5.6.3.1 Service Interface Model

AccessFeed Update Service (as shown in Figure 5-5) provides the functionality for getting up-to-date AccessFeeds from GeoChannel Service nodes (or the manager/provider/operator of resources), and for disseminating latest updates of AccessFeeds to clients. So several main interface methods should be provided by AccessFeed Update Service:

- notifyFeedUpdate

GeoChannel Service nodes publish latest AccessFeeds to the repository of an AccessFeed Update Service by using this interface method.

- getFeedUpdate

An AccessFeed client uses the Pull (request/respond) pattern to get latest updates of a specific AccessFeed by calling this getFeedUpdate interface of AccessFeed Update Service. Clients within a GeoObserveZone use the Pull pattern to request AccessFeed updates.

- subscribeFeedUpdate

An AccessFeed client uses the Push (subscribe/publish) pattern to subscribe to and then to be notified of the updates of a specific AccessFeed via this subscribeFeedUpdate interface of AccessFeed Update Service. Clients within GeoListenZone or GeoServeZone use the Push pattern to get AccessFeed updates.

#### 5.6.3.2 Information Model: Whole or Part Update

The AccessFeed information model (as codified in Appendix 3) also provides an <UpdateAccessFeed> element for updating an AccessFeed that has been loaded on a client side. This <UpdateAccessFeed> element format can function as the information model for the payload of response messages of the AccessFeed Update Service.

Two patterns applies to this update process, one is to replace an old version with a new version of the whole AccessFeed, and another is to incrementally modify the old version. Incremental update can contain any number of <Change>, <Create>, and <Delete> elements, which are processed in order.

This <UpdateAccessFeed> element, which is included in the full GeoAccessFeed schema model codified in Appendix 3, is also an extension to standard KML. Its syntax is shown as below:

Listing 5-5: The <UpdateAccessFeed> element in extended-KML for updating an AccessFeed using two patterns

**Syntax:** (only an outline is listed here, the details are described in its Schema in *Appendix 3*)

Pattern 1: whole replacement

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2"
      xmlns:geoww="http://www.geoww.net/geochannel">
  <geoww:UpdateAccessFeed feed_ID="AB435465768a" client_specific="false"
    newVersionID="08" src="http://.kml" timeSource="2012-12-17T09:30:47Z">
    <geoww:AccessFeed src="http://.kml" version="08" channel_ID="AB435465768a">
      <!-- here is the content of the new AccessFeed -->
    </geoww:AccessFeed>
  </geoww:UpdateAccessFeed>
</kml>
```

Pattern 2: incremental update

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2"
      xmlns:geoww="http://www.geoww.net/geochannel">
  <geoww:UpdateAccessFeed feed_ID="AB435465768a" client_specific="false"
    newVersionID="08" src="http://xx.kml" timeSource="2012-12-17T09:30:47Z">
    <geoww:targetVersionID>07</geoww:targetVersionID>
    <geoww:Create insideNodeID="node2" beforeNodeID="node23" afterNodeID="node22">

    </geoww:Create>
    <geoww:Change>

    </geoww:Change>
    <geoww:Delete>

    </geoww:Delete>
  </geoww:UpdateAccessFeed>
</kml>
```

Partly updating an AccessFeed can improve the communication efficiency, especially in the case that the original whole AccessFeed is a big message/file. And this incrementally-updating mechanism is especially useful, when necessary, for partly modifying the Geo-Off-On communication model parameters or GeoPolicy conditions so as to adjust the communication or access-organizing behaviour.

So by communicating a message that contains the information about a <UpdateAccessFeed> element, an existing AccessFeed can be updated in either pattern as described above.

#### **5.6.4 Geo-Off-On Communication Model for Updating AccessFeeds**

A smart AccessFeed-updating communication model with adaptability is expected, to achieve two objectives, i.e. timely updates-getting and optimal resource-using. The Geo-Off-On Communication Model designed in Chapter 4 just meets this demand and requirement. This section below specially elaborates the typical application of the Geo-Off-On communication model for delivering dynamic updates of AccessFeeds, by describing the detailed syntax with technical mechanism of its information model.

##### **5.6.4.1 Pull/Push Elements Information Model**

The Geo-Off-On as a hybrid communication model with two main communication patterns work for AccessFeed Update Service. The request/respond pattern based on HTTP-Get/Post communication can be used for getting latest AccessFeeds published by GeoChannel services (nodes), for getting subscription requests from GeoChannel clients, and for returning AccessFeed updates requested by GeoChannel clients. And the subscribe/publish pattern based on WebSockets or Server-Sent-Events can be used for notifying updates to AccessFeed clients. The two patterns can switch adaptively in a context-aware way.

A GeoChannel AccessFeed, when collaborating with a remote AccessFeed Update Service, functions as a client-side message endpoint which is responsible for dynamically detecting and communicating a client's GeoContext (such as the 2D/3D map browsing viewport or his/her physical location) to, and receiving returned (or pushed) updates from, this AccessFeed Update Service. In the Geo-Off-On communication model, the outbound messages report a client's GeoContext information to the server, and the inbound messages are usually about the updates to the <Accessibility> element, or the updates to other parts within an AccessFeed.

Listing 5-6 shows the detailed syntax, using the extended KML encoding schema as codified in Appendix 2, for the content of the <Pull> and <Push> elements of the Geo-Off-On communication model used within a GeoChannel AccessFeed, which functions as a client-side program component to work with a remote AccessFeed Update Service. The <Pull> and the <Push> elements may be both absent in the case that no AccessFeed Update Service is equipped/needed; or may be both present in the case that both pull and push communication patterns are needed; or either of them is present in the case that only pull or push pattern is needed.

Listing 5-6: The <Pull> and <Push> elements syntax for the Geo-off-on communication model

**Syntax:** (only an outline is listed here, the details are codified in its Schema in *Appendix 2*)

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2"
      xmlns:geoww="http://www.geoww.net/geochannel">
  <!-- there may be other standard KML data snippets here -->
  <geoww:GeoChannel channel_ID=" " <!-- the channel_ID attribute is an Globally Unique Identifier (GUID)-->
    <!-- the GeoChannel element may only contain
          either Record element or ChannelFeed element, or may contain both -->
    <name>...</name>
    <description>...</description>

    <geoww:Metadata>...</geoww: Metadata> <!--the GeoChannel metadata returned from
          a GeoChannel Discovery Service -->
    <geoww:AccessFeed src="http://..kml" version="2012-10-08" channel_ID="id123">
      <!-- the AccessControl element can be updated, with a version number. -->
      <name>...</name>
      <description>...</description>

      <GeoPolicy>...</GeoPolicy> <!-- a common space-time-aware condition for controlling the
          Pull/Push/Accessibility elements-->
      <adhocRefreshInterval>...</adhocRefreshInterval> <!-- float -->
      <geoww:Pull>

        <GeoPolicy>...</GeoPolicy> <!-- a space-time-aware condition for
          controlling Pull element -->
        <href>...</href> <!-- the URL for http-based pull (request/response) communication -->

        <refreshMode>...</refreshMode> <!-- refreshModeEnum: onChange, onInterval, or onExpire -->
        <refreshInterval>...</refreshInterval> <!-- float -->
        <geoww:viewRefreshMode>...</geoww:viewRefreshMode>
          <!-- viewRefreshModeEnum: never, onStop, onRequest, onPolicyToTrue -->
        <viewRefreshTime>...</viewRefreshTime> <!-- float -->
        <geoww:locationRefreshMode>...</geoww:locationRefreshMode>
        <geoww:remoteEvaluate>...</geoww:remoteEvaluate> <!-- boolean -->
        <viewBoundScale>...</viewBoundScale> <!-- float -->
        <viewFormat>...</viewFormat> <!-- string -->
        <httpQuery>...</httpQuery> <!-- string -->
        <geoww:clientContext> <!-- this element can contain some information about the client's context -->

        Feed=[channelID],[feedVersion];Location=[clientLocation]; Speed=[clientSpeed];Heading=[clientHeading];Time=[Time]
        </geoww:clientContext> <!-- the content inside this Context element is extensible -->
      </geoww:Pull>
    </geoww:AccessFeed>
  </geoww:GeoChannel>
</kml>
```

```

</geoww:Pull>

<geoww:Push>                                <!-- may be absent. If present, then connect to a pushing server -->
  <GeoPolicy>...</GeoPolicy>                <!-- a space-time-aware condition for
                                              controlling Push element -->
  <geoww:pushPattern>WebSockets</geoww:pushPattern> <!-- or
                                              ServerSentEvents-->
  <href>...</href>                          <!-- the URL for a client-side WebSocket or EventSource object to
                                              initiate connecting to a pushing server -->
  <refreshMode>...</refreshMode>
  <refreshInterval>...</refreshInterval>
  <geoww:viewRefreshMode>...</geoww:viewRefreshMode>
  <viewRefreshTime>...</viewRefreshTime>
  <locationRefreshMode>...</locationRefreshMode>
  <geoww:remoteEvaluate>...</geoww:remoteEvaluate> <!-- boolean -->
  <viewBoundScale>...</viewBoundScale>
  <viewFormat>...</viewFormat>
  <Query>...</Query>                        <!-- This Query element's content and syntax is similar to the
                                              "httpQuery" element within the "Pull" element -->
  <geoww:clientContext>...</geoww:clientContext>

</geoww:Push>

<geoww:Accessibility>... </geoww:Accessibility>

</geoww:AccessFeed>
</geoww:GeoChannel>
<!-- there may be other standard KML data snippets here -->
</kml>

```

The demand on the push pattern (based on WebSockets or Server-Sent-Events) rises from some working scenarios. If a GeoChannel's GeoContext (e.g. GeoServeZone) and/or its access-control policy change frequently or randomly, this GeoChannel's resources may become inaccessible to its existing clients, or become accessible to its potentially new clients that are interested in these resources. In this scenario, it is necessary to inform all these clients in a timely way about the updated <Accessibility> information, with which the client-side-running AccessFeed will disconnect the existing clients who become unentitled, or connect new clients who become entitled, for accessing certain GeoChannel resources. In addition to updating the <Accessibility> content, the server may also push to its clients about dynamic updates of GeoPolicies which can then enforce new controlling conditions on the different parts within an AccessFeed. Traditional pull pattern that uses regular polling requests to the server cannot satisfy this timely nature of demand on delivering real-time updates.

To some extent, in terms of syntax the content for the <Pull> or <Push> element within a GeoChannel AccessFeed is somewhat similar to a standard KML <Link> element (OGC 2008) (Google 2013a), as they are all derived from a standard KML <BasicLink> element type, just as shown in the class diagram in Figure 3-1. But the essential difference is that the AccessFeed

<Pull> and <Push> element have the <GeoPolicy> sub-element incorporated to cater for GeoChannel access-control functionality. They dynamically report a client's GeoContext to a remote AccessFeed Update Service and get the updates about the content of this <AccessFeed> element.

The following explains specific syntax content (sub-elements/parameters) within the <Pull> and <Push> elements.

Some sub-elements have variable parameters, for example, the custom query string (represented in the elements such as <viewFormat>, <httpQuery>/<Query>, <clientContext>, etc.) appended to the <href> URL. It is noted that, an earth/map browser (program) that supports the GeoChannel AccessFeed technology should be able to substitute the parameter variables, which are enclosed in the [ ] brackets, by using appropriate current values when constructing a query string.

#### **<href>**

A URL specifies the resource location. The <Pull> element uses its sub-element <href> to supply the URL address of the remote access-control service. The pull pattern works via HTTP-based request/response communication. Comparatively, the <Push> element uses its sub-element <href> to supply the URL address of server-side message endpoint which can push messages to the client using WebSockets or Server-Sent-Events technologies. In the push pattern, a client can firstly initiate the request to this server-side message endpoint to establish a WebSockets or Server-Sent-Events connection for subsequently receiving updates from the server.

#### **<geoww:pushPattern>**

This sub-element is specific to the <Push> element, for indicating the pushing communication methods that may choose two alternative values: WebSockets or ServerSentEvents, corresponding to two types of pushing communication techniques, i.e. WebSockets for bi-directional, or Server-Sent-Events for one-way communication.

#### **< geoww:GeoPolicy>**

The GeoPolicy condition specified for controlling the behaviour of the client-server communication between the AccessFeed and the Access-Control (AccessFeed-update) service. The GeoPolicy language model (codified in Appendix 1) can be used for expressing this

GeoPolicy condition.

Only if this GeoPolicy is evaluated true can the <Pull> or <Push> element function. That is, the <Pull> element can contact the server, or the <Push> connection (client-server) can exist for sending and/or receiving messages. Dynamic changes of the GeoPolicy will affect the communication accordingly. This mechanism can be used to achieve the adaptive Geo-Off-On communication model.

#### **<geoww:refreshMode>**

Specifies a time-based mode for contacting server, i.e. sending messages to the server, with a prerequisite that the <GeoPolicy> is true. This <refreshMode> value can be one of the following:

*onChange* - contact server when the AccessFeed is first loaded and whenever the Pull/Push parameters change. This is the default value.

*onInterval* - contact server every *n* seconds (specified in <refreshInterval>).

*onIntervalTimeSpan* - contact server every *n* seconds (specified in <refreshInterval>) during the period specified in <TimeSpan>. If the <TimeSpan> is not present, then this *onIntervalTimeSpan* value is equivalent to the *onInterval* value without bounded time period.

*onExpire* - contact server when the expiration time is reached. If a fetched file (i.e. a KML file containing the AccessFeed content) has a NetworkLinkControl, the <expires> time takes precedence over expiration times specified in HTTP headers. If no <expires> time is specified, the HTTP *max-age* header is used (if present). If *max-age* is not present, the *Expires* HTTP header is used (if present). The Section RFC261b of the *Hypertext Transfer Protocol - HTTP 1.1* (Fielding et al. 1999a) presents the details on HTTP header fields.

#### **<refreshInterval>**

Indicates to contact server every *n* seconds.

#### **<TimeSpan>**

This element is present when the <geoww:refreshMode> has the *onIntervalTimeSpan* value.

The syntax of the KML <TimeSpan> is employed for this element, specifying an extent in time



bounded by begin and end temporal values. At least one of the child elements kml:begin and kml:end shall be encoded. If <begin> or <end> is missing, then that begin or end of the period is unbounded. The <begin> and <end> value shall be encoding according to the kml:dateTimeType field type.

#### **<geoww:viewRefreshMode>**

Specifies how the <Pull>/<Push> contacts the server when the "camera" (i.e. the client's map viewport) changes.

Can be one of the following:

*never* (default) - Ignore changes in the view. Also ignore <viewFormat> parameters, if any.

*onStop* - Contact server *n* seconds after movement stops, where *n* is specified in <viewRefreshTime>.

*onRequest* - Contact server only when the client explicitly requests it. (For example, in Google Earth, the client right-clicks and selects Refresh in the Context menu.)

*onPolicyToTrue* - Contact server when the <GeoPolicy> becomes true, that is, at the moment when the GeoPolicy-evaluated result switches from false value to true value.

#### **<viewRefreshTime>**

After camera movement stops, specifies the number of seconds to wait before refreshing the view. (See <viewRefreshMode> and **onStop** above.)

#### **<geoww:locationRefreshMode>**

Specifies how the <Pull>/<Push> contacts the server when the series of "geoLocation" information (as listed in Table 3-1) of this client changes.

Can be one of the following:

*never* (default) - Ignore changes in the series of "geoLocation" information (as listed in Table 3-1). Also ignore the [clientLocation], [clientSpeed] and [clientHeading] parameters in the <clientContext> element, if any.

*onChange* – Contact server when the client's location changes, i.e. when a change event of the

“geoLocation” has been detected. The new location information encoded in the <clientContext> will be sent to the server.

#### **<geoww:remoteEvaluate>**

A Boolean value, true or false, indicates whether or not a remote policy evaluation facility, i.e. Access Policy Evaluation Service will be used. If the value is true, then the information encoded in the following parameters (such as <viewBoundScale>, <viewBoundScale>, <viewFormat>, <httpQuery> / <Query>, <clientContext>, etc.) may be used by the remote policy evaluation service; if the value is false, then these information parameters may be ignored, or even it is not necessary to send these parameters to the server.

This parameter <geoww:remoteEvaluate> can be used for switching access-policy evaluation function between local client and remote server.

#### **<viewBoundScale>**

Scales the BBOX parameters before sending them to the server. A value less than 1 specifies to use less than the full view (screen). A value greater than 1 specifies to fetch an area that extends beyond the edges of the current view.

#### **<viewFormat>**

Specifies the format of the query string that is appended to the <Pull>/<Push>’s <href> parameter before contacting server.

If the <geoww:viewRefreshMode> is **onStop** and no <viewFormat> tag is present, the following information is automatically appended to the query string:

BBOX=[bboxWest],[bboxSouth],[bboxEast],[bboxNorth]

If an empty <viewFormat> tag is present, no information is appended to the query string.

Custom set of viewing parameters can be added to the query string. If supplying a format string, it is used instead of the BBOX information. If the BBOX information (parameters) is also needed, these parameters can be added along with the custom parameters.

Any of the following parameters can be used in the format string (and an earth/map browser should substitute the appropriate current value at the time it creates the query string):

[**lookatLon**], [**lookatLat**] - longitude and latitude of the point that KML <LookAt> is viewing.  
[**lookatRange**], [**lookatTilt**], [**lookatHeading**] - values used by the KML <LookAt> element's <range>, <tilt>, and <heading>.

[**lookatTerrainLon**], [**lookatTerrainLat**], [**lookatTerrainAlt**] - point on the terrain in degrees/meters that <LookAt> is viewing.

[**cameraLon**], [**cameraLat**], [**cameraAlt**] - degrees/meters of the eyepoint for the camera.

[**horizFov**], [**vertFov**] - horizontal, vertical field of view for the camera.

[**horizPixels**], [**vertPixels**] - size in pixels of the 3D viewer.

[**terrainEnabled**] - indicates whether the 3D viewer is showing terrain.

<**httpQuery**>, <**geoww:Query**>

The <**httpQuery**> is used in the <Pull> element, and <**geoww:Query**> in the <Push> element, for appending additional information to the query string, based on the parameters specified. (An earth/map browser such as Google Earth substitutes the appropriate current value at the time it creates the query string.) The following parameters are supported:

[clientVersion]

[kmlVersion]

[clientName]

[language]

<**geoww:clientContext**>

To send some other information about the client's context, in addition to his/her viewport. The context information can include the parameters indicated below, and they are extensible. An earth/map browser substitutes the appropriate current value for these parameters at the time it creates the query string.

```
<geoww:clientContext>    <!-- this element can contain some information about the client's context -->
    Feed=[channel_ID],[feed_Version];Location=[clientLocation];Speed=[clientSpeed];Heading=[clientHeading];Time=[Time]
</geoww:clientContext>    <!-- the content inside this Context element is extensible -->
```

The Feed=[**channel\_ID**],[**feed\_Version**] parameters are usually necessary to be communicated, to report the identity of this specific AccessFeed to the server (i.e. a general GeoChannel AccessFeed Update Service), which is then able to identify and respond to this specific AccessFeed with this GeoChannel-specific information or services.

### 5.6.4.2 An Example of Updating an AccessFeed

With having introduced (in last section) the content parameters of the <Pull> and <Push> elements, the following exemplifies an AccessFeed with detailed content for the <Pull> element. A <Push> element also has this similar content model. Listing 5-7 gives a brief example with information instance about the <Pull> element within a GeoChannel AccessFeed.

Listing 5-7: GeoChannel AccessFeed <Pull> element: syntax and content exemplification

```
<geoww:AccessFeed src="http://..kml" version="2012-10-08" channel_ID="id123">
  <!-- the AccessControl element can be updated, with a version number. -->
  <name>...</name>
  <description>...</description>
  <GeoPolicy>
    <TimePrimitive>...</TimePrimitive>
    <Region>...</Region>
  </GeoPolicy>
  <geoww:Pull>

    <GeoPolicy>
      <TimePrimitive> <!-- the KML TimePrimitive element, e.g. TimeSpan or TimeStamp-->
        <begin>2012-09-22</begin> <!-- for time-aware access to this GeoChannel-->
        <end>2013-07-18</end>
      </TimePrimitive>
      <Region> <!-- the KML Region element for space-aware controlling the Pull elements-->
        <LatLonAltBox>
          <north>37.430419921875</north>
          <south>37.41943359375</south>
          <east>-122.080078125</east>
          <west>-122.091064453125</west>
        </LatLonAltBox>
        <Lod>
          <minLodPixels>128</minLodPixels>
        </Lod>
      </Region>
    </GeoPolicy>
    <href>http://www.example.com/geochannels/AccessControl</href>
    <geoww:refreshMode>onIntervalTimeSpan</geoww:refreshMode>
    <!-- refreshModeEnum: onChange, onInterval, or onExpire -->
    <refreshInterval>4</refreshInterval> <!-- float -->
    <TimeSpan>
      <begin>2012-08-17T09:30:47Z</begin>
      <end>2012-08-17T12:30:47Z</end>
    </TimeSpan>
    <geoww:viewRefreshMode>onStop</geoww:viewRefreshMode>
    <!-- viewRefreshModeEnum: never, onStop, onRequest, onPolicyToTrue -->
    <viewRefreshTime>4</viewRefreshTime> <!-- float -->
    <viewBoundScale>1</viewBoundScale> <!-- float -->

    <viewFormat>BBOX=\ <!-- string -->
```

```

[bbboxWest],[bbboxSouth],[bbboxEast],[bbboxNorth];CAMERA=\
[lookatLon],[lookatLat],[lookatRange],[lookatTilt],[lookatHeading];VIEW=\
[horizFov],[vertFov],[horizPixels],[vertPixels],[terrainEnabled]
</viewFormat>

<httpQuery>                                <!-- string -->
    clientVersion=[clientVersion]&amp;kmlVersion=[kmlVersion]&amp;
    clientName=[clientName]&amp;language=[language]
</httpQuery>

<geoww:clientContext> <!-- this element can contain some information about the client's context-->
    Feed=[channel_ID],[feed_Version];Location=[clientLocation];Speed=[clientSpeed];
Heading=[clientHeading];Time=[Time]
</geoww:clientContext>                    <!-- the content inside this Context element is extensible -->
</geoww:Pull>

<geoww:Push>...</geoww:Push> <!-- may be absent. If present, then connect to a pushing server -->

<geoww:Accessibility>... </geoww:Accessibility>

</geoww:AccessFeed>

```

## 5.7 Application Scenarios

AccessFeed is a core technique component in the GeoAccessFeed technology system which involves a set of techniques (such as GeoPolicy language, Geo-Off-On communication model, AccessFeed and Geoww-URI scheme, etc.). The AccessFeed technique can be used in a wide range of areas. The following just exemplifies some application scenarios.

### 1) Weaving things into the GeoChannel Web with dynamic relations

By analogy to the general Web made up of mostly pre-set and stable hyperlink relations between digital resources, the GeoChannel Web features dynamic and condition-based relations mapped for real-world things. AccessFeeds just work for dynamically correlating real-world things based on their latest context attributes and then weaving things into the GeoChannel Web with these dynamic relations.

The two use cases in Chapter 9 demonstrate the examples. In Use Case 1, AccessFeeds correlate individual moving vehicles with road segments and then map these dynamic relations to GeoChannel networks on the GeoChannel Web Engine. In Use Case 2, AccessFeeds are also

employed for controlling and mapping dynamic relation between panorama GeoChannels, audio commentary resources, offsite online viewers, beacon-identified exhibits and onsite visitors.

## **2) Controlling access to resources of interest**

AccessFeeds employ GeoPolicy language for encoding access-controlling conditions in a non-programming way. So the AccessFeed technique can function in various application domains where conditional-access to resources of interest is expected.

For example, Use Case 2 in Chapter 9 will demonstrate controlling the access to audio commentaries based on clients' contexts (e.g. offsite viewers' GeoChannel Geoww-URIs, view heading/pitch, or onsite visitors' physical positions, etc.); and controlling the access to inner-GeoCluster, inter-GeoCluster or cross-Real-Virtual-World connecting and chatting services based on clients' contexts (e.g. the proximity and context topology of offsite viewers and onsite visitors).

AccessFeeds empowered by GeoPolicy language with rich context-condition-expressing capability, dynamically-self-updating and continuously-evaluating mechanisms, and both-front-back-ends applicability, can overcome some related access-control solutions, for example, the Geo-tag, Geo-filter or (Geo)XACML techniques as reviewed in Section 2.5.

## **3) GeoClustering things for interaction**

The GeoChannel Web's mission "bridging things in context" is supported by AccessFeeds, which can dynamically gather and group clients with similar or related context attributes into clusters for discovering and connecting them for interaction.

The use cases in Chapter 9 will demonstrate this kind of functionality. AccessFeeds in Use Case 1 can group context-matched vehicles for each road segment (i.e. GeoChannel) into a GeoCluster, and can GeoBridge adjacent GeoClusters on the Context Topology Networks into a GeoClusters union, to achieve inter-vehicle interactions for communicating traffic information.

And Use Case 2 exemplifies a more complex and interesting application scenario, where AccessFeeds can conduct virtual-world, physical-world and cross-virtual-physical-world clustering for interaction. For example, firstly online (e.g. off-site) viewers can gather in separate panorama GeoChannels for interaction. Secondly, viewers in a GeoChannel may speak to or be heard by viewers in other panorama GeoChannels that are adjacent on the GeoContext Topology Network and within the valid range of voice spreading. This mechanism can form

complex combination of adjacent panorama GeoChannels which simulate real-world scenarios of crowd chatting in continuous proximity. Thirdly, an on-site visitor in the museum can join several nearby physical-world GeoChannels with exhibits and then interact with other visitors in these related GeoChannels. Fourthly and most interestingly, off-site viewers and on-site visitors can also be bridged across-virtual-physical-world based on their current context attributes. The Figures (from Figure 9-25 to Figure 9-28) in Chapter 9 illustrate these complex scenarios on GeoClustering and GeoBridging for interaction.

#### **4) AccessFeed technique for Augmented Reality**

AccessFeed can play a promising role in Augmented Reality (AR) which overlays digital content into real-world environments.

A key need and technical challenge for mixing digital items with real-world scenes exists in the accurate mapping or correspondence between the overlaid items and their real-world scene context (e.g. position, heading, pitch, etc.). A digital item should be put in the right location, viewed in appropriate perspective and loaded in an on-demand manner in a real-world scene. This task calls for condition-controlled augmentation of digital items into real-world environments.

Apart from descriptive ARML data (e.g. ARElements), the current ARML specification (Augmented Reality Markup Language) (OGC 2014) employs script (e.g. JavaScript) bindings to allow dynamic access of objects in AR scenes. This data/scripts mixing format usually makes ARML content complex and illegible. This kind of data-script hybrid format does not occur in other content-encoding formats such as GML, KML, etc., which only contain encoded data. Now the AccessFeed technique can become an alternative solution to ARML for dynamic and condition-controlled access or augmentation of data content objects in AR scenes.

Just as indicated in Section 5.4.1, the principle and mechanism of AccessFeed is basically encoding-neutral. ARML can be extended for representing AccessFeeds in their domain-specific applications. Similar to the extended-KML schema (as codified in Appendix 3) for embedding AccessFeeds content into KML, AccessFeeds can also be encoded into ARML by using an extended-ARML encoding model, to work with the host language of ARML for implementing context-aware and condition-controlled dynamic access or augmentation of AR content.

## 5.8 Summary

GeoAccessFeed is a term for grouping a set of techniques involving GeoPolicy language, Geo-Off-On communication model, AccessFeed and Geoww-URI scheme, etc. The GeoAccessFeed technology is centred on AccessFeed. This Chapter has designed the information model, working process model, and updating mechanism for AccessFeeds.

AccessFeed is a mechanism for dynamically correlating things in contexts, i.e. for operating dynamic relations between things based on their context conditions in a GeoChannel. From a resource-client perspective, AccessFeed technique can be used for controlling access to a resource and for clustering clients based on context attributes of the resource and/or its clients.

AccessFeed technique makes use of GeoPolicy language and Geo-Off-On communication model techniques, for defining context conditions, controlling correlation, and throttling communication between things. So AccessFeed can dynamically bridge resource with client, cluster clients and federate resources in a context-aware and automatic way. AccessFeed has self-updating mechanism, so it can cater for dynamic environment where things' contexts may change randomly.

This capability of AccessFeeds can be generally used for control-accessing pervasive services (e.g. LBS), for gathering users in similar context for participative interaction (e.g. social networking), for implementing large-scale real-time maps with massive dynamics, and for an extensive range of applications or services that feature dynamic context conditions.

The AccessFeed technique just meets the demand of the GeoChannel Web for mapping and applying dynamic and random relations between things in real-world context. In a global view, AccessFeeds manipulate GeoChannel networks for the GeoChannel Web, as they dynamically control the connection relations between things weaved in the GeoChannel Web.



## **6 Geoww-URI Scheme**

### **6.1 Introduction**

The general Web uses the “http://” (Fielding et al. 1999) URI scheme (RFC3986 2005) for identifying and locating web resources. This general URI scheme does not have a native mechanism for encoding or associating resources’ GeoContext information into their http URIs. The “geo:” URI (IETF 2010) scheme can add geography coordinates into information entities such as GeoSMS (OGC 2011c), vCards (IETF 2011), etc., nevertheless it is too simple to incorporate complex context attributes other than position information. Although GeoRSS (GeoRSS 2013) can encode geospatial content (e.g. geographical points, lines, and polygons of interest and related feature descriptions.) as part of a Web feed, it is still a onefold solution limited to geospatial features, lacking adequate mechanisms for supporting contextualization with other types of attributes. Actually, Chapter 2 has reviewed these existing solutions for contextualization and has identified their deficiency for the GeoChannel Web proposed in this research.

This chapter proposes a new URI scheme for the GeoChannel Web and for the general Web. Firstly Section 6.2.1 identifies the demands and expected capabilities of this new URI technique. Then the Geoww-URI scheme “geoww://” is designed in Section 6.2.2 with elaboration on its encoding structure model and information components. Section 6.3 and Section 6.4 exemplify the novel functionality of Geoww-URI for identifying, selecting, and targeting resources and clients. Section 6.5 illuminates the usage and significance of the Geoww-URI technique for the GeoChannel Web and for the general Web.

### **6.2 A New URI Scheme for the Web**

#### **6.2.1 Demands and expected Capability of a new URI Scheme**

Now in the context of accessing contextualized resources (e.g. information, applications, services, activities, interaction, etc.), some new and peculiar requirements and speciality occur as enumerated below, which motivate to design a new URI scheme to meet new demands and to

highlight particularity.

- uniquely identify a contextualized resource with emphasizing conditional access

The http-URI scheme is just a syntax format for encoding the expressions of locators of web resources, without involving expressing resources' context and access conditions. In the scenario of the GeoChannel architecture, a same general web resource (e.g. a dataset or web application) identified by an http-URI, may be Geo-Contextualized by some AccessFeeds with various access conditions respectively, so this general resource, which are now referenced within different Accessors of AccessFeeds, is becoming diversified resources from the views of users. So current http-URI scheme is not capable of identifying these geo-contextualized resources.

If a competent URI encoding scheme is available, any resources with corresponding GeoContexts can be identified, bookmarked and circulated, just like the current usage pattern that we can bookmark http-URI-identified general web resources, hyperlink them, or recommend them to others via emails/messages, etc. Furthermore, not only acting as an identifier, a URI encoded based on this desired competent URI scheme can also function as a condition expression/controller for controlling the delivering/running of a Geo-Contextualized resource.

- uniquely identify GeoClustered users

Social interaction is an integral functionality of the GeoChannel technology, as opposed to current GeoWeb without this kind of inbuilt facility. GeoAccessFeed technique can supply the capability of GeoClustering users based on the GeoContext conditions prescribed in AccessFeeds. Now a capable method is expected for identifying each GeoCluster of users grouped on a geo-contextualized resource, and for identifying each member within a specific GeoCluster, to facilitate social networking and work cooperation via these identities.

- combine, select and filter resources for access in an on-demand and flexible way

An AccessFeed may Geo-Contextualize and reference multiple resources, and sometimes only

one or some of them may be on demand or be of interest to a user or task. The <AccessorSelector> element in an AccessFeed is a solution to meeting this need, by using <AccessorSelector> element(s) prior to the full content of an <AccessFeed> element for choosing some part of resources for access. Now a more lightweight and flexible solution is desired for combining or/and selecting any parts of resources from any AccessFeeds without the need of re-presenting the content of these AccessFeeds. This desired solution is expected to be able to re-organize (e.g. group, customize and tailor) AccessFeeds-referenced resources without re-designing new AccessFeeds.

- automatically launch GeoChannel-supporting software/programs

By analogy with the traditional “http://” URL which, when present, can usually launch a web browser for loading/running a web resource. The similar paradigm is expected to trigger/launch a user-agent program (such as a certain web-based application) that can support dealing with GeoChannel resources, when a URI for a GeoChannel resource occurs. There will be this kind of user agent programs that are aware of this new URI scheme desired.

- a native URI scheme technology desired for the GeoWeb and for the GeoChannel Web

Physical objects in the real world are addressed by their geographic coordinates which are the fundamental elements for the GeoWeb. However, the current GeoWeb does not have a geospatially-specialized addressing technique for conditionally-accessible resources, other than the general http-URI scheme that originated from and works for the general Web. The http-URI scheme cannot cater for the GeoChannel Web. It is desired that the GeoWeb can have its native URI solution.

### **6.2.2 Geoww-URI: “geoww://” URI Scheme**

Here a new URI scheme, “geoww://”, is proposed for identifying, locating, selecting, combining and access-controlling Geo-Contextualized resources for the GeoWeb.

This “geoww://” URI scheme name is “geoww”, which is specially used for marking the expression of identifiers for geo-contextualized resources with space/time and

condition-controlled characteristics. The “**geoww://**” aims to become the GeoWeb-specific (native) URI scheme, as opposed to the “http://” scheme for the general Web. Several component parts are involved in this “**geoww://**” URI scheme. Figure 6-1 briefly illustrates the main structure of the “**geoww://**” URIs.

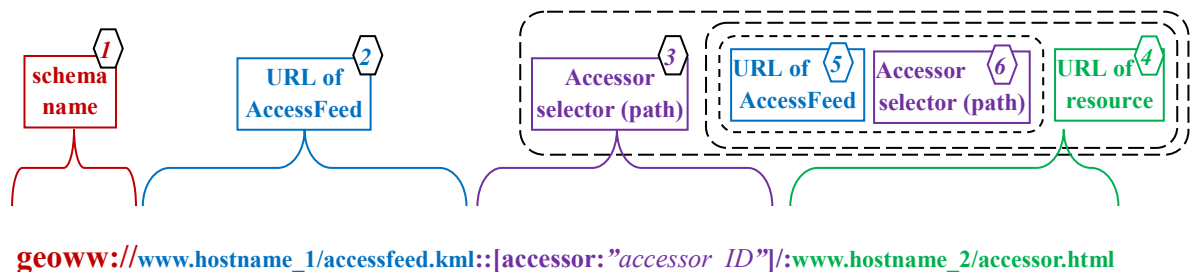


Figure 6-1: The Geoww-URI schema model “geoww://”: syntax & semantics for component parts

Some components are mandatory parts such as ① and ② in Figure 6-1; and some are optional parts which are outlined using dashed-line boundaries such as ③, ④, ⑤ and ⑥. Actually the Part ② and Part ⑤ are the same kind of components, and it is also the case for Part ③ and Part ⑥. The initial part marks the scheme name “geoww://”, which is followed by URL of an AccessFeed which points to the web location of this AccessFeed. An AccessFeed is a file/message of extended-KML encoding, so the URL of AccessFeed usually locates a file with kml extension name, or points to a web service that can return a kml-extension-name file. Part ③ is an Accessor selector for choosing a specific resource from this AccessFeed which usually includes multiple Accessors with references to different resources. The Accessor selected is identified by its “accessor\_ID” attribute value. Part ④ is just the URL of the resource referenced by the <href> element within the selected Accessor. Mostly a resource is usually a general (standard) kml-encoded dataset, or an html web program/application for accessing a target resource. In a more general sense, for chained AccessFeeds, the <href> within an <Accessor> element can also point to another AccessFeed. In this case the Part ⑤ with Part ⑥ will be applicable, which are used for chained AccessFeeds. This kind of chain may occur for multiple times, which implies there may be more cascaded components of Part ⑤ with Part ⑥ in a “geoww://” URI. Figure 6-1 just shows the component of Part ⑤ with Part ⑥ for one time for brevity of this illustration.

### 6.3 “geoww://” URIs for Identifying & Selecting Resources

This components structure pattern described above can represent various “geoww://” URIs with scalable complexity. The following exemplifies the syntax and semantics for expressing different “geoww://” URIs, which are based on three pieces of simplified AccessFeeds to facilitate the demonstration.

Listing 6-1: Examples of AccessFeeds used for exemplifying the “geoww://” URI scheme

*Listing 6-1-a: AccessFeed\_a*

```
<AccessFeed src="http:// www.example-a.com/accessfeeds/accessfeed02.kml">
  <Accessibility>
    <Accessor accessor_ID ="abc1234">
      <GeoPolicy>...</GeoPolicy>
      <href>http://www.example-b.com/data/dataset6.kml</href>
    </Accessor>
    <Accessor accessor_ID ="abc1235">
      <GeoPolicy>...</GeoPolicy>
      <href>http://www.example-b.com/accessors/accessor8.html</href>
    </Accessor>
    <Accessor accessor_ID ="abc1236">
      <GeoPolicy>...</GeoPolicy>
      <href>http://www.example-c.com/accessfeeds/accessfeed06.kml</href>
    </Accessor>
    <Accessor accessor_ID ="abc1237">
      <GeoPolicy>...</GeoPolicy>
      <Geometry>...</Geometry>
    </Accessor>
  </Accessibility>
</AccessFeed>
```

*Listing 6-1-b: AccessFeed\_b*

```
<AccessFeed src="www.example-c.com/accessfeeds/accessfeed06.kml">
  <Accessibility>
    <Accessor accessor_ID ="dfgc8656">
      <GeoPolicy>...</GeoPolicy>
      <href>http://www.example-d.com/data/dataset26.kml</href>
    </Accessor>
    <Accessor accessor_ID ="dfgc8658">
      <GeoPolicy>...</GeoPolicy>
    </Accessor>
  </Accessibility>
</AccessFeed>
```

```

        <href>http://www.example-d.com/accessors/accessor28.html</href>
    </Accessor>
    <Accessor accessor_ID ="dfgc8659">
        <GeoPolicy>...</GeoPolicy>
        <href>http://www.example-f.com/accessfeeds/accessfeed32.kml</href>
    </Accessor>
</Accessibility>
</AccessFeed>

```

Listing 6-1-c: AccessFeed\_c

```

<AccessFeed src="www.example-f.com/accessfeeds/accessfeed32.kml">
    <Accessibility>
        <Accessor accessor_ID ="fgha9216">
            <GeoPolicy>...</GeoPolicy>
            <href>http://www.example-g.com/accessors/accessor118.html</href>
        </Accessor>
    </Accessibility>
</AccessFeed>

```

Some examples for demonstrating the syntax and semantics of the new URI scheme “geoww://” are given below. Each example may demonstrate the combination of different component parts depicted in Figure 6-1.

**Example 1):** This “geoww://” URI can identify the AccessFeed\_a that is exemplified in Listing 6-1-a, and select all its Accessors with their referenced resources.

Components: ① + ②





**geoww://**www.example-a.com/accessfeeds/accessfeed02.kml

**Example 2):** This “geoww://” URI identifies the AccessFeed\_a and selects the inline resource (which may be some KML geometry feature) encoded within the Accessor that has the value “abc1237” for its accessor\_ID attribute.

Components: ① + ② + ③





**geoww://**www.example-a.com/accessfeeds/accessfeed02.kml::[accessor:abc1237]

**Example 3):** This “geoww://” URI identifies the AccessFeed\_a and selects the Accessor “abc1234” that reference a KML dataset on the URL of www.example-b.com/data/dataset6.kml.

Components:  +  +  + 







**geoww://**www.example-a.com/accessfeeds/accessfeed02.kml::[accessor:abc1234]/:www.example-b.com/data/dataset6.kml

**Example 4):** This “geoww://” URI identifies the AccessFeed\_a and selects the Accessor “abc1235” that references an Accessor program on the URL of www.example-b.com/accessors/accessor8.html for accessing its corresponding resource.

Components:  +  +  + 









**geoww://**www.example-a.com/accessfeeds/accessfeed02.kml::[accessor:abc1235]/:www.example-b.com/accessors/accessor8.html

**Example 5):** This “geoww://” URI demonstrates two chained AccessFeeds. Firstly this URI identifies the AccessFeed\_a and selects the Accessor “abc1236”, which points to another Accessor, i.e. Accessor\_b (depicted in Listing 6-1-b) that then selects its Accessor “dfgc8658” pointing to an Accessor program on www.example-d.com/accessors/accessor28.html for accessing target resource.

Components:  +  +  +   + 

**geoww://**www.example-a.com/accessfeeds/accessfeed02.kml::[accessor:abc1236]/:www.example-c.com/accessfeeds/accessfeed06.kml::[accessor:dfgc8658]/:www.example-d.com/accessors/accessor28.html

**Example 6):** This “geoww://” URI demonstrates three chained AccessFeeds. Firstly it identifies the AccessFeed\_a and selects the Accessor “abc1236”, which points to Accessor\_b that then selects its Accessor “dfgc8659”, which then points to Accessor\_c (as depicted in Listing 6-1-c) whose Accessor “fgha9216” points to an Accessor program on www.example-g.com/accessors/accessor118.html.

Components:  +  +  +  +  +  +  + 

**geoww://**www.example-a.com/accessfeeds/accessfeed02.kml::[accessor:abc1236]/:www.example-c.com/accessfeeds/accessfeed06.kml::[accessor:dfgc8659]/:www.example-g.com/accessors/accessor118.html

e-c.com/accessfeeds/accessfeed02.kml::[accessor:dfgc8659]/:www.example-f.com/accessfeeds/  
accessfeed32.kml::[accessor:fgha9216]/:www.example-g.com/accessors/accessor118.html

*Apart from* the functionality of identifying/choosing one resource from those referenced by AccessFeeds, this “geoww://” URI scheme also provides syntax for selecting multiple resources. The operators { } and | can be used for this kind of purpose. The example 7), 8) and 9) just demonstrate this kind of usage.

**Example 7):** This “geoww://” URI demonstrates selecting two resources (i.e. two Accessor programs) from different AccessFeeds. One is from AccessFeed\_a with its Accessor “abc1235”, and another is from AccessFeed\_b with its Accessor “dfgc8658”.

**geoww://**{www.example-a.com/accessfeeds/accessfeed02.kml::[accessor:abc1235]/:www.example-b.com/accessors/accessor8.html}|{www.example-c.com/accessfeeds/accessfeed06.kml::[accessor:dfgc8658]/:www.example-d.com/accessors/accessor28.html}

**Example 8):** This “geoww://” URI demonstrates selecting two resources (i.e. two Accessor programs) from the same AccessFeed, i.e. AccessFeed\_a.

**geoww://**www.example-a.com/accessfeeds/accessfeed02.kml::{|[accessor:abc1234]/:www.example-b.com/data/dataset6.kml}|{|[accessor:abc1235]/:www.example-b.com/accessors/accessor8.html}

**Example 9):** This “geoww://” URI demonstrates a more complex grouping/combining of multiple resources. Totally three resources are selected. One is from AccessFeed\_a, and the other two are from the chained AccessFeed\_b with selected Accessors “dfgc8656” and “dfgc8658”.

**geoww://**www.example-a.com/accessfeeds/accessfeed02.kml::{|[accessor:abc1234]/:www.example-b.com/accessfeeds/accessfeed09.html}|{|[accessor:abc1236]/:www.example-c.com/accessfeeds/accessfeed06.kml::{|[accessor:dfgc8656]/:www.example-d.com/data/dataset26.kml}|{|[accessor:dfgc8658]/:www.example-d.com/accessors/accessor28.html}|}

## 6.4 “user@geoww://” URIs for Identifying & Targeting Users

A resource identified by its “geoww://” URI can function as a medium for the interaction between the users that are accessing this resource. The GeoAccessFeed technology achieves this GeoClustering capability. Technically, a “geoww://” URI identifying a Geo-Contextualized resource can also be used for identifying the cluster of its current users. Two syntax patterns are



designed for identifying users.

1) [users]@geoww://

The prefix “[users]@” can represent the whole cluster of users gathering on the resource identified by the “geoww//” URI in a global web-wide scope. Collective operations can be conducted on a whole GeoCluster. For example, a member of any GeoClusters can send messages to all the members within a specific GeoCluster by using this “[users]@geoww://” for identifying the target collective receivers.

For example, the following expression:

[users]@geoww://www.example-a.com/accessfeeds/accessfeed02.kml::[accessor:abc1235]/:www.example-b.com/accessors/accessor8.html

can identify all the users as a whole on the resource that is identified in Example 4 given in Section 6.3.

2) [user: “local\_identity”]@geoww://

This URI pattern is for identifying a specific user by indicating its specific “local\_identity”, which is usually a temporary token assigned to a user by a GeoCluster when this user just joins this cluster, that is, when this user begins to match a resource’s GeoContext condition and is about to access this resource. This user’s identity is “local” as it is just unique within the scope of this GeoCluster. By being suffixed by this resource’s URI (i.e. geoww//...), the expression of [user: “local\_identity”]@geoww://... can uniquely identify a user in a global web-wide scope.

For example, the following expression:

[user:ACB2463758689]@geoww://www.example-a.com/accessfeeds/accessfeed02.kml::[accessor:abc1235]/:www.example-b.com/accessors/accessor8.html

can identify the user using its local identity “ACB2463758689” on the resource that is identified in Example 4 given in Section 6.3.

This “[user: ‘local\_identity’]@geoww://...” URI can solve the demand on identification of users for cross-resources/GeoChannels interaction. For example, a user GeoClustered in a resource can receive from and send/reply messages to another user GeoClustered on another resource. This identification solution can cater for the desired capability of Geo-diffuse social

networking on the GeoChannel Web.

## 6.5 The Usage of the “geoww://” URIs: GeoChannel Web Addresses

The functional essence of a “geoww://” URI is it can identify, reference and filter AccessFeed(s), which in its/their turn can Geo-Contextualize, identify, reference and filter web resources including other AccessFeed(s). That is, a “geoww://” URI can Geo-Contextualize and reference target resources. So the “geoww://” URIs can on-demand bring Geo-Contextualized resources to the GeoWeb, for example, for mashing up geospatial dynamics, services and Geo-Communities (interaction) into web maps.

The following just exemplifies some typical usage of the “geoww://” URIs, which can actually be applied to more scenarios by analogy to general web URIs.

- Separation of authoring and utilizing AccessFeeds

The “geoww://” URI scheme’ capability of referencing, combining and tailoring AccessFeeds provides simplicity and flexibility for authoring and utilizing AccessFeeds and Geo-Contextualized resources.

Firstly, it is flexible to incrementally design/author new AccessFeeds by exploiting existing AccessFeeds. This mechanism can maintain an inheritance relationship between base AccessFeeds and derived AccessFeeds, by analogy to the inheritance mechanism in Object-Oriented design pattern. The chaining-AccessFeed approach can use both Http-URIs and the Geoww-URIs for chained AccessFeeds. Actually the Geoww-URIs can provide more powerful flexibility, as a Geoww-URI can combine and tailor multiple AccessFeeds/resources, while an http-URI can only reference an existing AccessFeed with all contained resources without tailoring capability.

Secondly it is lightweight and convenient to achieve the separation of authoring and utilizing AccessFeeds. Once an AccessFeed has been authored, it (with its Geo-Contextualized resources) can be referenced anywhere, without the need to incorporate (in-line) the actual content of this AccessFeed into its user/caller environment. For example, the “geoww://” URIs can be called from within a general KML data set/file, as the extended usage of KML NetworkLink and Placemark balloon that is exemplified below.

- KML NetworkLink

Currently a standard KML `<NetworkLink>` element is for referencing/returning standard KML data content. Now we can extend its capability for delivering GeoChannel resource. For example, a “geoww://” URI can be assigned to the `<href>` sub-element within a `<NetworkLink>` element. In this way, a city’s dynamics, services or interaction in different areas or at different zoom levels can be delivered to map users.

- KML Placemark balloon

Currently a KML Placemark description balloon can display HTML format information that may contain some http-URIs hyperlinking to targets such as KML data files. Now we can extend the target types of these hyperlinks to support GeoChannel resources identified by the “geoww://” URIs. For example:

Listing 6-2: An example of supporting Geoww-URIs in KML Placemark balloons

```
<Placemark>
  <name>Bus Stop (WO35), Nottingham, UK</name>
  <description>
    <![CDATA[
      <a
        geowwURI="geoww://www.example-a.com/accessfeeds/accessfeed02.kml::[accessor:abc1235]/:www.ex
        ample-b.com/accessors/accessor8.html"
        type="application/vnd.GeoChannel-Accessor.html">
          Running buses that can be called at this bus stop: Wollaton, The Crown (Stop WO35).
        </a>
      ]]>
    </description>
    <Point>
      <coordinates> -1.190891, 52.955982</coordinates>
    </Point>
  </Placemark>
```

In this example, when a map user clicks this bus stop icon, and then clicks the hyperlink in the opened placemark balloon window, a Geoww-URI-aware client agent program (e.g. the GeoChannel Explorer web application that will be described in Chapter 8.) will retrieve the related AccessFeed file, and retrieve and run the Accessor component of the GeoChannel resource identified by this geoww:// URI to display real-time positions of the buses that can be called at this bus stop.

This usage of correlating “geoww://” URIs with KML Placemarks can supply a solution for discovering and running GeoChannel resources. This is one of the GeoChannel discovery

mechanisms.

- Identify, discover, bookmark, recommend and run/attend geospatial communities, services, activities, etc.

As opposed to the http-URI scheme for general web resources, the Geoww-URI scheme is adept at dealing with Geo-Contextualized resources, such as geospatial communities, location-based services/activities, etc. The Geoww-URIs can be bookmarked, circulated (e.g. recommended) and executed. These conveniences can facilitate a GeoChannel resources Web.

The discovery of GeoChannel resources of interest is just to get their Geoww-URIs. Chapter 7 will design the techniques for GeoChannel discovery that generally involve physical-world and virtual-world approaches, both of which return the Geoww-URIs of target resources.

- GeoWeb addresses: a native URI scheme technology for the GeoWeb and LBS

By analogy with the http-URI scheme for identifying and locating resources for the general web, now the Geoww-URI scheme can function as a native URI model for the GeoChannel Web, for dealing with Geo-Contextualized resources and user. In this sense, Geoww-URIs can be considered as the GeoWeb addresses, as the Geoww-URI scheme supplies a native addressing solution for conditionally-accessible GeoWeb resources, LBS (Location-Based Services) and their users.

This is just the fundamental significance of having designed this Geoww-URI scheme.

## 6.6 Summary

This research has invented a new URI model, termed Geoww-URI scheme, for identifying contextualized resources, clients, clusters or their relations.

Beyond traditional Http-URI scheme, the Geoww-URI scheme introduces context and access policy into URIs, making resources (or clients) function and used in specific contexts and constrained conditions. So, by analogy to Http-URIs for the general web, the Geoww-URI model aims to become a native URI technique for the GeoChannel Web which weaves things in real-world contexts, and can also act a native URI technique for the GeoWeb that has space/time attributes for objects. So by analogy to using Http addresses on the general Web, now we can

use Geoww-URI addresses on the GeoChannel Web and on the GeoWeb.

The Geoww-URI scheme is built on and works with the AccessFeed technique. In a Geoww-URI expression, there is (are) usually specific AccessFeed(s) referenced, which organize the access to contextualized target resources. So without the need of encoding GeoPolicy conditions inside the URI expression, the Geoww-URI scheme supplies a light-weight mechanism for a Geoww-URI to point to one or multiple contextualized resources (or users).

Some typical functionality and usages apply to this Geoww-URI technique:

- Identify, bookmark and recommend contextualized resources, users, clusters or their relations.

It is noted that, a same general Web resource identified a Http URL such as a data set file will become (i.e. be considered as) different GeoChannel resources if it is accessed from different AccessFeeds. There are different Geoww-URIs expressions differentiating these resources.

- Organize (e.g. select, filter, group, federate) resources in a on-demand and flexible way

The Geoww-URI schema model also provides applicable syntax and semantics for selecting and combining access of resources extracted from a same or multiple AccessFeeds. This supplies a light-weight way for re-organizing resources of interest, instead of having to create new AccessFeeds for the re-organization purpose.

- Facilitate discovery and access-control of pervasive services

GeoChannel Discovery technology (designed Chapter 7) returns Geoww-URIs via virtual-world or physical-world approaches. Geoww-URI is particularly useful for identifying, discovering and controlling access to pervasive services with their context conditions.

## **7 GeoChannel Discovery**

### **7.1 Introduction**

GeoChannels are a class of resources which organize and deliver various GeoChannel services, for example, event/information services, functional services, social/interaction services, control/manipulation services (which may be included in the category of functional service), etc. So identifying a GeoChannel implies corresponding services organized and brokered by this GeoChannel can be discovered. GeoChannels are web-distributed and may be crowd-sourced resources. The discovery of GeoChannels of interest is an essential step for subsequently utilizing and participating in the GeoChannel services.

This chapter designs a GeoChannel Discovery system, which involves developing a series of normative models for information, interfaces and workflow processes. Section 7.2 presents an overview of the components framework containing all functional entities and information flows involved in the GeoChannel resource discovery process. GeoChannel Catalogue Service centres on a unified GeoChannel catalogue metadata model (codified in Section 7.4.1) for interoperability of federated search, and features equivalent implementation of OGC CSW (Catalogue Service for the Web) interfaces for simplifying and democratizing GeoChannel discovery tasks. Section 7.5 incorporates the PubSubHubBub protocol into the GeoChannel architecture system with modelling GeoChannel metadata feeds format for real-time harvesting GeoChannel catalogue metadata. The OpenSearch technique is employed and further extended in Section 7.6 to cater for GeoContext specialization of searching GeoChannel resources. This chapter further extend standard KML language for encoding and presenting search results of GeoChannel Catalogue Service.

### **7.2 The Framework & Approaches**

GeoChannel discovery is to acquire the identifiers of GeoChannel resources of interest for subsequent access to the identified resources. The novel Geoww-URI schema model (designed in Chapter 6) is used for encoding the identifiers of GeoChannels (resources) with their access

GeoContext. So GeoChannel discovery is just to get the Geoww-URIs of GeoChannel resources.

### 7.2.1 The Component & Process Framework

GeoChannel Discovery involves a set of functional components and work flows. Figure 7-1 illustrates the components framework for GeoChannel Discovery and subsequent access to resources.

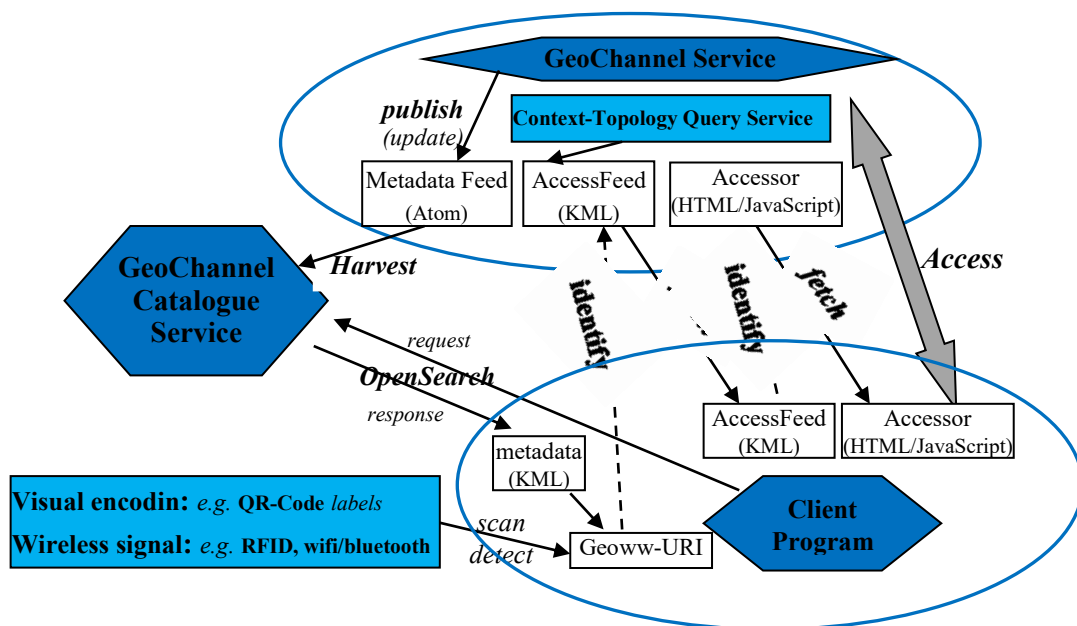


Figure 7-1: The components framework for GeoChannel discovery and subsequent access to resources

The left part in Figure 7-1 shows the two categories of discovery approaches, that is, the virtual-world approach to search GeoChannel Catalogue Service and the physical-world approach to detect some mediums (e.g. QR code labels, wireless signals, etc.) from our physical environment. The two approaches are for eventually getting the Geoww-URI for a GeoChannel resource. This Geoww-URI identifies this resource's AccessFeed, which will then be fetched to the client side for organizing access to the target resource. Once meeting the access condition (i.e. GeoPolicy), this target resource's Accessor (program) will be fetched to the client side for accessing this GeoChannel resource.

The following sections will elaborate the technical details about mechanisms, information models and process models for GeoChannel discovery.

## **7.2.2 Physical-world Approach & Virtual-world Approach**

Two types of approaches for GeoChannel discovery system are explored in this research. They can be generally called physical-world approach and virtual-world approach.

### **7.2.2.1 Getting Geoww-URIs from the Physical Environment**

The physical world is populated by various media for conveying information. For example, billboards for advertisement, symbol markers for notices, etc. It is also applicable for indicating or communicating Geoww-URIs in our ambient environment.

Acquiring Geoww-URIs of GeoChannel resources from our physical environment is considered as a physical-world approach for GeoChannel discovery. A Geoww-URI as a piece of information can be represented in various medium forms, such as text, graph, sound, radio signals, etc. The Geoww-URIs in different information media can be conveyed in different ways, for example, shared, recommended or posted via emails, labels, etc. Among these media, the QR-code, RFID and Eddystone beacon techniques supply convenient approaches for conveying Geoww-URIs information in visual graph/symbol or wireless signal ways, especially for discovering pervasive services populating our environment.

Since GeoChannels deliver contextualized resources with accessibility controlled by contextual attribute conditions, discovering GeoChannels in this physical-world way is just directly allied to accessing GeoChannels, as the processes of discovering and accessing GeoChannels are both based on the context attributes of physical-world clients and/or resources. Remarkably the physical-world approach for discovering and accessing GeoChannels can especially enable and facilitate a smart ambient environment populated by pervasive services.

This physical-world approach can enable us to smartly and pervasively interact with GeoChannels relevant to or associated with our immediate context to get pervasive services via the Web. These GeoChannels' Geoww-URIs are context-aware discoverable in our surrounding environment, and their accessibility is also context-aware controlled.

Section 7.3 will elaborate and exemplify the physical-world approach, involving QR-code, RFID (Radio Frequency Identification) and BLE (Bluetooth Low Energy) beacon signal for encoding and conveying Geoww-URIs for GeoChannel Discovery.



### 7.2.2.2 Search & Query Services getting Geoww-URIs

The virtual-world approach refers to web services of search and query that can return Geoww-URIs of GeoChannels. This research mainly designs GeoChannel Catalogue Service and Context-topology Query Service.

GeoChannel Catalogue Service is a centralized query service that can be analogized with current web search service, as illustrated in Figure 7-2. GeoChannel Discovery (Catalogue) Service acts as a broker between clients and GeoChannel resources. However, differing from the discovery service of web pages that are usually indexed and searched by keywords, GeoChannels are working units for organizing and delivering contextual resources and services, which need to be described by metadata information that can characterize the context attributes of GeoChannel resources. So the GeoChannel Catalogue service designed in this research is to support the functionality of both publishing/harvesting and discovery of GeoChannel metadata.

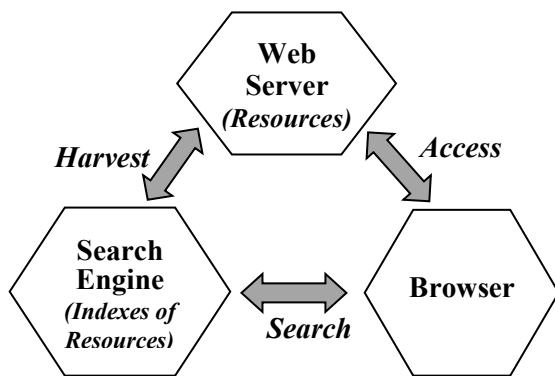


Figure 7-2 (a): How the Web works

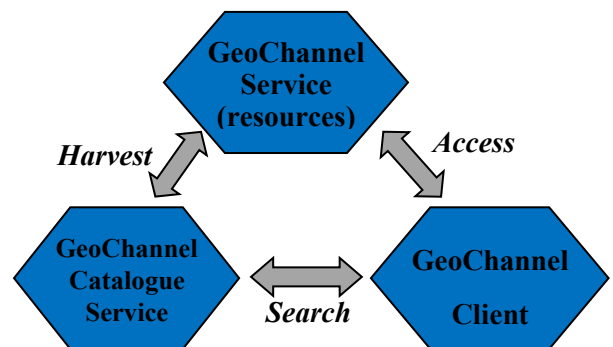


Figure 7-2 (b): How GeoChannel resources are discovered via Catalogue Service

Figure 7-2: The analogy of discovery mechanisms for the Web and for GeoChannel resources

This chapter is to design a series of normative models for information, interfaces and workflow processes for GeoChannel Catalogue Service, which mainly involves the following aspects:

- metadata information model for characterizing GeoChannel resources, especially their GeoContexts; and service interface model for standard-compliant interaction.
- mechanism and information model for harvesting timely (real-time) update of metadata, to cater for the dynamic nature of resources' GeoContext that may change randomly and frequently.
- information model for search service (capability/functionality) description, and for syntax expression of query requests and responses. This design will target technical simplicity to cater

for mass market users.

The subsequent sections will elaborate these technical details outlined above. As for the virtual-world approach, Section 7.4, 7.5 and 7.6 will design a series of information and workflow models for GeoChannel Catalogue Service, and Section 7.7 will describe Context-topology Query Service.

### 7.3 Physical-world Approach

This section (in its subsections) elaborates the physical-world approach that gets Geoww-URIs from the Physical Environment, which employs visual symbol or wireless signal media for conveying Geoww-URIs. The techniques involves encoding geoww-URIs in QR-code, RFID or signal frame of Bluetooth beacon, then scanning QR code labels, detecting RFID tags, or scanning Eddystone beacon signals to get Geoww-URIs of GeoChannel resources.

#### 7.3.1 QR-code for Geoww-URI

A Geoww-URI, expressed in text form using the encoding format designed in Chapter 6 can also be represented by using graphic form such as a barcode to become optical machine-readable. QR Code (Quick Response Code) (Wave 1994, 2011) is just competent for this task.



Figure 7-3: An example of encoding a Geoww-URI using QR Code for identifying and discovering a GeoChannel resource

[geoww://www.example-a.com/accessfeeds/accessfeed02.kml::\[accessor:abc1236\]/:www.example-c.com/accessfeeds/accessfeed06.kml::\[accessor:dfgc8658\]/:www.example-d.com/accessors/accessor28.html](geoww://www.example-a.com/accessfeeds/accessfeed02.kml::[accessor:abc1236]/:www.example-c.com/accessfeeds/accessfeed06.kml::[accessor:dfgc8658]/:www.example-d.com/accessors/accessor28.html)

For example, a Geoww-URI can be graphically represented using a QR Code as shown in Figure 7-3. A QR Code can be easily captured and read by camera-equipped devices such as smartphones. This approach can bring significance for discovering and circulating GeoChannel resources. As GeoChannels can refer to any GeoContextualized resources such as a geo-community (e.g. residence area, building, park), a vehicle (e.g. bus, train), etc., the Geoww-URIs in QR code labels posted on the objects/places in the physical world can facilitate discovering and accessing these specific GeoChannel resources. For example, people with smartphones, which have Geoww-URI-aware software (such as the GeoChannel Explorer described in Chapter 8) installed, can discover and may then access GeoChannel resources anywhere when they meet QR code labels about Geoww-URIs

### **7.3.2 RFID for Geoww-URI**

As a physical-world approach, RFID (Radio-Frequency Identification) (Landt 2001) can provide another way for discovering GeoChannel resources. The information about Geoww-URIs can be stored in the RFID media, which can be detected and read out by RFID reader devices such as smartphones that are equipped with RFID capability. Unlike the function process of a QR Code, the RFID tag does not necessarily need to be within line of sight of the reader, and may be embedded in the target object. By issuing detecting signals, smartphones can detect the RFID tags with Geoww-URIs stored which identify GeoChannel resources such as pervasive services in the physical world.

The media of RFID with Geoww-URI encoded can especially work for smart things/objects in our physical world. For example, in the application scenario of Smart Home (i.e. home automation systems), attached or embedded RFID tags with Geoww-URIs can automate services delivered or operated by GeoChannels in a context-aware and –controlled way. This can enable a smart ambient environment with the GeoChannel technology.

### **7.3.3 Physical Web with Geoww-URI**

#### **7.3.3.1 The Concept and Technique of Physical Web**

The Physical Web (Google 2013b) is an effort initiated by Google to extend the superpower of the web - the URL - to everyday physical objects. It is an open approach to enable quick and seamless interactions with physical objects and locations, as it provides a lightweight discovery and interaction layer to the WoT (Web of Things) (Guinard & Trifa 2009) (Guinard et al. 2011) or IoT (Internet of Things) (ITU 2005).

The Physical Web extends the web we know into the physical world around us. This involves creating an open ecosystem where smart devices can broadcast URLs into the area around them. At its base, the Physical Web is a discovery service: a smart object broadcasts relevant URLs that any nearby device can receive. This simple capability can unlock exciting new ways to interact with the Web, as the URL is the fundamental building block of the web.

Currently the practice of the Physical Web is encoding, broadcasting and discovering http-URIs of Web resources by employing BLE (Bluetooth Low Energy) beacons. Eddystone (Google 2014a) is an open beacon format from Google. As a protocol specification, Eddystone defines a BLE message format for proximity beacon messages. It describes several different frame types that may be used individually or in combinations to create beacons that can be used for various applications. Though similar to the iBeacon (Apple 2016) profile released by Apple in 2013 which is a proprietary protocol that mainly works for broadcasting IDs to nearby portable electronic devices, Eddystone as an open protocol format can be implemented without restriction, and can support a more variety of applications.



Figure 7-4: The Eddystone beacon protocol for a Physical Web

Currently the Eddystone protocol (Google 2016) defines the following frame types of BLE signal packets:

- Eddystone-UID for broadcasting beacons IDs. This 16-byte ID includes namespace and instance parts, which may be used for grouping a particular set of devices (e.g. beacons) and for identifying individual devices in the group respectively. A beacon ID may be useful in mapping a device to a record in external storage which points to a target resource such as web page or an application service. In some sense, it is similar to the iBeacon technique. An app installed on a smart device can use the identifier to trigger desired action.

- Eddystone-URL for broadcasting URL addresses. This is the peculiar capability distinct from iBeacon. Actually Eddystone-URL incorporates the ideas from the UriBeacon (Google 2014b) format from which it evolved. The http-URL could be a regular web page providing relevant information—e.g., a beacon next to a movie poster could broadcast a link to a YouTube trailer. It also could be a dynamic web application, with certain parameters embedded in the http URL—e.g., [http://my-restaurant.com/check-in?restaurant\\_id=6523](http://my-restaurant.com/check-in?restaurant_id=6523). The Eddystone-URL frame forms the backbone of the Physical Web, an effort to enable frictionless discovery of web content relating to one's surroundings. A Physical Web-enabled browser (or application) can detect Eddystone-URL packets and can then take corresponding actions based on the URLs. For example, a URL can be used by any client with access to the Internet. In some sense, Eddystone-URL is the BLE format for the Physical Web.

- Eddystone-TLM for beacon telemetry. This type of packets can broadcast telemetry information about the beacon itself such as battery voltage, device temperature, and counts of broadcast packets. This type of frames can work for fleet management such as maintaining beacons.

- Eddystone-EID for security. This frame broadcasts an encrypted ephemeral identifier that changes periodically at a rate determined during the initial registration with a web service. The broadcast ephemeral ID can be resolved remotely by the service with which it was registered, but to other observers appears to be changing randomly. This frame type is intended for use in security and privacy-enhanced devices. EID protects clients against two kinds of attacks: 1) Hijacking/Piggybacking - attack where a third party application is using the infrastructure to deliver content to the users, e.g. one retailer using beacon network of the other to deliver their own promotions. 2) Spoofing - attack where a third party creates a copy of the device used in the infrastructure to place it in different location, e.g. copy of the beacon originally deployed in a store is placed in a bus.

### **7.3.3.2 Conveying and Discovering Geoww-URIs via Bluetooth Beacons**

The Physical Web helps users discover Http-URLs relevant to their surroundings via beacons. This technical principle can also apply to the GeoChannel Web, which uses Geoww-URIs for identifying and accessing resources. This research employs and extends the Eddystone beacon technique for conveying and discovering Geoww-URIs. This is another so-called physical-world approach for GeoChannel Discovery, in addition to the QR-code and RFID means described in Section 7.3.1 and Section 7.3.2.

The Eddystone-URL frame can broadcast a Geoww-URL using a compressed encoding format in order to fit more within the limited advertisement packet. Any object or place can broadcast Geoww-URLs. Clients can discover GeoChannels with “geoww://” URIs from ambient things.

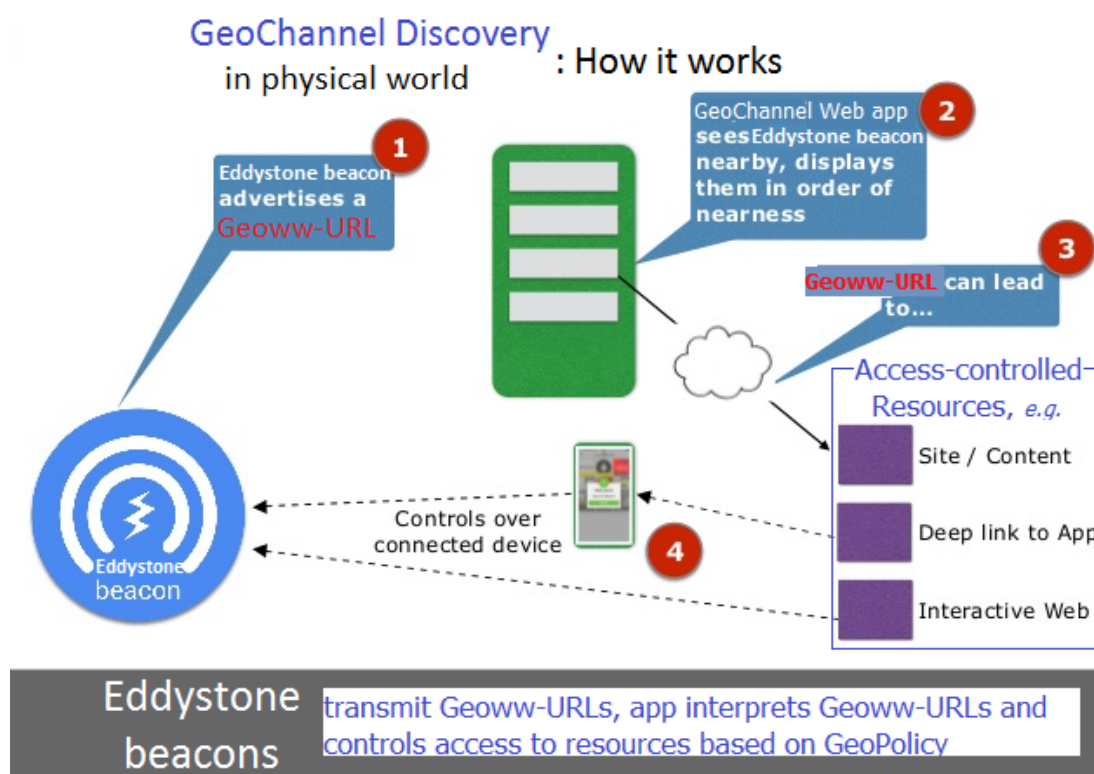


Figure 7-5: Eddystone beacons facilitate discovering GeoChannels in the physical world

There are mainly two notable aspects or differences when it comes to using the Eddystone BLE technique for discovering GeoChannel (resources):

1) Geoww-URI, which is designed in Chapter 6.

As opposed to Http-URIs that identify general web resources, Geoww-URIs can identify contextual resources. So it is more applicable and reasonable for mapping Geo-URIs to real-world things which usually have corresponding context attributes. The Eddystone-URL frames can encode and broadcast Geo-URIs identifying real-world things such as pervasive services, e.g. LBS (location based services) in the ambient environment surrounding clients.

2) GeoAccessFeed, which is designed in chapter 5.

The Physical-Web technique only supplies a discovery mechanism for Http-URIs, but does not

involve controlling access to target resources identified by these Http-URIs. Fortunately, the GeoChannel Web provides the GeoAccessFeed mechanism for organizing access to resources in a context-aware way. This technology can well satisfy achieving a smart environment where conditionally discovering and then conditionally accessing resources can work seamlessly. For example, clients can be aware of nearby resources via Eddystone-URI signals that broadcasting Geoww-URIs, and their access to these resources are filtered and controlled by AccessFeeds which only allow on-site access. This workflow can filter out other irrelevant resources/services to the ambient environment around the clients.

So the GeoChannel Web technology stack designed in this research working in conjunction with the Physical Web based on Eddystone beacon technique can achieve a smart physical environment, where anything can indicate and offer information, utility or services, and can control their access in a context-aware and automatic way. This technical conjunction can open endless possibilities for various innovative applications and services.

## **7.4 GeoChannel Catalogue Service**

This section designs GeoChannel Catalogue Service, which is a virtual-world approach for GeoChannel Discovery.

Catalogue Services are the systems that handle publishing and discovery of metadata entries. They are the key technological facilities for locating, managing and maintaining distributed resources (i.e. geospatial data, applications and services) (OGC 2007a).

The GeoChannel Catalogue Service technique is compliant with OGC Catalogue Service standard (OGC 2007b) that specifies the interfaces, bindings, and a framework for defining application profiles required to publish and access catalogues of metadata. This research designs the standard-compliant GeoChannel Catalogue Service system with extension and enhancement to cater for specific requirements in the context of this GeoChannel architecture. Normative Information model and interface model are defined to facilitate harvesting (publishing) and searching GeoChannel metadata.

### *Information Model:*

- Metadata model for describing GeoChannel resources for their discovery

Section 7.4.1 designs the GeoChannel catalogue metadata content model, which is codified in Appendix 6.

- Various message models (information structure and encoding formats/rules) for interaction with GeoChannel Catalogue Services.

Section 7.5.2 designs GeoChannel metadata feeds model for real-time harvesting GeoChannel catalogue metadata; Section 7.6 extends OpenSearch service description document format and request format, and models an extended-KML-encoded schema for responses of GeoChannel search operations.

#### *Interface Model:*

Section 7.4.2 outlines the equivalent implementation of interfaces corresponding to OGC CSW interface specification (OGC 2007b, 2007a).

### **7.4.1 The Metadata Model of GeoChannel Catalogues**

Metadata as generalised properties in catalogues represent resource characteristics that can be queried and returned through catalogue services for resource evaluation and, in many cases, invocation or retrieval of the referenced resource (OGC 2007b).

A normative and unified GeoChannel metadata framework can facilitate the interoperability between systems conforming to a consistent metadata profile. This can contribute some benefits. Client applications can discover GeoChannels from different GeoChannel catalogue service providers, which can also harvest and share GeoChannel metadata from each other in a distributed environment to logically form a federated system of GeoChannel catalogue services.

Current geospatial metadata standards basically intend to codify specifications for a set of metadata elements for describing geospatial data (or data service) resources for discovery purposes. There are currently some geospatial metadata specifications or related standards, for example:

Dublin Core Metadata codified by the Dublin Core Metadata Initiative (DCMI 2012);

ISO 19115:2003, Geographic information – Metadata (ISO 2003);

ISO 19115/Cor.1:2006, Geographic information – Metadata, Technical Corrigendum 1 (ISO 2006);

ISO 19119:2005, Geographic information – Services (ISO 2005);

ISO 19119:2005/Amd 1:2008, Extensions of the service metadata model (ISO 2008);



ISO/TS 19139:2007, Geographic information - Metadata - Implementation specification (ISO 2007);

OGC 07-006, OGC CSW, OGC™ Catalogue Services Specification, version 2.0.2 (Corrigendum Release 2) (OGC 2007b);

OpenGIS Catalogue Services Specification 2.0.2 - ISO Metadata Application Profile, Version 1.0.0, OGC 07-045. (OGC 2007c).

And some regional specifications/directives have significant influence currently, for example:

INSPIRE Metadata Implementing Rules--Technical Guidelines based on EN ISO 19115 and EN ISO 19119 (INSPIRE 2010);

INSPIRE Discovery Service (INSPIRE 2011);

UK Gemini 2.1: Specification for discovery metadata for geospatial data resources (AGI 2010).

GeoChannels, in a general sense, are also a kind of geospatial resource, but differ from traditional geospatial data (or data services supplying data) resources. GeoChannels are not limited to providing data snapshot services, but feature the capability of organizing, brokering and bridging real-time dynamics, location-based services and GeoClustered interaction and activities, all of which may frequently and randomly change on their context of space and time. So there is a need to extend and customize current geospatial metadata specifications to cater for describing GeoChannels for discovery and subsequent access purpose.

So this research designs a GeoChannel catalogue metadata information content model, which is presented in Appendix 1 of this thesis document. A set of metadata elements (fields) have been codified for characterizing GeoChannel resources. Here only some typical information fields are interpreted to highlight their function.

- geoChannelIdentifier

Every GeoChannel needs a unique identifier. Usually a GUID (Globally Unique Identifier) is used. This is meaningful for discovering and federating GeoChannel resources. For example, a search engine (such as GeoChannel catalogue service) can use Web crawling/spidering software to traverse the Web for collecting GeoChannel-related documents such as GeoChannel metadata feeds (designed in Section 7.5.2), GeoAccessFeeds (designed in Chapter 5) or other geo-data files (e.g. KML) with GeoChannel objects encoded in, to build and update a GeoChannel catalogue or AccessFeed repository. Auto-discovery or mutual-reference mechanisms have been designed for these file types that have GeoChannel information incorporated.

- resourceType

A GeoChannel as an organizational unit can involve multiple types of resources, for example, sensor dynamics, location-based services, social media interaction, remote control, etc.

- spatialFeature

The geometric representation or depiction for the spatial property and characteristic of a GeoChannel resource. The spatial feature may change over time. The spatialFeature as well as other space-related attributes such as spatialExtent and verticalExtent are usually the key metadata fields for space-based filter to discover GeoChannel resources.

- spatialExtent, verticalExtent

The spatialExtent represents the spatial coverage of a GeoChannel service. A detailed geometry, e.g. a polygon or multi-polygon, can be employed to outline the spatial area. BoundingBox is another simple method for this purpose. The verticalExtent is the vertical dimension of spatial extent if it is necessary to outline the vertical extent of this GeoChannel service's function domain.

Dynamic changes usually happen on the shape or size of spatial extent. Giving a moving vehicle as a GeoChannel resource for example, its space coverage for a shuttle bus may be a fixed area along a routine route, but for a private car it may roam within an area of a city district, a whole city or a country in different period.

The spatial coverage attribute usually acts as a parameter of access-organize conditions for dynamically correlating a GeoChannel resource with interested clients. Chapter 5 has designed the access-organize technology.

- TemporalExtent

Temporal range for the usage of this GeoChannel service, which will be available during this time period.

- geowwURI

This is a mandatory metadata field containing the Geoww-URI to identify the URI for this GeoChannel resource, so the geowwURI identifies the target GeoChannel of the discovery task.

The complete metadata model is codified in Appendix 6.

## 7.4.2 Catalogue Service Interface Model

Interface model can provide an external view about a system's functionality and the manner for interaction (access) from other components such as client applications. An interface is a named set of operations that characterize the behaviour of an entity (ISO 2005). As a part of OGC Catalogue Service (OGC 2007b) that defines a general interface model, Catalogue Service for the Web (CSW) focuses on HTTP Protocol binding. The general interface model categorizes several classes of interfaces, e.g. OGC\_Service class, Discovery class and Manage class, while the CSW specifies these interfaces into specific operations with corresponding HTTP request encoding modes, as mapped in the left and middle parts of Table 7-1.

Table 7-1: Interface model mapping between OGC CSW and GeoChannel Catalogue Service

<b>Interface Class</b>	<b>CSW(T) ISO Operation</b>	<b>Request encoding (HTTP)</b>	<b>Implementation (equivalent action)</b>
OGC_Service Interface	GetCapabilities	KVP (GET)	get (retrieve) an OpenSearchDescription document, which describes service metadata, query syntax and URL templates.  OpenSearch search request
CSW_Discovery Interface	DescribeRecord	KVP (GET)	
	GetDomain	KVP (GET)	
	GetRecords	KVP (GET)	
CSWT_Manage Interface	GetRecordById	KVP (GET)	
	Harvest	KVP (GET)	harvest metadata via PubSubHubBub protocol
	Transaction	XML (POST+XML)	

As the GeoChannel architecture aims to democratize the utilization of GeoChannel-organized resources, the GeoChannel Catalogue Service system tries to simplify the complexity of highly specialized systems to cater for the neogeography (Turner 2006, Haklay et al. 2008) paradigm, by introducing and extending some mass-market technology (such as OpenSearch). The GeoChannel Catalogue Service provides equivalent implementation to OGC CWS models of information and interface. As shown in the right part of Table 7-1, the OpenSearch (that will be designed in Section 7.6) description document can describe the service metadata and query syntax with parameterized templates for constructing OpenSearch requests to search GeoChannel catalogue repository. And a real-time metadata harvesting system (in Section 7.4) based on PubSubHubBub protocol can be used for publishing, collecting and updating GeoChannel catalogue metadata. The detailed information and interface models will be

illustrated in the following sections.

## **7.5 Real-time Harvesting of GeoChannel Metadata**

Harvesting GeoChannel metadata is a task and functionality for GeoChannel Catalogue Service. The dynamic nature of GeoChannel resources, which may change their GeoContext or other characteristics randomly and frequently, makes it necessary to timely acquire and update GeoChannel catalogue metadata that reflects the changing features of GeoChannels. An up-to-date GeoChannel catalogue can facilitate the search service with accurate results. For example, the associated geometry (e.g. a circle) characterizing a moving vehicle as a GeoChannel is changing dynamically, which makes the need to timely update the metadata record maintained on the repository of the catalogue service.

This section designs a real-time GeoChannel metadata harvesting system, which is based on the PubSubHubBub (Fitzpatrick et al. 2010) protocol, with new extension on the information model and component roles model for disseminating up-to-date GeoChannel metadata.

### **7.5.1 PubSubHubBub: a Real-time PubSub Protocol**

Low-latency communication is always a main demand for the GeoChannel architecture to deal with comprehensive dynamics and pervasive services. GeoChannel Services can employ WebSocket based protocol for server-client communication to deliver real-time resources to end users which are GeoClustered clients that can interact with each other via bi-directional brokering by GeoChannel nodes. Now the task of harvesting GeoChannel metadata involves the interaction between GeoChannel nodes and catalogue services nodes, which is a type of server-server communication. So a competent working model is expected to cater for this task.

PubSubHubBub (Google 2013a) is an open, server-to-server and webhook-based (Leggetter 2012) pubsub (publish/subscribe) protocol. This protocol extends the Atom (and RSS) protocols for communicating near-instant notifications of information that may be encoded in Web feeds delivered between servers.

PubSubHubbub enables pushed Web feed (e.g. Atom, RSS) update notifications, making subscribers be directly notified when there's an update to their interested information, rather than periodically pulling a feed server at some time interval. The diagram in Figure 7-6 illustrates the main workflows:

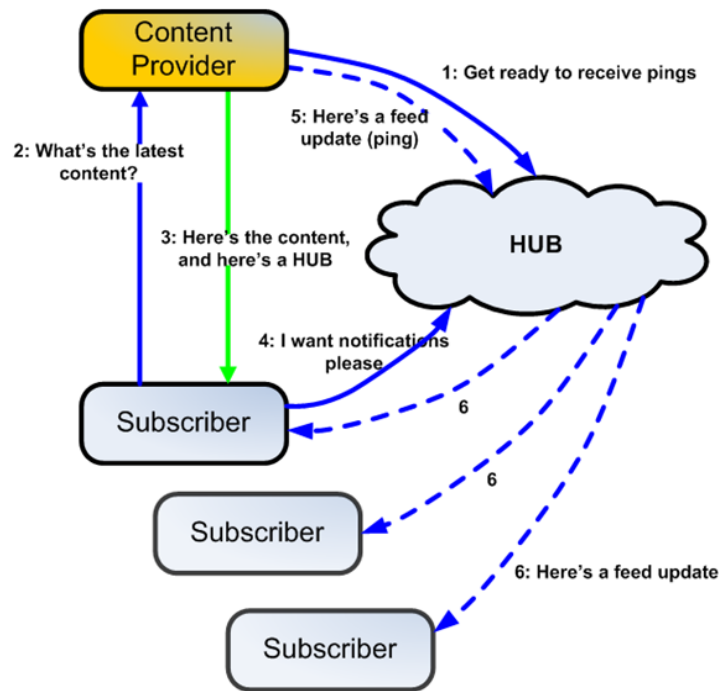


Figure 7-6: The roles and workflow in the PubSubHubBub protocol  
(Smith 2009)

In the PubSubHubbub protocol, all parties (i.e. publishers, subscribers, and hubs) run on servers. Subscribers can get near-instant notifications (via webhook callbacks) when a topic (feed URL) they're interested in is updated. The protocol in brief is as follows (Google 2013a):

- A publisher declares the Hub server(s) for brokering its resource (a "topic") feed in its Atom or RSS XML file, via `<link rel="hub" ...>`. The hub(s) can be run by the publisher of the feed, or can be a community hub that anybody can use.
- A subscriber (a server that's interested in a topic), initially fetches the Atom URL as normal. If the Atom file declares its hubs, the subscriber can then avoid repeated polling of the URL and can instead register with the feed's hub(s) and subscribe to updates.
- The subscriber subscribes to the Topic URL from the Topic URL's declared Hub(s).
- When the Publisher next updates the Topic URL, the publisher software pings the Hub(s) notifying that there's an update.
- The hub efficiently fetches the published feed and multicasts the new/changed content out to all registered subscribers.

In general, two main aspects feature the PubSubHubBub protocol, one is the extended feed format which indicates the Hub URL, and another is the workflows between participant

publishers, subscribers, and hubs. To meet the need of harvesting real-time GeoChannel catalogue metadata, some specific extension and enablement are needed. The following sections will design the information model for GeoChannel metadata feeds, and the component roles model for harvesting GeoChannel metadata.

## 7.5.2 The Information Model for GeoChannel Metadata Feeds

Two main parts of information content feature a GeoChannel catalogue metadata feed in the context of using PubSubHubBub. Apart from specifying the Hub URL in the PubSubHubBub way, incorporating GeoChannel catalogue metadata into a Web feed needs a specific encoding model embedded into existing feed formats. This section extends XML feed (e.g. Atom, RSS) encoding to support PubSubHubBub protocol and GeoChannel metadata information for real-time harvesting GeoChannel metadata for GeoChannel catalogue services.

Section 7.4.1 has designed the GeoChannel metadata content model, which comprises the metadata fields as listed in Appendix 1. Now they are encoded as extended elements into Web feeds. These extended elements are modelled as part of the information model codified in Appendix 2: “GeoChannel & Discovery: extended-KML & -XML Schema”.

Listing 7-1: GeoChannel metadata encoding model extending Atom/RSS for harvesting GeoChannel catalogue metadata

### ***Syntax:***

*(only some elements are listed here, the complete information model Schema is in **Appendix 2**)*

```
<geochannel:CatalogMetadata>

  <!-- more metadata elements can be included under the Catalogue element.?
  <!-- here only some of them are listed.?

  <geochannel:title>String</geochannel:title>
  <geochannel:geoChannelIdentifier>String</geochannel:geoChannelIdentifier>
  <geochannel:spatialFeature>String</geochannel:spatialFeature>
  <geochannel:spatialExtent>
    <geochannel:TimeSpan>
      <geochannel:start>DateTime</geochannel:start>
      <geochannel:end>DateTime</geochannel:end>
    </geochannel:TimeSpan>
  </geochannel:spatialExtent>
  <geochannel:TemporalExtent>String</geochannel:TemporalExtent>
  <geochannel:geowwURI> AnyURI</geochannel:geowwURI>
  <geochannel:metadataDateTime>dateTime</geochannel:metadataDateTime>
```

```
<!-- there may be other metadata elements ?
```

```
</geochannel:CatalogMetadata>
```

As shown in the Listing 7-1, the sub-elements included in the <CatalogueData> element encode GeoChannel metadata information content (listed in Appendix 1). These encoded elements can then be incorporated into Web feeds. Among these metadata, some spatial attribute elements such as <spatialFeature> and <spatialExtent> can employ WKT (Well-Known Text) (OGC 2006) format for encoding vector geometry objects. The <geowwURI> is a mandatory element for identifying a GeoChannel's AccessFeed, which is an extended-kml-encoded file for organizing (controlling) access named GeoAccessFeed that has been elaborated in Chapter 5. Listing 7-2 gives a simple example of GeoChannel metadata feed.

Listing 7-2: An example of GeoChannel metadata feed for harvesting GeoChannel catalogue metadata via PubSubHubBub protocol

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:geochannel="http://www.geoww.net/geochannel"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.geoww.net/geochannel
http://www.geoww.net/geochannel/GeoChannel_and_Discovery--extended-KML_and_XML_Schema.xsd">
  <title>Example Feed</title>
  <link rel="hub" href="http://myhub.example.com/endpoint"/>
  <link rel="self" href="http://example.org/feed/" />
  <link href="http://example.org/" />
  <id>urn:uuid:60a76c80-d399-11d9-b91C-0003939e0af6</id>
  <updated>2012-12-11T18:30:02Z</updated>
  <entry>
    <geochannel:CatalogMetadata>
      <!-- only part of GeoChannel metadata elements are present here -->
      <geochannel:title>a moving vehicle in London</geochannel:title>
      <geochannel:geoChannelIdentifier>
3F2504E0-4F89-11D3-9A0C-0305E82C3301</geochannel:geoChannelIdentifier>
      <geochannel:spatialExtent>
POLYGON((1 1,5 1,5 5,1 5,1 1),(2 2,2 3,3 3,3 2,2 2))</geochannel:spatialExtent>
      <geochannel:geowwURI>
http://example.org/access_feed_6.kml</geochannel:geowwURI>
    </geochannel:CatalogMetadata>
  </entry>
</feed>
```

In this example, the <link rel="hub" href="http://myhub.example.com/endpoint"/> element indicates the Hub server's endpoint URL, the attribute rel="hub" is specific to the PubSubHubBub protocol, and the atom:link element with rel="self" value identifies the feed

URL (a "topic"). The up-to-date GeoChannel metadata information elements included in the <CatalogueMetadata> element as payload is carried by this feed, which is delivered via this Hub that bridges the publisher (i.e. a GeoChannel node) and some subscribers (i.e. some server nodes for GeoChannel Catalogue Service).

Automatic discovery potential is built in this GeoChannel metadata feed information model. The namespace identifier "http://www.geoww.net/geochannel" in conjunction with the element name "CatalogMetadata" can be used to identify a GeoChannel metadata feed file type. In order to collect more GeoChannel resources metadata, a GeoChannel catalogue server can crawl the Web, just like search engines, to discover new GeoChannel metadata feeds and then to subscribe to their updates from brokering Hubs. In this way, a GeoChannel catalogue server can increasingly collect and build a comprehensive GeoChannel metadata repository to serve GeoChannel clients for discovering these GeoChannel resources.

### 7.5.3 The Component Roles Model for Harvesting GeoChannel Metadata

With the disseminated-feed information model designed already, this section illustrates components workflow for harvesting GeoChannel catalogue metadata. The PubSubHubBub protocol only introduces a general framework including some abstract roles such as publisher, hub and subscriber, how actual GeoChannel architectural components fit in with these roles to jointly work for metadata harvesting task is a question to design the roles model for GeoChannel architectural components involved in the process of catalogue metadata harvest.

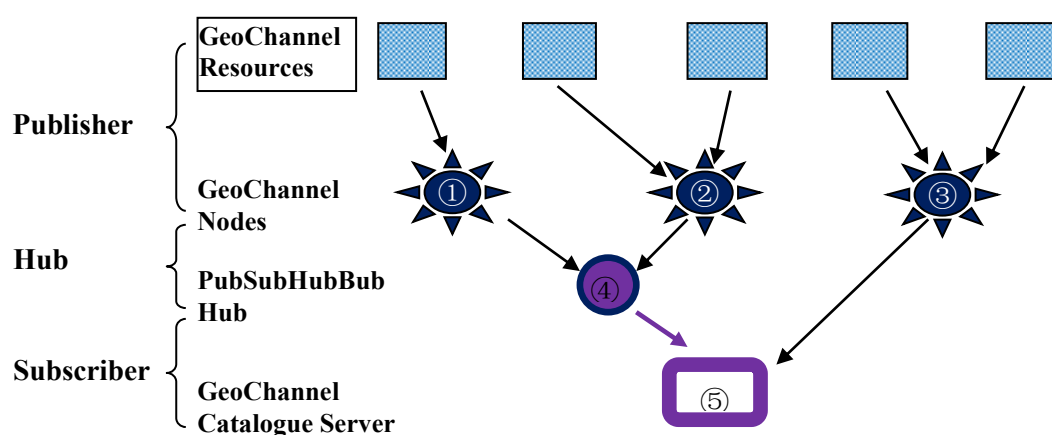


Figure 7-7: The component roles and workflow for harvesting GeoChannel catalogue metadata

(Note that, for brevity and clarity, this figure with workflow here only indicates the delivery directions of GeoChannel metadata feeds, but purposely omits other interactive communication flows which can refer to Figure 7-6.)

The Figure 7-7 illustrates this component roles model with workflow process. There are two



practical schemas to configure GeoChannel architectural components for harvesting GeoChannel metadata. One scheme model is based on the generally conventional PubSubHubBub framework where dedicated PubSubHubBub hub(s), are employed for disseminating GeoChannel feeds. In Figure 7-7, the component assembly (①, ②)-④-⑤ exemplifies this schema model, where the GeoChannel resource sources (e.g. owners, managers, providers of services or information) and GeoChannel nodes act as the role of Publisher, and a dedicated PubSubHubBub hub (e.g. ④ in Figure 7-7) as the role of Hub notifies feed updates to GeoChannel catalogue server ⑤ that is the role of Subscriber.

Another schema is a more GeoChannel-architecture-native model. As GeoChannel nodes themselves in nature are broker components for publish/subscribe message systems, a viable solution is merging the functionality of a conventional PubSubHubBub hub into a GeoChannel node which can then function as the role of a PubSubHubBub hub. Here we term it GeoChannel-native PubSubHubBub hub, in contrast with a dedicated PubSubHubBub hub. As exemplified in Figure 7-7, the GeoChannel node ③ plays as the role of a GeoChannel-native Hub, and the GeoChannel catalogue server ⑤ subscribes to and receives feed updates directly from the GeoChannel node ③.

GeoChannel-native PubSubHubBub hub schema makes more multifunctional GeoChannel nodes. A GeoChannel node is a complex component for organizing and brokering resources, and for bridging GeoClustered clients. Now additional function of real-time disseminating GeoChannel Metadata is further integrated. A general usage is that, a GeoChannel-native PubSubHubBub hub usually works for disseminating metadata of local GeoChannel resources that are currently plugged in this GeoChannel node; while a dedicated PubSubHubBub hub can usually deal with all GeoChannels distributed on the Web.

For working with the different schema models (i.e. component roles models) identified above, the information model codified in Section 7.5.2 will discriminate the “href” attribute values for the atom:link element whose “rel” attribute has the value “hub” value. For example, the `<link rel="hub" href="http://myhub.example.com/endpoint"/>` element (in Listing 7-2) that uses the “href” attribute containing the endpoint URL may point to a dedicated PubSubHubBub hub, or to a GeoChannel node that is equipped with this kind of Hub function. This difference is just worth caring to feed creators (e.g. GeoChannel resource providers); but is transparent to GeoChannel catalogue servers which just subscribe to feed updates via this endpoint URL, without having to know it is a dedicated or a general Hub broker.

## **7.6 OpenSearch for GeoChannels: GeoContext and KML Enablement**

OGC's Catalogue service can fall into the camp of Search Web Services (SWS) (OASIS 2013a). This OASIS's SWS standard describes an OpenSearch (OASIS 2013b) protocol binding for general Search/Retrieve Web Services. This research here employs and extends the OpenSearch approach for GeoChannel search services, mainly involving extending OpenSearch service description document model to support GeoChannel-specific metadata query, and modelling an extended-KML schema for encoding GeoChannel OpenSearch results.

### **7.6.1 OpenSearch with Geospatial & Temporal Extension**

#### **7.6.1.1 An Introduction of OpenSearch**

The OpenSearch technique originated with an intention to allow syndication of search results from distributed search service websites. The basic concept of OpenSearch is to specify how to query a web resource, and additional metadata in the results to support syndicating the results. The OpenSearch specification (Clinton 2012) mainly defines the formats for an OpenSearch description document, query parameters and URL templates syntax, and OpenSearch response elements and applicable formats.

OpenSearch provides a lightweight and specialized schema for describing REST (Fielding 2000) based web search services, as opposed to some general and complex description languages such as WSDL (Web Service Description Language) (W3C 2007) or WADL (Web Application Description Language) (W3C 2009). OpenSearch only describes search query URL templates and some response elements. An OpenSearch description document describes a search engine/service's public interface in the form of parameterized URL templates that indicate how a search client should make search requests. A URL template may contain mandatory or optional query parameters. Search engines can use OpenSearch response elements to add search metadata to results in a variety of content formats. An "OpenSearchDescription" document can provide necessary information for search clients to learn about the interface features, build query requests, and process response formats.

OpenSearch description features brevity and simplicity, so it is a current mass-market technique for discovery on the Web. To employ OpenSearch protocol approach for geospatial search services can reduce complexity and cater for the mass market. This is the motive to extend the OpenSearch protocol to support the discovery of GeoChannel resources in this research.

#### **7.6.1.2 Geospatial & Temporal Extension**

The general OpenSearch specification only involves keyword-based search services. To introduce basic geographic and temporal search facility to the OpenSearch protocol, there is an OCC's proposal (as shown in Table 7-2) of spatial and temporal extension (OGC 2011b), which mainly adds new parameters of space and time conditions for geographically constraining search results.

Two space/time namespaces were provided for use in URL templates that form part of the OpenSearch Description Document:

- <http://a9.com/-/opensearch/extensions/geo/1.0/>
- <http://a9.com/-/opensearch/extensions/time/1.0/>

Under the two namespaces, some new parameters were introduced into OpenSearch URL templates syntax:

With these geo-/time- extension, search services that support the OpenSearch Specification and the Geospatial and Temporal extension are called OpenSearch Geospatial Services (OGC 2011b).

Table 7-2: OGC proposal of geospatial & temporal extension to OpenSearch query parameters in a search request (OGC 2011b)

Name (parameter example)		Definition	Data type and values	Multiplicity and use
<b>Geospatial</b> Extension ( <a href="http://a9.com/-/opensearch/extensions/geo/1.0/">http://a9.com/-/opensearch/extensions/geo/1.0/</a> )				
<b>box</b> bbox={geo:box}		Geographic bounding box	The box is defined by "west, south, east, north" coordinates of longitude, latitude, in a EPSG:4326 decimal degrees	One (optional)
<b>geometry</b> geom={geo:geometry}		Geographic area (geometry)	The geometry is defined using the Well Known Text and supports the following 2D geographic shapes: POINT, LINESTRING, POLYGON, MULTIPOINT, MULTILINESTRING, MULTIPOLYGON The Geometry shall be expressed using the EPSG:4326.	One (optional)
<b>uid</b> id={geo:uid}		Unique identifier of the record in the repository context	Character String	One (optional)
<b>lat</b>	lat={geo:lat} &lon={geo:lon}	The latitude of a given point	Latitude in decimal degrees in EPSG:4326	One (optional)
<b>lon</b>	& radius=	The longitude of a	Longitude in decimal degrees in	One (optional)

	{geo:radius}	given point	EPSG:4326	
<b>radius</b>		A search radius from a lat-lon point	The distance in meters along the Earth's surface.	One (optional)
<b>relation</b> rel={geo:relation}		Spatial relation to result set	Character String; One of “overlaps”, “contains”, “disjoint”	One (optional) default is “overlaps”
<b>name</b> loc={geo:name}		A string describing the location to search.	Character String	One (optional)
<b>Temporal</b> Extension ( <a href="http://a9.com/-/opensearch/extensions/time/1.0/">http://a9.com/-/opensearch/extensions/time/1.0/</a> )				
<b>start</b> startdate={time:start }		A string describing the start of the temporal interval to search.	Character String; must match the RFC-3339 (also used by the Atom syndication format).	One (optional)
<b>end</b> stopdate={time:end}		A string describing the end of the temporal interval to search.		One (optional)

## 7.6.2 Description of GeoChannel Search Service

GeoChannel Search Service description is to describe the capability, interface and usage of a GeoChannel catalogue service. This section employs and extends OpenSearch schema for modelling GeoChannel Search Service description documents, from which GeoChannel clients can learn about services’ space/time coverage, GeoChannel OpenSearch URL templates syntax with GeoChannel-metadata-specific parameters for building search requests to query GeoChannel catalogue services for discovering GeoChannel resources of interest.

### 7.6.2.1 Extending OpenSearch Description Format

To cater for the speciality of GeoChannel catalogue metadata (as modelled in Appendix 1) and the space/time features of GeoChannel resources, the general OpenSearch description document model needs to be extended to incorporate GeoChannel specialities, which mainly involve the extensions to parameters and elements, by using a new namespace: <http://www.geoww.net/geochannel>.

- Extending parameters

Current OpenSearch with OGC geo-extension to query parameters has some usage limitations or inapplicability for searching GeoChannel resources.

For server-side implementation of a GeoChannel Search service, OpenSearch’s general query

parameter named `searchTerms` need to be mapped to appropriate GeoChannel catalogue fields. As GeoChannel catalogue metadata has many fields (as listed in Appendix 1), this one-to-many mapping cannot achieve exact corresponding filter when a user wishes to only search against a specific catalogue field. This issue also affects space-based search, as a general spatial-filter query condition (as listed in Table 7-2) needs to correspond to multiple space-related fields (such as `spatialFeature`, `boundingBox`, `spatialExtent`, etc) of GeoChannel catalogue metadata. However, basically these multiple space-related fields of GeoChannel catalogue have different semantics, while current OGC's geo-extension only allows one spatial query condition can be used at a search request. This problem can cause ambiguity of spatial query conditions.

For improving exactness and speciality of GeoChannel catalogue search services, here the OpenSearch (including OGC's geo/time-extension) request parameter model is extended to incorporate specific parameter types of GeoChannel catalogue metadata fields. These newly-introduced parameters can be used in the OpenSearchDescription document `<Url>` element's template expression, and can be used as the attributes of the Query element in OpenSearchDescription documents or in OpenSearch response messages. The formal information model for these extended parameters is codified in Appendix 2.

Now with this extension, multiple spatial conditions can be defined in one search request, for example, to simultaneously specify `spatialFeature` and `spatialExtent` parameters for discovering GeoChannel resources of interest. And now the "SearchTerms", which is a mandatory parameter in OpenSearch specification, can become optional as now more GeoChannel metadata fields have become individually queryables parameters for exact search. Now with these exact parameters, the general parameters (such as `SearchTerms`, `geo:box`, `geo:geometry`, etc.) with ambiguous semantics can be superseded or they can just function as fuzzy query conditions for generally searching GeoChannel catalogue metadata.

- Extending elements

General OpenSearch service description does not involve an indication about a geographic coverage of a search service. GeoChannel (or other types of) resources usually have explicit geospatial context, so a GeoChannel catalogue service usually has a space/time coverage in total with GeoChannel resources metadata in its repository. For explicitly informing search clients which can then avoid performing queries out of service's geographic domain, here a `SearchCoverage` element is proposed to extend OpenSearchDescription document. The `SearchCoverage` can involve spatial and/or temporal extents of a GeoChannel search service. The format of GeoRSS Simple is employed for encoding the `SpaceExtent`, which can then be

visualized to inform clients of spatial involvement of this search service. This selection of GeoRSS encoding for OpenSearchDescription documents differs from GeoChannel metadata feeds (as designed in Section 7.5.2) that use WKT (Well-Known Text), and as well differs from GeoChannel search responses that uses KML, for encoding geometry features.

Listing 7-3: The syntax for the information structure of GeoChannel OpenSearch service coverage included in OpenSearchDescription document

**Syntax:** (only an outline is listed here, the information model Schema is in *Appendix 4*)

```
<geochannel:SearchCoverage>

  <geochannel:SpaceExtent>
    <georss:box> </georss:box>
    <georss:polygon> </georss:polygon>
    <georss:circle> </georss:circle>
  </geochannel:SpaceExtent>

  <geochannel:TimeExtent>

    <geochannel:TimeSpan>
      <geochannel:start> </geochannel:start>
      <geochannel:end> </geochannel:end>
    </geochannel:TimeSpan>
    <geochannel:TimeSpan> </geochannel:TimeSpan>

  </geochannel:TimeExtent>

</geochannel:SearchCoverage>
```

- Example

Here a simple example is given for exemplifying GeoChannel OpenSearch description documents.

Listing 7-4: an example of GeoChannel OpenSearch description document

```
<?xml version="1.0" encoding="UTF-8"?>
<OpenSearchDescription xmlns="http://a9.com/-/spec/opensearch/1.1/"
  xmlns:geo="http://a9.com/-/opensearch/extensions/geo/1.0/"
  xmlns:time="http://a9.com/-/opensearch/extensions/time/1.0/"
  xmlns:georss="http://www.georss.org/georss/1.1" xmlns:geochannel="http://www.geoww.net/geochannel"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.geoww.net/geochannel
http://www.geoww.net/geochannel/GeoChannel_and_Discovery--extended-KML_and_XML_Schema.xsd">

  <ShortName>GeoChannel find</ShortName>

  <Description>Use Example.com to search the metadata of GeoChannels.</Description>
```

```

    <Url type="application/opensearchdescription+xml" rel="self"
template="http://example.com/OpenSearchDiscriptionDocument.xml"/>
    <Url type="application/vnd.google-earth.kml+xml"
template="http://example.com/search/geochannel/kml/ ?q={searchTerms?}&start={time:start?}&s
top={time:end?}&title={geochannel:title?}&atitle={geochannel:alternativeTitle?}&abstract={ge
ochannel:abstract?}&topic={geochannel:topicCategory?}&id={geochannel:geoChannelIdentifier?}&a
mp;keyword={geochannel:keyword?}&type={geochannel:resourceType?}&feature={geochannel:spa
tialFeature?}&bbox={geochannel:boundingBox?}&scope={geochannel:spatialExtent?}&rel={ge
o:relation?}&vscope={geochannel:verticalExtent?}&tscope={geochannel:TemporalExtent?}&res
p={geochannel:responsibleOrganisation?}&access={geochannel:limitationOnPublicAccess?}&count=
{count?}&startPage={startPage?}&startIndex={startIndex?}"/>

    <LongName>GeoChannel Search Service</LongName>
    <Query role="example" geochannel:keyword="vehicle" geochannel:boundingBox="10,10,12,12"/>
    <Query role="example" geochannel:resourceType="dynamics" geochannel:spatialExtent="POLYGON((1
1,5 1,5 5,1 5,1 1),(2 2,2 3,3 3,2 2 2))" geo:relation="contains"/>
    <Query role="example" searchTerms="flight" geochannel:spatialFeature="MULTILINESTRING((3 4,10
50,20 25),(-5 -8,-10 -8,-15 -4))" geo:relation="overlaps"/>
    <Query role="example" searchTerms="traffic" geochannel:responsibleOrganisation="TfL"
time:start="2010-09-01T00:00:00" time:end="2013-09-01T23:39:59"/>

    <!-- Other service metadata-->
    <Developer>Xuelin He, www.Geoww.net</Developer>
    <Attribution>Copyright 2012-2013, Geoww.net</Attribution>
    <SyndicationRight>open</SyndicationRight>
    <Language>en-uk</Language>

    <geochannel:SearchCoverage>
        <geochannel:SpaceExtent>
            <georss:box>-90 -180 90 180</georss:box>
        </geochannel:SpaceExtent>
        <geochannel:TimeExtent>
            <geochannel:TimeSpan>
                <geochannel:start>2010-12-17T09:30:47Z</geochannel:start>
                <geochannel:end>2012-12-17T09:30:47Z</geochannel:end>
            </geochannel:TimeSpan>
            <geochannel:TimeSpan>
                <geochannel:start>2013-12-17T09:30:47Z</geochannel:start>
                <geochannel:end>2015-12-17T09:30:47Z</geochannel:end>
            </geochannel:TimeSpan>
        </geochannel:TimeExtent>
    </geochannel:SearchCoverage>

</OpenSearchDescription>

```

### 7.6.2.2 Construction of Search Requests

According to the URL template(s) defined in an OpenSearchDescription document, OpenSearch requests can be constructed. All parameter values in a request message should be URL (percent) encoded. URL-encoding (RFC3986 2005) converts characters into a format that can be transmitted over the Internet. Spatial coordinates encoding for some parameter values in the geo-enabled OpenSearch requests introduces space or comma (,) characters. A space character must have a percent-encoded value of %20, or it can be encoded as a plus sign '+'. The comma (,) used to separate spatial coordinate values can be but not necessarily replaced by its percent-encoded equivalent %2C.

Here are some examples of geometries in WKT (Well-Known Text) encoding which introduces many space or comma characters:

```
POINT(6 10)
LINESTRING(3 4,10 50,20 25)
POLYGON((1 1,5 1,5 5,1 5,1 1),(2 2,2 3,3 3,2 2))
MULTIPOINT(3.5 5.6, 4.8 10.5)
MULTILINESTRING((3 4,10 50,20 25),(-5 -8,-10 -8,-15 -4))
MULTIPOLYGON(((1 1,5 1,5 5,1 5,1 1),(2 2,2 3,3 3,2 2)),((6 3,9 2,9 4,6 3)))
```

So a request as below:

```
http://example.com/kml/?q=traffic&spatialExtent=POLYGON((1 1,5 1,5 5,1 5,1 1),(2 2,2 3,3 3,2 2))
```

can be URL-encoded into:

```
http://example.com/kml/?q=traffic&spatialExtent=POLYGON((1%201,5%201,5%205,1%205,1%201),(2%202,2%203,3%203,3%202,2%202))
```

GeoChannel OpenSearch requests are performed via HTTP GET communication to query GeoChannel catalogue services to get back search results.

### 7.6.3 KML-extension for Search Responses

Currently the typical media formats for returning OpenSearch search results are XML syndication formats, such as RSS and Atom. The current OGC's proposal of OpenSearch Geo-extension also chooses Atom as a default format, although other formats are also allowable. This research extends KML language for encoding GeoChannel OpenSearch results. Two main considerations stated below motivate this KML extension.



In some sense, KML-encoded search responses can make OpenSearch more “geospatial” and integratable. By employing a GeoWeb-native language such as KML for encoding, OpenSearch responses information can be easily augmented (e.g. aggregated, mashed up, merged) with normal geospatial data. So a unified encoding format language can bridge and fuse the OpenSearch technique into the GeoWeb spectrum of technology.

A more compelling motive is to facilitate seamlessly integrating the workflows of the GeoChannel architecture. In order to link up the process flows of GeoChannel discovery (chapter 7) and access organization (Chapter 5), designing their information and process models (i.e. the encoding schemas for GeoChannelSearch, GeoAccessFeed, GeoChannel, etc.) desires a consistent encoding language with geospatially-native capability. So this research extends current KML language to fulfil this objective. As shown in Figure 3-1, actually a series of KML extensions have been conducted in this research for supporting this GeoChannel architecture.

This section here designs the information model for OpenSearch results (responses) of GeoChannel Search Service by extending current KML encoding language. The normative information model is codified in Appendix 2. Figure 7-8 illustrates the structure of some main element types derived from existing KML components.

To facilitate the clarity and modularity of expression, Figure 7-8 is organized by using three separate parts, which can in essence be in a whole diagram as these elements have nesting relationship. The elements GeoChannelSearch and GeoChannel extend KML’s AbstractContainerType and AbstractFeatureType respectively, implying they can appear in the corresponding positions in a normal KML file where standard KML Container type (such as Folder or Document) or Feature type (such as Placemark, NetworkLink, and Container) can appear. This derivation mechanism makes GeoChannel search results and GeoChannel objects able to fuse into KML media seamlessly. GeoChannel element (s) can be nested within a GeoChannelSearch element for representing search-returned collection of GeoChannel metadata that matches a search condition; GeoChannel elements can also appear outside a GeoChannelSearch element, that is, GeoChannels as a new type of resource (i.e. dynamics, services, interactions) objects can appear in the positions of normal KML Feature objects. GeoChannel elements are the most important information objects in this GeoChannel architecture system. They can include the sub-element of AccessFeed (as shown in Figure 7-8 (b)) for organizing resource access, which is a core technology that has been designed in Chapter 5.

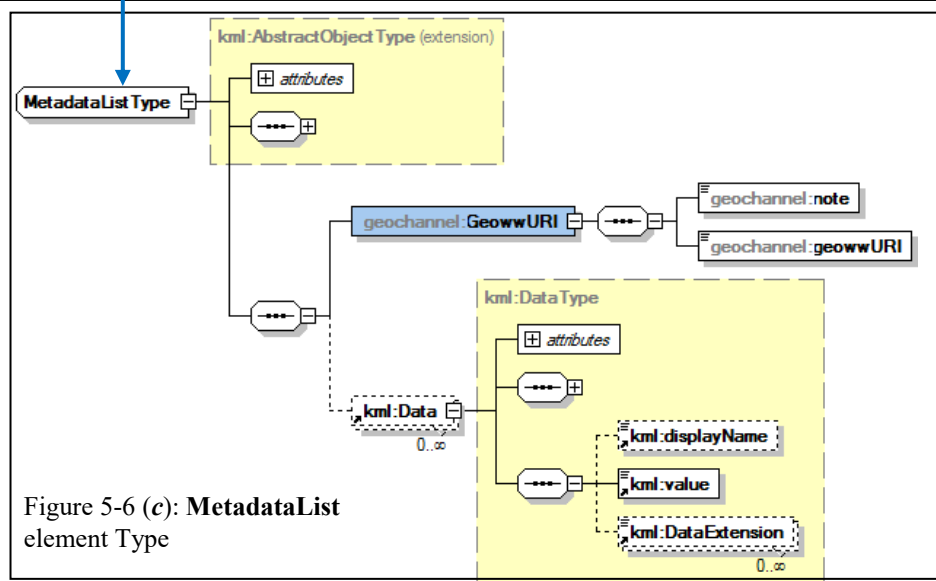
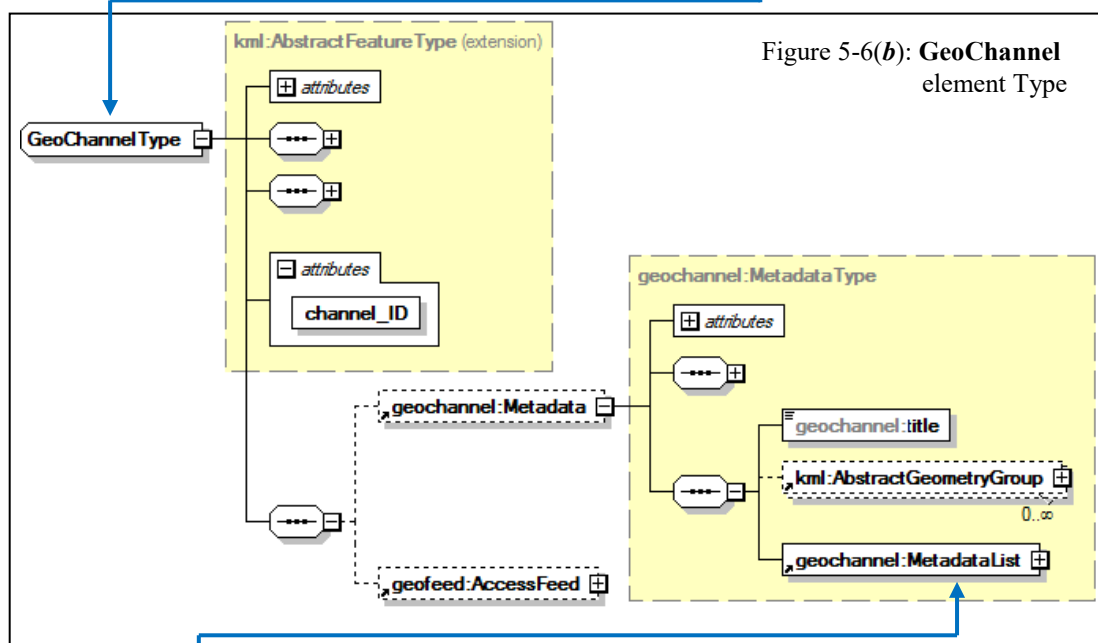
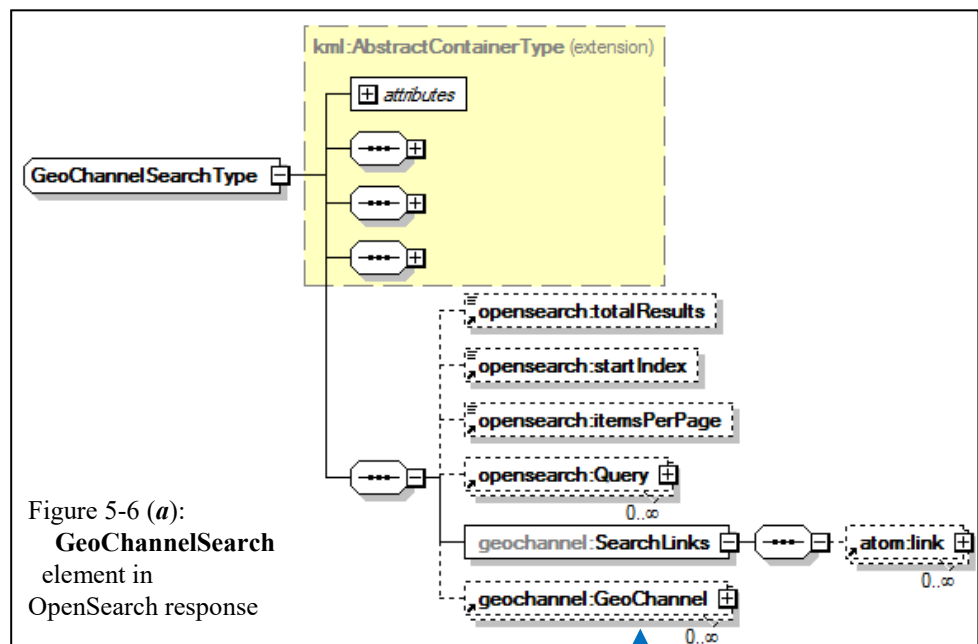


Figure 7-8: Extended-KML encoding model for OpenSearch responses & GeoChannel Objects

A GeoChannelSearch element also include some elements defined by OpenSearch specification (as shown in Figure 7-8 (a)), among which the Query element(s) is the similar information type as it appear in an OpenSearchDescription document (as illustrated in Section 7.6.2). When nested within a GeoChannelSearch element, the GeoChannel elements have their sub-element Metadata that further nests the MetadataList element for grouping the actual GeoChannel metadata with different information fields (as listed in the Metadata Information Model in Appendix 1). Here, as shown in the lower part of Figure 7-8 (c), the format of standard KML “Data” type is employed for encoding these GeoChannel metadata fields. The most mandatory metadata returned from a GeoChannel Catalogue Service is expressed by the “GeowwURI” element, which identifies the Geoww-URI of this GeoChannel resource. A Geoww-URI (as designed in Chapter 6) usually references the AccessFeed for organizing access to this GeoChannel resource. Access-Organize is a key work process following the step of GeoChannel Discovery.

Listing 7-5: The syntax for the information structure of GeoChannel OpenSearch responses (results) in extended-KML encoding

**Syntax:** (only an outline is listed here, the information model Schema is in *Appendix 4*)

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2"
  xmlns:geoww="http://www.geoww.net/geochannel"
  xmlns:geofeed="http://www.geoww.net/geofeed"
  xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
  xmlns:geo="http://a9.com/-/opensearch/extensions/geo/1.0/"
  xmlns:time="http://a9.com/-/opensearch/extensions/time/1.0/">
  <!-- there may be some other standard KML data snippets here -->
  <geoww:GeoChannelSearch>
    <opensearch:totalResults>4230000</opensearch:totalResults>
    <opensearch:startIndex>21</opensearch:startIndex>
    <opensearch:itemsPerPage>10</opensearch:itemsPerPage>
    <opensearch:Query role="" searchTerms=" " startPage="" geo:box=""/>
    <geoww:SearchLinks>
      <atom:link rel=" " href=" " type=" " title=" "/>
      <atom:link rel=" " href=" " type=" " title=" "/>
    </geoww:SearchLinks>
    <geoww:GeoChannel channel_ID="!">
      <name> </name>
      <description> </description>
      <geoww:Metadata>
        <geoww:title> </geoww:title>
        <Geometry>...</Geometry>
        <geoww:MetadataList>
          <geoww:GeowwURI>
            <geoww:note> </geoww:note>
            <geoww:geowwURI> </geoww: geowwURI>
          </geoww:GeowwURI>
          <Data name=" ">
            <value> </value>
```

```

        </Data>
        <Data name=" ">
            <value> </value>
        </Data>
    </geoww:MetadataList>
</geoww:Metadata>
<geofeed:AccessFeed src=" " channel_ID="!!!">
    <geofeed:Accessibility/>
</geofeed:AccessFeed>
</geoww:GeoChannel>
<geoww:GeoChannel channel_ID=" "> </geoww:GeoChannel>
</geoww:GeoChannelSearch>
    <!-- there may be some other standard KML data snippets here -->
</kml>

```

The syntax for encoding GeoChannel OpenSearch responses (results) in extended-KML encoding is shown in Listing 7-5. The formal syntax model is codified in Appendix 2 that can be used for guiding and validating the encoding of actual KML files (messages) of GeoChannel OpenSearch responses. For brevity, Listing 7-5 only presents some element information. It purposely simplifies or omits the content of the element of AccessFeed, which is the research focus for Chapter 5 “GeoAccessFeed”.

In Listing 7-5, the <Geometry> element under <Metadata> element is a placeholder that can present any KML geometry objects for depicting spatial features/characteristics of a GeoChannel to facilitate choosing GeoChannels of interest by clients. It is noted that the included information, i.e. the <GeowwURI> sub-element inside the <Metadata> element indicates the source address of a KML file that contains the information about this GeoChannel’s AccessFeed. After the process of the GeoChannel discovery, this AccessFeed KML file is fetched to the client to form (or update) the <AccessFeed> element within the <GeoChannel> element. So in this way GeoChannel access-control functionality follows discovery service seamlessly.

## 7.7 Context-topology Query Service

Apart from the GeoChannel Catalogue Service designed in previous sections, this section describes Context-topology Query Service as another virtual-world approach for finding GeoChannel resources of interest.

### 7.7.1 The concept of Context Topology

Actually Chapter 5 that designs the GeoAccessFeed technology has already involved the technique of context topology, as Section 5.4.3 designs the information model of TopologyLink module within AccessFeed. This section here just conceptualizes this technique.



Figure 7-9: Context topology network – an indoor street view example

Context Topology of GeoChannels refers to the context relations between GeoChannels relevant to each other. The relevance between these GeoChannels is about the GeoContexts of their resources in terms of the similarity or proximity on their GeoContext conditions (i.e.

GeoPolicies) defined in AccessFeeds that correlate clients with resources.

By analogy to the general Web that features web resources' hyperlink topology relations which are usually based on the relevance of content, topics, categories or workflows, the GeoChannel Web features context topology relations that characterize the similarity, adjacency or proximity between context attributes of GeoChannel resources which can use AccessFeeds for representing and controlling these context attributes and conditions.

Figure 5-4, which illustrates TopologyLink concept on traffic networks using AccessFeed technique, can actually exemplify the context topology concept as well, and here a street view example can further illuminate this concept. The pictures in Figure 7-9 are about some indoor panoramas of a hotel. The panoramas as visual resources for viewing have their context attributes respectively, which can be parameterized by location coordinates, heading and pitch, etc. The proximity relations between panoramas are actually the adjacency between the context attributes of corresponding panoramas. These relations, which are used for navigation in street view maps, can form a context topology network as shown below.

A GeoChannel context can be based on and can be parameterized by any types of attributes. In a generalized sense and in broad application scenarios, context topology relations are not necessarily limited to space/time topology, but can refer to various context attribute relations other than space/time context. The GeoPolicy Language designed in Chapter 3 can work for various context attribute scenarios. For the purpose of being more intuitive and easily understandable, this thesis document mostly exemplifies space/time topology in illustrations.

### **7.7.2 Querying Context Topology Network**

From a mathematical perspective, topology relations can be represented by graph networks, which are termed GeoContext graphs (or context graph generally) in GeoChannel technology where a vertice/node represents a certain context condition defined by an AccessFeed of a GeoChannel resource, and an arc/edge represents the similarity or adjacency between GeoChannel contexts.

A large-scale graph topology service named GeoChannel Web Engine is designed in the Geonoon platform (as described in Chapter 8) which is a prototype implementation of the GeoChannel technological architecture. It can traverse the context graphs to query context conditions and to identify GeoChannel resources based on context topology relations. So this Context-topology Query Service can function as a virtual-world approach apart from GeoChannel Catalogue Service for discovering GeoChannels in a dynamic and context-aware environment.

Section 8.5 will describe neighbour-querying discovery which is another topology-based approach for GeoChannel Discovery. Chapter 9 will exemplify two use cases which demonstrates the context topology networks for the New York taxi-trip project and the British museum street view project.

## **7.8 Summary**

This chapter designed the technical system for discovering GeoChannel resources. Two categories of approaches were elaborated, both of which are to eventually get the Geoww-URIs for GeoChannel resources of interest. The physical-world approach is to directly obtain Geo-URIs from visual or wireless-signal media such as scanning QR code labels, detecting RFID tags or WiFi/Bluetooth signals that have Geoww-URIs encoded. This approach is particularly suitable for indicating and finding pervasive resources such as location-based services in our physical environment. The virtual-world approach is to search GeoChannel Catalogue Services based on keywords or GeoContext conditions, or to query GeoChannel Context-topology Query Service based on similarity or proximity of context attributes of different GeoChannel resources.

A series of normative models for information, interfaces and workflow processes have been designed for GeoChannel Discovery:

- the metadata model of GeoChannel catalogues, as codified in Appendix 6, provides a unified metadata content model for characterizing GeoChannel resources for their discovery.

- an equivalent implementation of OGC CSW (Catalogue Service for the Web) interfaces is designed for simplifying GeoChannel discovery tasks.
- various message models for interaction with GeoChannel Catalogue Services. These mainly include: GeoChannel metadata feeds model for real-time harvesting GeoChannel catalogue metadata; the extended OpenSearch service description document format and request format; and the extended-KML-encoded schema for responses of GeoChannel search operations.
- the workflow process models for discovering of GeoChannel resources using physical/virtual-world approaches, and for real-time harvesting of GeoChannel catalogue Metadata.

Federated catalogue services for GeoChannel discovery can be achieved from the normative design of unified and consistent metadata set, search operation interface, and normalized formats for query messages and result sets. The OpenSearch technique is employed and further extended for democratizing and catering for spatial-temporal specialization of searching GeoChannel resources.



## **8 The GeoChannel Web & a Prototype Platform**

### **8.1 Introduction**

The previous chapters have elaborated the concepts and practical techniques for identifying, discovering, bridging, clustering and interacting with things by employing their context attributes. This chapter will present a systematic perspective and a full architecture view underpinned by the constituent concepts and techniques designed heretofore for a new type of Web, i.e. the GeoChannel Web.

The GeoChannel Web is expected for mapping and applying dynamic and random relations between real-world things based on their context attributes. Based on the GeoAccessFeed technology, this chapter will design new technical mechanisms for forming GeoChannel networks, and will illuminate the features of the GeoChannel Web.

This chapter will also develop a prototype platform, named Geonoon, for materializing and implementing the new concepts, models and techniques proposed or designed in this research. This Geonoon platform is about to support the use case projects that will be developed and presented in next chapter.

### **8.2 Dynamic GeoChannel Networks of Resources & Users**

A GeoChannel resource gathers clients that commonly meet its access-control GeoPolicy condition, and a client may access multiple GeoChannel resources simultaneously based on its interest and the matching of accessing conditions of these interested resources. Figure 8-1 illustrates some GeoChannel resources with their respective GeoServeZones depicted as simple boundary shapes. The resource-user relationships depicted in Figure 8-1 can form the access relationship topology as illustrated in Figure 8-2. This is a resource-user relationships network named a GeoChannel network on which resource nodes bridge user nodes and user nodes bridge resource nodes.

The topology depicted in Figure 8-2 can also be represented in Figure 8-3 which gives another view arranging resources and users in two tiers for the clarity of examining the resource-user relationships. A user accessing a resource via this resource's Accessor. A GeoChannel Accessor

Bus functions as an intermediate facility for plugging and bridging Accessors which connect to their corresponding GeoChannel resources respectively. In Figure 8-3, taking the user number 13 for example, its Accessor Bus incorporates four Accessors from resources B, C, D and E.

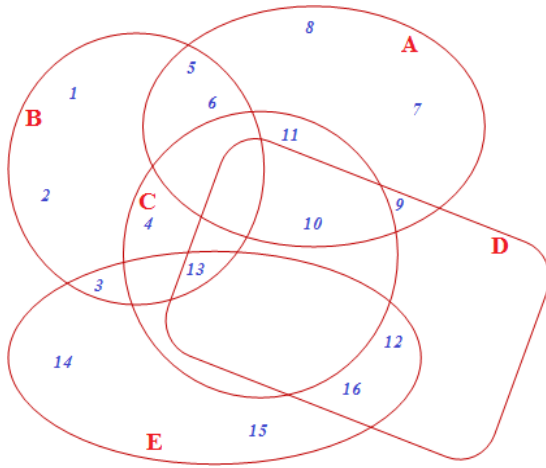


Figure 8-1: An example depiction of GeoChannel resources with users matching access-organize conditions

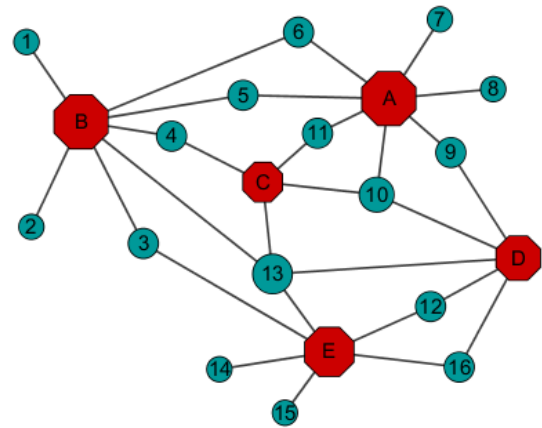


Figure 8-2: : An example of GeoChannel network with connected nodes of users and resources

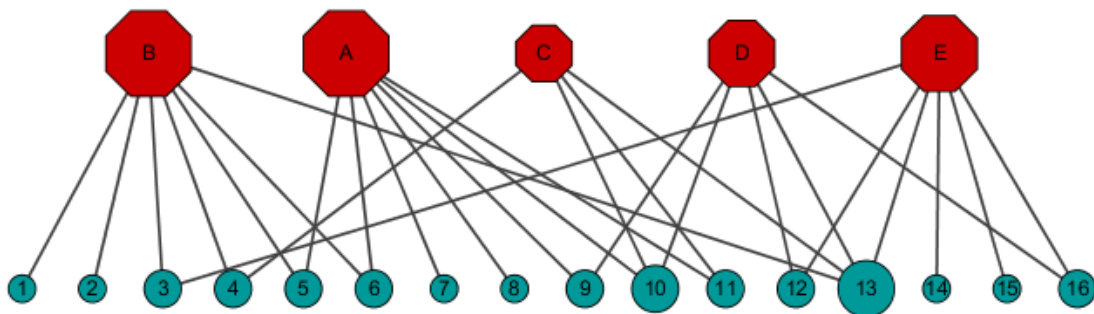


Figure 8-3: An example of GeoChannel network bridged by resource and users  
(another view of the same topology depicted in Figure 8-2)

This kind of resource-/user-bridged GeoChannel networks have highly dynamic topology. When the changes of users' interest, the GeoContexts of resources/users, or access-organize conditions happen, those user-resource connection relationships may change (i.e. become connected or disconnected) anytime.

Here we refer to this kind of dynamic networks as ad hoc or opportunistic GeoChannel networks for characterizing the dynamic nature of network topology. The term ad hoc network (Perkins & others 2001) or opportunistic network (Pelusi et al. 2006) is originally and mostly used in wireless communication technology domain, but now these concepts can analogically

characterize the highly dynamic and opportunistic features about the connectivity of topology consisting of resource and user nodes.

There are two metrics proposed here for quantifying the resource-user accessing relationships. The “Popularity” represents the current number of a resource’s users, and the “Sociability” represents the current number of a user’s accessing resources. For example, in Figure 3, the “Popularity” of resource A is 7, and the “Sociability” of user ⑩ is 3. Only when a user’s Sociability exceeds 1 can this user bridge resource nodes for a GeoChannel network. As more popular GeoChannel resources (e.g. events, activities or services) can usually gather more people for participating in, and in turn more sociable participants with various interests may usually bridge more resources together, this user-bridged networking mechanism can form some highly dense GeoChannel networks geographically corresponding to hot events or popular activities happening in the real world.

Table 8-1: Geo-Clustering clients and federating resources on a GeoChannel network

Resource Client	A	B	C	D	E
1		√			
2		√			
3		√			√
4		√	√		
5	√	√			
6	√	√			
7	√				
8	√				
9	√			√	
10	√		√	√	
11	√		√		
12				√	√
13		√	√	√	√
14					√
15					√
16				√	√

The dynamic GeoChannel networks are mapping and reflecting the real-time connection relationships between resources and users. The matrix in Table 8-1 just records this kind of client-resource-client and resource-client-resource relations at a certain time point/period for the

same GeoChannel network example that is illustrated in Figure 8-1, 8-2 and 8-3.

The relations between resources and clients can be easily analysed from the matrix presented in Table 8-1. The each column outlined by red-colour line reflects a group of clients that are accessing a same resource. This group is referred to as a GeoCluster. And each row outlined by blue-colour line reflects some resources that are concurrently accessed by a same client. This status is referred to as federating resources via a client. The techniques about GeoClustering clients (i.e. resource-bridging clients) and federating resources (i.e. client-bridging resources) will be elaborated in section 8.3 and in Section 8.4 respectively.

### **8.3 GeoCluster & GeoBridge**

Gathering and bridging users for interaction is an anticipated capability of the GeoChannel Web, to facilitate building and running participative and cooperative applications or services, and to supply a new social-networking paradigm. This section (with its subsections) elaborates effective mechanisms for organizing client-client dynamic correlation. GeoCluster and GeoBridge are a couple of techniques for achieving this functionality.

#### **8.3.1 GeoClustering Users for Different GeoServeZones**

##### **8.3.1.1 Resource, GeoCluster and Clients Relationship**

The AccessFeed technique solves the demand on client-resource dynamic correlation. Users interested in a GeoChannel resource and matching its access-organize condition get access to this resource, so they fall into a user group named GeoCluster.

A GeoCluster, also named GeoCommunity, is a user-group unit that contains some users that currently meet the access-control GeoPolicy condition of a GeoChannel resource. GeoClustering refers to the mechanism and process for dynamically gathering users, which are interested in the same GeoChannel resource and can meet its access-control condition, into a group, i.e. a GeoCluster.

GeoClustered clients are referred to as resource-bridged clients/users, because a resource functions as an intermediary/medium for gathering and bridging those interested clients with similarity or proximity in terms of their GeoContexts that meet this resource's access-control condition. It is noted that, usually there is certain meaning difference between the word client and the word user in this thesis document. Users are special clients that currently meet a resource's access condition and have become the members of the GeoCluster, while clients

generally refer to all the objects/people that are interested in and are running the AccessFeed of a resource, but not necessarily meet its access condition currently. Their status of memberships is changeable by the GeoClustering process, which dynamically incorporate a client into or exclude a user from the GeoCluster.

A GeoCluster is associated to a GeoChannel resource. They have two main components, i.e. GeoPolicy condition and Geoww-URI. The GeoPolicy condition evaluates the GeoContext entitlement of clients for accessing this resource; as a result, those clients that can currently meet this condition are grouped into this GeoCluster. So as an effect, the GeoPolicy condition controls both accessing resource and GeoClustering users, namely, it controls both client-resource and client-client correlation. A GeoChannel resource and its user GeoCluster has the same Geoww-URI, which is encoded using the Geoww-URI scheme model. Their Geoww-URI plays a very important role for GeoClustering mechanism during the process of client-client interaction, which is meant to be elaborated later in this section.

Within a GeoCluster, clients can interact with each other for cooperative tasks or social networking. Normally, a user can only interact with other users in the same GeoCluster, but cannot conduct cross-GeoCluster interaction. In the case that a client may access multiple GeoChannel resources concurrently, as exemplified in Figure 8-2 or Figure 8-3, this client can belong to multiple GeoClusters that are identified by different Geoww-URIs. So this client will have multiple different user Geoww-URI identifiers, each of which identifies this client in a GeoCluster. Chapter 6 has designed and exemplified the Geoww-URI technique for identifying a user in a GeoCluster. This client can use different user Geoww-URI identifiers to interact with other users distributed in corresponding GeoClusters. The next section will design a GeoCluster-isolating mechanism based on the Geoww-URI technique.

#### **8.3.1.2 A Same Target Resource but for Different GeoChannels & GeoClusters**

As illuminated in Chapter 2 on its concept, a GeoChannel has some main components, i.e. target resource, access condition and current users (GeoCluster). A general Web resource can become various GeoChannel resources with their respective GeoContexts and user clusters, as it can be specified, in different AccessFeeds, to act as the target resource of different GeoChannels with different access-control conditions. This diversity is referred to as “Polymorphism” capability of a same target resource acting as different GeoChannel resources with different user GeoClusters.

This Polymorphism can be exemplified below. Listing 8-1 gives the example of four

AccessFeeds that GeoContextualize four different GeoChannel resources with their respective GeoPolicy conditions, but using (i.e. controlling access to) a same target resource, [www.example-site.com/a.html](http://www.example-site.com/a.html), which is a general Web application.

Listing 8-1: The AccessFeeds example of a same target resource for different GeoChannel resources and user GeoClusters

Listing 8-1-1: AccessFeed<sub>1</sub>

```
<AccessFeed src="www.accessfeed-site1.com/accessfeed-1.kml">
  <Accessibility>
    <Accessor accessor_ID="dfg81">
      <GeoPolicy>...</GeoPolicy>
      <href>www.example-site.com/a.html</href>
    </Accessor>
  </Accessibility>
</AccessFeed>
```

Listing 8-1-2: AccessFeed<sub>2</sub>

```
<AccessFeed src="www.accessfeed-site2.com/accessfeed-2.kml">
  <Accessibility>
    <Accessor accessor_ID="dfg82">
      <GeoPolicy>...</GeoPolicy>
      <href>www.example-site.com/a.html</href>
    </Accessor>
  </Accessibility>
</AccessFeed>
```

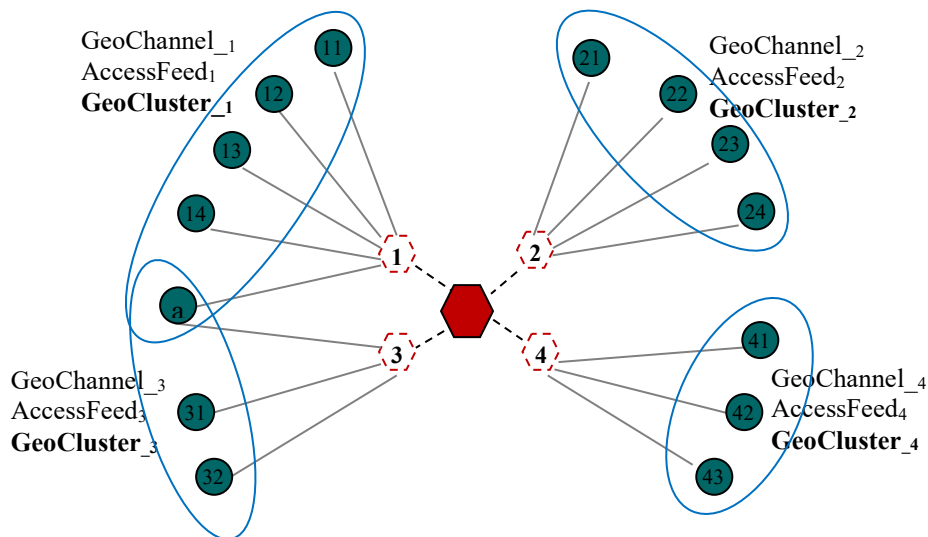
Listing 8-1-3: AccessFeed<sub>3</sub>

```
<AccessFeed src="www.accessfeed-site3.com/accessfeed-3.kml">
  <Accessibility>
    <Accessor accessor_ID="dfg83">
      <GeoPolicy>...</GeoPolicy>
      <href>www.example-site.com/a.html</href>
    </Accessor>
  </Accessibility>
</AccessFeed>
```

Listing 8-1-4: AccessFeed<sub>4</sub>

```
<AccessFeed src="www.accessfeed-site4.com/accessfeed-4.kml">
  <Accessibility>
    <Accessor accessor_ID="dfg84">
      <GeoPolicy>...</GeoPolicy>
      <href>www.example-site.com/a.html</href>
    </Accessor>
  </Accessibility>
</AccessFeed>
```

Figure 8-4 illustrates the GeoClusters of these different GeoChannel resources that are using the same target resource, i.e. a Web application in this example. This example has four GeoChannels that are GeoContextualized by their AccessFeeds (as shown in Listing 8-1) which form four GeoClusters containing respective users. Even if the User<sub>a</sub> is accessing both GeoChannel<sub>1</sub> and GeoChannel<sub>3</sub> concurrently, it has independent memberships in GeoCluster<sub>1</sub> and in GeoCluster<sub>3</sub>.



**Geoww-URIs for these GeoChannels/GeoClusters:**

**GeoCluster<sub>1</sub>:** [Geoww:// www.accessfeed-site1.com/accessfeed-1.kml::\[accessor:dfg81\]:www.example-site.com/a.html](http://www.accessfeed-site1.com/accessfeed-1.kml::[accessor:dfg81]:www.example-site.com/a.html)

**GeoCluster<sub>2</sub>:** [Geoww:// www.accessfeed-site2.com/accessfeed-2.kml::\[accessor:dfg82\]:www.example-site.com/a.html](http://www.accessfeed-site2.com/accessfeed-2.kml::[accessor:dfg82]:www.example-site.com/a.html)

**GeoCluster<sub>3</sub>:** [Geoww:// www.accessfeed-site3.com/accessfeed-3.kml::\[accessor:dfg83\]:www.example-site.com/a.html](http://www.accessfeed-site3.com/accessfeed-3.kml::[accessor:dfg83]:www.example-site.com/a.html)

**GeoCluster<sub>4</sub>:** [Geoww:// www.accessfeed-site4.com/accessfeed-4.kml::\[accessor:dfg84\]:www.example-site.com/a.html](http://www.accessfeed-site4.com/accessfeed-4.kml::[accessor:dfg84]:www.example-site.com/a.html)

Figure 8-4: A same web resource acts as different GeoChannel resources for grouping and working for clients in different GeoClusters

It is desired that this same Web application, without custom code logic for specific GeoChannels, can create and maintain a separate GeoCluster for each GeoChannel (resource), and can bridge users' interaction within each GeoCluster, without mixing all these clients/users' memberships and messing up their interaction. This desired GeoCluster-isolating capability has practical justification. As a same target resource may be used by arbitrary number of GeoChannels, and each GeoChannel aims to gather and connect their clients with specific-range of GeoContext status, it would make no sense if this target resource could not identify membership affiliations of clients, could not partition different GeoServeZones of different GeoChannels, or needs GeoChannel-specific code logic for dealing with inter-client interaction

for a specific GeoChannel. For example, suppose that this target resource is a social-networking application, messing up communication between irrelevant people is not a desired feature.

So now the technical question to be addressed is, how can the same target resource, a Web application in this case, function as a seemingly-exclusive resource to be used by each of GeoChannels with their respective GeoPolicy conditions and GeoClusters of users?

Now the Geoww-URI technique can function as a key solution to this question. This research designs the GeoClustering and inter-clients interaction mechanisms based on the novel Geoww-URI technique designed in Chapter 6. Chapter 8 will elaborate the Geonoon platform that supplies an implementation of a common GeoCluster Service, and Chapter 9 will demonstrate use cases about GeoClustered user interaction. Here only a brief description is given about Geoww-URI based GeoClustering and inter-clients interaction mechanisms. The GeoCluster Service acts as a common service infrastructure for automatically and dynamically GeoClustering clients and bridging their communication interaction for any GeoChannel resources, which do not need to have their own facility for this function. The communication messages between any clients with a target resource are marked by Geoww-URIs, via which the target resource node/application can identify a client's dynamic membership affiliations to specific GeoChannel resources (GeoClusters). The GeoCluster Service facility can isolate GeoClusters and correctly broker clients' interaction within GeoClusters in a transparent way. Normally, even if multiple GeoChannels' clients are interacting with a same target resource (e.g. a Web application/service), a client can only know and interact with its own GeoCluster members, and other GeoClusters are invisible to this client. This isolating mechanism can achieve an effect that each client or GeoCluster seems to exclusively use this target resource, which has the "Polymorphism" capability to serve clients in different GeoServeZones with respective GeoContext conditions.

Thanks to the Geoww-URI technique for identifying GeoChannel resources, GeoServeZones, GeoClusters and users, now the users of a same target resource but of different GeoChannels (resources) can be separated into different GeoClusters. The users within a GeoCluster can interact with each other, without being messed up by other GeoClusters' users interaction, although all these GeoClusters' users are concurrently using a same target resource, e.g. a general web application. Figure 8-4 illustrates inner-GeoCluster clients' relation, which is in contrast with cross-GeoCluster clients' relation illustrated in Figure 8-5 that depicts the GeoBridge mechanism which will be designed in next section.



## **8.3.2 GeoBridge Mechanism for GeoClusters Union**

### **8.3.2.1 The demand on GeoClusters Union**

The GeoCluster-isolating requirement and technical mechanism has been described in last section; however, there is also a practical demand on connecting multiple GeoClusters for cross-GeoCluster communication to propagate crowdsourced dynamics, relay client-provided services and further-clustering clients' interaction. This demands especially arise when a same target resource are used by multiple GeoChannels with different GeoContext conditions that form different GeoServeZones for this same target resource.

For example, an organization sometimes needs to bridge the activities distributed in multiple space/time scopes. For instance, a university has several campuses, suppose a certain internal service, e.g. mobile social networking service is purposely confined on each campus as the GeoServeZone of a GeoChannel, and sometimes there is a need to connect these separate GeoChannels to make cross-campus interaction among students that are distributed on different campuses.

Technically, this kind of demands raise the need to, on one hand, conduct a union of GeoClusters and GeoServeZones of multiple GeoChannels; but on the other hand, still keep the independence of individual GeoClusters and GeoServeZones of GeoChannels.

### **8.3.2.2 The Technical Mechanism for Bridging GeoClusters**

A technique named GeoBridge is designed to meet the demand on union of GeoClusters of multiple GeoChannels, which use a same target resource, for achieving cross-GeoChannel interaction.

GeoCluster supplies gathering and isolating mechanism for bringing together users but grouping them in separate units, while GeoBridge can provide a connecting mechanism for bridging the isolated GeoClusters on a same target resource to form a context-scope bigger GeoCluster, which can facilitate propagating dynamics, relaying services and super-clustering interaction between these bridged GeoClusters. In effect, GeoBridging can result in the union/merge of multiple GeoServeZones and their respective users into a bigger GeoServeZone and GeoCluster.

The GeoBridge mechanism mainly involves two aspects of tasks. Firstly, to declare the GeoCluster-bridging instruction in each of AccessFeeds of GeoChannels to be bridged, and then

the common GeoCluster Service facility can create/maintain these individual GeoClusters and manage their cross-GeoCluster communication according to the bridging-instruction declared in their AccessFeeds. The following exemplifies the GeoBridge mechanism.

Firstly, the Listing 8-2 gives the example of three AccessFeeds (simplified for brevity), which mutually declare their agreement on GeoClusters union on a same target resource.

Listing 8-2: An example of some AccessFeeds that mutually declare the GeoBridging of GeoServeZones/GeoClusters on a same target resource

Listing 8-2-a: AccessFeed<sub>1</sub>

```
<AccessFeed src="www.accessfeed-site1.com/accessfeed-1.kml">
  <Accessibility>
    <Accessor accessor_ID="dfgc85">
      <GeoPolicy>...</GeoPolicy>
      <href>www.example-site.com/a.html</href>
      <bridgeWith>
        geoww://www.accessfeed-site2.com/accessfeed-2.kml::[accessor:dfg86]/:www.example-site.com/a.html
      </bridgeWith>
      <bridgeWith>
        geoww://www.accessfeed-site3.com/accessfeed-3.kml::[accessor:dfg87]/:www.example-site.com/a.html
      </bridgeWith>
    </Accessor>
  </Accessibility>
</AccessFeed>
```

Listing 8-2-b: AccessFeed<sub>2</sub>

```
<AccessFeed src="www.accessfeed-site2.com/accessfeed-2.kml">
  <Accessibility>
    <Accessor accessor_ID="dfgc86">
      <GeoPolicy>...</GeoPolicy>
      <href>www.example-site.com/a.html</href>
      <bridgeWith>
        geoww://www.accessfeed-site1.com/accessfeed-1.kml::[accessor:dfg85]/:www.example-site.com/a.html
      </bridgeWith>
      <bridgeWith>
        geoww://www.accessfeed-site3.com/accessfeed-3.kml::[accessor:dfg87]/:www.example-site.com/a.html
      </bridgeWith>
    </Accessor>
  </Accessibility>
</AccessFeed>
```

Listing 8-2-c: AccessFeed<sub>3</sub>

```
<AccessFeed src="www.accessfeed-site3.com/accessfeed-3.kml">
  <Accessibility>
    <Accessor accessor_ID="dfgc87">
      <GeoPolicy>...</GeoPolicy>
      <href>www.example-site.com/a.html</href>
      <bridgeWith>
        geoww://www.accessfeed-site1.com/accessfeed-1.kml::[accessor:dfg85]/:www.example-site.com/a.html
      </bridgeWith>
    </Accessor>
  </Accessibility>
</AccessFeed>
```

```

        <bridgeWith>
geoww://www.accessfeed-site2.com/accessfeed-2.kml:[accessor:dfg86]/:www.example-site.com/a.html
        </bridgeWith>
    </Accessor>
</Accessibility>
</AccessFeed>

```

As described in Chapter 5 about AccessFeed information model, and exemplified at Listing 8-2 in this section, the `<bridgeWith>` element is for declaring the union of GeoClusters. An `<Accessor>` element can contain multiple `<bridgeWith>` elements, each of which indicates the GeoBridging of a pair of GeoClusters, i.e. between this GeoChannel resource's GeoCluster defined in this AccessFeed and another GeoChannel resource's GeoCluster that may be defined by its own AccessFeed, which also needs to correspondingly declare its GeoCluster bridging with this GeoCluster. Two GeoClusters must have bilateral declarations for their union/GeoBridging. For example, the AccessFeed<sub>1</sub> in Listing 8-2-a declares its GeoCluster's GeoBridging to those GeoClusters that are defined in AccessFeed<sub>1</sub>, AccessFeed<sub>2</sub> and AccessFeed<sub>3</sub>, each of which reversely declares its GeoCluster's GeoBridging to the GeoCluster of AccessFeed<sub>1</sub>. In this way, every pair of GeoChannel resources (that use the same target resource) mutually make an agreement on their GeoClusters' union. It is noted that, bilateral declarations are required for the security of information that are passed across GeoClusters. Unilateral declaration is invalid and will be ignored by the GeoAccessFeed processing system. Now the GeoCluster<sub>1</sub>, GeoCluster<sub>2</sub> and GeoCluster<sub>3</sub> have been declared for GeoBridging.

With the GeoBridging instructions declared already, then the following will explain the technical process for executing GeoClusters union and cross-GeoCluster communication. The common GeoCluster Service as a common service facility works for automatically and dynamically creating and maintaining GeoClusters for all GeoChannel resources, and for inter-client communication. When GeoChannel clients (which are required to do so in a random or regular way) inform it of their status about access-condition matching with their interested GeoChannel resources, the communication messages contain the Geoww-URIs information that may have attached addition information such as the GeoBridging declarations. For example, a client of the AccessFeed1 shown in Listing 8-2-a can report a message containing the following message to the GeoCluster Service:

```

Geoww://www.accessfeed-site1.com/accessfeed-1.kml:[accessor:dfg81]/:www.example-site.com/a.html
<bridgeWith>geoww://www.accessfeed-site2.com/accessfeed-2.kml:[accessor:dfg86]/:www.example-site.com/a.html</bridgeWith><bridgeWith>geoww://www.accessfeed-site3.com/accessfeed-3.kml:[accessor:dfg87]/:www.example-site.com/a.html</bridgeWith>

```

So the GeoCluster Service can know the latest declarations about GeoClusters union for GeoChannels. It can register and update the metadata of GeoClusters with their latest GeoBridging declarations, via which a GeoBridging network graph can be created and updated to represent the GeoBridging relations between all GeoClusters that are dealt with by the GeoCluster Service. Figure 8-5 shows the GeoBridging relations that are declared by those AccessFeeds exemplified in Listing 8-2.

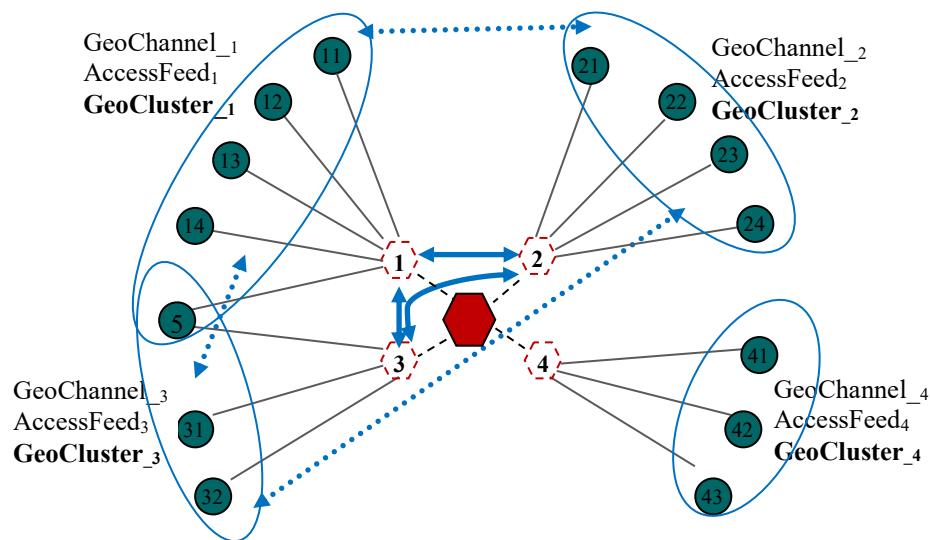


Figure 8-5: GeoBridge for union of multiple GeoClusters on a same target resource

Cross-GeoCluster connections mediated by the GeoBridge mechanism can form client networks with both inner-GeoCluster and cross-GeoCluster client relations, as illustrated in Figure 8-7 which is in contrast to the inner-GeoCluster client networks as shown in Figure 8-6.

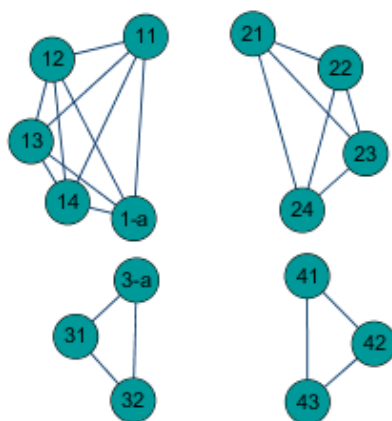


Figure 8-6: The example of client networks within GeoClusters

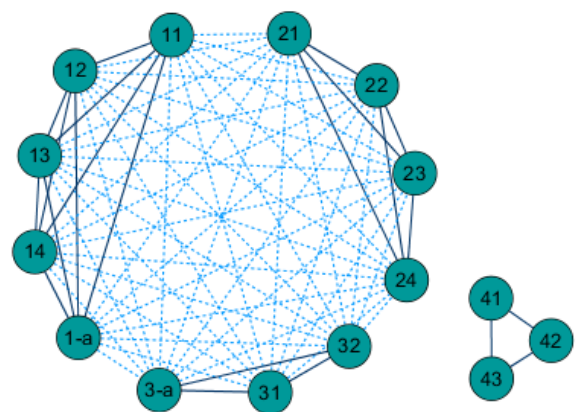


Figure 8-7: The example of cross-GeoCluster client networks enabled by the GeoBridge mechanism

There is no need to change or re-program the target resource (application/service) for supporting GeoBridging GeoClusters. As the normal procedure for processing inner-GeoCluster clients' interaction described in Section 8.3.1, the target resource node still delivers its client's interaction information, even if it is actually for a cross-GeoCluster interaction, to its own GeoCluster that hosted on the GeoChannel Service server. The GeoChannel Service can identify cross-GeoCluster interaction messages by checking their destination Geoww-URIs, and then can dispatch a message to its destination GeoCluster(s). Figure 8-7 illustrates part of the GeoBridging network graph involving the nodes ①, ② and ③ that represent GeoCluster\_1, GeoCluster\_2 and GeoCluster\_3 respectively, and shows the cross-GeoCluster messaging relations.

Actually, GeoBridge is a flexible, easy and dynamic mechanism for organizing/re-organizing a new ad-hoc GeoServeZone/GeoCluster unit from discrete units of originally separate GeoServeZones/GeoClusters, without impairing their autonomy and independence. By modifying the AccessFeeds that declare GeoBridging instructions, a union of GeoClusters can be dynamically changed, without a need to re-program the target resource application/service.

## **8.4 Federation of Geo-Compatible GeoChannel Resources**

The techniques of AccessFeed, GeoCluster and GeoBridge, as designed heretofore, can conduct client-resource and client-client dynamic correlation. Now this section describes the mechanism for achieving resource-resource dynamic correlation for federating resources based on GeoContext conditions, that is, for Geo-Federating GeoChannel resources.

Resource federation refers to the integration or cooperation of Geo-Compatible resources via a client-side Accessor Bus to achieve composite functions or services of these individual GeoChannel resources. Geo-Compatible resources refer to those GeoChannel resources whose access-control GeoPolicy conditions can be simultaneously met by a client's GeoContext. That is, these resources' GeoServeZones have overlaps where a client can fall to access these resources simultaneously. A client accesses resources via these resources' Accessor programs. When these Geo-Compatible resources' Accessor programs are concurrently alive on a client-side Access Bus, there is the potential or opportunity for interoperation between these resources, if their Accessors have been programmed for interaction to provide this client with certain integrated services/functions.

The Listing 8-3 and Figure 8-8 exemplify the concepts and process of Geo-Compatibility and Geo-federation of GeoChannel resources. The Listing 8-3 gives an AccessFeed example.

Listing 8-3: An AccessFeed example for illuminating Geo-Compatibility and federation of GeoChannel resources

```
<AccessFeed src="www.accessfeed-site1.com/accessfeed-1.kml">
  <Accessibility>
    <Accessor accessor_ID="A">
      <GeoPolicy>...</GeoPolicy>
      <href>www.example-site.com/A.html</href>
    </Accessor>
    <Accessor accessor_ID="B">
      <GeoPolicy>...</GeoPolicy>
      <href>www.example-site.com/B.html</href>
    </Accessor>
    <Accessor accessor_ID="C">
      <GeoPolicy>...</GeoPolicy>
      <href>www.example-site.com/C.html</href>
    </Accessor>
    <Accessor accessor_ID="D">
      <GeoPolicy>...</GeoPolicy>
      <href>www.example-site.com/D.html</href>
    </Accessor>
  </Accessibility>
</AccessFeed>
```

The Figure 8-8 illustrates the changing process of federation of GeoChannel resources.

The AccessFeed in Listing 8-3 Geo-Contextualizes four GeoChannel resources that have respective access-control GeoPolicy conditions, which form their GeoServeZones denoted as A, B, C, and D as shown in Figure 8-8, where some clients and time points/periods are depicted. The pictures of red-colour geometry shapes represent the GeoServeZones relations status of these resources, and clients' access status, at different time points. The middle part of Figure 8-8 shows resources federation status on different clients (Accessor Bus). It is noted the federation relations shown here just imply the related resources can have space/time opportunity to be federated, but are not necessarily be federated. The bottom part of Figure 8-8 depicts the total resources federation relations, i.e. the resources network bridged by these clients at different time points/periods.

This example just gives the scenario of one AccessFeed with three clients for organizing and federating the four resources. In practice, this AccessFeed may have more clients which can also federate/bridge the four resources, and there may be other AccessFeeds which organize resources that also involve some of all the four resources organized in this AccessFeed, consequently, that practical scenario may result in a more extensive resources network involving more clients that bridge more GeoChannel resources for their federation.

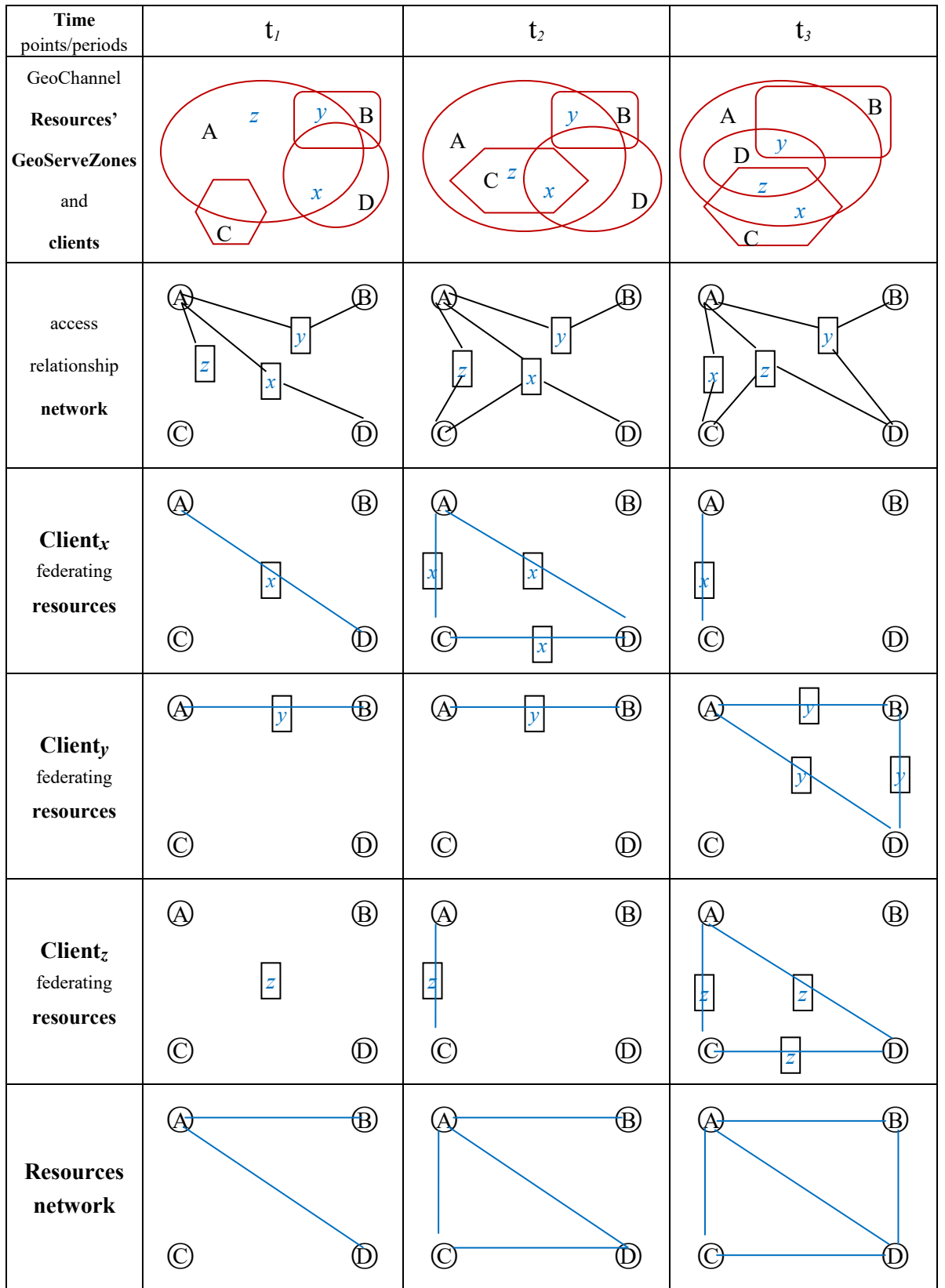


Figure 8-8: An example of federating Geo-Compatible resources on different clients at different time points/periods and forming a network of resources

This mechanism of client-bridging-resources can be referred to as GeoFederating resources, as opposed to GeoClustering clients (i.e. resource-bridging-clients) elaborated in last section.

Some information and functional points about GeoFederating resources are worth noting as below:

- AccessFeed as organizer and Accessor Bus as connector

Resource federation basically involves the target resources that are referenced within an AccessFeed, which functions as an organizer of an integrated application/service that uses these resource modules. This is in contrast with a GeoCluster that is an organizational unit for clients' interaction. An AccessFeed Geo-Contextualizes the resource modules by specifying their access-control conditions, which can trigger the activation/deactivation of these resources' Accessors to achieve the purpose of conditionally assembling and tailoring resource modules at proper space/time contexts.

Client-side GeoChannel Accessor Bus is the connector for communication between active Accessors of resource modules. This is in contrast with server-side resource nodes or common GeoCluster Service for bridging client-client interaction within or across GeoClusters. Accessor Bus can mediate the communication between the Accessors of Geo-Compatible resources for loosely coupling their functions.

There is usually a need for coordinating federation of resource modules. Usually a common practice is to, among these target resources (modules) organized within an AccessFeed, have a main/coordinator resource module that can organize the interoperation between all these modules. This coordinator resource usually remains GeoCompatible with other resources in this AccessFeed, thus it can keep active/running so as to manage other resources' activities. For example, the Resource<sub>A</sub> shown in Figure 8-8 functions as the coordinator role, which has a broad GeoServeZone that can overlap other resources, so mostly it can federate with any other resource modules in this AccessFeed.

- Dynamic resource-resource correlation: resources Geo-Compatibility and clients as bridges

Geo-Compatibility between resources is usually dynamically-changeable. GeoChannel resources can become Geo-Compatible or Geo-Incompatible over time. As each resource's access-control GeoPolicy condition and its GeoContext may change, its GeoServeZone then changes correspondingly. These changes may lead to the changes of relations between these resources' GeoServeZones, which may become overlapped or disjoint, as exemplified in Figure 8-8 that depicts the changing process of resources GeoCompatibility at different time point/periods.



Geo-Compatible GeoChannel resources can be bridged for their federation via a client's Access Bus. As shown in Figure 8-8, at a specific time point, the possibility for federating resources may be different on different client side. Only there is a client located in the overlap of two GeoServeZones can the two Geo-Compatible resources have opportunity for federation. In this sense, a client functions as a bridge role.

So, both requirements must be concurrently met for resource federation: being mutually Geo-Compatible and having common client(s). Two/multiple GeoChannel resources, even if they become Geo-Compatible sometimes, but if there is not a common user for bridging these resources, they cannot be federated at this time. For example, as illustrated in Figure 8-8, at the time  $t_1$ , the resource B and resource C are Geo-Compatible, but they cannot interact with each other at this time point/period as there is not a common client (Access Bus) for bridging their interaction.

Consequently, dynamic resource-resource correlation is affected by the dynamic GeoContexts status of both resources and clients, as resources' GeoContexts affect their mutual GeoCompatibility, and clients' GeoContexts affect their roles for bridging GeoCompatible resources.

- Mashing up pervasive services: the actual implication of GeoFederating resources

In this section (Section 8.4) the intention of analysing the process about GeoFederating resources is to find out related technical mechanism that can be used on practical applications. Now the mechanism identified here for conducting dynamic resource-resource correlation is useful for organizing and mashing up pervasive (e.g. location-based) service resources.

Some points that can guide practice can be outlined as following:

- 1) Crowdsourcing mashable (GeoChannel) resources

Crowdsourcing can achieve massive sets of resources involving various context coverages and functional categories. A resource may be a pure data set such as a KML file, or a Web application program which is architected using the GeoChannel program framework that basically consists of the components of a service node and an Accessor. The Accessor exposes APIs for the interoperation with other resources' Accessors via the Accessor Bus of a client.

This resource may be GeoContextualized by an AccessFeed (as described in the step 2 below) for specifying its access-control GeoPolicy condition, or may be an unconditionally-accessible

resource. These crowdsourced resources are deployed on the Web for being mashed up by their clients.

## 2) AccessFeeds for organizing mashups for an integrated service (application)

For an integrated pervasive service application, an AccessFeed is defined for organizing multiple target resource modules (described in Step 1 above) with respective GeoPolicy conditions to arrange context opportunities for their activation/deactivation and interoperation, based on resource GeoFederating mechanisms. There may be a need to program an application-specific organizer module for coordinating the interoperation of these mashed up resources. The GeoFederating and coordinator mechanism have been identified in this Section 8.4 heretofore.

## 3) Using pervasive services, engaging in participative and cooperative activities

A pervasive service (resource) can be discovered via the physical-world or virtual-world approaches, to get its AccessFeed, which can automatically and dynamically control the running of this pervasive application to aggregate and interoperate GeoContext-aware resources for cooperative services. Users are dynamically grouped into the GeoClusters of different resources for inter-clients interaction, and Geo-Compatible resources bridged by clients can federate their functions/services. This kind of GeoChannel network with resource and clients based on client-resource, client-client and resource-resource dynamic correlation can supply a GeoContext-aware environment for clients to engage in participative and cooperative activities.

# 8.5 Discovery & Navigation of Resources & Users

Chapter 7 had designed the technology of metadata-querying GeoChannel discovery by searching resources catalogue based on querying keywords or context features against GeoChannel metadata, and now the GeoChannel network technology can provide another mechanism for discovering and navigating GeoChannel resources or users of interest. This approach is called neighbour-querying discovery.

Figure 8-9 illustrates the principle of the neighbour-querying discovery mechanism. Figure 9-a is the actual resource-user network topology. Originally resource nodes are not connected directly, but are bridged by some user nodes. A resource node can query each of its current users, which can inform this resource node about other resources that this user is accessing currently. Then a resource node can know the existence of other resource nodes, named

neighbour resources/nodes that this resource's users are connected with. In this way, in a GeoChannel network all resource nodes can know the existence of their respective neighbour resource node(s). Between any two inter-neighbour resource nodes is one or multiple paths. Situated on every path is a user node that bridges the two inter-neighbour resource nodes. This neighbour-querying mechanism can result in a GeoChannel node network as shown in Figure 9-b. For example, now the resource node B knows its neighbour nodes A, C, D and E. Only one path with the User node No. 13 bridges B with D, but there are two paths respectively for B-A, B-C and B-E.

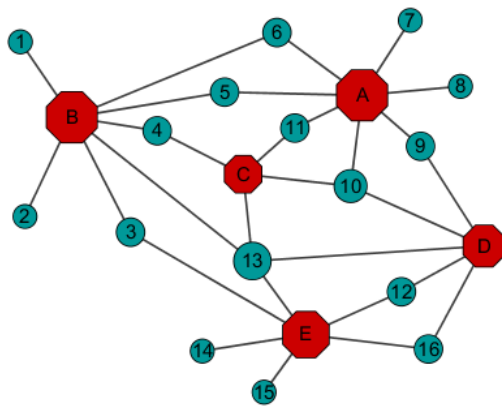


Figure 8-9-a: An example of GeoChannel network with connected nodes of users and resources

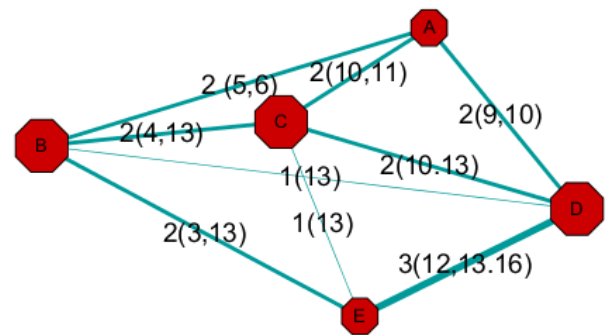


Figure 8-9-b: An example of GeoChannel network with GeoCompatible resources bridged by clients

Figure 8-9: An example of GeoChannel network for discovering and navigating resources

Every resource node can dynamically keep a local record of the existence of its neighbour resource nodes, and inter-neighbour resource nodes can further on-demand query opposite-party's local record information to learn about more nearby resource nodes. A user can query its connected resource node(s) to discover other GeoChannel resources, and a resource node can also actively recommend its locally-recorded information with corresponding "Popularity" metrics to its current users. For example, now the user No. ① connected with resource B can know the existence of resource A, C, D and E.

In some sense, this discovery mechanism is somewhat like goods-recommending technique for online shopping which, based on past customers' purchase behaviour records, may suggest or prompt "other customers buying this product had ever also taken some other goods like ...". And in terms of navigating users through the Web, there is also similarity to the Document Web which uses inter-linked document mechanism for guiding users. Now a GeoChannel network can navigate users to find and access resources networked.

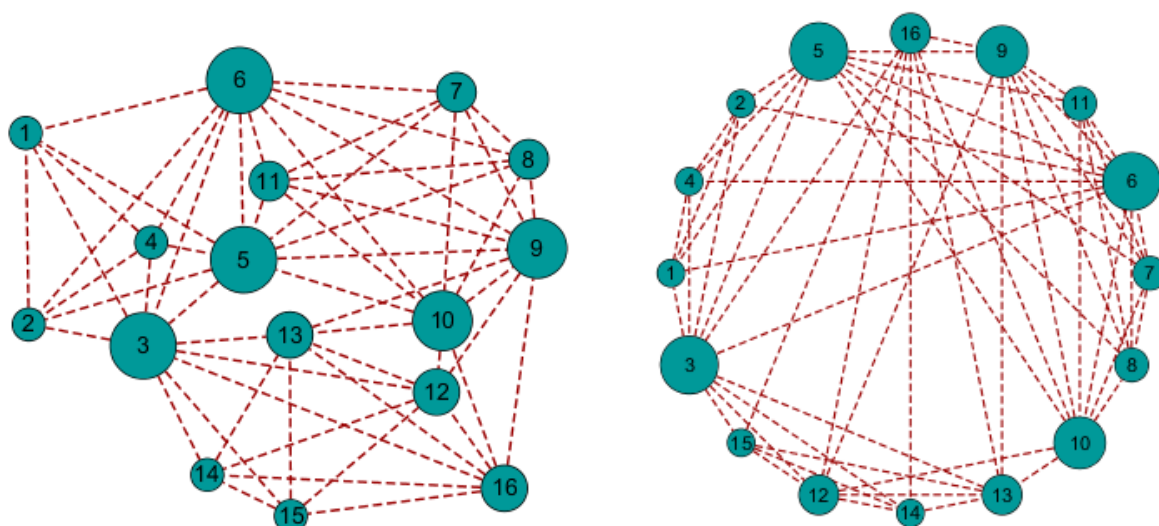


Figure 8-10: An example of GeoChannel network for discovering and navigating clients

Discovering and navigating clients can also use the neighbour-querying discovery mechanism. Figure 8-10 illustrates the client-client relationships bridged by resource nodes on the GeoChannel network that had been exemplified in Figure 8-9-a. The two parts in Figure 8-10 (on the left and right) show the same client-client relationships in different views to facilitate examining their connections. Firstly, every client (user) can query its GeoCluster about the existence of other clients in the same GeoCluster, namely neighboured clients. A client that belongs to multiple GeoClusters can learn about its neighboured clients distributed in these GeoClusters. Secondly, a non-cross-GeoCluster client can learn from the cross-GeoCluster client about the existence of the clients in other GeoClusters. So this querying process is based on passing on neighbour information.

Actually, this kind discovery and navigation functionality can be implemented by the GeoCluster Service and the GeoChannel Web Engine Service.

The neighbour-querying discovery and navigation method works with GeoContexts of resources and users. As GeoChannel resources are usually accessed based on GeoContext conditions prescribed in their AccessFeeds, there is usually similarity or proximity on the GeoContexts or functionality of inter-neighboured resources that are bridged by the common user(s). So this neighbour-querying discovery and navigation method has realistic sense for finding and accessing context-similar or functionally-cooperative resources of interest, for example, local popular and hot events, activities and service facilities, etc.

## 8.6 The GeoChannel Web

### 8.6.1 GeoChannel Web: a Global Dynamic GeoChannel Network

The GeoContext-based mechanisms for resources/clients identifying (by Geoww-URIs), access control (by AccessFeeds), GeoClustering (i.e. resource-bridging-clients) and GeoFederating (i.e. client-bridging resources) can conduct client-resource, client-client and resource-resource dynamic correlation, which makes clients and resources dynamically involved into GeoChannel networks as depicted in previous sections. The GeoChannel networks of local access-relation topology can be bridged by their common users/resources when they are across multiple local networks, to form an extensive GeoChannel network on a global scale.

As described in Chapter 2, this research proposes a new concept, the GeoChannel Web for characterizing this global-scale dynamic GeoChannel network. The GeoChannel Web targets mapping and applying dynamic, random and context-based relations between real-world things. Figure8-11 illustrates a conceptual depiction about this massive network of the GeoChannel Web. The GeoChannel Web takes GeoChannels and clients as network nodes, with correlations as the connections making resources and users related each other.

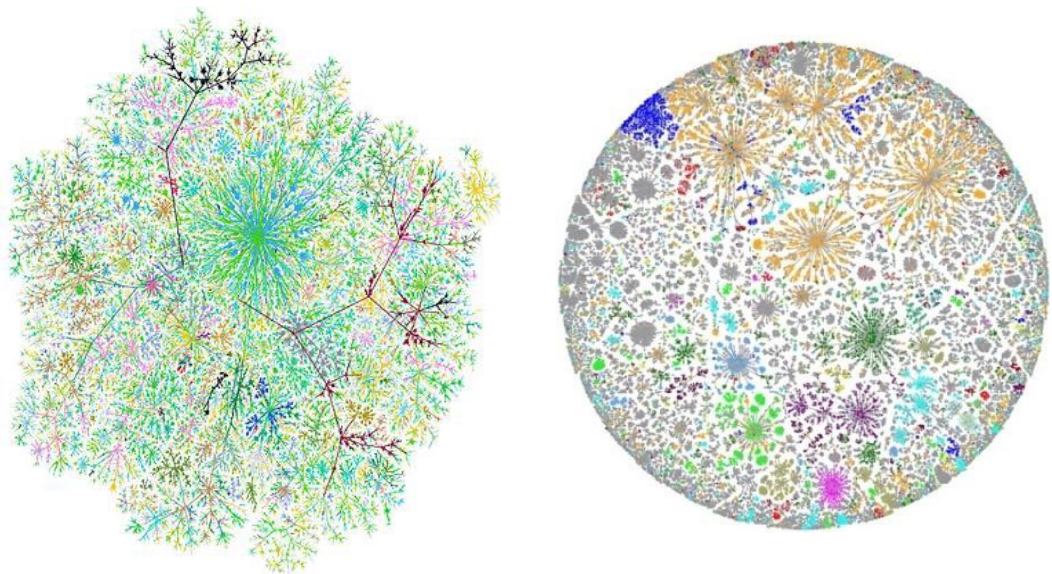


Figure 8-11: A conceptual depiction of the GeoChannel Web networks

Philosophically, the concept and principle of the GeoChannel Web is consistent with the “First Law of Geography” (Tobler 1970), which states "Everything is related to everything else, but

near things are more related than distant things". In the context of the GeoChannel Web, the nearness is represented as the proximity or similarity of GeoContexts (e.g. space/time properties of GeoChannel resources and clients), which affect the relatedness of these resources and clients. The relatedness embodies two scopes. In inner-GeoChannel scope, clients with same or similar GeoContext can be GeoClustered into a GeoChannel, so they are more related. This is GeoChannel-clustered relatedness of users. And in inter-GeoChannel scope, when an user can concurrently participate (be GeoClustered) into multiple GeoChannels, these GeoChannels are more related in terms of their GeoContexts proximity which can let this user meet their respective access conditions simultaneously. This is user-bridged relatedness for GeoChannels. It is the GeoContext proximity or similarity for making the nearness and relatedness that can achieve the usability (as elaborated in Section 8.2, 8.3, 8.4 and 8.5) for discovery, interaction and federation of GeoChannel resources and users.

Technologically, the GeoChannel Web objective and practice can comply with and extend the vision and techniques of establishing a global Social Graph. As the idea of Social Graph envisioned by the currently biggest social networking platform Facebook's founder Mark Zuckerberg (Zuckerberg 2010), if we map out all the connections between people and the things they care about, it would form a graph that connects everyone together. Facebook has focused mostly on mapping out the part of the graph around people and their relationships. At the same time, other web sites and services have been mapping out other parts of the graph so we can get relevant information about different types of things. For example, Yelp (<http://www.yelp.com/>) maps out the best local businesses and Pandora (<http://www.pandora.com/>) maps out which songs are related to each other. All of these connections are important parts of the social graph. And now the GeoChannel Web vision can further enrich and extend the concept and practice of mapping a global graph (network) of people and things. Firstly, GeoChannel can represent and deliver comprehensive resources (i.e. any type of things, e.g. dynamics, services and interaction, etc.) other than the contents (such as web document articles, media content e.g. songs/photos/videos, and products, etc.) that are mainly coped with by current social media. Secondly, as opposed to the relatively fixed or pre-established connection relationships around people dealt with by current social media, the highly-dynamic nature of the connections (triggered by changing GeoContexts and interest) among resources and users can make a truly real-time relationship network for connecting people and things.

And realistically, the global dynamic network of the GeoChannel Web has a notable applicability for mirroring and reflecting the real-world activity-centric social interaction, compared with and complementing current people-centric social networking systems. In the real

world, interactive actions among people and things happen anytime and usually centre on specific activities (events). This activity-based interaction (social networking) usually takes place improvisationally without having to know the real identities of or keep permanent connectivity with the activity or its participants. On the GeoChannel Web network, a resource (activity) as a medium brings together its visitors (participants) which in their turn may link this resource with other resources (activities) being participated in concurrently. So this GeoChannel Web is a naturalistic paradigm for organizing, discovering, federating, correlating and bridging people and things.

## 8.6.2 Features of the GeoChannel Web

This section identifies some characteristics of the GeoChannel Web network, and indicates its implicit association with current GeoWeb (Geospatial Web) technology.

### 8.6.2.1 The GeoChannel Web vs. other Speciality Webs

Actually only one general Web exists physically, but with multiple aliases indicating functional and perspective diversification into speciality Webs, as reviewed in Chapter1, for example, the Resource Web, Social Web, GeoWeb, and now the GeoChannel Web concept proposed in this research. Different speciality Webs feature specific networking mechanisms and technologies for organizing and processing their corresponding Web constituents. By making some comparison with other speciality Webs, we can identify some essential features of the GeoChannel Web.

Table 8-2: The concepts comparison between the general Web, Social Web and GeoChannel Web

	the general Web	the Social Web	the GeoChannel Web
<b>Networking based on (affected by)</b>	relevance/relatedness of resources/functions	friend(follower)ships/memberships	GeoContext (e.g. space/time attributes functioning as network switches)
<b>Addressing/ Identifier</b>	http://www.... ws://www.... etc.	User: e.g. account@ facebook account @Twitter Resource: e.g. http://www....	Geoww-URI scheme, <i>for</i> things, clusters and their relations
<b>Networked elements/content</b>	resources/services e.g. web pages, multimedia content, web services, program code, etc.	relational networks of friends, followers, members, etc., for people and their own content	resources & clients/users with GeoContext (e.g. space/time attributes)
<b>Changes of network/topology relations</b>	general	relatively static, fixed or stable	highly dynamic nature based on context conditions.

<b>Platform dependence</b>	the Internet/Web platform	proprietary platforms/services providers	platform neutral, account-free.
<b>Connecting/ Networking</b>	navigate & link resources (data, applications, services)	social networking relations for connecting people	dynamic correlation between things (e.g. client-resource, client-client, resource-resource)
<b>Technology facility</b>	Web protocols/ techniques/ infrastructure	social graph, media repository, messaging platform, etc.	GeoAccessFeed (AccessFeed, GeoPolicy language, Geoww-URI model, Geo-off-on communication model, AccessFeed Update Service), GeoChannel Web Engine, GeoCluster Service, GeoChannel Discovery Service, GeoChannel Accessor Bus, etc.
<b>Usage</b>	for all general purposes, is the base for all other networking missions	connect people, share interests or status	facilitate achieving on-demand and context-based dynamics, pervasive services and GeoClustered interaction

The Table 8-2 gives some concepts comparison between the general Web, Social Web and GeoChannel Web. The general Web as a generic platform and infrastructure is the base of any other “speciality Webs”. Its name “Web” just reflects its networking mechanism by using hyperlinks (e.g. based on http address) to network all kinds of information/service resources. The Social Web uses social relationships as its networking paradigm. Essentially the current GeoWeb (Geospatial Web) has not introduced any native mechanism for networking, other than employing the general Web’s networking mechanism for linking resources. Apart from geo-tagging and geo-filtering (as reviewed in Chapter 2) for organizing resources based on their geo-references, basically the current GeoWeb does not have its own mechanism for establishing or organizing connection relationships between things/people. Now the GeoChannel Web, namely a GeoChannel-enabled and networked Web, is meant to add the networking functionality to the GeoWeb.

The GeoChannel Web is a complement to existing Webs, but featuring peculiar capability of organizing dynamic relationships of client-resource, client-client and resource-resource by making use of GeoContext conditions. Just as outlined in Table 8-2, a set of new techniques and mechanisms characterize this GeoChannel Web. The following describes some main points.

- Geo-contextual sensitiveness and dependency

The topology of GeoChannel Web network is highly sensitive to and dependent on the GeoContexts (e.g. space/time attributes) of both GeoChannel resources and users. GeoChannels are usually contextualized resources (activities) by the GeoAccessFeed technology. In one



aspect, usually a user needs to meet prescribed context condition to connect to a resource to become a GeoClustered user in this local GeoChannel's users network. And in another aspect, only if two (or multiple) GeoChannel resources are geo-compatible, i.e. their accessing conditions can be simultaneously met by a same user, can these GeoChannel resources be bridged by this user to form an expanded GeoChannel network.

In terms of geographic attributes, the Resource Web and Social Web have no intrinsic space/time dependency for connection relationships between resources or between people. The GeoWeb is essentially sensitive to spatial attributes for organizing geo-referenced information; however, the current GeoWeb does not have inbuilt mechanism for bridging resources and people.

- The “**geoww://**” URI scheme for GeoChannel Web resources and Users

A new URI scheme, “**geoww://**”, has been designed in Chapter 6 for the GeoChannel Web. As opposed to the general Web “**http://**” URI scheme that is just a syntax format for identifiers or locators for Web resources, this “**geoww://**” URI scheme works in conjunction with the GeoAccessFeed technology for identifying, locating and organizing (e.g. selecting and access-controlling) Geo-Contextualizing resources (e.g. information, applications, services, activities, interaction, etc.) and users. So by analogy with the “**http://**” URI scheme working for the general Web, the GeoChannel Web will feature this “**geoww://**” URI scheme, which will become a native and integral technical component for the GeoChannel-enabled and networked Web.

- Network constituents: resources, users and relationships (e.g. access, GeoCluster, GeoBridge, GeoFederate, etc.)

Current general Web originated from the Document Web which is a system of interlinked hypertext documents. This linking method uses the anchors (hyperlinks) on web pages for organizing, discovering and navigating resources. So interlinked resources are the constituents of network nodes of the Document Web. The Social Web focuses on the connections between people for social networking, so people with friend relationships are the elements of Social networks. Basically the current GeoWeb centres on geospatial information and services, but does not weave users into GeoWeb networks.

From the perspective of constituents, now the GeoChannel Web incorporates resources (activities), users (people) and relationships (e.g. access, GeoCluster, GeoBridge, GeoFederate,

etc.) into a global network as its constituents of nodes and connections. A resource as a medium for clustering and bridging its users, and a user accessing multiple resources can correlate these resources. In some sense and to some extent, this networking mechanism is a fusion of the Document Web and the Social Web for connecting resources and people, and also takes the geospatial sensitiveness of the GeoWeb. Differing from the users of the Document Web and the GeoWeb, the users on the GeoChannel Web become integral components for bridging local pieces of GeoChannel networks into a global network. As elaborated in Section 8.2, the users (especially those who connect to multiple GeoChannel resources) play a role of expanding network topology. People as clients can also act as the providers of resources and services thanks to the symmetrical (real-time and bi-directional) communication framework. So the network nodes (either GeoChannels or people) are both providers and consumers of resources and services. As such, access relationships as the connections of nodes are substantially weave people into this GeoChannel Web network, where a GeoChannel node can cluster users, and a user node can bridge GeoChannel nodes.

- Dynamic network

From a perspective of time, a highly dynamic nature pertains to the topology of the GeoChannel Web network, whose connectivity between nodes (of resources and users) changes randomly and maybe frequently. As illustrated in the “user-bridged GeoChannel network” example in Section 8.4, when a user begins or terminates the access to a resource, the user-resource connection can be established or destroyed, and the change of this connection affects the reachability of communication between this user with other users GeoClustered on this same resource node, and may affect the connectivity between this resource node with other resource nodes in the case this user is a common visitor to these multiple resources.

So the GeoChannel Web is an ad hoc dynamic network. In contrast, the connection relationships are relatively fixed or stable on the Document Web (hypertext links) and on the Social Web (social links).

It is noted that, the GeoChannel Web is a complex system of massive pieces of networks, some of which may become attached to or detached from others anytime. A specific user or resource may become an isolated node when it does not access any resource or does not be accessed by any user. So a user or resource usually gets on/off the GeoChannel Web network dynamically.

- Automatic evolution

From an administrative perspective of topology, the GeoChannel network evolves (changes) automatically and distributedly for self-organizing and self-assembling, without an administrative or central role for controlling this network.

The dynamical and automatic process of topology evolution happens endlessly at two levels. The connectivity between network nodes is up to the interest/intention of a user that may or may not access a resource, and is determined by the GeoContext condition defined by a resource's AccessFeed, which provides a manner for automatically processing connectivity. At a resource level, the GeoAccessFeed technology can enable a GeoChannel resource to GeoCluster its users. A user running an AccessFeed of a GeoChannel resource can be automatically engaged or disengaged (i.e. connected or disconnected) into/from this resource node when this user begins to match or unmatch the GeoPolicy condition defined in this AccessFeed. And at cross-resource level, multiple resource nodes can be automatically bridged or detached when a same user begins to match or unmatch the conditions of their AccessFeeds.

By contrast, general users cannot change Document Web hyperlinks which are managed by the authors of web pages. Friend relationship connections on social networks can be managed by users' manual updates.

#### **8.6.2.2 GeoChannel-enabled & -networked Web**

The Internet is a worldwide communication and computing infrastructure, built on which the general Web (World Wide Web) is a global application layer platform for various application domains. As identified in last section, the Document Web is based on hyperlinked web pages which are connected resources, and social networks are centred on people's connections via Social Graph (Zuckerberg 2010) for recommending friends and for interaction. However, the current GeoWeb seemingly does not have the native nature of "inter-linked objects" that the original Web (i.e. Document Web) features, as the current GeoWeb basically works for dealing with geo-referenced information, which has its peculiar self-organizing capability. Spatial information objects can have their absolute earth geographic coordinates that can directly be used to facilitate discovering (searching such as querying, filtering and retrieving ), presenting (e.g. overlaying on maps) and navigating. So the current GeoWeb does not employ a relative-relationship mechanism of inter-linking objects in the way that the Document Web and the Social Web deal with documents or people. This self-organizing capability of geo-referenced information for the GeoWeb works in the scenarios of general resource-centric

geospatial applications.

Web applications are for serving people eventually, so more cooperative and personalizable ways for building and utilizing the Web is on demand. The surge of the social media phenomena is just a proof. People are shaping how information moves through their connections and are increasingly discovering information from the people and things they care about (Zuckerberg 2010). The big vision (as described in Section 8.6.1) of mapping out a global graph (network) of people and things drives new technological strategy for providing more social and personalized functionality and experiences.

Now the GeoChannel Web concept introduces new strategy and technology for building and utilizing a new type of Web. The GeoChannel Web can be briefly interpreted as GeoChannel-enabled and networked Web. The terming of the GeoChannel Web concept in this research is mainly for characterizing and highlighting this GeoChannel architecture for organizing and connecting things (e.g. resources and people) in dynamic contexts on a global scale. Two aspects of technological strategies and components are included for extending the current Web.

1) **GeoChannel-enabled:** Using GeoChannels as working units can organize, broker and interact with resources and people. GeoAccessFeed technology works for Geo-contextualizing resources and for Geo-Clustering users. These technical facilities can bring the capability of delivering comprehensive dynamics, pervasive services and participatory interaction into the general Web.

2) **GeoChannel-networked.** GeoChannel networks are resource networks plus user networks. As a complement to the self-organizing capability of geo-referenced information in the GeoWeb, a new capability of networking resources and people is added. People (users) now become integral constituents other than resources that current GeoWeb largely focuses on. In a highly dynamic and automatic way, a resource can GeoCluster users and a user can bridge multiple resources to form and change a GeoChannel Web network. This mechanism can result in a real “webbing”, i.e. inter-connecting capability for resources and people that is absent in current GeoWeb. As elaborated in Section 8.6.2.1, the networking capability can bring an array of functionalities, for example, propagating dynamics, relaying services, super-Geo-Clustering interaction, new ways for discovering and navigating resources and users, context-organized social networking, and integrating and interoperating geo-compatible resources across GeoChannels, etc.

*To sum up*, the new term GeoChannel Web refers to GeoChannel-enabled and -networked Web. The fundamental implication of the concept, technology and practice of the GeoChannel Web is it aims to map out and reflect the real-time and dynamic connection relationships among things and people with their specific GeoContexts. In this sense, the GeoChannel Web can mirrors and assists the real-world activities and interactions. This introduces new strategies, mechanisms and facilities for building and utilizing a more discoverable, sociable, cooperative and interactive Web for both resources and users.

## **8.7 The Full Architecture Model**

This section presents the full model of the GeoChannel Architecture, which comprises all the functional components elaborated heretofore. They connect and work jointly becoming a new type of infrastructure for the Web. As a whole this GeoChannel Architecture forms a Context-Organize Layer added into current Web layer stack.

### **8.7.1 Bring All Together: the Complete Architecture**

Chapter 2 had presented an overview of this GeoChannel architecture in term of the demands on functional module structure. From Chapter 3 to Chapter 8 all these architectural components with respective facilities and mechanisms have been designed in detail. Now it is time to assemble them into a complete GeoChannel architecture model, which is illustrated in Figure 8-12.

#### **8.7.1.1 The Full View with Connected Components**

The top and bottom parts in Figure 8-12 represent traditional Web application elements: resources and clients (users), between them consist of all the functional components newly designed by the GeoChannel architecture which, for clear interpretation purpose, are represented as twelve tiers as marked in Figure 8-12. Work flows (i.e. information, control, etc.) make these components connected and integrated into a complete system. It is noted that Figure 8-12 illustrates the fully possible connections between architectural components, while an actual GeoChannel application may only have part of these workflows.

The GeoChannel architecture model involves server-side and client-side components, which can basically be grouped into resource-organize facilities (including ①, ②, ③, ⑦), access-organize facilities (including ④, ⑤, ⑥, ⑧, ⑨, ⑩), and access-interact facilities (including (11), (12)), as illustrated in Figure 8-12. The following mainly illuminates their

workflow relationships, without re-describing individual components' technical details which can be consulted from corresponding chapters in this thesis.

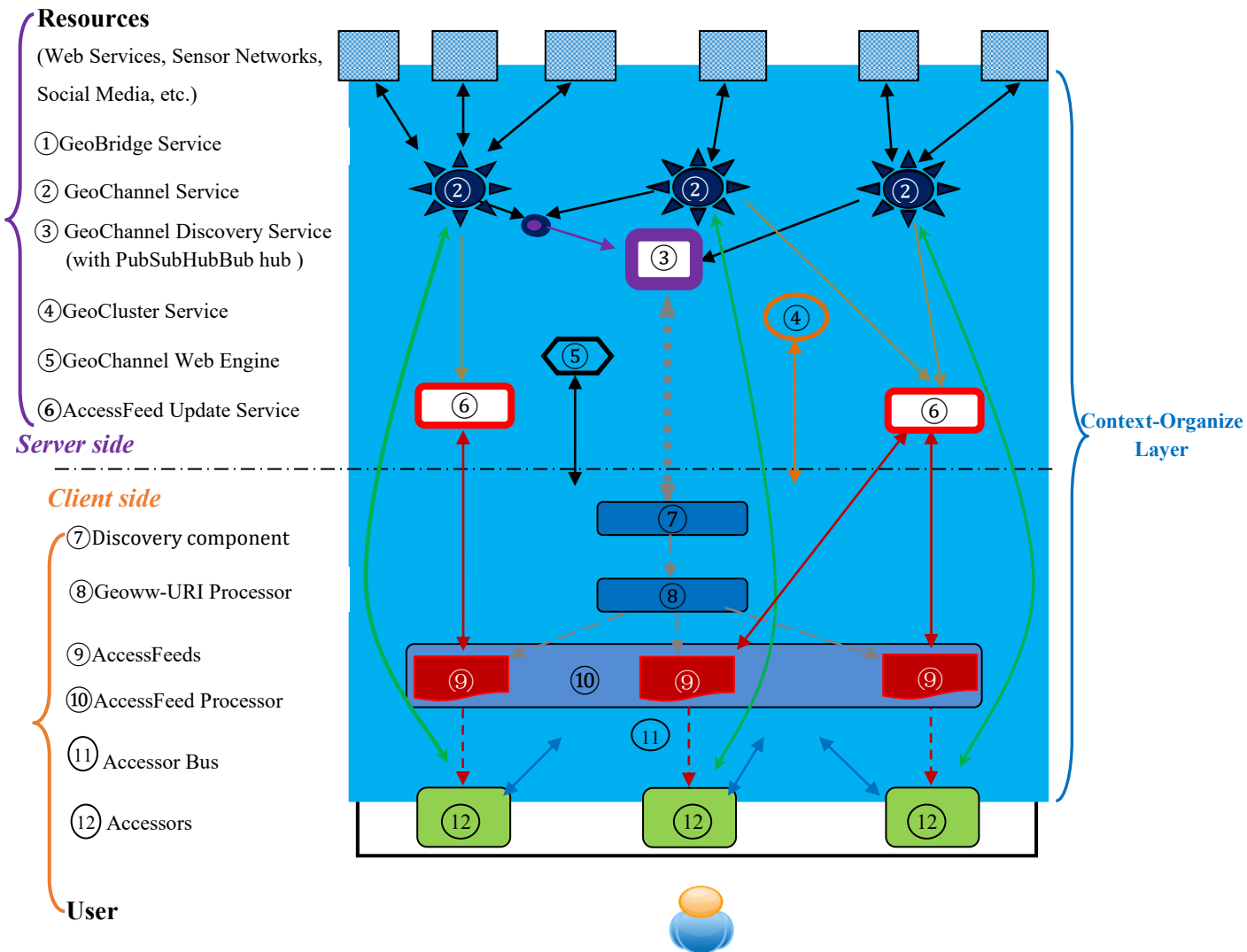


Figure 8-12: A full view for the GeoChannel Architecture Model

### Server-side components:

#### ► GeoChannel Service (at Tier 2)

GeoChannel nodes provide GeoChannel Services that deliver various dynamics, pervasive services and GeoClustered interaction. GeoChannels are the working units for organizing and brokering resource, and for dynamically bridging users. GeoChannel nodes provide a Plug&Play mechanism for assembling and connecting heterogeneous resources and users. GeoChannel nodes can publish their latest discovery metadata and AccessFeed updates to GeoChannel Catalogue Service and AccessFeed Update Service correspondingly, can query

GeoChannel Web Engine for exploring the topology information about GeoChannel networks, and can employ GeoCluster Service for enabling participative and cooperative applications/services or social networking for users. GeoChannel nodes (services) are accessed by GeoChannel Accessors (programs).

► **GeoChannel Catalogue Service** (at Tier 3)

GeoChannels as geo-contextualized resources can be characterized by their metadata (as modelled in Appendix 6). GeoChannel Catalogue Service (designed in Chapter 7) can harvest real-time updates of GeoChannel resources by subscribing to metadata updates from GeoChannel nodes directly, or from PubSubHubBub hubs; and can provide keyword- or space-based query from a client for discovering GeoChannel resources of interest, whose Geoww-URIs are then delivered to this client.

► **GeoCluster Service** (at Tier 4)

GeoCluster Service provides native social networking capability, to facilitate building and running participative and cooperative Web applications/services without having to resort to external proprietary social media platforms for organizing users' interaction. As a common service facility, it gets latest information from clients about their access/participation relationships with resources, to automatically create, maintain (e.g. update) and cancel a dynamic cluster (or namely GeoCommunity) that gathers and bridges the current users for a GeoChannel resource. Any resource identified by a Geoww-URI has its own GeoCluster. The users gathered in a GeoCluster can perform interactive activities such as social networking or participative cooperation. GeoChannel nodes or clients can make use of the GeoCluster Service.

► **GeoChannel Web Engine** (at Tier 5)

This novel backend service facility works for constructing, updating and querying a dynamic topology graph of GeoChannel networks on a global scale, i.e. a complete view of the GeoChannel Web networks. This global topology graph has promising application potential for discovering and federating resources, for geo-diffuse social networking, etc. The GeoChannel Web Engine gets the latest information about user-resource access relationships from client programs, and provides query service to GeoChannel nodes or clients for exploring topology relationships.

► **AccessFeed Update Service** (at Tier 6)

GeoChannel resources may dynamically change their GeoContexts and access conditions, which are represented in AccessFeeds. AccessFeed Update Service harvests the latest AccessFeed from a GeoChannel resource (node), and then disseminates this AccessFeed to this resource's clients.

***Client-side components:***

► **Client-side Discovery Component** (at Tier 7)

This Discovery Component is for discovering GeoChannel resources from virtual and physical world. The virtual-world approach is to search GeoChannel Catalogue Services by using a set of new information models designed in this research; and the physical-world approach to scan/detect QR-code labels, RFID tags or Bluetooth signals from our physical environment. Both approaches are eventually to get Geoww-URIs that identifies GeoChannel resources of interest.

► **Geoww-URI processor** (at Tier 8)

Geoww-URI is a novel URI schema invented in Chapter 6 for identifying geo-contextualized resources. The Geoww-URIs returned from GeoChannel Discovery are interpreted by the Geoww-URI processor, which then fetches target AccessFeed(s). Geoww-URI processor can also work for managing (e.g. bookmarking) and sharing Geoww-URIs of GeoChannel resources.

► **AccessFeed** (at Tier 9)

GeoAccessFeed (as elaborated in Chapter 5) is most crucial technology for the whole GeoChannel architecture. GeoAccessFeed technology mainly includes the technical components of AccessFeed and Geoww-URI models. AccessFeed functions as an access-organize system. AccessFeed provides novel mechanisms for geo-contextualizing resources, dynamically correlating a resource with targeted clients, and Geo-Clustering clients that have common interest and similar GeoContext. AccessFeed information model features GeoPolicy condition language, Geo-off-on communication model and Accessibility control mechanism. A runtime AccessFeed gets real-time updates from AccessFeed Update Service by using the adaptive Geo-off-on communication model, and dynamically controls the accessibility and participability of a GeoChannel to interested clients based on their GeoContexts.



#### ► **AccessFeed Processor** (at Tier 10)

The AccessFeed Processor is a complex facility that fulfils the AccessFeed technology. It implements all the information and working models for GeoPolicy condition language, Geo-off-on communication model and AccessFeeds. It interprets AccessFeed files, turning them into runtime executable units for organizing access to resources and for updating AccessFeeds.

#### ► **Accessor** (at Tier 12)

GeoChannel Accessor is the actual client-side working unit for accessing a GeoChannel resource. A GeoChannel AccessFeed identifies and controls the running of an Accessor, which communicates with a GeoChannel node for accessing resources and for interacting with other GeoClustered users via bi-directional and real-time communication usually.

#### ► **Accessor Bus** (at Tier 11)

Accessor Bus supports mashing up and interoperating GeoChannel Accessors for aggregating and federating resources (e.g. information, services) from multiple GeoChannels, and provides a novel capability for networking Geo-Compatible resources to enable user-bridged GeoChannel networks.

### **8.7.1.2 Transparent & Simple to Users**

This full GeoChannel architecture model, as illustrated in Figure 8-12, involves a set of functional components connected and cooperating in a seemingly complex way, which may bring a concern on the complexity that may affect the practicality of these constituent facilities. However, actually this architectural technology is quite easy for use.

This GeoChannel architecture is modelled in structured components which are modularized, autonomous and loosely-coupled, with normative models for service/function interfaces, information formats and process/work flows. Besides resource-specific components such as AccessFeeds and Accessors, most of the GeoChannel architectural facilities are designed and can be deployed as commonly available service infrastructure. This enables interoperability and availability for these GeoChannel architecture facilities.

Not every GeoChannel resource or access activity needs to involve all these components and the full workflows as manifested in Figure 8-12. Usually these individual components can function separately and independently. For example, an AccessFeed may not change over time and it

evaluates GeoPolicy conditions locally, implying it does not need to work with AccessFeed Update Service and Access-Policy Evaluation Service.

GeoChannel architectural components and workflows are transparent to end users. Actually, an end user only directly interfaces with GeoChannel Accessors, which function similar to general client program modules existed in traditional Web applications, to interact with GeoChannel resources or with other users. All the workflows behind are automatically processed and are not necessarily aware by end users, for example, what GeoChannel nodes are organizing and brokering these resources or activities, what and how an AccessFeed is correlating resources with this client based on latest GeoContext conditions, etc.

As for developers or managers of GeoChannel resources or applications, it is also an easy process for building, organizing and deploying application tasks. They can make use of publicly available GeoChannel architecture infrastructure.

### **8.7.2 Context-Organize Layer: an Abstraction & Infrastructure**

How do these GeoChannel architectural facilities fit into the current Web architecture? This question raises a need to observe this GeoChannel architecture in a whole view. Virtually this GeoChannel architecture introduces a Context-Organize Layer into the current Web fabric, and brings new infrastructure for building and running the GeoChannel Web.

#### **8.7.2.1 A Systematic Perspective & High-level Abstraction**

Essentially, all these architectural components designed in this research can constitute a Context-Organize Layer (as indicated in Figure 8-12), which features dynamic GeoContext as a medium for identifying and organizing resources, users and activities.

In the real world, GeoContext plays an important role for engagement of activities that involve people and things; however, the current Web makes use of GeoContext (e.g. location information) for organizing resources and activities in a very rudimental way. Currently the most practice and usage is geo-tagging/referencing general information objects to facilitate their discovery and filtering for use geographically. Basically this simplex paradigm (i.e. geo-tagging/referencing) caters for static GeoContext and is mostly POI (Place of Interest) based, but falls short of the competence for dynamic correlation of resources and clients, both of which may change their GeoContexts randomly. In addition, other than existing search/filter practice by geo-indexing individual information objects without considering their relative

relationships in terms of access conditions or users' interest, etc., more complex and advanced functionality is expected. For example, to explore real-time access relationships among crowds of resources with users, just as the usage expected in the GeoChannel Network technology, to identify Geo-Compatible resources or similar-minded people for federation or interaction.

As GeoContext attributes or conditions generally exist among the tasks and applications of the GeoWeb as well as sensor networks and social media, a common, unified and reusable architectural system for dealing with GeoContext medium can facilitate building and running context-related applications and services. Now this GeoChannel architectural model just meets the demand of constructing a common intermediate layer for the Web to makes use of GeoContext medium.

A new concept as well as architectural facility, the Context-Organize Layer, is introduced into the Web fabric. This perspective enables a systematic abstraction of GeoChannel architectural components. An advantage of the concept of Context-Organize Layer is to clearly hierarchize the Web stack, without messing up current Web architecture but can work well with it, with introducing an additional layer for contextual discovery, access control, service-client correlation and mashup, and for users' interaction.

This GeoChannel architecture is virtually and mostly an organizing system based on GeoContext of resources and users. This Context-Organize Layer works for organizing, correlating and bridging resource-client access and client-client interaction dynamically and conditionally. These architectural components mainly involve the functionalities:

- **Resource organize:** working for resources federation supported by components: GeoChannel Service (node), Accessor Bus, GeoChannel Catalogue Service, Geoww-URI, AccessFeed, and GeoChannel Web Engine.
- **Access organize:** working for resource-user correlation, by components: Geoww-URI, AccessFeed, and GeoChannel Web Engine.
- **User organize:** working for users' interaction, by components: GeoCluster Service, Geoww-URI, AccessFeed, and GeoChannel Web Engine.

The GeoAccessFeed technology, including AccessFeed (which involves GeoPolicy language, Geo-Off-On communication Model, etc.) and Geoww-URI techniques, is the core and most fundamental facility, which underpins other facility components, for enabling the Context-Organize Layer featuring dynamic GeoContext as organizing mediums.

### 8.7.2.2 A New Infrastructure for the Web

The materialization of this Context-Organize Layer implements the constituent components of this GeoChannel architecture model into functional facilities. These interconnected structural facilities that implement and operate the GeoChannel architectural components comprise a new type of infrastructure which provides services and function for building, operating and utilizing GeoChannel resources and activities.

Firstly, some GeoChannel architectural components can be deployed as common service facilities. For example, GeoChannel (hosting/brokering) Service, GeoChannel Catalogue Service, AccessFeed Update Service, GeoChannel Web Engine and GeoCluster Service, etc. These common service facilities are independent from specific GeoChannels, applications and organizational users, and may open for general public uses. For example, a certain GeoChannel application may freely make use of a hosting service for brokering GeoChannel resources and bridging their users, dynamically publish latest metadata to a catalogue service for exposing this GeoChannel, and connect to an AccessFeed Update Service for updating this AccessFeed to current users. Widely deploying and supplying these common service facilities and making them publically accessible can democratize the GeoChannel technology, especially can cater for the proliferation of crowdsourced GeoChannel resources and applications.

In addition to backend service facilities, frontend technical components also constitute parts of this GeoChannel architectural infrastructure. For example, in client-side environment, Geo-URI processor for interpreting and bookmarking Geoww-URIs, AccessFeed processor for supporting access organization, GeoChannel Accessor Bus for mashing up and interoperating Accessors, etc.

A set of information models, as the key outcomes of this research, make up the core part of this GeoChannel architectural infrastructure. The Geoww-URI model functions as a novel URI scheme for Web addresses for identifying geo-contextualized resources. The GeoChannel Markup Language (as codified in Appendix 5), which is actually a collection and integration of some individual encoding models that are codified in the Appendixes of this thesis, provides normative models for information and processes. For example, GeoPolicy Language, Geo-off-on communication model, GeoAccessFeed model, etc. They are essential constituents for standardizing this GeoChannel architectural infrastructure.

These individual facility components (as exemplified above) can also be fused into current Web fabric, to become new infrastructure for the Web. They can be supplementary to and working

jointly with current social media, sensor networks, SDI (Spatial Data Infrastructure) services, standards and library components such as WMS, WFS, WCS and general Web mapping systems, Digital Earth, GeoBrowsers, etc. This GeoChannel architectural infrastructure works for the GeoChannel Web environment featuring the capability to support comprehensive dynamics, pervasive services and GeoClustered interaction.

## **8.8 A Prototype Implementation: the Geonoon Platform**

Based on the full architecture model as illustrated in Figure 8-12, this research develops a prototype implementation of the GeoChannel architecture with the new technical mechanisms and models designed in previous chapters.

The following sub-sections describe the common software facilities for the GeoChannel platform. Section 8.8.1 identifies the software components and their interactive workflow. Section 8.8.2, Section 8.8.3 and Section 8.8.4 elaborate the front-end and back-end facilities with their software implementation respectively.

### **8.8.1 Components Workflow Framework**

The GeoChannel architecture, from a general overview, basically involves three parts of components, i.e. GeoChannel Platform, GeoChannels (resources), and users. As illuminated in Section 8.6 on the technology about GeoChannel Networks and GeoChannel Web, users are also constituent elements that are woven into the GeoChannel Web (Networks). Users and resources both have GeoContexts, based on which user-resource interaction takes place dynamically and on demand. Users can use the GeoChannel platform to access and mash up GeoChannel resources, participate in social networking, and supply client-provided services.

The GeoChannel platform refers to a set of common architectural facility components for providing and using GeoChannel services. This research gives a specific name, i.e. Geonoon, to this GeoChannel platform. The name Geonoon (Geo-noon) implies context attributes (such as but not limited to space/time) for organizing various dynamics, pervasive services and GeoClustered interaction. In this research work, the Geonoon is a prototype implementation of the GeoChannel architectural technology and facilities, and Geonoon expects to become a public and even commercial product and service working for a global scope in the future, to enable the GeoChannel Web as expected in this research.

The Geonoon platform involves backend and frontend facilities. The backend part is usually

transparent to GeoChannel resources' users, which may not be aware of backend facilities but only interact and interface with the frontend part. This is analogical to current web mapping platforms such as Google Maps/Earth, whose backend infrastructure is not visible to end users. For the convenience of specific indication, here a specific name, i.e. GeoChannel Explorer, is given for referring to the frontend part of this Geonoon platform.

This GeoChannel architectural components framework mainly involves some front-end technical components such as GeoAccessFeed technology and Accessor Bus component, and some back-end common services as depicted at the left part of Figure 8-13. There are associations between these components for joint work.

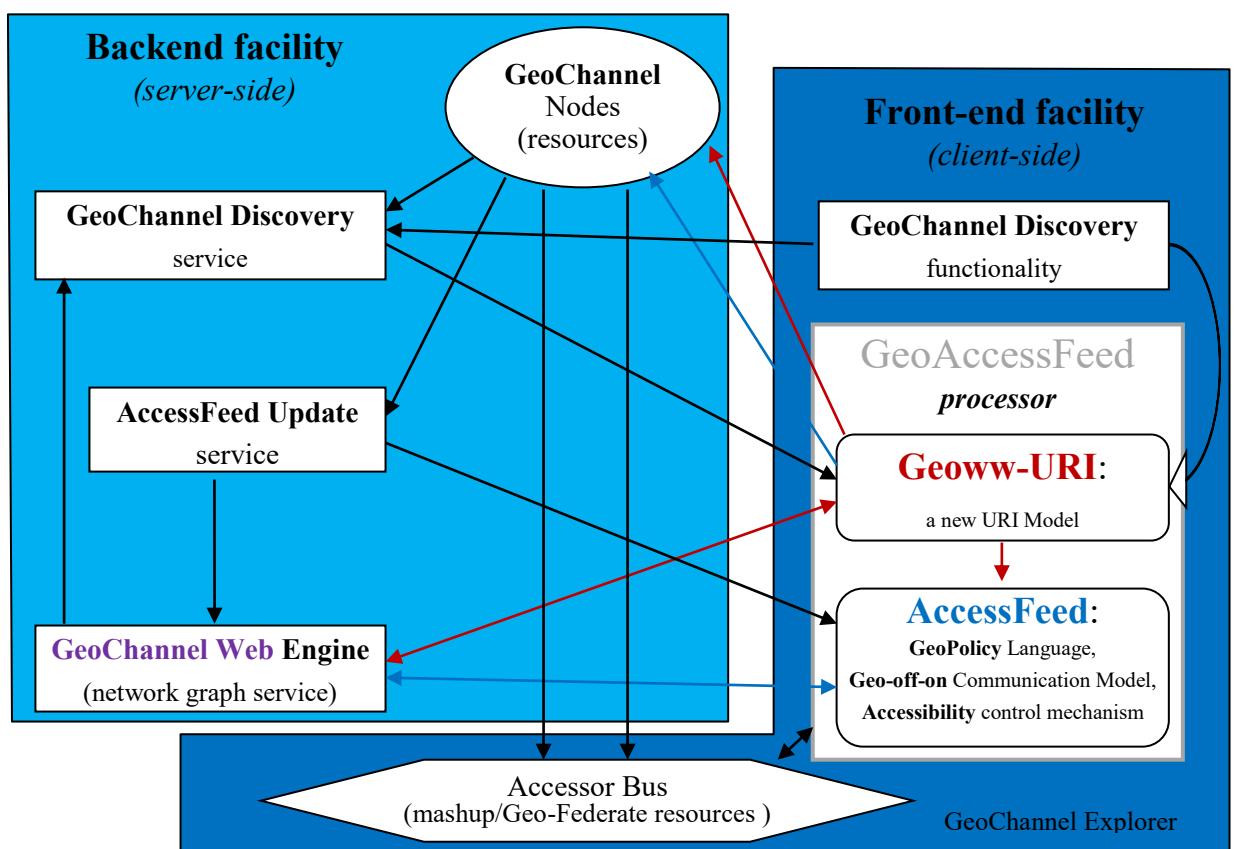


Figure 8-13: The components workflow framework for Geonoon platform

The GeoAccessFeed technology mainly involves the Geoww-URI and AccessFeed techniques. GeoChannel Discovery service/functionality returns Geoww-URIs, which can identify AccessFeeds, GeoChannel resources with GeoContext, GeoClusters, GeoServeZones, and resource/clients nodes on GeoChannel Web (networks). AccessFeeds point to target GeoChannel resources, control GeoClustering users, affect client-resource relationships on GeoChannel networks, and get updates from AccessFeed Update Service. GeoChannel resources publish their discovery metadata to catalogue service and AccessFeed updates to

AccessFeed Update Service; and are mashed up or Geo-Federated via Accessor Bus

The subsequent sections will introduce the software implementation of the prototype platform, mainly involving functional description, software architecture stack, development (programming) languages, etc. Section 8.8.2 is about front-end components, which are followed by Section 8.8.3 to Section 8.8.5 that are about back-end facilities.

## 8.8.2 Front-end Facility

The Geonoon platform frontend named GeoChannel Explorer itself is a Web application, by analogy with current Web Mapping programs such as Google Maps/Earth. User can run the web-browser-based GeoChannel Explorer in desktop computers or mobile devices such as smartphones with a web browser. Functionally GeoChannel Explorer is a platform, browser, portal and also an application/service interface for GeoChannel resources, which can be discovered, run or aggregated (mashed up) in the GeoChannel Explorer environment.

Other than acting as a standalone web program, alternatively GeoChannel Explorer (or its individual components) can also serve as software library that can be integrated into other applications/systems which can then become GeoChannel-enabled applications/systems.

The GeoChannel Explorer (or its software library) is an integrated environment, which implements some common runtime components and utilities as introduced in the following subsections.

### 8.8.2.1 GeoAccessFeed processor

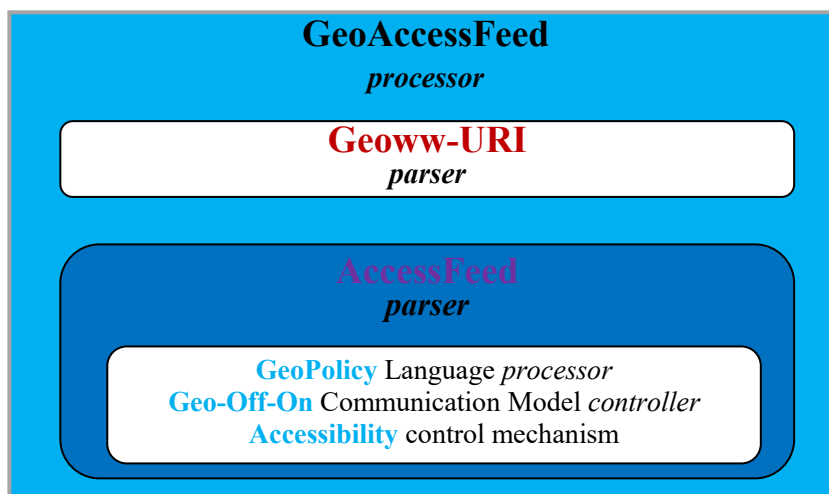


Figure 8-14: Component modules of GeoAccessFeed technology

The core facility for the Geonoon platform frontend is the processor for GeoAccessFeed technology, which involves technical constituents such as GeoPolicy Language, Geo-Off-On Communication Model, AccessFeed and Geoww-URI scheme, etc., as showed in Figure 8-14.

Table 8-3: The software components of GeoAccessFeed processor (at front end)

<b>GeoPolicy Language</b> <i>processor</i>	language elements	as designed in Table 3-1
	extended KML class diagram	as designed in Figure 3-1, Figure 3-2
	extended-KML schema	as codified in Appendix 1
<b>Geo-off-on</b> <b>Communication Model</b> <i>controller</i>	as designed in Chapter 4, Section 5.6.2, Section 5.6.4	
	principle & design	as illustrated in Figure 4-2, Figure 4-3, Figure 5-2, Figure 5-6
	Pull/Push elements Information Model	as designed in Listing 5-6, codified in Appendix 2, and in Table 5-1 for workflow model
	implemented by	GeoChannel Explorer (or its programmable library)
<b>AccessFeed</b> <i>parser</i>		as designed in Chapter 5, and codified in Appendix 3
<b>Geoww-URI</b> <i>parser</i>		as designed in Chapter 6, and illustrated in Figure 6-1
programming & runtime environment		JavaScript, Web Browsers, Desktop OS; Java, Android OS

The GeoAccessFeed processor mainly implements its software components for corresponding techniques and models as listed in Table 8-3. The following subsections will describes some details.

#### 8.8.2.1.1 *GeoPolicy Language processor*

GeoPolicy Language (designed in Chapter 3) is a fundamental component for the GeoAccessFeed technology. The GeoPolicy language processor is for parsing and processing GeoPolicy condition statements encoded within AccessFeeds. It can parse and validate GeoPolicy condition statements based on its language syntax model (as codified in Appendix 1), evaluate the condition statements using built-in operator functions in this processor, observe real-time updates of condition parameters and then re-evaluate the GeoPolicy conditions. Programming the GeoPolicy language processor was really a solid and time-consuming work in



this research, as it had to deal with many complexities in terms of software implementation. GeoPolicy language involves rich language elements and flexible expression. For example, as for its extended-KML encoding schema (codified in Appendix 1), GeoPolicy elements can mix with standard KML elements, by embedding or being embedded by each other, or nesting in any depth. Recursive methods are employed for parsing GeoPolicy statements based on syntax schemas of multiple languages involved. This processor includes a rich set of operator functions, e.g. space/time analysis, mathematics/logic/relation calculation, etc., for evaluating parameters of GeoPolicy conditions. In addition, this processor can deal with dynamic changes of GeoPolicy condition statements or their parameters. The context attributes of resources or clients may involve various parameters, some of which may come from sensor observations, for example, real-time sensor data from smart phone devices. So the GeoPolicy Language processor should support acquiring sensor data, observing their random updates and then triggering re-evaluation of GeoPolicy conditions.

The component modules and workflow of the GeoPolicy Language process is briefly illustrated in Figure 8-15.

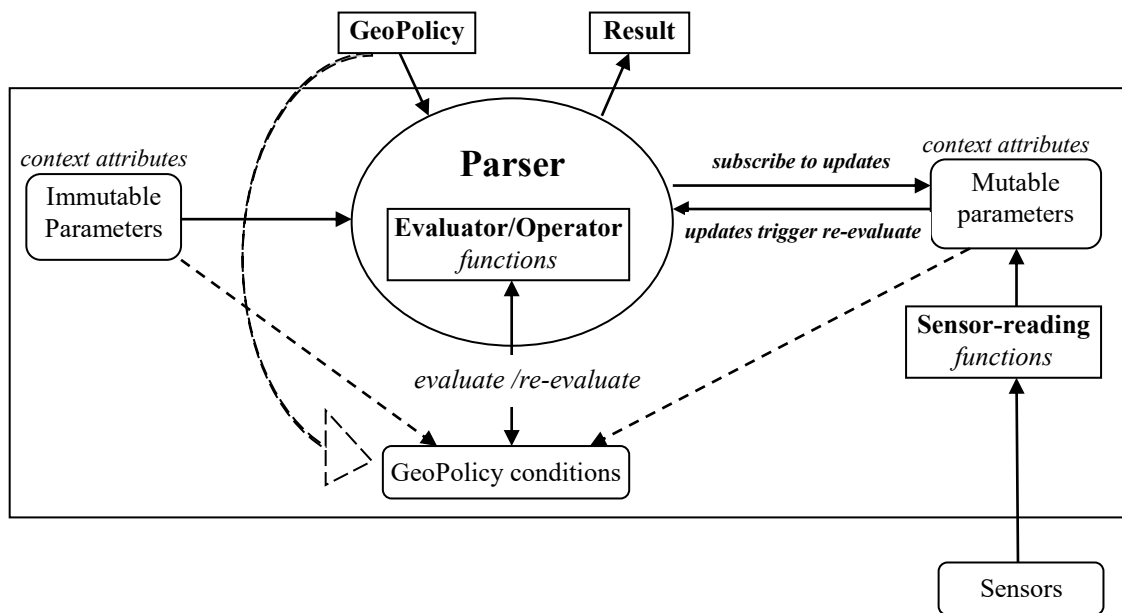


Figure 8-15: GeoPolicy processor: component modules and workflow

#### 8.8.2.1.2 Geo-Off-On Communication Model controller

Chapter 4 has designed the Geo-Off-On Communication Model for supporting adaptive communication in a context-aware and on-demand way. The Geo-Off-On communication

model employs GeoPolicy language for expressing controlling conditions for throttling networking. Figure 8-16 illustrates the principle and workflow of the Geo-Off-On Communication Model controller.

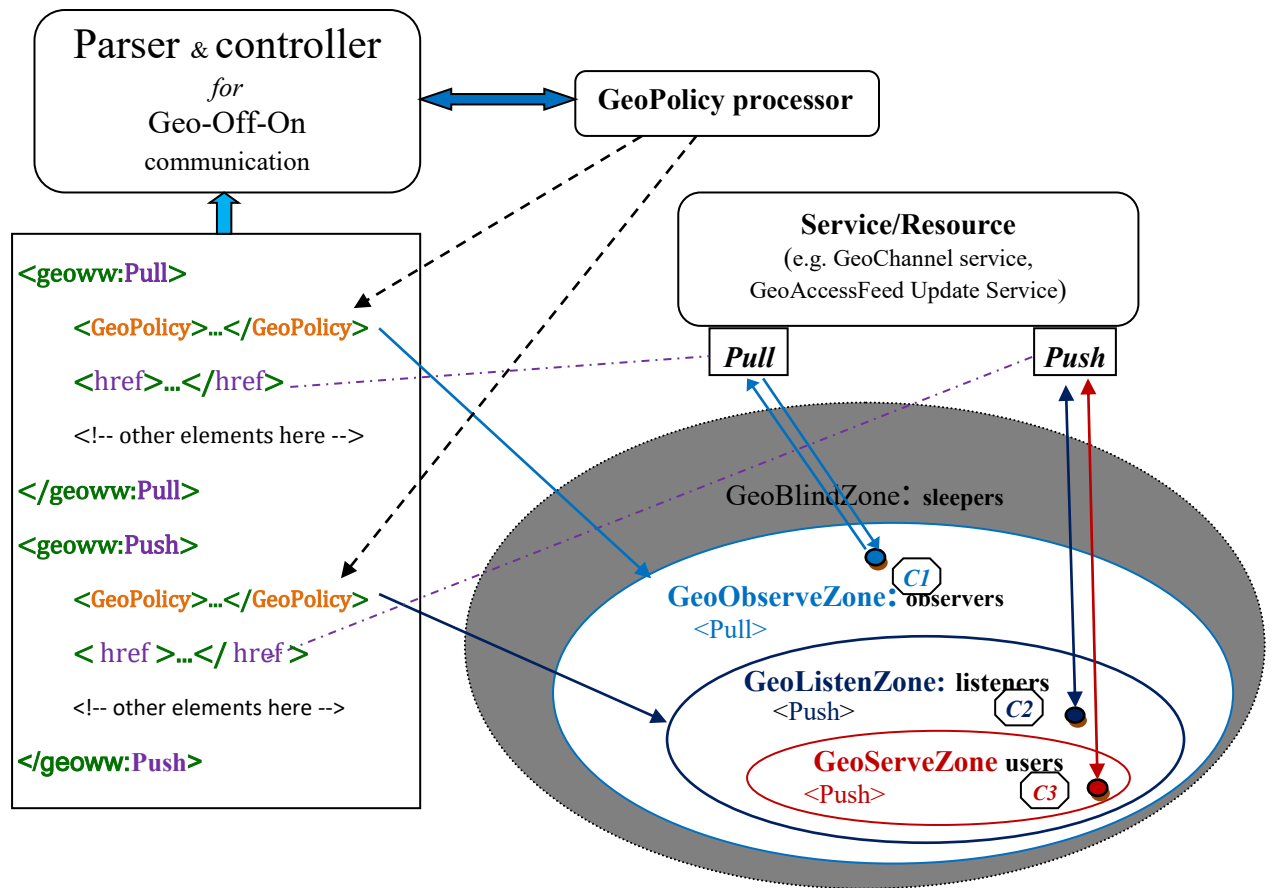


Figure 8-16: Controller for Geo-Off-On communication model with adaptive pull & push patterns

The Geo-Off-On communication model can dynamically and automatically discriminate clients in different GeoContexts to use switchable communication patterns with different levels of priority. Empowered by GeoPolicy processor (as described in last section) that can evaluate GeoPolicy conditions, the controller can switch between Pull and Push communication patterns, based on the context attributes of resources and/or clients. So clients can use discriminated priorities of communication to interact with GeoChannel resources. Notably a Geo-Off-On communication control has self-update capability, as it can get dynamic updates of GeoPolicy condition statements and/or context attribute parameters, so as to evolve to new control patterns of communication.

And further, the Geo-Off-On Communication Model controller implements the sophisticated mechanisms for controlling communication interactions, as expressed by the information model

showed in Listing 8-4 for the Pull and Push elements as described in Section 5.6.4.1.

Listing 8-4: The controlling mechanisms encoded in <Pull> and <Push> elements for the Geo-off-on communication model

```

<geoww:Pull>

  <GeoPolicy>...</GeoPolicy>                                <!-- a space-time-aware condition for
                                                                controlling Pull element -->
  <href>...</href>                                           <!-- the URL for http-based pull (request/response) communication -->

  <refreshMode>...</refreshMode>
                                                                <!-- refreshModeEnum: onChange, onInterval, or onExpire -->
  <refreshInterval>...</refreshInterval>                    <!-- float -->
  <geoww:viewRefreshMode>...</geoww:viewRefreshMode>
                                                                <!-- viewRefreshModeEnum: never, onStop, onRequest, onPolicyToTrue -->
  <viewRefreshTime>...</viewRefreshTime>                    <!-- float -->
  <geoww:locationRefreshMode>...</geoww:locationRefreshMode>
  <geoww:remoteEvaluate>...</geoww:remoteEvaluate>         <!-- boolean -->
  <viewBoundScale>...</viewBoundScale>                      <!-- float -->
  <viewFormat>...</viewFormat>                              <!-- string -->
  <httpQuery>...</httpQuery>                                <!-- string -->
  <geoww:clientContext> <!-- this element can contain some information about the client's context -->

  Feed=[channelID],[feedVersion];Location=[clientLocation]; Speed=[clientSpeed];Heading=[clientHeading];Time=[Time]
  e]
  </geoww:clientContext>                                     <!-- the content inside this Context element is extensible -->
</geoww:Pull>

<geoww:Push>
  <GeoPolicy>...</GeoPolicy>                                <!-- a space-time-aware condition for
                                                                controlling Push element -->
  <geoww:pushPattern>WebSockets</geoww: pushPattern> <!-- or
                                                                ServerSentEvents-->
  <href>...</href>                                           <!-- the URL for a client-side WebSocket or EventSource object to
                                                                initiate connecting to a pushing server -->
  <refreshMode>...</refreshMode>
  <refreshInterval>...</refreshInterval>
  <geoww:viewRefreshMode>...</geoww:viewRefreshMode>
  <viewRefreshTime>...</viewRefreshTime>
  <locationRefreshMode>...</locationRefreshMode>
  <geoww:remoteEvaluate>...</geoww:remoteEvaluate>         <!-- boolean -->
  <viewBoundScale>...</viewBoundScale>
  <viewFormat>...</viewFormat>
  <Query>...</Query>    <!--This Query element's content and syntax is similar to the
                                                                "httpQuery" element within the "Pull" element -->
  <geoww:clientContext>...</geoww:clientContext>

</geoww:Push>

```

### 8.8.2.1.3 AccessFeed processor

The AccessFeed processor is a high-level and practical facility underpinned by GeoPolicy processor and Geo-Off-On controller. It mainly implements the Accessibility controller and

TopologyLink controller. Figure 8-17 illustrates the relation between these functional components.

The AccessFeed processor functions as the interpreter, executor, updater and aggregator of AccessFeeds. It is capable of processing GeoPolicy language for evaluating GeoPolicy conditions, performing Geo-off-on communication model for adaptively updating AccessFeeds, and controlling the accessibility of Accessors to dynamically organize GeoClustering of users, connecting/disconnecting to dynamics sources or pervasive services.

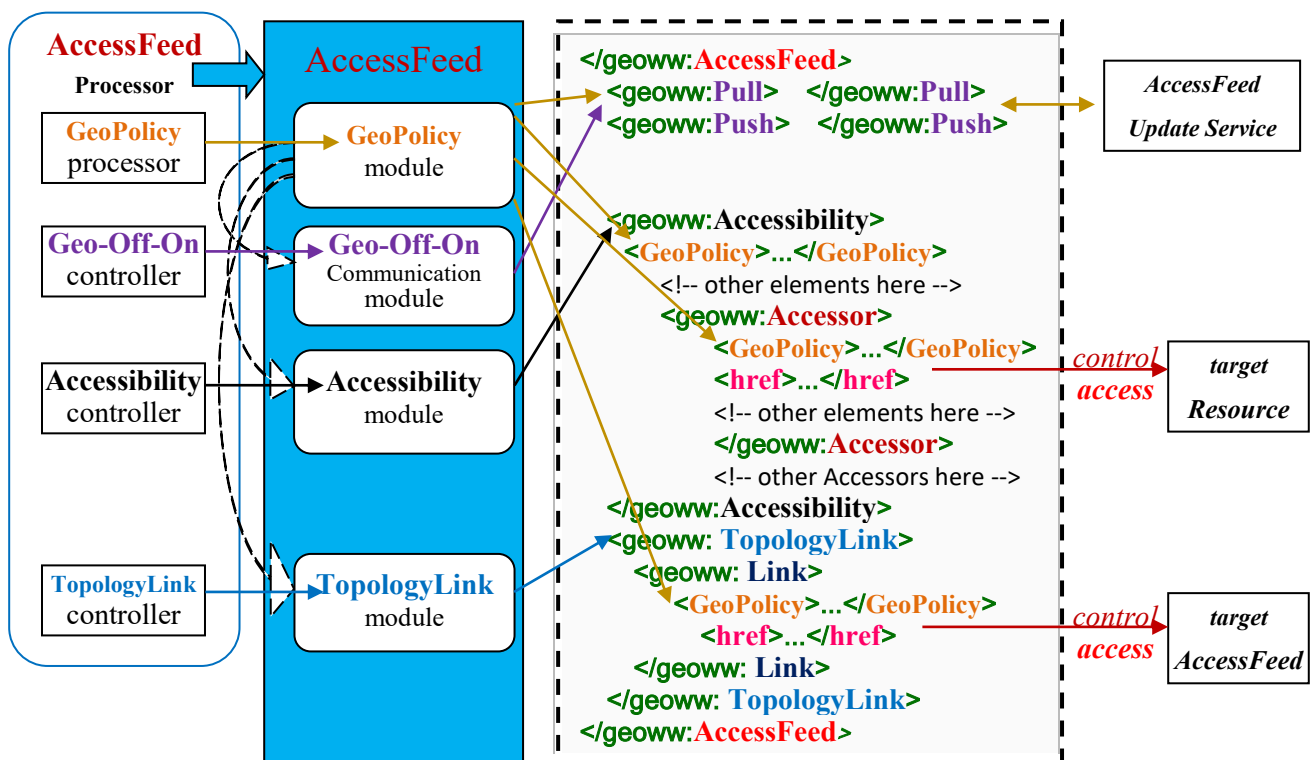


Figure 8-17: AccessFeed processor components relational model

The Accessibility controller and TopologyLink controller implement the working mechanisms as designed in Section 5.4.2 and Section 5.4.3.

#### 8.8.2.1.4 Geoww-URI parser

The Geoww-URI parser is for interpreting Geoww-URIs encoded by using the Geoww-URI scheme, a new URI model (designed in Chapter 6) for contextualizing, identifying, selecting and even gathering resources, clients, clusters, etc. Geoww-URIs ( `geoww://` ) can function as GeoChannel Web addresses, by analogy to HTTP-URIs for the general Web.

The upper part of Figure 8-21 illustrates a simplified structure of the Geoww-URI scheme, below which is a simple geoww-URI sample. It is noted that the Geoww-URI scheme supports a mechanism of chaining AccessFeeds, as denoted by ② and ⑤ in Figure 8-21. An actual Geoww-URI with multiple chained AccessFeeds may become longer and more complex than this sample. So the Geoww-URI parser can interpret Geoww-URIs with arbitrary complexity, and can implement the functions as designed in Chapter 6.

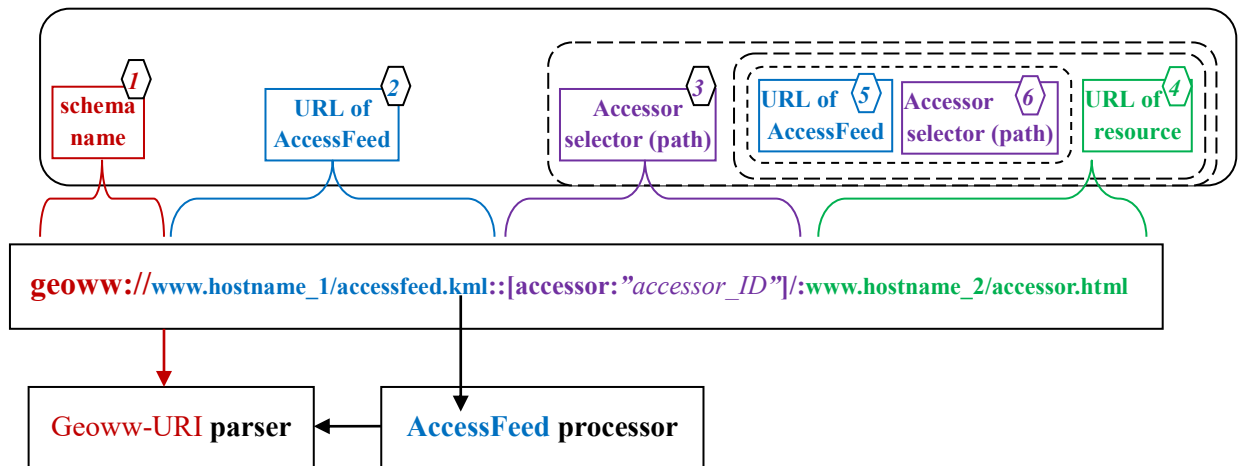


Figure 8-18: Geoww-URI parser for processing Geoww-URIs

### 8.8.2.2 Accessor Bus

GeoChannel Accessor Bus works for plugging resource Accessors to manage mashups of GeoChannel resources, bridging interaction between Accessors of GeoChannels to interoperate and federate Geo-compatible GeoChannel resources, and networking GeoChannel resources and users to form GeoChannel networks.

In its implementation, GeoChannel Accessor Bus employs an event-bus facility for inter-Accessor communication and interaction. This event-driven mechanism can cater for dynamic updates from/to GeoChannel resources (or clients) plugged in GeoChannel Accessor Bus.

### 8.8.2.3 Client-side components for discovery functionality

GeoChannel Discovery functionality designed in Chapter 7 involves some front-end and back-end facilities. In terms of their implementation, server-side GeoChannel Catalogue Service and Context Topology Service will be introduced in Section 8.8.3 and Section 8.8.5. There are also client-side components for GeoChannel discovery functionality.

The discovery components are for discovering GeoChannel resources, i.e., getting their Geoww-URIs, from virtual and physical world. On client side, two module parts are involved. Firstly, a client side provides a search module (i.e. virtual-world approach) that can search/query backend GeoChannel Catalogue Service or Context Topology Query Service. And secondly, another module (i.e. physical-world approach) is for detecting visual or wireless-signal media (from our physical environment) which indicates GeoChannel resources such as pervasive services. Both approaches are to get the Geoww-URIs that identify GeoChannel resources of interest.

Client-side components for discovery functionality are implemented by GeoChannel Explorer, which directly obtains Geoww-URIs from visual or wireless-signal media such as scanning QR code labels, detecting RFID tags or WiFi/Bluetooth signals that have Geoww-URIs encoded, or queries back-end services by using a set of new information models designed in Chapter 7.

### 8.8.3 GeoChannel Discovery Services

The Geonoon platform includes a set of backend components, which will be introduced from this section onwards in terms of their software implementation.

Table 8-4: The software implementation of GeoChannel Discovery Services

Discovery Component & Process Framework		as designed in Figure 7-1
<b>GeoChannel Catalogue Service</b>	metadata model of GeoChannel catalogues	as codified in Appendix 6
	Catalogue Service interface model	as designed in Table 7-1
	GeoChannel metadata feeds model	as designed in Listing 7-1, Appendix 4
	extended OpenSearch service description document format and request format models	as codified in Listing 7-3, Appendix 4
	extended-KML-encoded schema for responses of GeoChannel search operations.	as designed in Figure 7-8, Listing 7-5, Appendix 4
	workflow process model for real-time harvesting GeoChannel catalogue Metadata	as illustrated in Figure 7-7
	programming & runtime environment	Java, web service, Tomcat, Ubuntu
<b>GeoChannel Context-topology Query Service</b>		included in GeoChannel Web Engine and described in Section 7.7, Section 8.8.5.2

Chapter 7 has designed the technical systems for discovering GeoChannel resources. Two categories of approaches were elaborated, both of which are to eventually get Geoww-URIs for

GeoChannel resources of interest. The physical-world approach is implemented by front-end component of GeoChannel Explorer (as described in 8.8.2.3). The virtual-world approach involves GeoChannel Catalogue Service and GeoChannel Context-topology Query Service. This section mainly describes the software implementation of GeoChannel Catalogue Service, while GeoChannel Context-topology Query Service is a functionality of GeoChannel Web Engine that will be illustrated in Section 8.8.5.

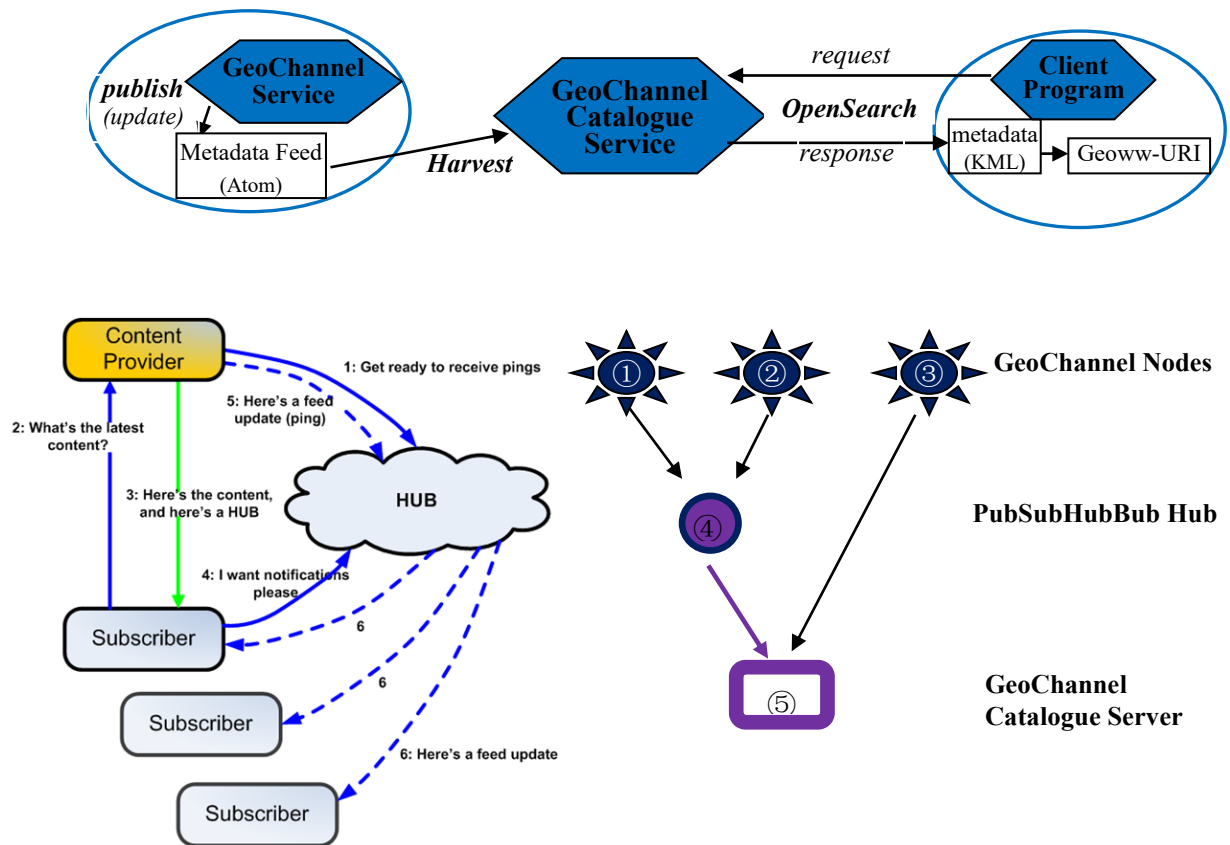


Figure 8-19: The components framework for GeoChannel Catalogue Service

GeoChannel Catalogue Service is a search/query service based on keywords or GeoContext conditions. This service software mainly implement a set of information and workflow models (as designed in Chapter 7 and listed in Table 8-4) for harvesting, registering and searching real-time metadata information of GeoChannel resources. These models involve:

- the metadata model of GeoChannel catalogues, as codified in Appendix 6 providing a unified metadata content model for characterizing GeoChannel resources for their discovery.
- an equivalent implementation model of OGC CSW (Catalogue Service for the Web) interfaces for simplifying GeoChannel discovery tasks.
- various message models for interaction with GeoChannel Catalogue Services. These

mainly include: GeoChannel metadata feeds model for real-time harvesting GeoChannel catalogue metadata; the extended OpenSearch service description document format and request format; and the extended-KML-encoded schema for responses of GeoChannel search operations.

- the workflow process model for real-time harvesting GeoChannel catalogue Metadata.

Figure 8-19 illustrates the software components framework and workflow for GeoChannel Catalogue Service that implements a series of unified and consistent metadata models, search operation interfaces, and normalized formats for query messages and result sets. Particularly the OpenSearch technique is employed and further extended for democratizing and catering for spatial-temporal specialization of GeoContext for searching GeoChannel resources.

#### **8.8.4 AccessFeed Update Service**

Adaptively updating AccessFeeds is a critical work for dynamically correlating things so as to organize geospatial dynamics, pervasive services and client-client interaction. Section 5.6 has designed multi-level mechanisms (as illustrated in Figure 5-5) for on-the-fly updates, which mainly involving cooperative interactions between GeoChannel Explorer (or its programmable library components) and backend AccessFeed Update Service. This section describes the software implementation of the multi-level updating mechanisms with AccessFeed Update Service.

Table 8-5 lists the implementation of components involved for updating AccessFeeds. AccessFeed Update Service provides the functionality for getting up-to-date AccessFeeds from GeoChannel service nodes, and for disseminating latest updates of AccessFeeds to clients. This service component implements the service interface model designed in Section 5.6.3.1 and the information model designed in Listing 5-5 for accessing service functions and for delivering messages with whole or part/incremental update of AccessFeeds.

Geo-Off-On communication model (designed in Chapter 4) plays a key role for adaptively updating AccessFeeds. Technically the Geo-Off-On module (described in Section 8.8.2.1.2) is mainly a client-side component for implementing the Geo-Off-On communication model that can generally conduct adaptive communication with any kinds of resources. As particularly it is largely used for condition-controlled updating of AccessFeeds, AccessFeed Update Service should collaborate with Geo-Off-On communication model. For example, AccessFeed Update Service provides HTTP-based pull pattern (request/respond) basically for non-pressing



communication, and also supports WebSocket-based push pattern (subscribe/publish) for real-time communication. This switchable pull/push communication model that can adaptively throttle communication between clients and AccessFeed Update Services can achieve timely and resource (i.e. networking and computing) -efficient updating of AccessFeeds.

Table 8-5: The software implementation of updating AccessFeeds

<b>Multi-level mechanisms</b> <i>for updating AccessFeeds</i>	as illustrated in Figure 5-5, designed in Section 5.6.1	
<b>Geo-Off-On communication Model</b> <i>for updating AccessFeeds</i>	as designed in Chapter 4, Section 5.6.2, Section 5.6.4	
	principle & design	as illustrated in Figure 4-2, Figure 4-3, Figure 5-2, Figure 5-6
	Pull/Push elements Information Model	as designed in Listing 5-6, and in Table 5-1 for workflow model
	implemented by	GeoChannel Explorer (or its programmable library) in JavaScript
<b>AccessFeed Update Service</b>		
Service Interface Model	<ul style="list-style-type: none"> <li>• notifyFeedUpdate</li> <li>• getFeedUpdate</li> <li>• subScribeFeedUpdate</li> </ul>	as designed in Section 5.6.3.1
Message Information Model	as designed in Listing 5-5, for whole or part/incremental update of AccessFeeds	
programming & runtime environment	Java, web service, Tomcat, Ubuntu	

In addition, AccessFeed Update Service supports multi-level updating mechanisms (as illustrated in Figure 5-5) that can enable clients to wake up in GeoBlindZone or to catch up in GeoLagZone to avoid missing proper updates.

### 8.8.5 GeoChannel Web Engine

The GeoChannel Web is expected for mapping and applying dynamic and random relations between real-world things so as to bridge thing in context. These kind of dynamic relations can constitute large-scale and high-dynamic topology networks. Section 8.6 has introduced the concepts and features of GeoChannel networks and the GeoChannel Web, and illuminated technical mechanisms (e.g. GeoCluster, GeoBridge and GeoFederation) for building the

GeoChannel Web networks. This section here describes the software architecture stack for implementing GeoChannel Web Engine, a core facility and service of the Geonoon platform for constructing, updating and querying a large-scale dynamic topology graph of GeoChannel networks on a global scale, i.e. a complete view of the GeoChannel Web networks.

Current general web search engines use web crawlers to crawl the web for retrieving web pages to their servers, and then analyse the interlink relationships between web pages to build a global view of the hyperlinks topology of the interlinked web content (resources). By analogy to this kind of topology-viewing capability of web search engines, the Geonoon platform with the GeoChannel web Engine component can, but by using different strategy and techniques (e.g. GeoCluster, GeoBridge and GeoFederation) , build and maintain a real-time graph of global GeoChannel networks that involve all GeoChannel resources, users and access (participation or connection) relationships. This global up-to-date graph just reflects actual dynamic correlation among clients and/or GeoChannel resources.

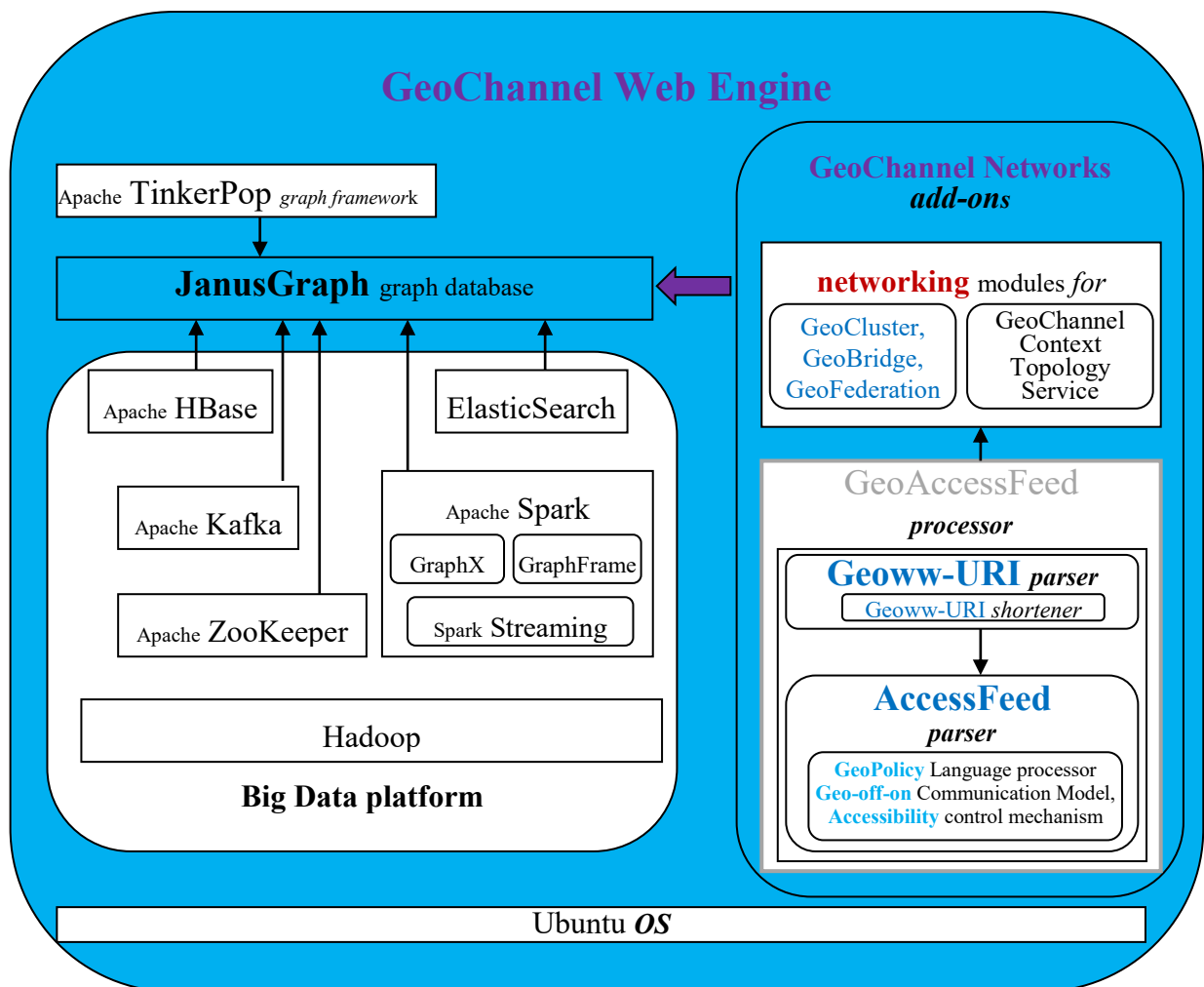


Figure 8-20: The software stack for GeoChannel Web Engine

The GeoChannel Web Engine is a common service facility for GeoChannel frontend (e.g. GeoChannel Explorer) or backend programs to query and update the topology of the GeoChannel Web networks. It caters for the highly dynamic-relation feature that demands the performance in a large-scale data set. By querying and navigating the access/participation relationships between resources and clients, there is promising potential for novel applications. For example, to discover geographically local resources (e.g. pervasive services), or users with similar context; to identify Geo-compatible (see Section 8.4) GeoChannel resources for federating them to achieve cooperative functionality; to perform geo-diffuse social networking; etc.

The GeoChannel Web Engine is underpinned by general graph-processing and big-data systems, with extension components for supporting GeoChannel-network-specific functionality. Figure 8-20 illustrates the software architecture stack of the GeoChannel Web Engine.

#### **8.8.5.1 Graph-processing and big-data systems**

To be capable of supporting large-scale networks with consideration on good performance and specific needs of GeoChannel networks, this architecture selects graph-processing and big-data system components from wide-range options available currently, as an optimal solution is showed at the left part of Figure 8-20.

The central component of the big graph-processing system is JanusGraph, with other jointly working components that support the big-data environment based on Hadoop (Apache 2016). The following illuminates their functionalities.

##### **1. JanusGraph: the scalable graph database**

JanusGraph (Linux-Foundation 2016) is a scalable graph database optimized for storing and querying large graphs containing hundreds of billions of vertices and edges distributed across a multi-machine cluster, and is a transactional database that can support thousands of concurrent users, complex traversals, and analytic graph queries. Aside from scalability, performance and reliability benefits (including cross-data-center HA), JanusGraph promises support for a wide variety of open source storage backends, analytics engines and search engines. Acting as a constituent, JanusGraph is a good choice for the GeoChannel Web Engine to be able to deal with large volumes of graph-structured data about dynamic relationships on a global scale.

TinkerPop (Apache 2016) is a graph computing framework for both graph databases (OLTP) and graph analytic systems (OLAP). JanusGraph implements and integrates with the TinkerPop

graph stack.

## 2. **HBase**: the storage backend for JanusGraph

In this architecture, JanusGraph uses HBase as its backend storage. HBase (Apache 2016) is an open-source, distributed, versioned, non-relational and scalable data store modeled after Google's Bigtable (Google 2006). HBase provides Bigtable-like capabilities on top of Hadoop and HDFS, and supports random, real-time read/write access to Big Data.

## 3. **ElasticSearch**: the search engine for JanusGraph

Elasticsearch (Elastic 2016) is a flexible and powerful open source, distributed, real-time search and analytics engine. Elasticsearch is chosen here as an index backend for JanusGraph and supports geo, numeric range and full-text search functionality.

## 4. **Spark**: the graph analytics engine for JanusGraph

Spark (Apache 2016) is a fast and general engine for large-scale data processing. The GeoChannel Web Engine architecture mainly employs some Spark's components, for example, using Spark GraphX for graphs and graph-parallel computation; using Spark GraphFrame for supporting DataFrame-based Graphs; and using Spark Streaming to support fault-tolerant processing of real-time data streams.

## 5. **Kafka**: a distributed stream platform

Kafka (Apache 2016) is a real time, fault tolerant, scalable messaging system for moving data in real time. It gets used for two broad classes of application: building real-time streaming data pipelines that reliably get data between systems or applications; and building real-time streaming applications that transform or react to the streams of data. The GeoChannel Web Engine architecture employs Apache Kafka and Spark Streaming to communicate real-time updates of the context properties of GeoChannel-resource and -client nodes.

## 6. **ZooKeeper**: a distributed coordination service

The GeoChannel Web Engine architecture also employs ZooKeeper (Apache 2016), a distributed, open-source coordination service for distributed processes. It provides a distributed configuration service, synchronization service, and naming registry for large distributed systems.

### 8.8.5.2 GeoChannel Networks add-ons

The peculiar components to the GeoChannel Web Engine are the GeoChannel Networks add-ons, which extend the general graph-processing and big-data systems, for supporting GeoChannel-network-specific functionality. This subsection describes the software implementation of the GeoChannel networks add-ons, as listed in Table 8-6.

Table 8-6: The software implementation of GeoChannel Networks add-ons

<b>GeoAccessFeed</b> <i>processor</i> <b>GeoPolicy</b> language processor <b>Geo-Off-On</b> Communication Model processor <b>AccessFeed</b> parser <b>Geoww-URI</b> processor <i>(including Geoww-URI shortener)</i>	as designed in Chapter 3, 4, 5, 6
<b>GeoCluster &amp; GeoBridge</b> <i>functionality</i>	as designed in Section 8.3, and illustrated in Figure 8-4, Figure 8-5
<b>GeoFederation</b> <i>functionality</i>	as designed in Section 8.4
<b>GeoChannel Context Topology Service</b> Context-topology Build Service Context-topology Query Service	as designed in Section 7.7
programming & runtime environment	Java, web service, big-data platform (as described in 8.8.5.1), Ubuntu

#### 1) GeoAccessFeed processor

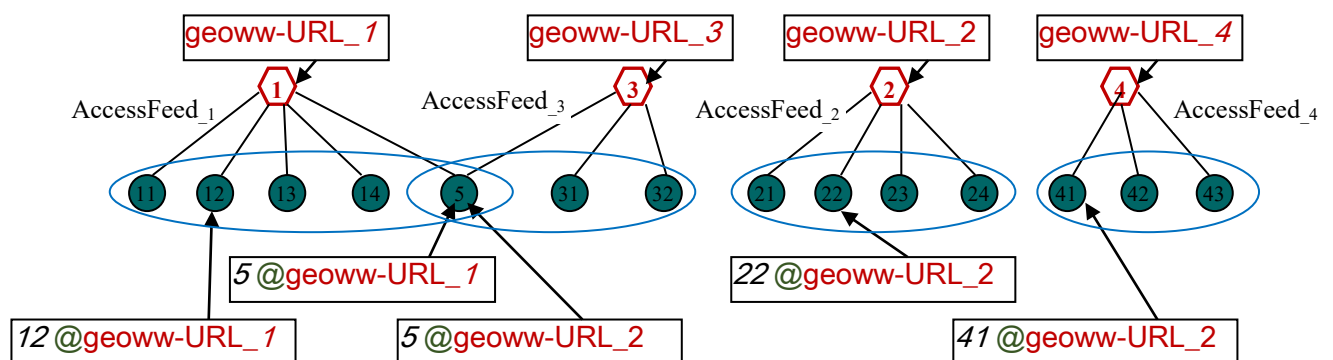


Figure 8-21: Geoww-URLs for identifying graph nodes (of resources and clients) and AccessFeeds for controlling their connection relations in GeoChannel Web (networks) dynamically

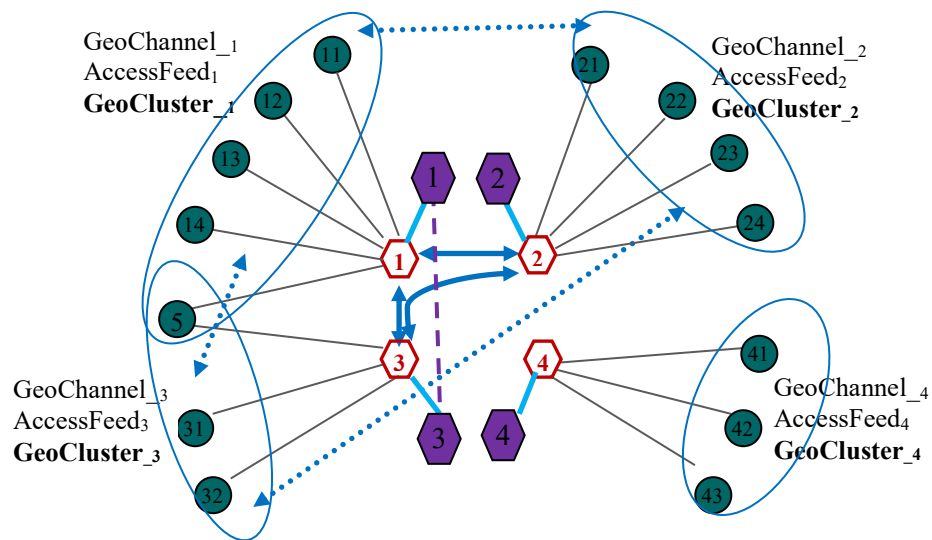


Figure 8-22: GeoBridge of GeoClusters and GeoFederation of resources

(e.g. GeoCluster\_1, GeoCluster\_2 and GeoCluster\_3) and (e.g. resource 1 with 3 federated by client 5)

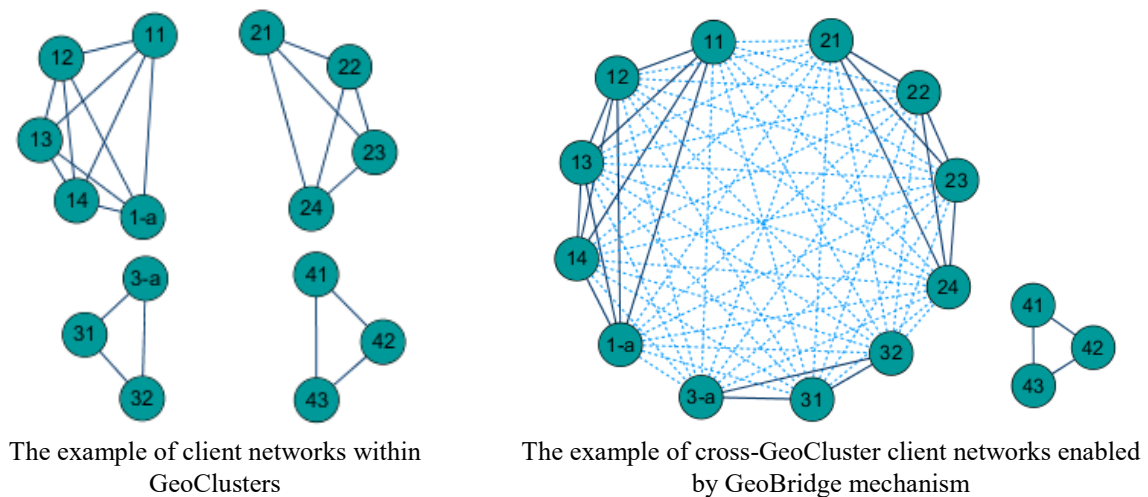


Figure 8-23: The example of GeoChannel networks enabled by GeoCluster and GeoBridge mechanism

Firstly, there is a back-end software implementation of the GeoAccessFeed technology, as opposed to its front-end implementation that has been described in Section 8.8.2.1. The back-end GeoAccessFeed processor works for the GeoChannel Web Engine. Similar to the front-end GeoAccessFeed processor (as illustrated in Figure 8-14), some component modules are involved, such as GeoPolicy language processor, Geo-Off-On Communication Model processor, AccessFeed parser, Geoww-URI processor, etc. They are programmed in Java code for back-end runtime environment.

As opposed to the general Web that employs http-URIs for identifying web resources and uses hyperlinks for representing inter-resource relations, the GeoChannel Web Engine employs Geoww-URIs for identifying resource/client nodes and uses AccessFeeds for controlling correlation of nodes.

As exemplified in Figure 8-21, peculiar to the GeoChannel Web, the nodes of GeoChannel resources and clients in the GeoChannel networks are identified by geoww-URIs, a new URI scheme designed in Chapter 6. The relations between resources and clients are dynamically controlled by AccessFeeds, a dynamically-correlating mechanism designed Chapter 5. From a perspective of graph data structure, the GeoChannel Web Engine takes the contexts (e.g. space/time or other sensor attributes) of resources and/or clients as the properties of the graph nodes, and takes AccessFeeds as the properties of graph edges. Internally, AccessFeeds take context attributes as parameters that are involved in the correlation conditions expressed by using GeoPolicy language.

## **2) GeoCluster, GeoBridge and GeoFederation**

Figure 8-22 exemplifies the mechanisms of GeoCluster, GeoBridge and GeoFederation for gathering clients into groups, for bridging groups into groups union, and for federating resources into integrated resources respectively, so as to constructing extended GeoChannel networks (as exemplified in Figure 8-23) with dynamic topology sensitive to the changes of context attributes of resources/clients.

The GeoChannel network add-ons implement the mechanisms of GeoCluster, GeoBridge and GeoFederation as built-in functionality for the GeoChannel Web Engine.

## **3) Context Topology Service**

GeoChannel networks add-ons for the GeoChannel Web Engine also implement Context Topology Service. This service includes two functions (or sub-services), i.e. Context-topology Build Service and Context-topology Query Service. The former is to construct context-topology networks, and the latter is to query context-topology networks. Both of the two sub-services are based on similarity or proximity of context attributes of GeoChannel resources or clients. Section 7.7 has introduced the concept of context-topology network and has exemplified Context-topology Query Service. Section 5.4.3 has designed TopologyLink information model for TopologyLink modules built in AccessFeeds. Figure 5-4 and Figure 7-9 exemplify context-topology networks.

A context-topology network is a topology graph about context attributes relation of GeoChannel resources or clients. A GeoChannel network can involve several types of graph networks, e.g. client-resource access relations graph, resource-resource context relations graph, client-client context relations graph, etc. The last two are just context-topology networks, while the first one (i.e. client-resource access relations graph, as exemplified in Figure 8-21) is largely involved in AccessFeed technology (designed in Chapter 5) that can control client-resource relations dynamically.

Context-topology Query Service (as exemplified in Section 7.7) can be used for another virtual-world approach to GeoChannel Discovery, apart from GeoChannel Catalogue Service implemented in Section 8.8.3. Not limited to GeoChannel Discovery, Context-topology Query Service can also be used for some other application scenarios where the contexts of resources or clients are involved, for example, client community detection, resources federation, etc.

The software implementation of Context Topology Service is based on existing graph network technology, with custom extension for working with context attributes of GeoChannel resources and clients. Context-topology Build Service for constructing context-topology networks employs the graph database JanusGraph. A context-topology graph can be derived from the access-relation graph of a GeoChannel network. Context-topology Query Service is basically about network-neighbour query using a graph traversal algorithm, a general function available in current graph-analytics systems. The software implementation of Context Topology Service is part of GeoChannel networks add-ons for GeoChannel Web Engine.

## 8.9 Summary

This chapter explored some mechanisms for forming GeoChannel networks which make up the GeoChannel Web on a global scale. A prototype platform, Geonoon, was developed for materializing and implementing the new concepts, models and techniques proposed or designed in this research.

This chapter viewed the relationships of client-resource, client-client and resource-resource from a perspective of GeoChannel networks, which involve resource-bridged clients via GeoChannel node and client-bridged resources via GeoChannel Accessor Bus. So GeoChannel networks are made up of resources, clients and access relationships. Based on AccessFeed technique that can control client-resource correlation, this chapter illuminated the GeoCluster and GeoBridge techniques for correlating clients, and the GeoFederate technique for integrating resources. The techniques of AccessFeed, Geoww-URI and GeoCluster Service can work



jointly to support the GeoCluster, GeoBridge and GeoFederate techniques.

GeoCluster can group and bridge clients which have common interest in a resource and have similarity or proximity in their GeoContexts for interaction and cooperation. GeoCluster can function as a new paradigm for social networking and for building or running participative and cooperative applications or services.

GeoBridge is a mechanism for bridging multiple GeoClusters to form a bigger GeoCluster (i.e. GeoClusters union) with extended context (e.g. space/time) coverage (namely merged GeoServeZones). This GeoBridge mechanism can work for propagating dynamics, relaying services and connecting users across individual GeoClusters.

GeoFederate can integrate Geo-Compatible resources to achieve composite functions or services. GeoFederation is organized by an AccessFeed that orchestrates resources, bridged by the GeoChannel Accessor Bus of a client, and conducted via the GeoChannel Accessors of these resources. GeoFederate works for resource-resource dynamic correlation (for composite functions or services), which is in contrast to access control for client-resource correlation (for pervasive services), and in contrast to GeoClustering and GeoBridging for client-client correlation (for social networking or cooperative applications).

The GeoChannel Web as a new type of Web is a global-scale dynamic GeoChannel network for mapping and applying dynamic relations between things. So it features highly dynamic topology and context-based relations. The GeoChannel Web technology can weave things into a global network with dynamic connections among them. The significance of proposing this GeoChannel Web concept is to inspire and consolidate a new vision of mapping and reflecting a full live status view of the real-world activities and interactions with their GeoContexts, to facilitate discovering, federating and interacting with resources or clients of interest.

Currently the Geonoon platform as a prototype implementation of the GeoChannel Web architecture mainly involves some backend and frontend facility components, which are about to support the use case projects described in next chapter. It is noted that the software development of this prototype platform has taken up a considerable part of time and efforts of the whole research work. The Geonoon expects to become a productive platform to support extensive applications and services for the GeoChannel Web.

## 9 Use Cases

### 9.1 Introduction

The previous chapters have proposed and designed a serial of concepts, models and detailed techniques for a new type of Web, the GeoChannel Web, for mapping and applying real-world dynamic and random relations to support various innovative application and services. Technically, the GeoChannel Web technology supplies a set of facilities and mechanisms for conducting dynamic correlation between client-resource, client-client and resource-resource in a context-aware, on-demand and scalable way, so functionally it can be generally used in a wide range of applications or services that involve context (i.e. space/time or other sensor attributes) nature.

This chapter is to exemplify the usefulness, demonstrate the functionality and examine the capability of the GeoChannel architectural technology designed and implemented in this research. Some notes are made here for these use cases:

- 1) Actual projects: a substantial work and a big part of time in this research has been involved in designing and developing two use case projects, i.e. Mass Vehicles Motion with inter-vehicle Interaction in New York City and 9.3 Virtual-Real-World Services and Interaction in British Museum. For implementing and presenting the original technology of the GeoChannel Web, much challenging software development work with cutting-edge programming techniques has been conducted. The demonstration of the two projects is available online for open access.
- 2) Integrative technologies: both of the projects demonstrates multiple aspects of the GeoChannel Web architecture and functionality, with different highlights as well as common components respectively.
- 3) Innovative and representative: the use cases representatively demonstrate novel applications and services powered by the original technology of the GeoChannel Web, whose concepts, models and techniques materialize in real applications.

4) Generalizable and extensible: the solutions to the use case here are applicable for an extensive range of application scenarios where the GeoChannel Web can involve, which can only be limited by our imagination. For example, although the use cases here mainly involve space/time as context attributes, any other types of contexts that can be expressed by the GeoPolicy Language can function in the GeoChannel Web architecture for achieving a smart context-aware environment.

5) Geonoon based: The use cases projects are developed and tested in the support of the Geonoon (described in Chapter 8) platform, which provides some prototype implementation of the new technical components designed in this research.

Section 9.2 and Section 9.3 will describe the use cases. For brevity of depiction, the examples presented in this chapter just demonstrate some basic application scenarios. The use cases are practical examples that have been built and tested in this research work, and some simulated experiments are involved, as their technical principles and mechanisms can apply to real-world complex practical applications. The use cases also involve certain application scenarios that are conceived for further illuminating the usage of the GeoChannel Web technology. Each example or scenario will indicate technical focuses that are mainly demonstrated.

## **9.2 Mass Vehicles Motion with inter-vehicle Interaction in New York City**

### **9.2.1 Background & Objectives**

This use case project is based on a real big data set about New York taxi trips, which includes 1.1 billion records covering a period from 2009 to 2015. This dataset (NYC & TLC 2015) is collected and released as open data by the NYC Taxi and Limousine Commission (TLC).

This open dataset is quite welcome by data researchers who are interested in extracting stories and meaning from the data. Some scientists or technologists working in big data field explore and investigate this big dataset in different aspects and for different purposes. For example, to depict the daily life of taxi drivers (Whong 2014), to clarify the problem of lacking taxi services (Wellington 2015b) in some time slots of days, to study on the potential of taxi shareability

(MIT 2014), to investigate the habits or custom of paying extra tips to taxis drivers (Wellington 2015a), and to analyze the neighbourhood with activities such as nightlife (Schneider 2015), etc.

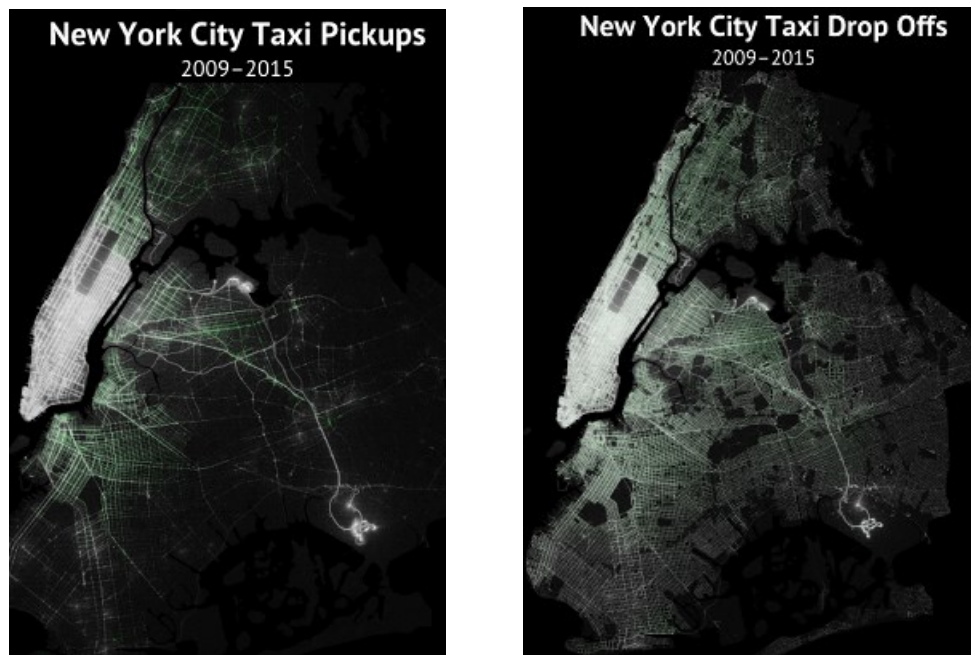


Figure 9-1: Mapping pickup and dropoff spots of taxi trips in New York City from 2009–2015 (Schneider 2015)

For the research work mentioned above, these researchers mostly investigate this big data set from sliced or snapshot space/time perspectives or based on statistic analytics, but lack exploration on the process of all the taxi trips. For example, quite a lot studies on this big data set are based on pickup and drop-off points with their timestamp to observe and infer certain questions. Figure 9-1 shows such an example. So far there seems to be no research or practice on explore or apply the mass dynamics and mobility of vehicles during their running processes. Actually, if taken as a whole, the detailed trip-level data is more than just a vast list of taxi pickup and drop-off coordinates, but can tell various stories about the metropolis of New York.

The research here attempts to reproduce vast continuous processes with massive random dynamics of taxi vehicles, by employing the GeoChannel Web technology, to implement large-scale dynamic maps and to explore dynamic vehicle-position or vehicle-vehicle relations, for supporting more novel means to extract or interpret stories about the New York city, or for supporting future innovative application and services that can facilitate smart-city or smart-environment practice.

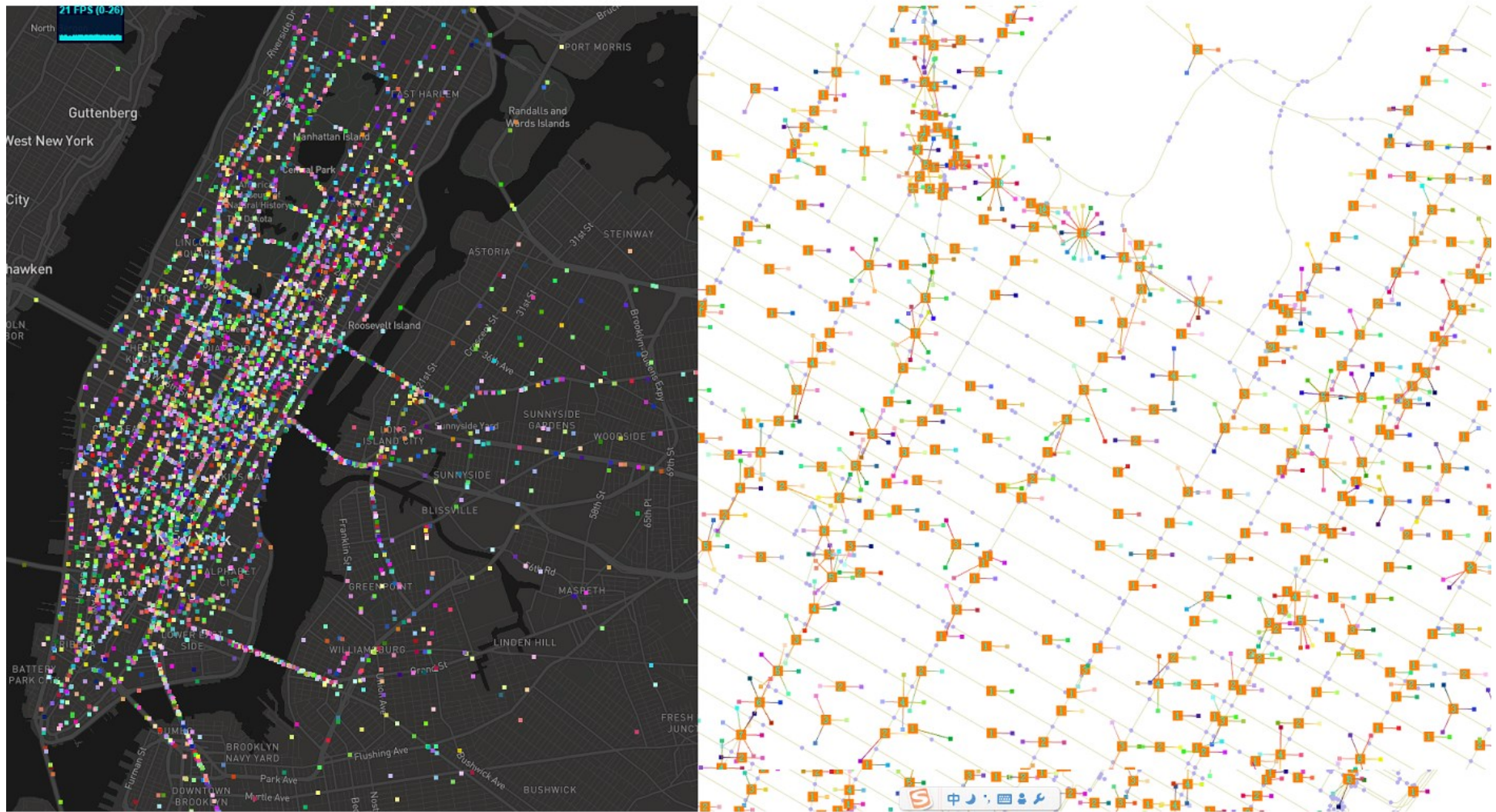
For brevity and clarity, this use case is to address some points listed as following:

- 1)** To examine the GeoChannel techniques on its mechanisms for generating dynamic relations and GeoClusters. This involves two aspects: reproducing dynamics by using historical data; or correlating vehicle-road, vehicle-vehicle for real-world vehicles.
- 2)** To examine the GeoChannel Web on its capability of efficiently mapping real-world massively dynamic and random relations. Here is to capture, change, analyze and present road-road, road-vehicle and vehicle-vehicle relations in frequently-changing and densely-happening contexts.
- 3)** To explore the application potential of the dynamic and random relations mapped by the GeoChannel Web. Here is to exemplify inter-vehicle networking interaction by bridging vehicles based on their contexts topology.

This use case demonstration is now online for public exploration. Figure 9-2 gives a screen snapshot of this complex system. The subsequent sections will elaborate the implementation of this use case demonstration.

Online Demos: [www.geonoon.net/demos](http://www.geonoon.net/demos)

## Large-scale and real-time map and dynamic network of vehicles & roads for New York City



Online demo: [www.geonoon.net/demos](http://www.geonoon.net/demos)

Figure 9-2: Demo project: Large-scale real-time map and dynamic network of vehicle & roads for New York City



## 9.2.2 Functionality Components Architecture

For fulfilling the points of objectives listed in last section, the functionality components architecture for this use case demo is identified, as illustrated in Figure 9-3.

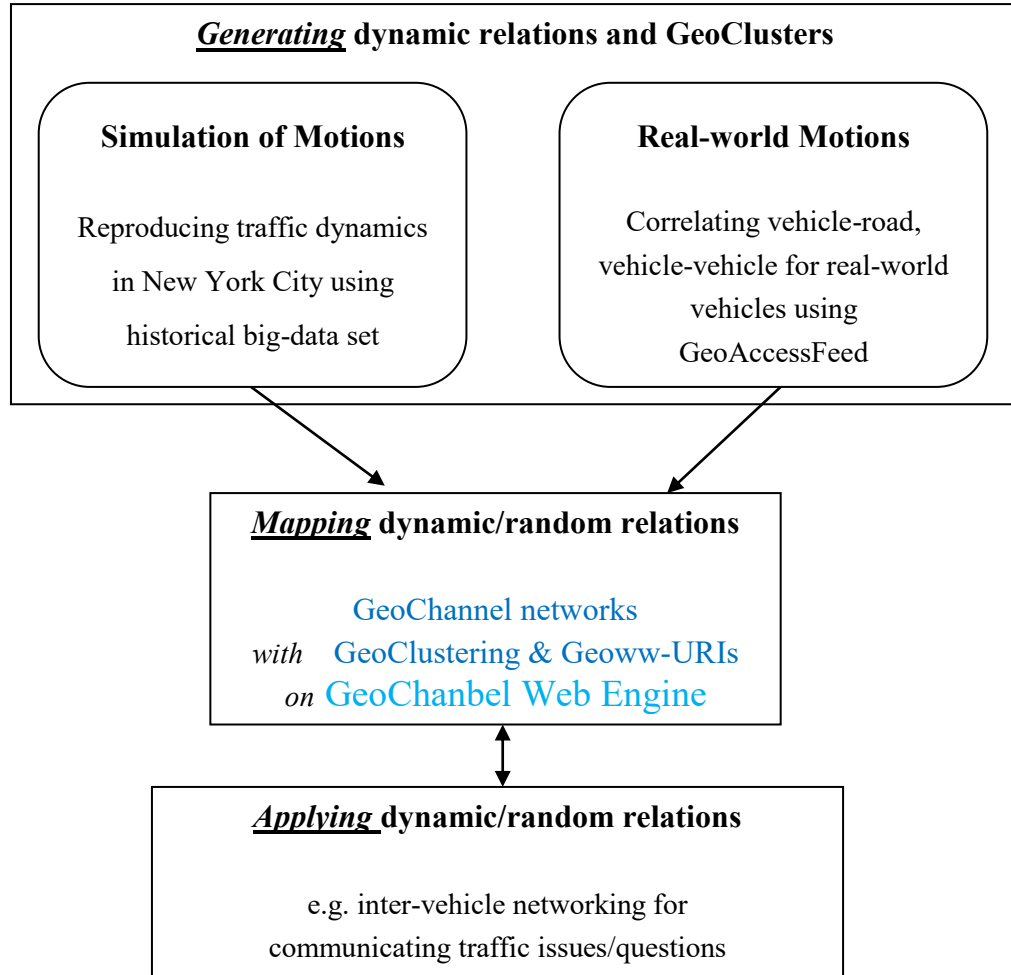


Figure 9-3: Functionality components architecture for Use Case 1

Firstly, this use case develops a large-scale simulation system for reproducing traffic dynamics in New York City using historical big-data set. Secondly, for real-world individual vehicles, the GeoAccessFeed technology is used for producing correlation dynamics between vehicle-road, and vehicle-vehicle. All these simulated or real-world dynamic relations are mapped by the GeoChannel Web Engine on GeoChannel networks with GeoClustering and Geoww-URI techniques. And finally an application scenario is exemplified for applying these dynamic and random relations.

The subsequent sections will expand on software implementation of each functionality component outlined in Figure 9-3.

### 9.2.3 Simulation: Reproducing Mass Vehicles motion process

Section 9.2.1 has described the big data set with about 1.1 billion records for historical taxi trips in New York area. This simulation here is to generate, transmit and present large-scale dynamic data and maps for reflecting continuous processes of all individual taxi trips in real space-time context happened in New York city. This work implements dynamic maps with vast number of vehicles in motion.

#### 9.2.3.1 Implementation for reproducing mass vehicles motion processes

A very complex system has been developed to use the real historical big data for reproducing mass vehicles motion processes. Figure 9-4 illustrates the software components framework that implements the functionality component “Simulation of Motions” outlined in Figure 9-3.

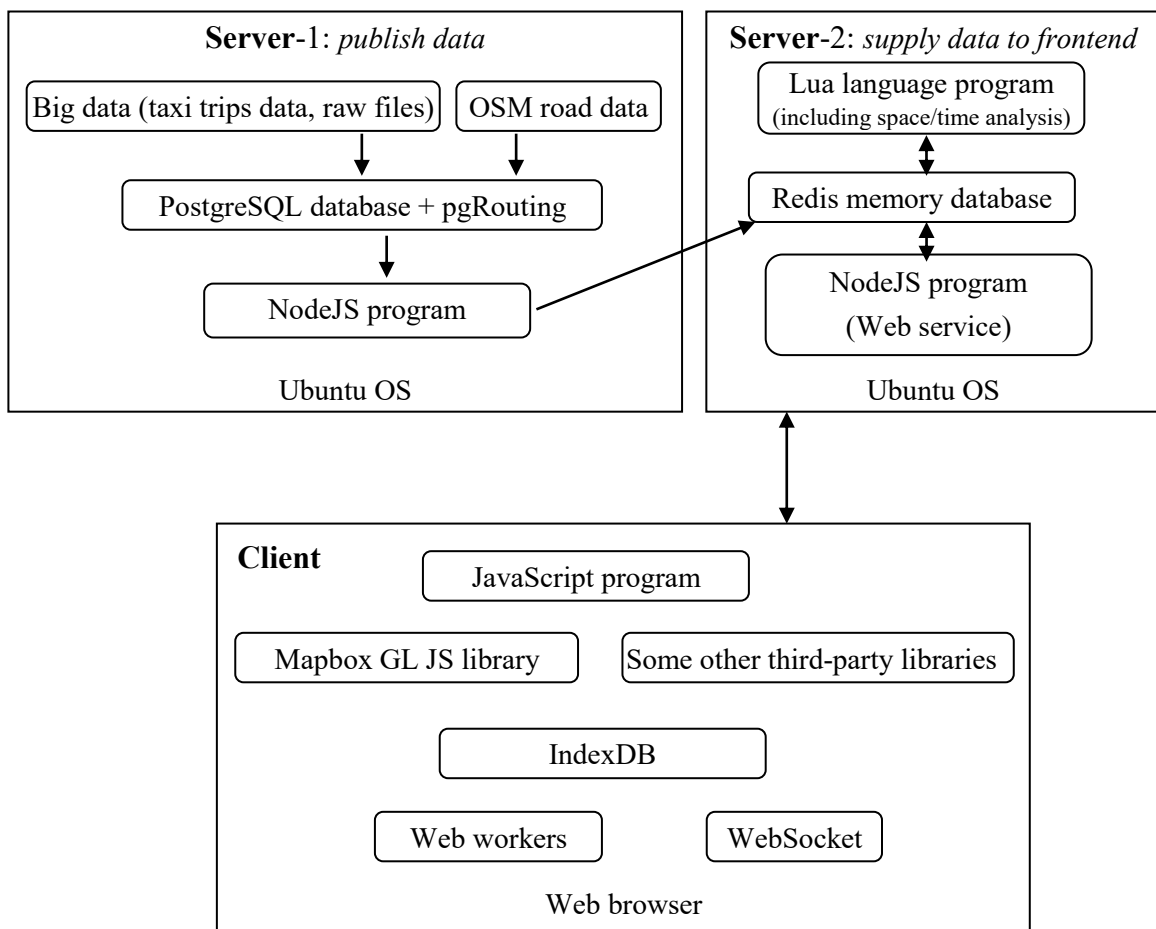


Figure 9-4: Software components framework for reproducing mass vehicle motion processes



Each component showed in Figure 9-4 is actually a quite complex software program. It was really a hard work and great effort to have developed these software programs for simulating vehicle motions in a city scale by using real historical big data, as this work has overcome many challenges in terms of optimizing the performance of computation, storage, communication and presentation (or visualization). This section here does not document much detail of programming, but just outline some main points about workflow and software modules.

Mainly two servers and one web browser-based software systems are involved in the motion simulation system for reproducing mass vehicle motion processes. The work for Server\_1 mainly involves the following tasks:

1. Importing raw data

The official TLC (NYC & TLC 2015) trip record dataset contains data for over 1.1 billion taxi trips from January 2009 through June 2015. Each individual trip record contains precise location coordinates for where the trip started and ended, timestamps for when the trip started and ended, plus a few other variables including fare amount, payment method, and distance traveled. The raw data is imported into a PostgreSQL database on Server 1 (as showed in Figure 9-4). The unindexed database takes up 330 GB on disk. Adding indexes for improved query performance increases total disk usage by another 100 GB.

2. Cleaning the big dataset.

This work firstly removes some variables information (such as taxi fare amount, payment method, etc.) unneeded for this use case, and then clean the errors coming from the raw dataset that has incorrect information about space, time or other attributes.

3. Identifying a proper OSM (OpenStreetMap) dataset and importing it into a database to function as the road network of New York City.

4. Routing all taxi trips to get planned routes.

Although providing a large number of taxi trip records, this data set only indicates the spatial coordinates of the pickup and drop-off positions with corresponding time stamps for all

taxi trips, but does not supply the information about the real processes (such as routes) of trips. The pgRouting software component is employed for routing all taxi trips to get planned routes.

#### 5. Sequentially publishing taxi trips data in timeline to Server\_2

Based on a common timeline and each trip starting timestamp, sequentially publishing taxi trips data in timeline to Server\_2 that will serve the web client with trip process data.

The work on Server\_2 centers on a Redis based memory database that can serve the client in a highly efficient way. Lua language is employed for programming space/time analysis logic on the data in Redis database to speed up computation, which can supply different data of taxi trips to different clients based on their map viewport scope or status. The Web service programmed in NodeJS stream data to clients in a push/pull adaptive way.

The client-side JavaScript program on a web browser involves many functional components and sophisticated processing logic that are mainly listed as below:

1. WebSocket connection is established with Server\_2 for real-time communication. Web Workers are used for dealing with computation or communication –dense tasks. These measures can relieve the main UI thread from these burdens.
2. IndexedDB technique is employed for Web local storage of road network edges data for New York City area.
3. Many efficient algorithms are programmed for some time-critical tasks. For example, line-interpolate-points for reproducing taxi trip processes; data serialization and compression encoding for efficient transmission; etc.
4. Presentation and visualization. It is really a challenge for visualizing massive and continuous dynamic processes of the vehicles in a big city scope in an interactive way. WebGL technique is employed and advanced algorithms are experimented and optimized repeatedly for improving the performance of large-scale visualization.

#### **9.2.3.2 Large-scale dynamic maps of taxi trips in New York City**

As reviewed in Chapter 1, the current GeoWeb mostly deals with static maps or stable space/time relations between things. Now this use case here has overcome many challenges and then exemplified implementing dynamic maps with massive dynamics that can reproduce

large-scale continuously dynamic processes of a vast number of vehicles in a big city with a broad scope and along with a long time line.

Figure 9-5 gives a screen snapshot of the dynamic map that can reflect the continuous processes with massive random dynamics of vehicles. This demo project is online for interactive exploration: [www.geonoon.net/demos](http://www.geonoon.net/demos)



Figure 9-5: Large-scale dynamic and real-time maps of taxi trips in New York City

#### 9.2.4 Real Vehicle Motion Process with GeoAccessFeed Technology

Section 9.2.3 developed a simulation system for reproducing traffic dynamics in New York City using a historical big dataset. When it comes to real-world vehicles, now this section here makes use of GeoAccessFeed technology for producing correlation dynamics between vehicle-road, and vehicle-vehicle. This is to apply GeoAccessFeed technology to design the working mechanisms for identifying, filtering, correlating and connecting vehicles and road segments in

a dynamic and adaptive way. This work involves the techniques about GeoPolicy language, AccessFeed, Geoww-URI scheme, GeoChannel Discovery Service, AccessFeed Update Service, GeoChannel Web Engine, etc.

A travel route for an individual vehicle usually consists of some road segments, which can be represented as GeoChannels respectively. So a travel motion along this route is actually the transition process between context-connected GeoChannels represented on a Context Topology network, whose concept has been illuminated in Section 7.7.

Each GeoChannel (i.e. road segment here) can be identified by a Geoww-URI, and can be access-controlled by an AccessFeed. A travel route can be represented by using a set of related AccessFeeds, which employ the TopologyLink module Information Model (designed in Section 5.4.3) for linking and transitioning between. Each AccessFeed controls the correlation between this vehicle with a specific road segment (i.e. a GeoChannel), and controls the transition between adjacent segments (i.e. connected GeoChannels on a GeoChannel Context Topology network).

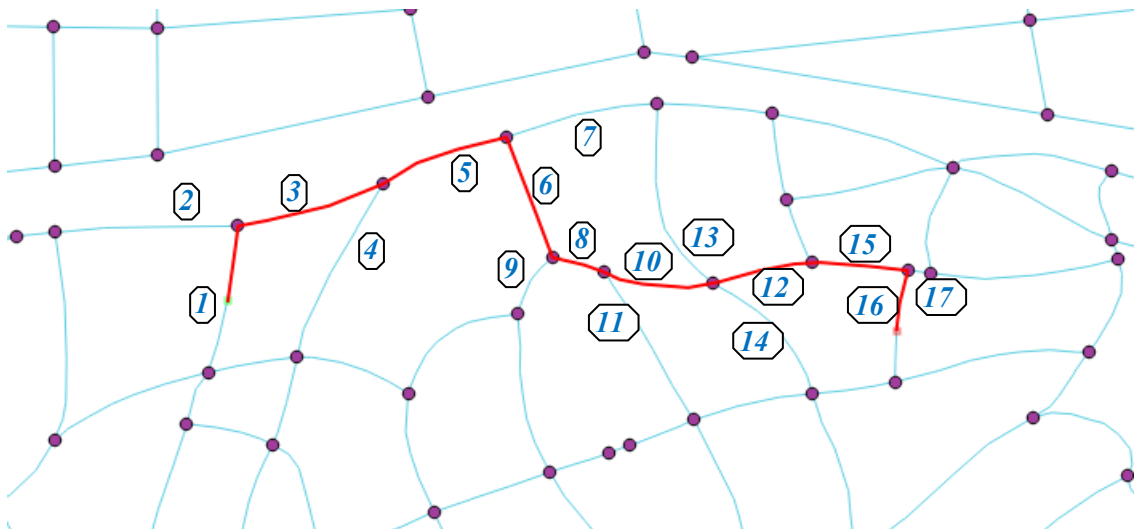


Figure 9-6: A travel route with segments represented by a set of related AccessFeeds with TopologyLink modules for controlling vehicle-segment correlation and segment-segment transition

Figure 9-6 exemplifies this kind of context-topology relations network with related AccessFeeds. Usually several road segments (i.e. GeoChannels) join with each other at a node where some branches exist for possible forward legs of a journey. Each possible branch (i.e. road segment) has its Geoww-URI and AccessFeed. Adjacent segments have linked AccessFeeds by using the

TopologyLink mechanism. For example, for a travel route (depicted as a red line), totally about 17 AccessFeeds are involved in this trip, but it actually uses part of these AccessFeeds, for example, those denoted as the number 1, 3, 5, 6, 8, 10, 12, 15 and 16. Anytime a GeoPolicy condition statement evaluates the real-time position of the moving vehicle against the road segment context. When entering a new segment/branch, this vehicle gets a new AccessFeed chosen and fetched by last AccessFeed, from a set of optional AccessFeeds available at the transition node.

For example, Listing 9-1 and Listing 9-2 exemplify AccessFeed\_1 and AccessFeed\_3, which are corresponding to segment\_1 and segment\_3 respectively. In these examples, we assume all the road segments are for one-way traffic; otherwise, more AccessFeeds with more <Link> branches will be involved.

Listing 9-1: Example: AccessFeed\_1 with TopologyLink mechanism for controlling vehicle-segment correlation and segment-segment transition

```
<geoww: AccessFeed feed_ID="1">

    <geoww:Pull>...</geoww:Pull>
    <geoww:Push>... </geoww:Push>
    <geoww: Accessibility>... </geoww: Accessibility>

    <geoww: TopologyLink>

        <geoww: Link>
            < geoww: GeoPolicy>...</ geoww: GeoPolicy>
            ②<href> http://... </href>          <!-- URL for fetching AccessFeed_2 -->
        </geoww: Link>

        <geoww: Link>
            < geoww: GeoPolicy>...</ geoww: GeoPolicy>
            ③<href> http://... </href>          <!-- URL for fetching AccessFeed_3 -->
        </geoww: Link>

    </geoww: TopologyLink>
</geoww:AccessFeed>
```

Listing 9-2: Example: AccessFeed\_3 with TopologyLink mechanism for controlling vehicle-segment correlation and segment-segment transition

```
<geoww: AccessFeed feed_ID="3">

    <geoww:Pull>...</geoww:Pull>
    <geoww:Push>... </geoww:Push>
    <geoww: Accessibility>... </geoww: Accessibility>

    <geoww: TopologyLink>
```

```

<geoww: Link>
  < geoww: GeoPolicy>...</ geoww: GeoPolicy>
  ④ <href> http://... </href>      <!-- URL for fetching AccessFeed_4 -->
</geoww: Link>

<geoww: Link>
  < geoww: GeoPolicy>...</ geoww: GeoPolicy>
  ⑤ <href> http://... </href>      <!-- URL for fetching AccessFeed_5 -->
</geoww: Link>

</geoww: TopologyLink>
</geoww: AccessFeed>

```

By analogy to HTML that hyperlinks digital-world resources for navigation on the general Web, AccessFeed can context-link and bridge real-world things mapped by the GeoChannel Web for discovery, access control, correlation, clustering and connection.

The software components framework for mapping real-world things' dynamics is illustrated in Figure 9-7. A mobile app with the client module as showed in the right part of Figure 9-7 can running on a mobile device (e.g. a driver's smart phone or an on-board equipment) at a real-world vehicle, which is a participant in reflecting and mapping dynamic and random relations in context (i.e. space/time in this use case example).

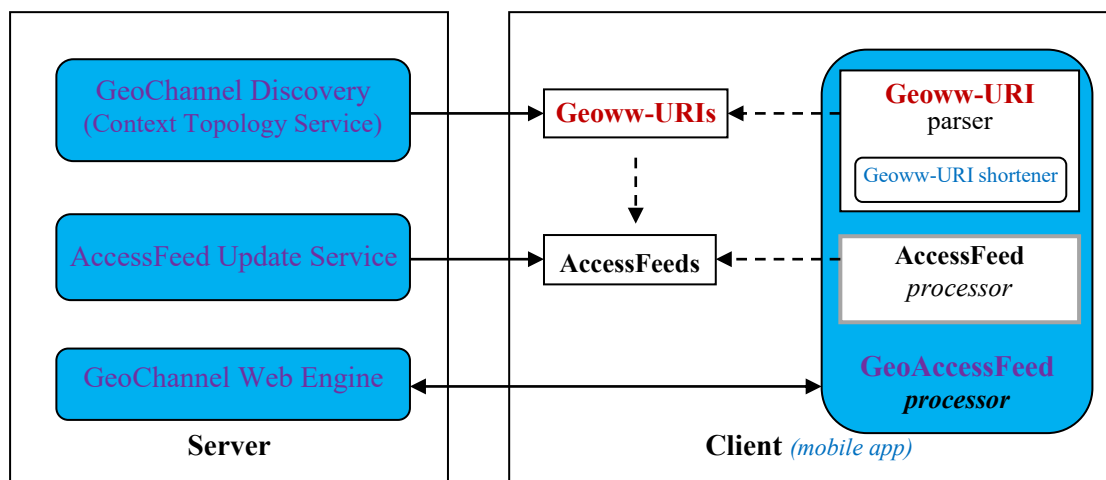


Figure 9-7: Software components framework for mapping vehicle-road relations along connected road segments

At the travel start of a journey, this client app gets the first AccessFeed for the starting road segment from GeoChannel Discovery Service. When this vehicle is moving along a travel route with connected road segments, the current AccessFeed controls the correlation between this vehicle and the road segment (i.e. a GeoChannel) which this vehicle is located on, and groups this vehicle in this GeoChannel's GeoCluster. The correlation and GeoCluster information is reported to the GeoChannel Web Engine which is mapping this kind of dynamic relations. By using its context TopologyLink mechanism, this AccessFeed can automatically fetch a new AccessFeed once the vehicle transfers to a next road segment, and then a new correlation and GeoCluster process happens and is mapped by the GeoChannel Web Engine. If all participant vehicles run this mobile app onboard, their motion with vehicle-GeoChannel correlating and vehicle-vehicle GeoClustering processes can be mapping into the GeoChannel Web Engine.

### **9.2.5 Mapping Dynamic & Random Relations to GeoChannel Networks**

The GeoChannel Web supports various context attributes (e.g. space, time, direction, speed, or any other sensor attributes), based on which real-world things' correlation and GeoCluster relations can be mapped. This use case mainly involves space/time contexts for vehicle trips motion. So we firstly construct context topology graph of road networks of the city, and then use GeoChannel technology to map dynamic and random relations. The context topology networks and the dynamic relations networks constitute GeoChannel networks that are mapped by the GeoChannel Web Engine (as indicated in Figure 9-7 and designed in Figure 8-20).

#### **9.2.5.1 Building Context Topology Networks**

OSM (OpenStreetMap) map data about New York City is chose as the raw data. Some parts that are beyond the coverage of historical taxi trips are removed for optimizing the project scope. The map data is imported into a PostgreSQL database which then transforms the map data into topology graph of the road networks. And then the graph data is transferred to the GeoChannel Web Engine system that turns the data into Context Topology Networks, which is made of more than 100,000 GeoChannels with their contexts topology.

The following figures (Figure 9-8, Figure 9-9, Figure 9-10 and Figure 9-11) illustrate some samples of the space-based Context Topology Networks for this use case.





Figure 9-8: An overview of the road networks in New York City

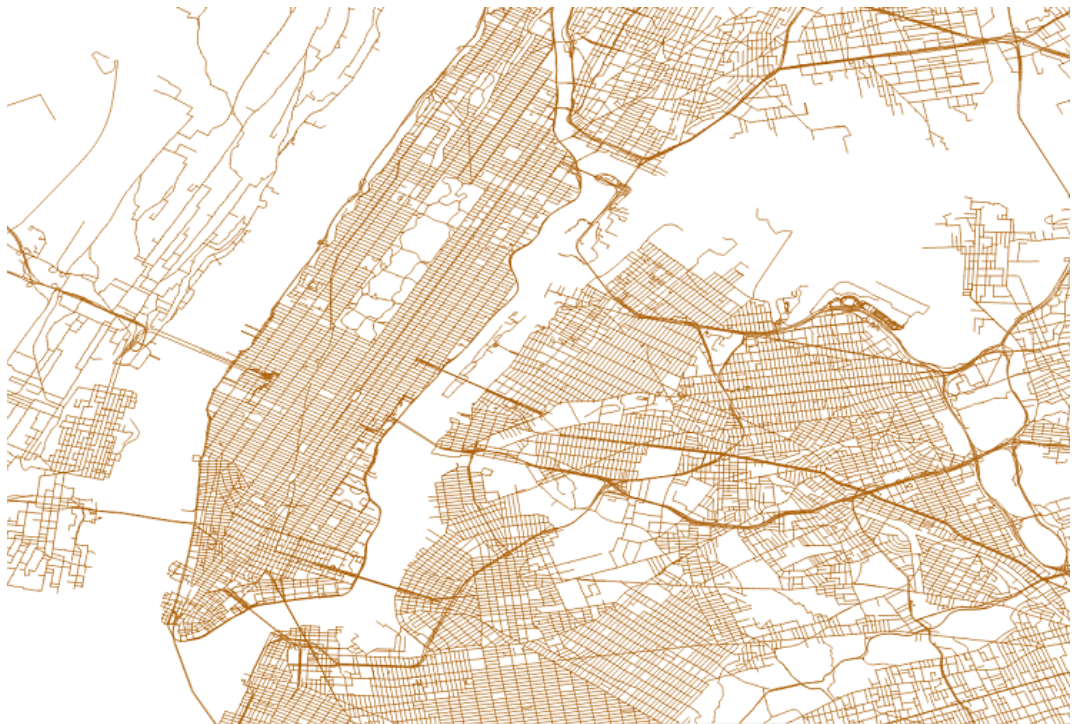


Figure 9-9: Road networks need to be transformed into GeoChannel context topology networks



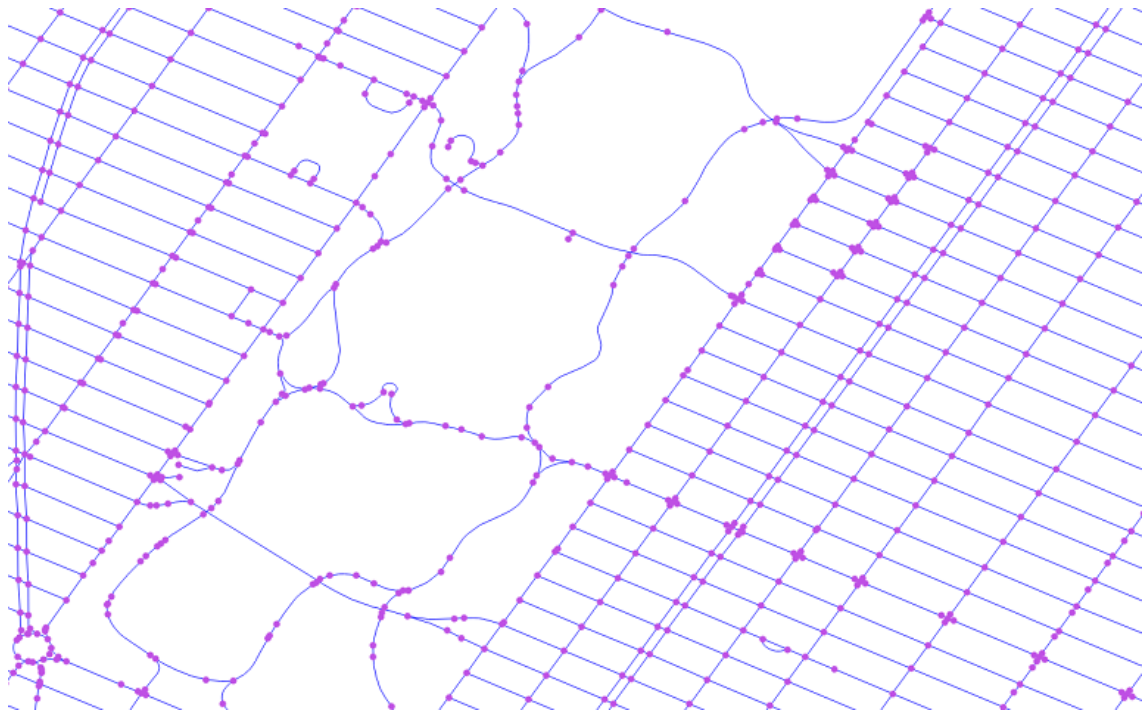


Figure 9-10: Sample 1: GeoChannel context topology network for the taxi-trip roads in New York City



Figure 9-11: Sample 2: GeoChannel context topology network for the taxi-trip roads in New York City

### 9.2.5.2 GeoClustering Vehicles with Dynamic & Random Relations

Both the large-scale motion simulation system (by using real historical big dataset) developed in

Section 9.2.3 and the AccessFeed-controlled system developed in Section 9.2.4 for real-world vehicle participants, can gather and group running vehicles on different road segments (i.e. GeoChannels) into corresponding GeoClusters. These dynamic vehicle-GeoCluster (/GeoChannel) relation networks join the context topology networks into GeoChannel networks mapped by the GeoChannel Web Engine.

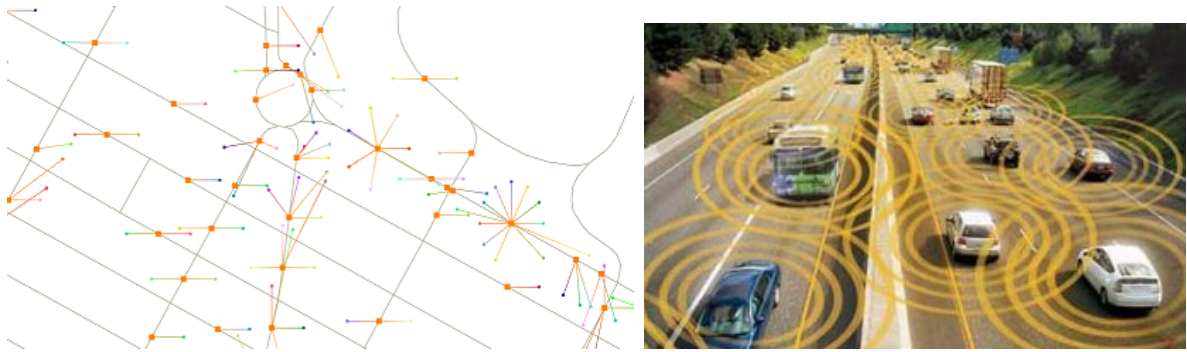


Figure 9-12: Example 1: Mapping dynamic & random relations into GeoChannel networks for roads and vehicles



Figure 9-13: Example 2: Mapping dynamic & random relations into GeoChannel networks for roads and vehicles

Figure 9-12 and Figure 9-13 exemplify the GeoClustering process that is forming dynamic GeoChannel networks, which are mapped by the GeoChannel Web Engine.

### 9.2.6 Application: Inter-vehicle Networking Interaction

Heretofore, the above sections described the work for mapping dynamic relations into the GeoChannel Web, and now this section exemplifies applying the dynamic relations for supporting innovative applications. This section does not present much detail on implementing application-specific functions, as they are beyond the research scope of core functionality focused on by this research for the GeoChannel Web.

In an ITS (Intelligent Transport System), inter-vehicle communication is a common demand. Now the GeoChannel Web can facilitate inter-vehicle interaction in a context-aware and context-relevant way. The dynamic relations between vehicles and road segments mapped by the GeoChannel Web Engine can support discovering, GeoClustering and connecting vehicles of interest. A common application scenario is about the interaction between vehicles with context relevance. For example, when a traffic congestions occur along a road or route, the vehicles involved in this jam can communicate or query related information.

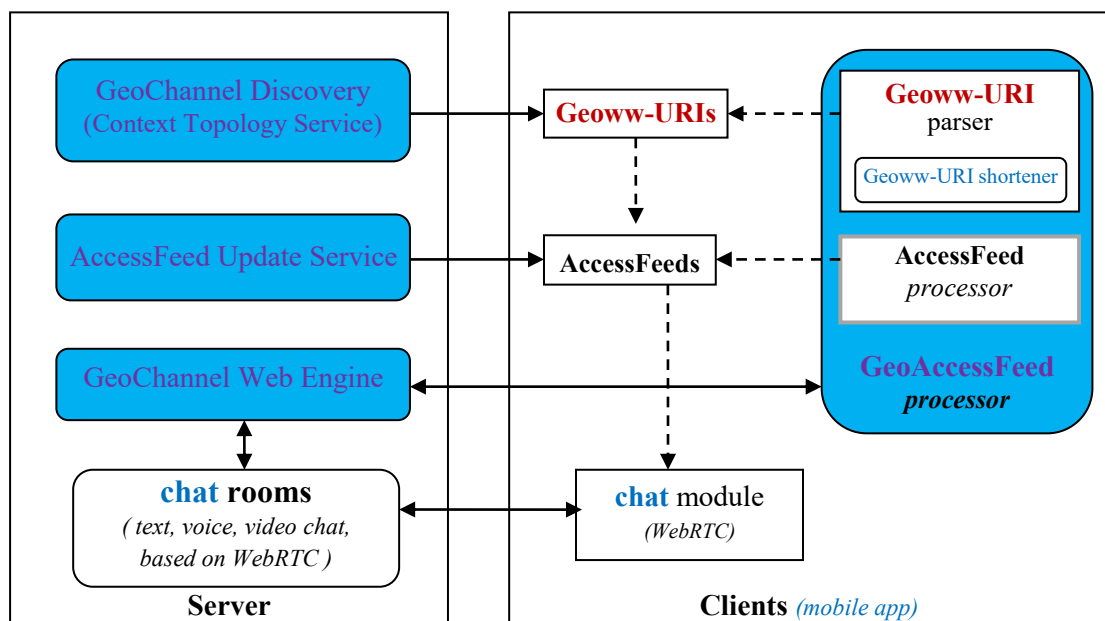


Figure 9-14: Inter-vehilce networking interaction: application of dynamic relations on GeoChannel networks

Figure 9-14 illustrates the software components framework for supporting inter-vehicle networking interaction. Drivers (or passengers) on vehicles can act as the participants in mapping dynamic vehicle-segment relations (as showed in Figure 9-7) and act as the interactive members of communication. For example, the people (on a jammed vehicle) located at one road segment can identify and connect other vehicles on the segments ahead along a route to query traffic status or the cause of this congestion, based on the dynamic relations mapped by the GeoChannel Web Engine service. In this application scenario, the GeoClusters for different road segments have their participant members who may move between GeoClusters randomly. In this case, the GeoChannel network actually functions as a dynamic social network with dynamic memberships for account-free social networking. Another use case described in Section 9.3 will exemplify the similar interaction examples on dynamic social networks with GeoClustered members.

## 9.3 Virtual-Real-World Services & Interaction in British Museum

### 9.3.1 Background & Objectives

As one of the most world-famous museums, the British Museum has developed and is providing some advanced means to facilitate on-site or online touring its vast collection of world art and artefacts. Currently, the approaches available involve using its audio guide devices for on-site tour, or using Google indoor street view maps for online exploration.

#### 9.3.1.1 Audio Guide System for on-site Tour in British Museum

The audio guide is based on dedicated mobile devices with bespoke app that delivers expert commentaries with audio, video, text and images about the items exhibited in the British Museum. An interactive map on this app can help visitors find their way around to take self-guided tours for exploring this museum.

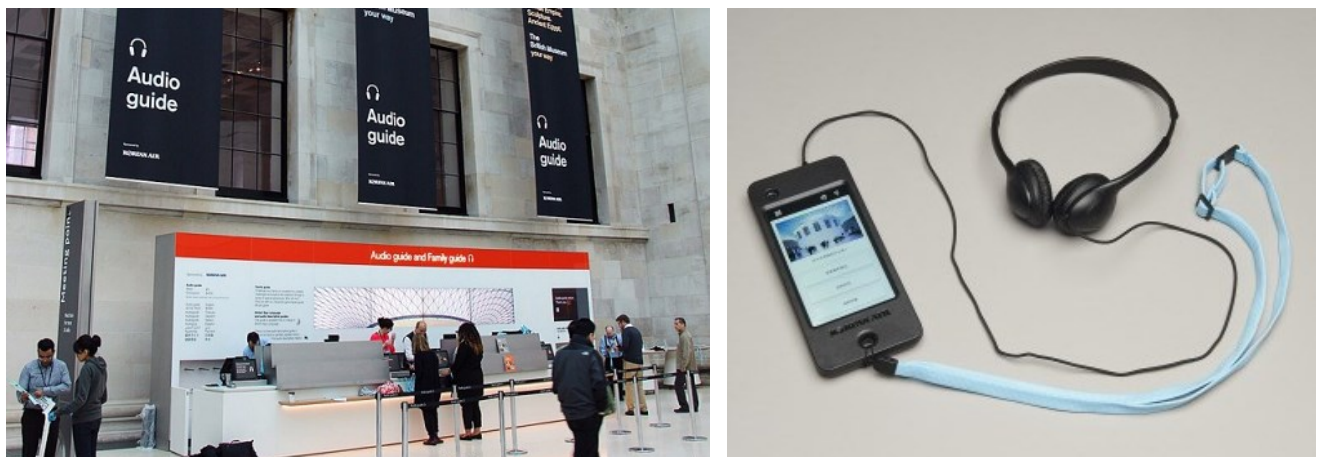


Figure 9-15: The audio guide system in the British Museum

For on-site tour, this audio guide device with app is actually not convenient or smart enough. Firstly it is embarked on dedicated devices that need to be borrowed/returned (with service price) by visitors and to be maintained by staff. Nowadays with the popularity of smart mobile phones, the prevalence of the BYOD (Bring Your Own Device) model (Buettner & Ricardo 2015) calls for accessing touring resources by using visitors' own devices.

Moreover, this system is not smart for perceiving and for controlling-access exhibits in a context-aware and automatically way. It is just a custom software for users to manually choose topics or items listed for viewing or listening. Except for a map or visual signs on screen that can be referenced for identifying the relations between the content pieces on the device and the objects in the physical space, this device itself do not have ability to sense and interact with the



ambient environment. For example, it itself does not know what objects there are around, and cannot start or end an audio-playing session automatically based on the position relation between a client and an exhibit.

### 9.3.1.2 Google indoor Street Maps for British Museum

As a joint effort, Google Cultural Institute has helped photograph the British Museum (Google 2015a) as an indoor street view project and make it online. Now a virtual walk-through enables anyone in the world with an Internet connection to explore the artefacts and collections on display.



Figure 9-16: British Museum collection viewable on Google Street View Maps

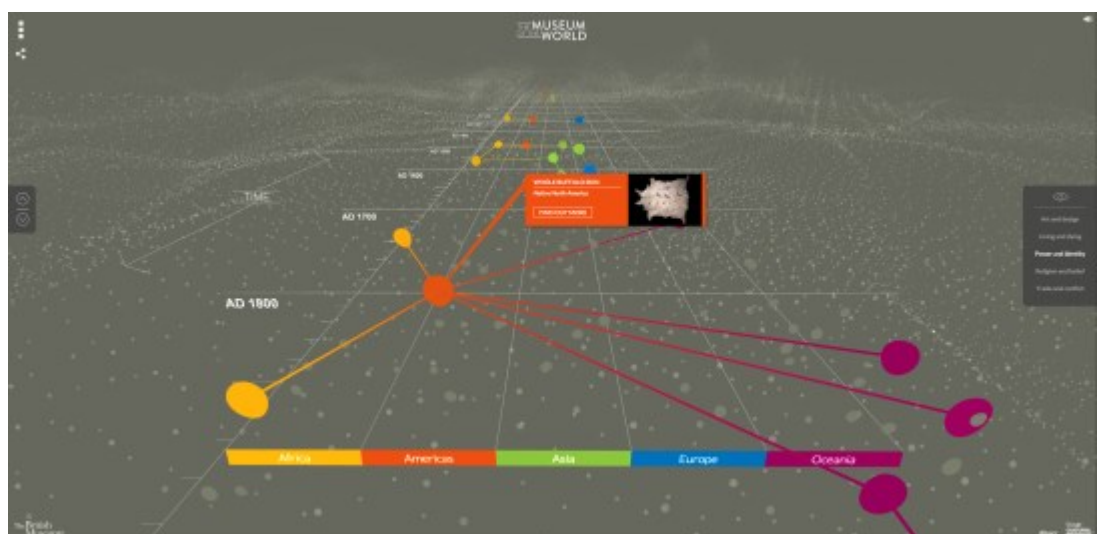


Figure 9-17: 'The Museum of the World' microsite for exploring connections between the world's cultures.

In addition to street view map for virtual tour, Google also helped develop a project named “The Museum of the World” (Google 2015b), as shown in Figure 9-17, which allows users to explore and make historical connections between the world’s cultures. This is virtually a website leveraging WebGL visualization technology that provides a graphic interface of linking objects on timelines for interactive experience.

Currently street view services mainly function on supplying visual content, but lack a flexible mechanism for incorporating other types of resources apart from visual panorama imagery, and also lack a feasible practice on connecting things (e.g. clients) based on their contexts. As a virtual world that mirrors the physical world, there is a potential as well as demand for street view services to link more physical-world resources with the virtual-world context, and to bridge things with specific context attributes in or across virtual and physical world.

#### **9.3.1.3 The Objectives for extending and improving the functionality**

The GeoAccessFeed technology provides a series of techniques (e.g. GeoPolicy language, Geo-off-on communication model, AccessFeed and Geoww-URI, etc.) for defining, identifying and access-controlling context-based resources and for bridging things (e.g. resources and clients).

This use case here attempts to improve or extend the functionality of the current systems for physical-world and for virtual-world touring of the British Museum by employing the GeoChannel technology. The GeoAccessFeed technology is not intended to completely supersede existing products for tour guides, but can fuse into them as a universal component for bridging things (e.g. resources with visitors, and visitors with visitors) in a context-aware way.

In brief, it is to address some points listed as following:

- 1) To exemplify GeoChannel techniques on its mechanism of dynamic client-resource correlation for context-aware organizing access to pervasive (e.g. location based) services. For online or on-site accessing the British Museum, this use case experiments on dynamically incorporating other types of resources apart from visual images into streetview maps, and on automatically discovering and controlling access of ambient resources for on-site visitors, in a flexible and context-aware way.
- 2) To demonstrate GeoChannel Web networking techniques on clustering and connecting things in similar or related context for supporting participative and cooperative activities. Here is to gather clients who are making on-site tour, or are making remote viewing of streetview

panoramas (i.e. GeoChannels), and then bridge them for interaction.

3) To demonstrate the novel capability of the GeoChannel Web for linking things across physical–virtual worlds. Here is to exemplify the GeoChannel techniques for supporting interaction between on-site visitors in British Museum with off-site viewer of its streetview maps.

### 9.3.2 Pervasive Services in Virtual & Physical World

#### 9.3.2.1 Context-aware Services by extending Streetview-Map Functionality

Street view maps supply panoramic images with 360-degree views for positions. This kind of services become increasingly popular as they can provide an immersive user experience for touring sites. Even not in 3D, panoramas can map our environment in somewhat a virtual-reality manner. Technically, during a viewing process, a street view panorama is relating to some context attributes, taking Google Street View maps for example, parameters can include geographic coordinates, heading and pitch of a camera or sight, etc. Figure 9-18 shows a spherical coordinate system for panorama, where the sphere centre refers to the position of a camera or an observer, and line OA refers to the perspective, i.e. a line of sight. Figure 9-18 also shows an equirectangular projection of a panorama image (i.e. the Great Court in British Museum). When viewing this panorama in Google street view maps, the context attributes of the viewer can be represented by panorama ID or position (geographic coordinates latitude, longitude), heading ( $\lambda$ ) and pitch ( $\phi$ ).

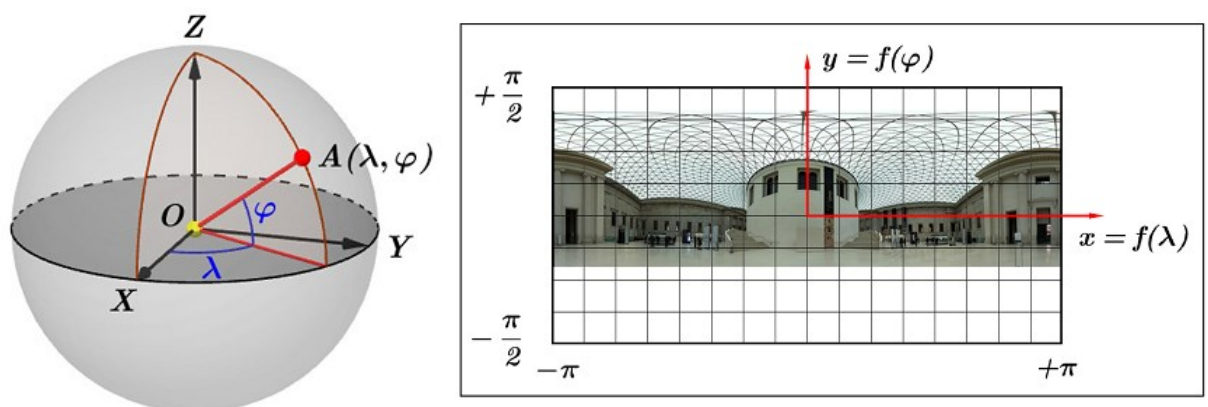


Figure 9-18: Spherical coordinate system and Equirectangular projection of panorama

Just as the principle for geo-referencing resources on the GeoWeb that can deliver resources based on location attributes, the imagery in street view panoramas is also context-referenced (or tagged). These context attributes can be used for correlating viewers with corresponding images.



This principle can apply to introduce other types of resources (e.g. audios, apart from visual images) into street view map services. There is actually this kind of practice, for example (Amplifon 2015) that does programming to add audio content into street view panoramas. This is a dedicated solution that specially works for audio content and for simple context conditions.

The GeoAccessFeed technology can work as a universal solution. It is powered by GeoPolicy language, Geo-Off-On communication model, AccessFeed and Geo-URI techniques that constitute a complete solution for any context-based application and services. It can not only dynamically correlating any resources with clients based on any context conditions using AccessFeeds in a non-programming way, but can also identify and access-control resources, cluster and connect resources or clients based on context conditions.

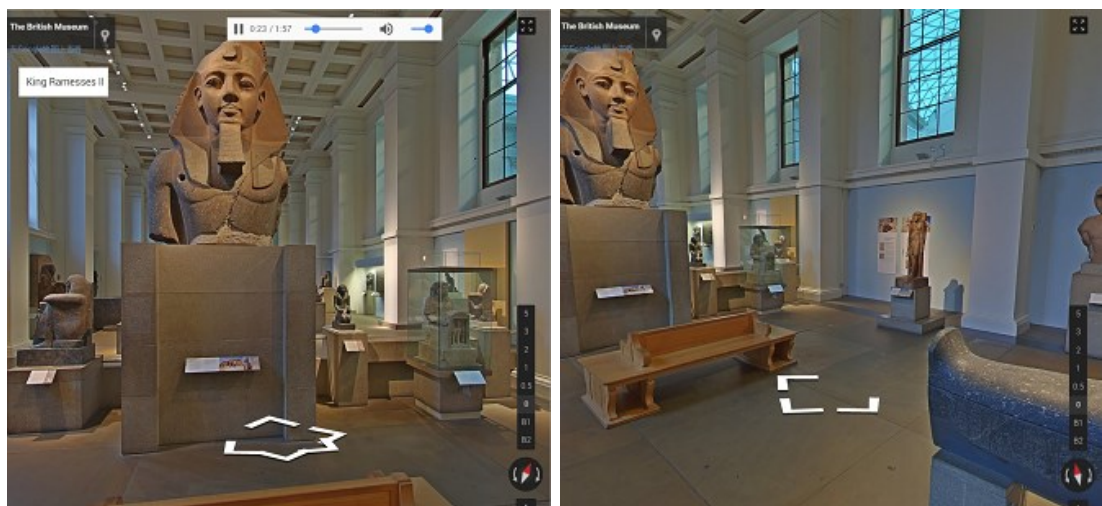


Figure 9-19: Different contexts (e.g. location, heading, pitch) of users viewing panoramas

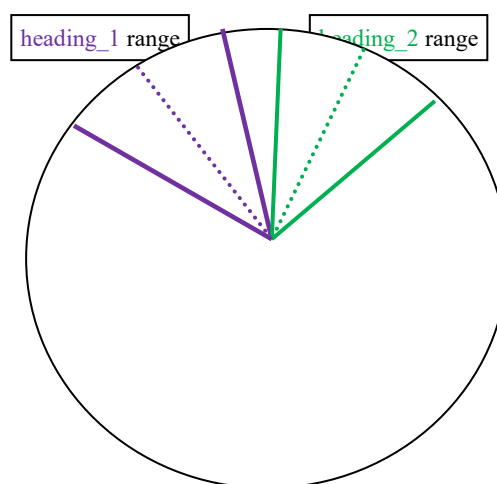


Figure 9-20: Different context (e.g. location, heading, pitch) conditions for audio resources

The above figures illustrate the principle that using GeoAccessFeed technology for correlating specific audio resources with clients in specific contexts. In Figure 9-19, the panorama scope

involving the left image has a related audio resource, but the scope involving the right image does not have. Figure 9-20 illustrates two different heading ranges. The GeoPolicy can represent the context conditions. For the target audio resource for the left image scope, an AccessFeed can use GeoPolicy language to define the context condition, which designates that a viewer can access the audio resource when his/her view heading is in the range denoted by heading\_1 range; otherwise, he/she does not access this audio resource. This AccessFeed essentially attaches this audio to a specific range of panorama view, namely it can correlate a resource with clients in a specific context (range), and can also control clients' access to this resource. And further, it can also cluster those clients that meets this context condition. A Geoww-URI can be used for identifying this audio resource with its context condition, and for identifying a cluster of clients matching this context condition.

So this section exemplifies the GeoAccessFeed technology for pervasive services in a virtual world. Here the specific example is about defining, correlating, accessing and identifying audio resources based on specific context conditions for street view maps.

#### **9.3.2.2 Pervasive Services in Physical World**

The GeoAccessFeed technology can also improve the user experiences for on-site visitors touring the British Museum. As reviewed in 9.3.1.1, the current audio guide system for on-site visitors is not smart for perceiving and for controlling-access exhibits in a context-aware and automatically way. The GeoAccessFeed technology can function as a solution to achieve pervasive services in the physical space in the British Museum.

In contrast with a virtual-world such as the street view map example (described in Section 9.3.2.1) where the viewers' context attributes (such as position, heading and pitch parameters) can be obtained by street view maps, there is a need to equip the physical space or the clients with some facilities for providing or capturing context attributes so as to configure and access pervasive services in a physical world. In the case of the British Museum, the GeoAccessFeed solution involves the following facilities:

- ◆ Devices (sensors) for the physical space or exhibits to broadcast Geoww-URIs.

The Geoww-URIs are for identifying resources and for localizing visitors. As designed in Chapter 6, the Geoww-URI technique provides a novel scheme for encoding URIs with context

conditions for identifying and for controlling-access to resources. Here Eddystone beacons are used for transmitting Geoww-URIs information, as the physical-world approach described in Chapter 7 for discovering GeoChannels. Each exhibit (e.g. artefact) is expected to have an Eddystone beacon attached for identifying itself. This beacon can also function for localizing visitors nearby.

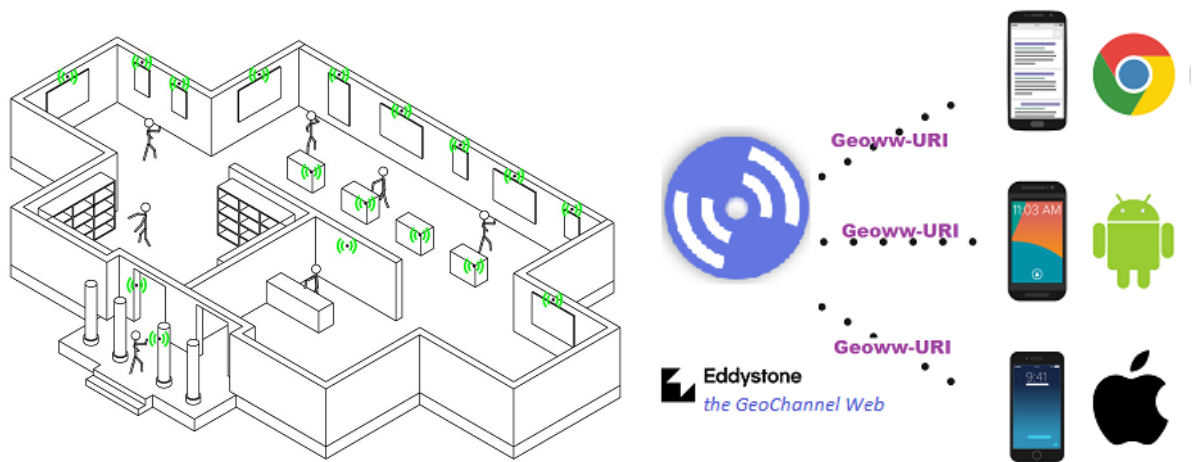


Figure 9-21: Eddystone beacons deployed in the museum for broadcasting Geoww-URIs



Figure 9-22: Vagile Eddystone beacons with reconfigurable Geoww-URIs for testing pervasive services in physical world

- ◆ Devices (sensors) for visitors to discover resources and obtain context attributes.

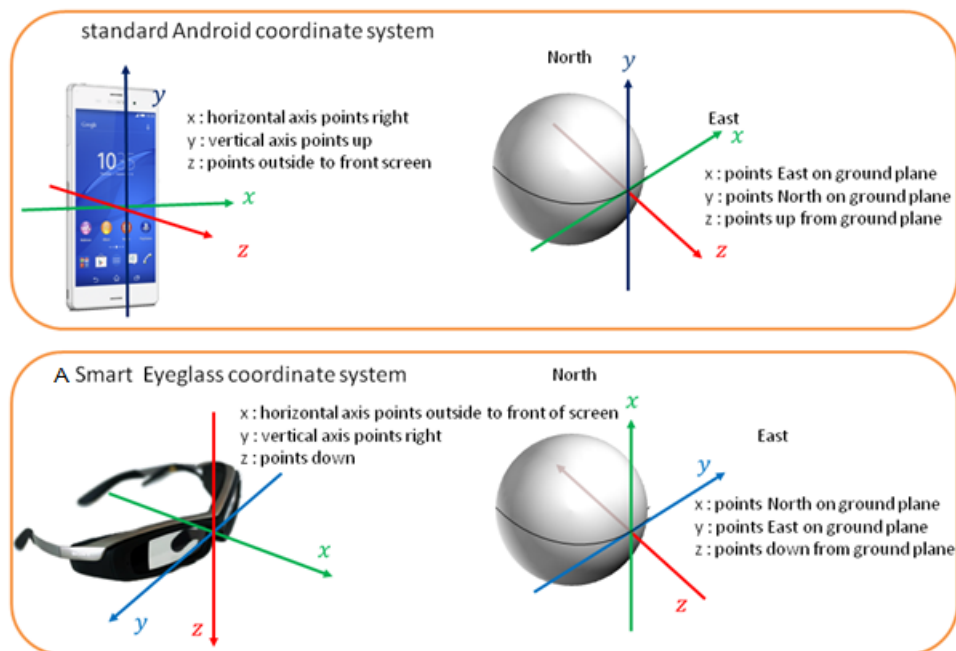


Figure 9-23: Coordinate systems for smart devices to get context attributes for on-site tour in British Museum

There are some options for client-side devices, as exemplified in Figure 9-23. Smart phones are capable of capturing visitors' context for this use case where the attributes such as position and orientation parameters are expected. The phone can scan Bluetooth signals broadcasted by Eddystone beacons related to each exhibit, so as to achieve localizing the visitors nearly. A visitor can take a mobile phone with the phone head pointing to or with the screen facing the exhibit object when he/she is looking at this artefact, then the azimuth angle of this visitor can be determined by the built-in sensors such as gyroscope and accelerator which can derive the heading and pitch parameter of this visitor's sight line. Another more suitable device is smart eyeglasses, which can sense the heading and pitch of a visitor's sight line conveniently.

The workflow process for physical-world pervasive services in the British Museum can be described as following:

- 1) Eddystone beacons broadcast Geoww-URIs that identify specific items respectively.
- 2) On-site visitors with smart mobile devices running a GeoChannel app (or a GeoChannel-enabled web application) can get the Geoww-URIs related to nearby items

automatically.

3) The GeoChannel app can parse a Geoww-URI to fetch an AccessFeed. This AccessFeed involves the GeoPolicy about a target resource (e.g. an audio commentary) and the context condition for accessing this target resource.

4) The GeoChannel app can dynamically captures a visitor's context attribute (e.g. position, heading, etc.) based on Geoww-URIs scanned from ambient environment and the sensor data from smart devices (e.g. mobile phone or smart eyeglasses).

5) The GeoChannel app can automatically and continuously evaluate real-time context condition so as to connect to or disconnect from the target resource for delivering or stopping pervasive services (e.g. audio guide).

6) Moreover, the GeoAccessFeed technology can cluster and connect on-site visitors and even off-site map-viewers for interaction. Section 9.3.3 and Section 9.3.4 will illuminate this novel functionality.

### **9.3.3 Clustered Interaction: Bridging Things in Context**

As the GeoChannel concept defined in Chapter 2, a GeoChannel generally has three constituents: a target resource, the condition for accessing this resource, and clients that match this condition and use this resource. The GeoChannel technology can not only correlate resources with clients, but also cluster clients and bridge their interaction, based on latest context. In this way, resources, clients and their relations and interaction can constitute a network named a GeoChannel network. This section exemplifies context-based social networking.

#### **9.3.3.1 For physical-world clustering**

Eddystone beacons deployed in the physical space play as a role of physical-world GeoChannels which can cluster on-site visitors nearby for interaction.

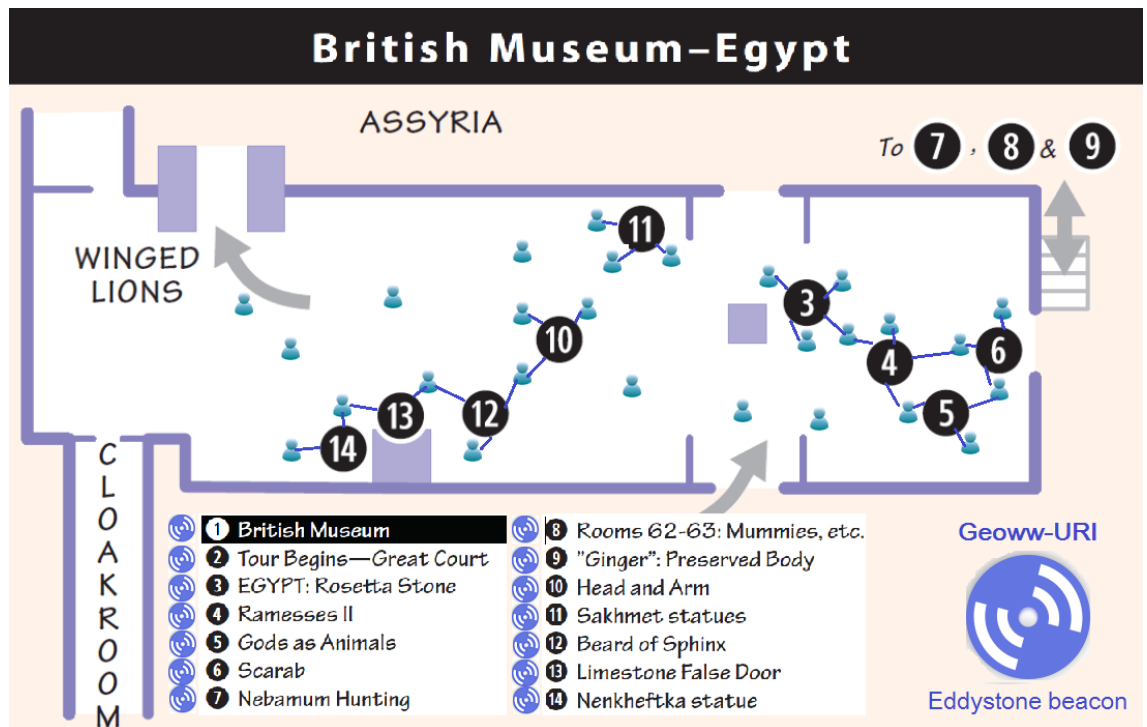


Figure 9-24: Eddystone beacons identifying objects can cluster visitors nearby

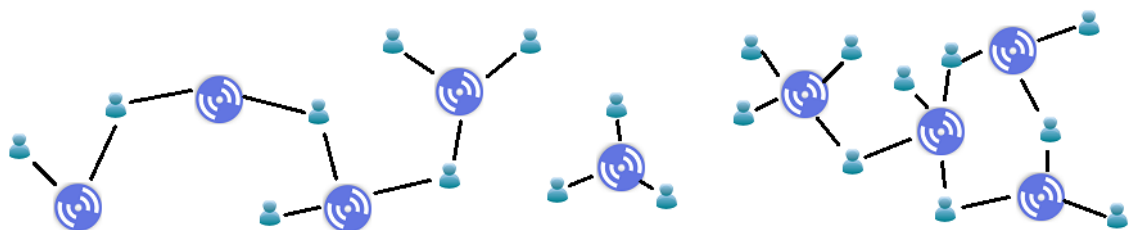


Figure 9-25: The GeoChannel network with resources and clients in physical world

### 9.3.3.2 For virtual-world clustering

As for online viewers of the indoor street view maps, they can be clustered mainly in two ways. Firstly, each panorama is virtually a virtual-world GeoChannel that can cluster viewers for interaction. When viewing a same panorama, they can comment on or talk about this scene (or exhibit), or interact with each other for exchanging ideas or opinions. And further, this use case designs another interesting way for extended clustering and social networking. The viewers who are seeing the panoramas that are adjacent to each other have the opportunity for interaction by messaging (e.g. text, voice, video messages) or chatting (voice or video chatting). This cross-panorama interaction is to simulate the real-world scenario where a person can hear or talk



to nearby people or crowds. As a real-world scene (e.g. an exhibit) may occur in several panoramas in close proximity, online viewers clustered in separate panoramas may discuss a topic about a same exhibit being seen in several adjacent panoramas.

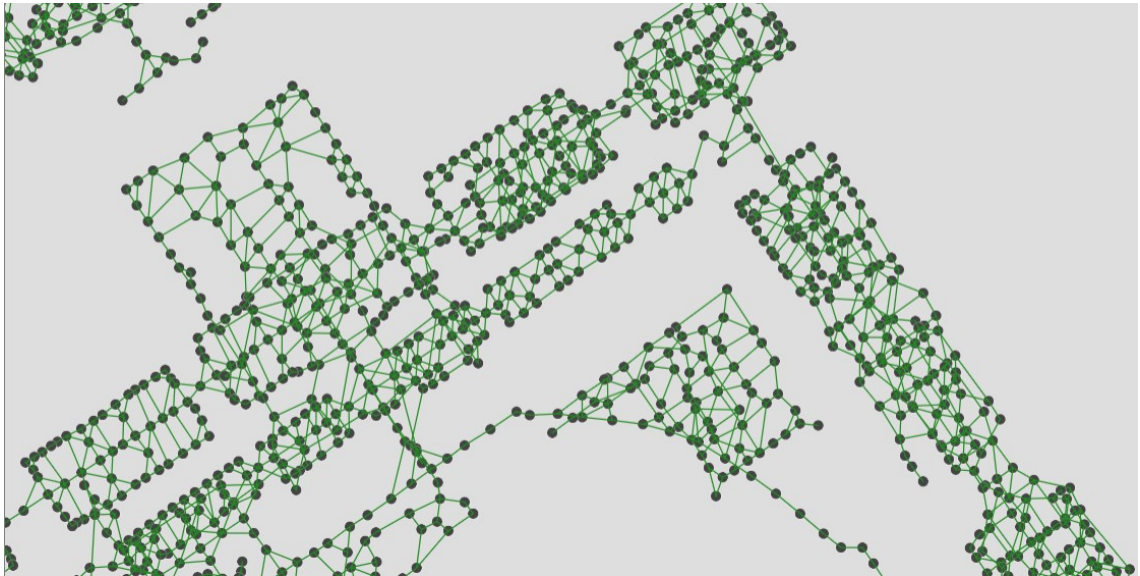


Figure 9-26: The GeoChannel context topology network for online viewers

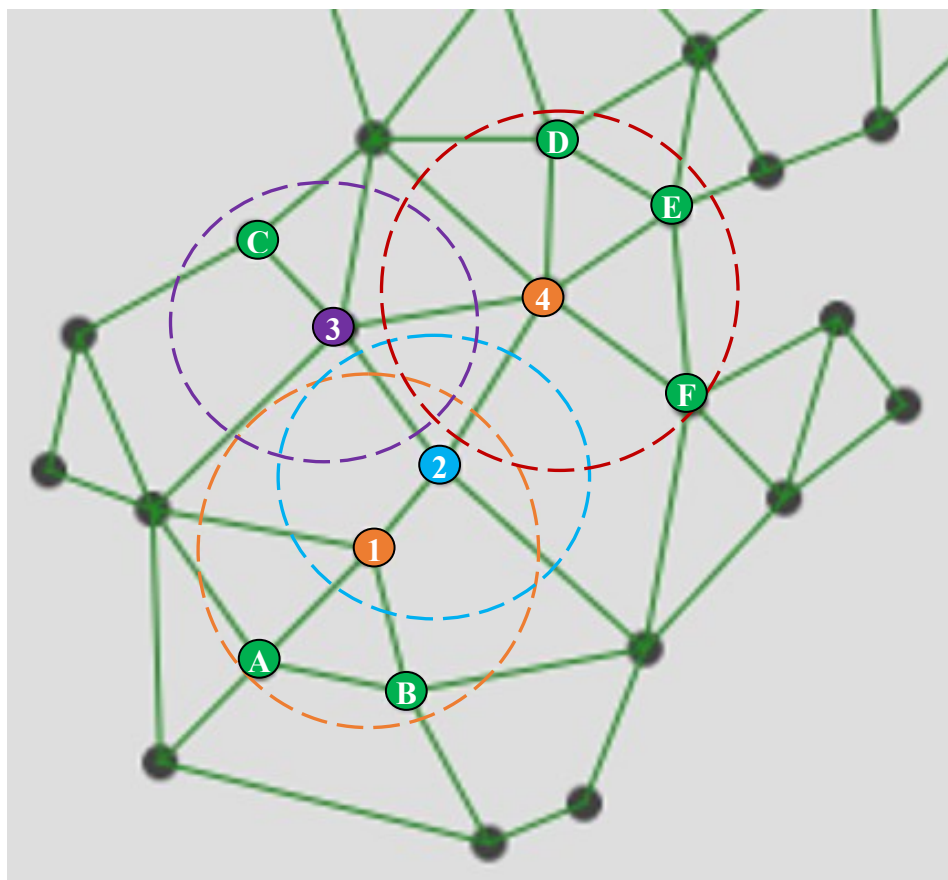


Figure 9-27: GeoBridging nearby GeoClusters on the context topology network for cross-GeoCluster interaction

Figure 9-26 illustrates the GeoChannel context topology network for online viewers. Each panorama is a GeoChannel which can gather online viewers into a GeoCluster for interaction. Figure 9-27 depicts a sample part of the GeoChannel context topology network, and exemplifies GeoBridging nearby GeoClusters for extended interaction. Not only proximity in planar distance, other real-world scenarios can also affect the extended GeoClustering of individual GeoClusters. For example, the walls that isolate seemingly adjacent GeoChannels in plan space may prevent from spreading voice into next rooms. For the brevity of description, only a small set of GeoChannels/GeoClusters, i.e. those denoted by letter or number labels, are depicted in Figure 9-27. For example:

GeoCluster ① GeoBridged with GeoCluster ②, ①, ②;  
 GeoCluster ② GeoBridged with GeoCluster ①;  
 GeoCluster ③ GeoBridged with GeoCluster ③;  
 GeoCluster ④ GeoBridged with GeoCluster ④, ⑤, ⑥

An explanation here is to further clarify GeoBridging (i.e. extended GeoClustering). For example, in the example listing above, the members (i.e. the online viewers) clustered in GeoCluster ① can interact with (e.g. speak to, hear or be heard by) the members in GeoCluster ②, ① and ②. This does not necessarily imply that the members in GeoCluster ②, ① and ② can do cross-GeoCluster interaction.

This kind of cross-GeoCluster interaction is supported by the GeoBridge mechanism in AccessFeeds that will be exemplified in Section 9.3.5.

### **9.3.4 Joining Virtual & Physical World: new patterns for Clustering & Connecting**

The GeoChannel technology can link virtual-world with physical-world involving resources, clients and interaction.

For example, Eddystone beacons deployed in the physical word and the indoor street map panoramas deployed in the virtual world can be correlated based on their similar contexts, such



as proximity positions. Then on-site visitors clustered in Eddystone beacon GeoChannels and off-site viewers clustered in panoramas can discover and interact with each other.

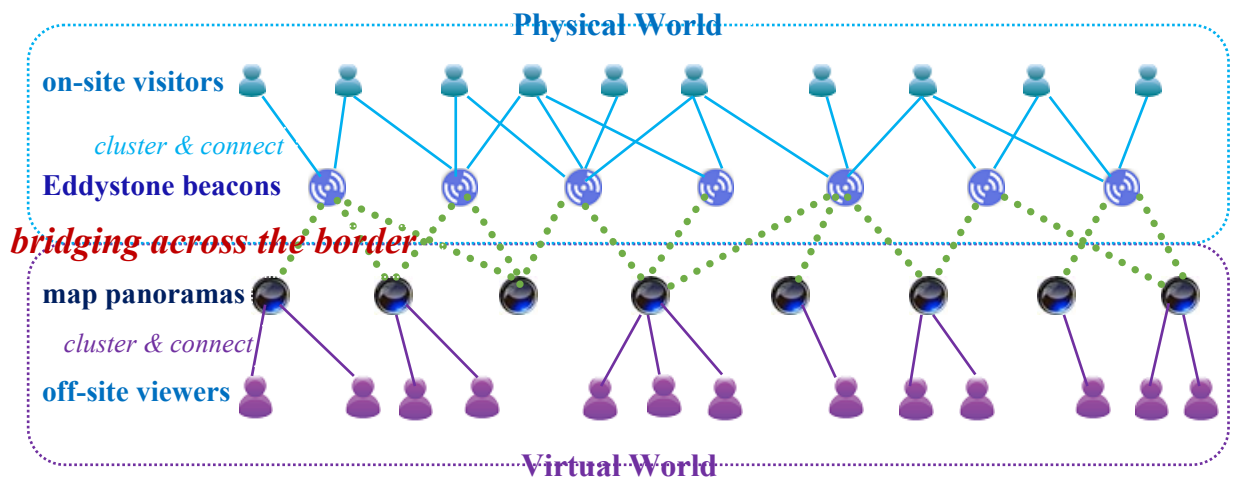


Figure 9-28: Joining the physical and virtual world ---- GeoChannel network for clustering and connecting resources & clients for pervasive services and participative interaction

### 9.3.5 Software Implementation of this use case

Based on the function design described in previous sections, this section introduces the implementation on software components for this use case.

#### 9.3.5.1 Software components framework

An integrated software components framework for fulfilling the objectives of this use case is illustrated in Figure 9-29. It involves the GeoChannel Web platform as commonly designed in previous chapters, and application-specific components to this use case for virtual-world and physical-world functions.

##### 9.3.5.1.1 Back-end servers

As showed in the left part of Figure 9-29, GeoChannel Web Engine is included for mapping various relations involved in this use cases. For example, virtual-world GeoChannels Context Topology networks (e.g. panorama and audio resources), online viewers with context attributes such as view heading and pitch, physical-world GeoChannels (e.g. beacon-identified exhibits), on-site visitors, etc. GeoChannel Discovery Service are AccessFeed Update Service are used for getting geoww-URIs and AccessFeeds of GeoChannels.

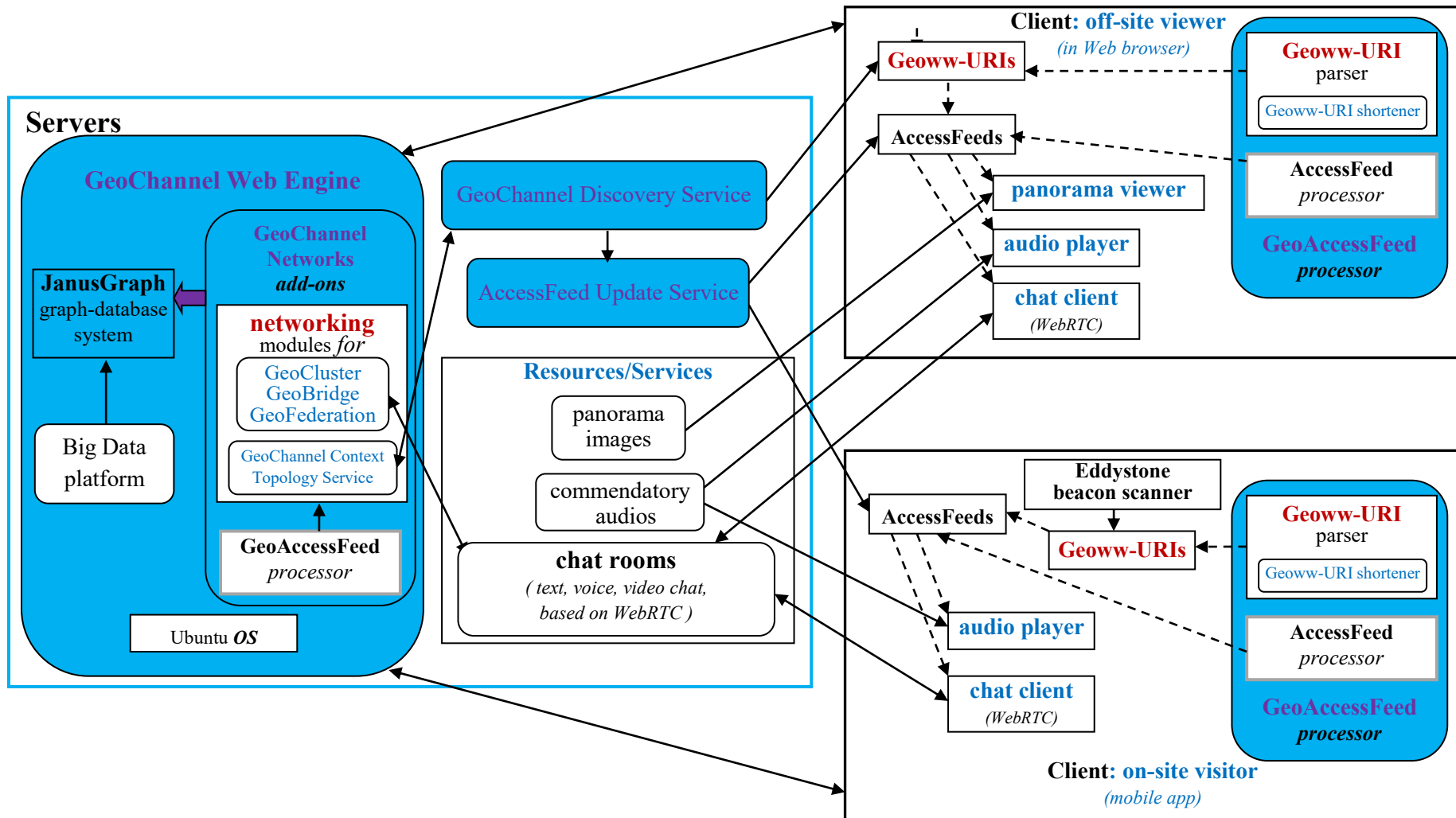


Figure 9-29: The software components framework for the Use Case 2

The back-end servers also include application-specific components. For example, the services for supplying resources of panorama images and commendatory audios; a WebRTC-based chat room service with text, voice and video chatting functionality.

#### ***9.3.5.1.2 Web app for off-site viewers***

A web browser based application (as showed in the upper right part of Figure 9-29) has been developed for off-site visitors (i.e. online viewers) to browse panoramas, listen to commendatory audio episodes and interact with other off-site visitors or with on-site visitors via messaging or chatting (e.g. text, voice and video).

The access to GeoChannel resources and networking for client-client interaction are controlled by GeoAccessFeed technology with its technical components commonly implemented in last chapter.

#### ***9.3.5.1.3 Mobile app for on-site visitors***

On-site visitors in the physical world can run a mobile app on their smart devices (e.g. mobile phones). The lower right part of Figure 9-29 shows its software components. This app can scan Eddystone beacons deployed for identifying the exhibit resources so as to locate and GeoCluster on-site visitors. The chat client module works for connecting backend chat-room service to interact with off-site viewers or with on-site visitors. The audio-player module is for using audio guide on exhibits.

Similar to the web application for off-site clients, the mobile app for on-site visitors is also underpinned by GeoAccessFeed technology.

### **9.3.5.2 AccessFeed & Geoww-URI samples**

This use case is widely based on GeoAccessFeed technology. For example, Geoww-URIs for identifying resources, GeoClusters and clients, and AccessFeeds for controlling their relations and interactions, etc. This section here especially exemplifies some AccessFeeds for GeoBridging GeoClusters into a GeoClusters (e.g. chat rooms) union that can dynamically group and bridge offsite viewers to make cross-GeoCluster interaction when they are browsing nearby panorama GeoChannels, as the application scenario depicted in Figure 9-27.

Listing 9-3, corresponding to Figure 9-27, gives some AccessFeed samples for GeoBridging nearby GeoClusters into GeoCluster unions, which are identified by the Geoww-URIs samples.

These AccessFeeds use the GeoBridge mechanism designed in Section 8.3.2 for mutually declaring GeoBridge relations to form a GeoClusters union. For example, the GeoCluster\_1 union consists of the members from GeoCluster\_1, GeoCluster\_2, GeoCluster\_a and GeoCluster\_b. This union is identified by Geoww-URI\_1\_union; the GeoCluster\_2 union consists of the members from GeoCluster\_2 and GeoCluster\_1, and is identified by Geoww-URI\_2\_union.

Listing 9-3: The samples of some AccessFeeds and Geoww-URIs for Use Case 2 on GeoBridging and identifying nearby GeoClusters for cross-GeoCluster interaction

Listing 9-3-1: AccessFeed\_1 for GeoCluster\_1 union

```
<AccessFeed src="www.accessfeed-site.com/accessfeed-1.kml">
  <Accessibility>
    <Accessor accessor_ID="dfgc81">
      <GeoPolicy>...</GeoPolicy>
      <href>www.example-site.com/s.html</href>
      <bridgeWith>
        geoww://www.accessfeed-site.com/accessfeed-2.kml::[accessor:dfg82]/:www.example-site.com/s.html
      </bridgeWith>
      <bridgeWith>
        geoww://www.accessfeed-site.com/accessfeed-a.kml::[accessor:dfg8a]/:www.example-site.com/s.html
      </bridgeWith>
      <bridgeWith>
        geoww://www.accessfeed-site.com/accessfeed-b.kml::[accessor:dfg8b]/:www.example-site.com/s.html
      </bridgeWith>
    </Accessor>
  </Accessibility>
</AccessFeed>
```

Geoww-URI\_1\_union for GeoCluster\_1 union:

```
Geoww://www.accessfeed-site.com/accessfeed-1.kml::[accessor:dfg81]/:www.example-site.com/s.html<
bridgeWith>geoww://www.accessfeed-site.com/accessfeed-2.kml::[accessor:dfg82]/:www.example-site.c
om/s.html</bridgeWith><bridgeWith>geoww://www.accessfeed-site.com/accessfeed-a.kml::[accessor:df
g8a]/:www.example-site.com/s.html</bridgeWith><bridgeWith>geoww://www.accessfeed-site.com/acce
ssfeed-b.kml::[accessor:dfg8b]/:www.example-site.com/s.html</bridgeWith>
```

Listing 9-3-2: AccessFeed\_2

```
<AccessFeed src="www.accessfeed-site.com/accessfeed-2.kml">
  <Accessibility>
    <Accessor accessor_ID="dfgc82">
      <GeoPolicy>...</GeoPolicy>
      <href>www.example-site.com/s.html</href>
      <bridgeWith>
        geoww://www.accessfeed-site.com/accessfeed-1.kml::[accessor:dfg81]/:www.example-site.com/s.html
      </bridgeWith>
    </Accessor>
  </Accessibility>
</AccessFeed>
```

Geoww-URI\_2\_union for GeoCluster\_2 union:

```
Geoww://www.accessfeed-site.com/accessfeed-2.kml::[accessor:dfg82]/:www.example-site.com/s.html<
bridgeWith>geoww://www.accessfeed-site.com/accessfeed-1.kml::[accessor:dfg81]/:www.example-site.c
```

[om/s.html](#)</bridgeWith>

Listing 9-3-3: AccessFeed\_3

```
<AccessFeed src="www.accessfeed-site.com/accessfeed-3.kml">
  <Accessibility>
    <Accessor accessor_ID="dfgc83">
      <GeoPolicy>...</GeoPolicy>
      <href>www.example-site.com/s.html</href>
      <bridgeWith>
geoww://www.accessfeed-site.com/accessfeed-c.kml::[accessor:dfg8c]/:www.example-site.com/s.html
      </bridgeWith>
    </Accessor>
  </Accessibility>
</AccessFeed>
```

Geoww-URI\_3\_union for GeoCluster\_3 union:

**Geoww://www.accessfeed-site.com/accessfeed-3.kml::[accessor:dfg83]/:www.example-site.com/s.html<**  
**bridgeWith>geoww://www.accessfeed-site.com/accessfeed-c.kml::[accessor:dfg8c]/:www.example-site.c**  
**om/s.html</bridgeWith>**

Listing 9-3-4: AccessFeed\_4

```
<AccessFeed src="www.accessfeed-site.com/accessfeed-4.kml">
  <Accessibility>
    <Accessor accessor_ID="dfgc84">
      <GeoPolicy>...</GeoPolicy>
      <href>www.example-site.com/s.html</href>
      <bridgeWith>
geoww://www.accessfeed-site.com/accessfeed-d.kml::[accessor:dfg8d]/:www.example-site.com/s.html
      </bridgeWith>
      <bridgeWith>
geoww://www.accessfeed-site.com/accessfeed-e.kml::[accessor:dfg8e]/:www.example-site.com/s.html
      </bridgeWith>
      <bridgeWith>
geoww://www.accessfeed-site.com/accessfeed-f.kml::[accessor:dfg8f]/:www.example-site.com/s.html
      </bridgeWith>
    </Accessor>
  </Accessibility>
</AccessFeed>
```

Geoww-URI\_4\_union for GeoCluster\_4 union:

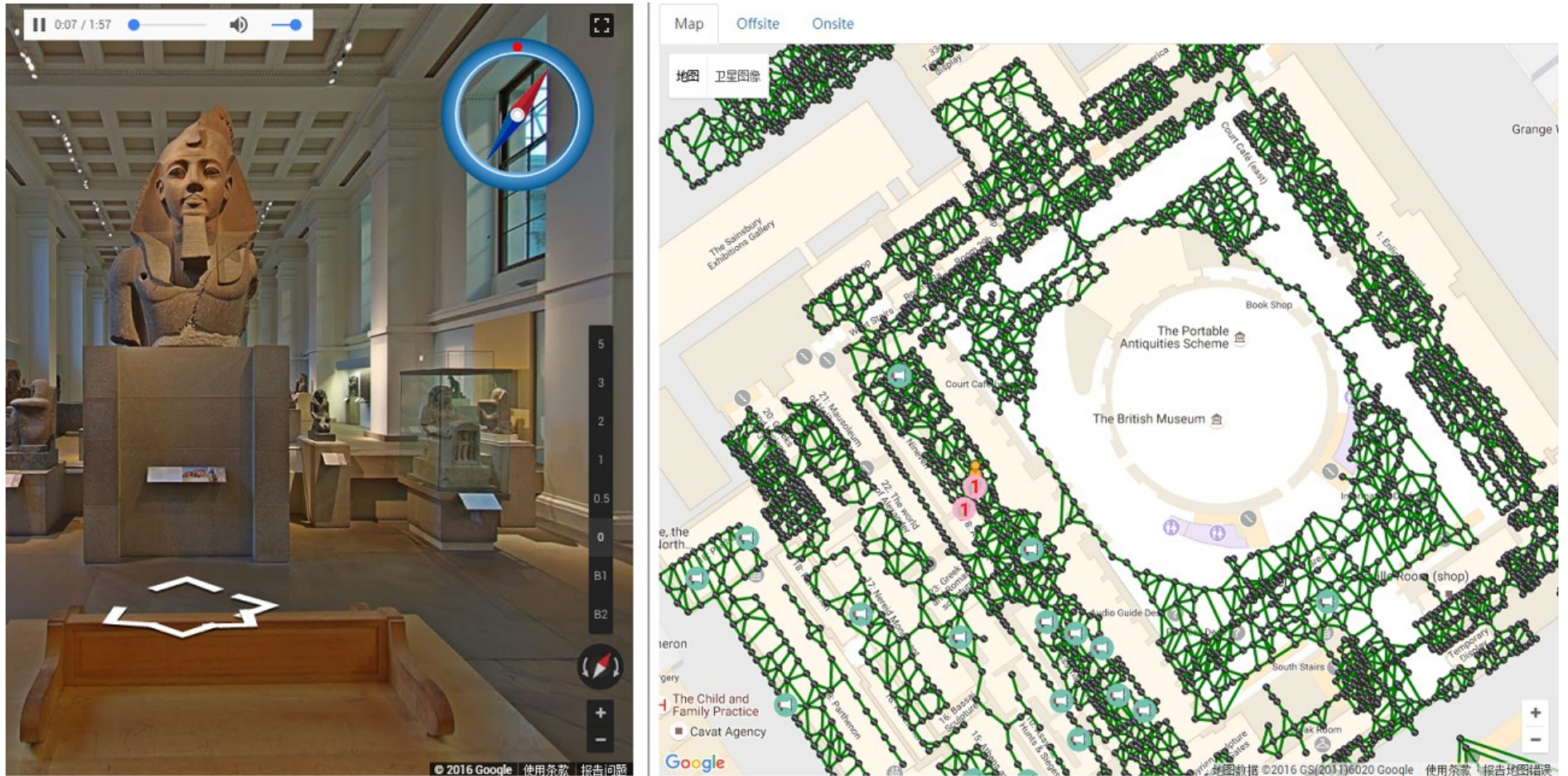
**Geoww://www.accessfeed-site.com/accessfeed-4.kml::[accessor:dfg84]/:www.example-site.com/s.html<**  
**bridgeWith>geoww://www.accessfeed-site.com/accessfeed-d.kml::[accessor:dfg8d]/:www.example-site.c**  
**om/s.html</bridgeWith><bridgeWith>geoww://www.accessfeed-site.com/accessfeed-e.kml::[accessor:df**  
**g8e]/:www.example-site.com/s.html</bridgeWith><bridgeWith>geoww://www.accessfeed-site.com/acce**  
**ssfeed-f.kml::[accessor:dfg8f]/:www.example-site.com/s.html</bridgeWith>**

### 9.3.1 Online Demonstration

This use case project is online for exploration: [www.geonoon.net/demos](http://www.geonoon.net/demos)



## Demo Project: Virtual-Real-World Services & Interaction in British Museum



Online demo: [www.geonoon.net/demos](http://www.geonoon.net/demos)

Figure 9-30: Demo project: Virtual-Real-world services and interaction in the British Museum

## **9.4 Assessment of Use Cases and Technology**

The two use cases (described in previous sections) have demonstrated and verified the GeoChannel Web technology on its functionality for mapping and applying dynamic and random relations between real-world things with their context attributes. This section here concludes by assessing the techniques and functions involved in these use cases.

### **9.4.1 Proven Technology of the GeoChannel Web**

#### **9.4.1.1 A review on the features of the use cases**

These use case projects feature the following points:

- ◆ Representative and relevant

The two use cases are appropriate and typical for demonstrating the new techniques invented in this research work.

- ◆ Clear and unambiguous

The experiment purposes and tested elements are explicitly indicated.

- ◆ Integrated, complicated and challenging

Both use cases consist of comprehensive functional components that are complicated and challenging in terms of their implementation.

- ◆ Novel and innovative

Creative ideas, thoughts and solutions are presented in the use case projects.

- ◆ Practical and useful

They are real-world projects and tested with real-world data and application scenarios.

- ◆ Available online [www.geonoon.net/demos](http://www.geonoon.net/demos)

The use case project demo systems are deployed online for public exploration.

#### 9.4.1.2 Proven techniques in the use cases

A serial of GeoChannel Web techniques have been employed in the use case applications. Now Table 9-1 gives a rollup of the involved content.

Table 9-1: The use cases demonstrate the functionality, capability and usage of the GeoChannel Web technology

<b>Use case in Section</b>	<b>Example Topics</b>	<b>The GeoChannel Web Technique Components involved</b>	<b>Technical Objectives verified</b>
Section 9.2	Mass Vehicles motion with inter-vehicle Interaction in New York City	GeoPolicy language; AccessFeed; Geoww-URI scheme; GeoChannel; GeoCluster; GeoBridge; GeoChannel networks; GeoChannel Web Engine	1) Generating & mapping massive dynamic and random relations.  2) Applying dynamic and random relations.
Section 9.3	Virtual-Real-World services & interaction in British Museum	GeoPolicy language AccessFeed; Geoww-URI scheme; GeoChannel; GeoCluster; GeoBridge; GeoChannel networks; GeoChannel Web Engine	1) client-resource dynamic correlation for pervasive services; 2) context-based interaction; 3) Linking things across physical-virtual world for interaction.

These use cases are mainly for demonstrating, testing and validating the GeoChannel Web technology itself with constituent techniques (or facilities) designed and implemented in this research. That is, the technology for mapping and applying dynamic and random relations to support bridging things in context.

The following gives a brief review on the GeoChannel Web techniques demonstrated and proven by the use cases.

#### **GeoPolicy language:**

As a condition-expressing language for defining context conditions, in the two use cases the GeoPolicy language is mainly used for representing GeoPolicy condition statements involving dynamic context (e.g. space, time, speed, direction, view heading/pitch, viewport status or maps scale level, interested GeoChannels, sensor readings for beacon scanning, etc.) parameters



changing randomly. Here this language features some capabilities:

**Non-programming:** It is independent on programming language, and can defining context policy conditions in a non-programming way.

**Flexibility:** It can seamlessly fuse into other data-representing languages (e.g. KML, ARML, etc.) for adaption into specific application domains.

**Expressiveness:** It supplies a rich set of expressive language elements for defining GeoPolicy conditions, supporting expressing virtual-world sensors (e.g. maps, panoramas) context and physical-world sensors context, etc.

**Extensibility:** By extending the language elements, it support defining domain-specific context conditions.

#### **AccessFeed:**

By analogy to HTML that hyperlinks digital-world resources for navigation on the general Web, AccessFeed can context-link and bridge real-world things mapped by the GeoChannel Web for discovery, access control, correlation, clustering and connection.

AccessFeeds play key roles in the two use cases for generating, controlling and bridging dynamic relations between things, based on their context conditions defined by GeoPolicy language.

#### **Geoww-URI scheme:**

By analogy to Http-URIs for the general web, the Geoww-URI model aims to become a native URI technique for the GeoChannel Web. As a new URI model, Geoww-URI scheme works for identifying contextualized resources, clients, clusters or their relations. Geoww-URIs are widely used in the two use cases.

#### **GeoCluster & GeoBridge:**

The two use cases have both demonstrated GeoCluster and GeoBridge mechanisms. The former is for gathering and grouping things into clusters based on context conditions, and the latter works for bridging multiple GeoClusters to become a Geocluster union for extending interaction between participants.

**GeoChannel network:**

By analogy to the general Web made up of hyperlinked-resource networks, a GeoChannel network can weave things with context attributes into a dynamic-relation network for supporting various novel applications and services, e.g. pervasive service based on context conditions. The two use cases are both centered on GeoChannel networks built and evolving dynamically.

**GeoChannel Web Engine:**

Underpinned by big-data systems, graph-processing systems and realtime-streaming technology, the GeoChannel Web Engine (as illustrated in Figure 8-20) works for mapping large-scale dynamic and random relations between real-world things with their context attributes. Its competence with scalability has been proved in the two use cases. In use cases 1, GeoChannel Context Topology Networks (involving about 365,000 GeoChannels with about 262,000 connections) and city-range 10,000 vehicles in motion weave a big dynamic-relation network. And in use cases 2, about 4,000 virtual-world GeoChannels and any number of physical-world GeoChannel with online viewers and on-site visitors in different contexts (e.g. view ID, view heading, view pitch, or physical position, etc.) make up a dynamic network; and further cross-GeoCluster and virtual-real-world interaction between clients can instantly assemble massive connection relations. There were no problems of scalability when mapping and querying the dynamic and random relations in the two use cases.

**9.4.2 Limitations and Drawbacks**

The process of developing and testing the two use cases has also identified some issues that may reflect the limitations or drawbacks to be overcome.

**1) Lengthy encoding of GeoPolicy language**

This research defined the GeoPolicy language in an extended-KML schema model (as codified in Appendix 1). This XML-based encoding for GeoPolicy language is lengthy, causing some overhead on storage and networking efficiency. Essentially, GeoPolicy Language designed in this research (as showed in Table 3-1) is encoding-neutral, as it can be represented by any encoding formats in host application environments. An improvement consideration is to employ JSON, which is more compact, for encoding GeoPolicy. However, JSON lacks some key features such as schema and namespace supports, which are sometimes crucial for validating

GeoPolicy statements and for extending GeoPolicy language with multiple sources of application domains. Nevertheless, in simple application scenarios, JSON format can be employed for encoding GeoPolicy statements that had already been validated. Fortunately there are existing tools for conversion between XML and JSON. So it is not a problem in terms of encoding formats for GeoPolicy language.

## **2) Frequent evaluation of GeoPolicy conditions**

As a condition-expressing language, GeoPolicy language supports dynamically-changing parameters included in GeoPolicy condition statements, which are re-evaluated once changes happen to parameter values that are sometimes captured by real-world sensors. In the case where updates of parameters happen frequently, the re-evaluation of GeoPolicy conditions becomes frequently as well. Over-frequent computation for evaluating GeoPolicy conditions may cause over-consuming resources such as computing and battery capacity in mobile devices. For example, the readings of inbuilt sensors on a smart phone usually produce continuous updates, and even just a little changes in readings will trigger a re-evaluation of a GeoPolicy condition involved. So a certain mechanism needs to be designed to throttle re-evaluation computation when it is unnecessary. Some solutions may be inspired by the tactics used in geo-fencing computation inbuilt in some mobile operation systems (e.g., Android or iOS) that have optimized algorithms.

## **3) Sensor dependence on mobile native code**

The GeoChannel Web maps and applies dynamic relations of real-world things based on their dynamic context attributes, which are usually captured by sensors integrated in mobile devices. Mobile operation systems usually provide native code APIs for reading onboard sensors, while current web browsers still lack full support for interacting with sensors. This state of the art hinders the GeoChannel Web (which is mainly web-browser-based) from gaining ground. Fortunately, there is a trend that mobile Web browsers are increasingly enhanced by adding native supports to sensors, for example, WebRTC (for media sensors e.g. camera, microphone), Web NFC, Web Bluetooth, WebAR, WebVR, etc. Before these prevalence, the current workaround can be a hybrid solution that can combine mobile native-program code and Web application components.

## **4) Application-specific scalability**

In the two use cases, some application-specific software components or logic for the provision,

consumption or use of application-specific services, resources or functionality have been developed, but essentially they are beyond the scope of the core techniques of the GeoChannel Web to be tested. So the performance of application-specific software components is technically irrelevant to the GeoChannel Web technology itself.

For example, for achieving large-scale and real-time dynamic maps in Use Case 1, the issue of latency may occur when a great deal of vehicles are involved in a whole city area coverage. In this case, intensive computing on both server and client sides for context (e.g. space/time) analysis, densely networking for communicating context status changes, fine-grain and large-scale rendering the updates of the maps, etc., may cause certain time lags for visualization of the maps.

And in the Use Case 2, as for the mixing-voice module for cross-GeoCluster crowd chatting among nearby GeoClusters, the issue of scalability may happen when many GeoClusters are GeoBridged to form a big GeoClusters union with many members whose voices are being mixed and then being dispatched to every participants.

The work of assessing the use cases here has not quantitatively measured the scalability performance of the application-specific software components, as they are beyond the scope of the research focus. Instead, this assessment lays the focus on the core technical components of the GeoChannel Web technology, as reviewed at Table 9-1 in last section.

## **9.5 Summary**

This chapter has exemplified two integrative use cases with respective highlights for demonstrating the GeoChannel Web technology, and has assessed the techniques involved.

Actually, it was really a great effort to have developed and tested these two use case projects, which involve big work content on the conception, planning, data preparation, software design, development and testing of these projects. Many challenges have been overcome in this process. For example, the challenges on dealing with efficient computing, networking and rendering big data for large-scale dynamic maps for the New York city; the challenges on sorting out creative ideas and solutions to cross-physical-virtual-world services/interaction for the British Museum.

These use case have further interpreted the concepts, models and techniques proposed and invented in prior chapters, and also presented, demonstrated and verified the functionality, usefulness, usage of the GeoChannel Web technology.

## **10 Research Summary and Future Work**

This concluding chapter of this thesis summarizes technological innovations conducted in this research work, reviews the research questions addressed and the research objectives achieved, and identifies future work to be undertaken for improving and utilizing this technological architecture towards a GeoChannel Web.

### **10.1 Contributions: new Techniques & Facilities**

This research has contributed an architectural model, named GeoChannel Architecture. A series of original knowledge and technologies have been introduced into the Web fabric. They involve new concepts, models, techniques and facility implementations about this GeoChannel Architecture. This section (in its subsections) just summarizes the innovative content designed and implemented in all previous chapters.

It is particularly noted that some main techniques are reflected via their information and working models for practical applications. For example, the GeoPolicy Language, Geo-off-on Communication Model, AccessFeed working mechanism, Geoww-URI scheme model, and GeoChannel Discovery functionality, etc., all of which have been designed in normative encoding models, as codified in Appendix 1, Appendix 2, Appendix 3, Appendix 4, Appendix5, Appendix 6 and Figure 6-1, etc. These are the practical outcomes of this research work.

#### **10.1.1 Framework of new Techniques and Facilities**

A general diagram, as shown in Figure 10-1, depicts the main components framework of the GeoChannel technology. This diagram can facilitate sectional summary in later sections for individual components. Figure 10-1 can be viewed in correspondence with Figure 2-8 (in Section 2.4), Figure 8-12 (in Section 8.7.1) and Figure 8-13 (in section 8.8.1) for consistent comprehension.

Centred on Figure 10-1 is “the GeoChannel Web”, which features a set of new technique and facilities as its native components. These can be analogized to the general Web that has its own technical facilities.

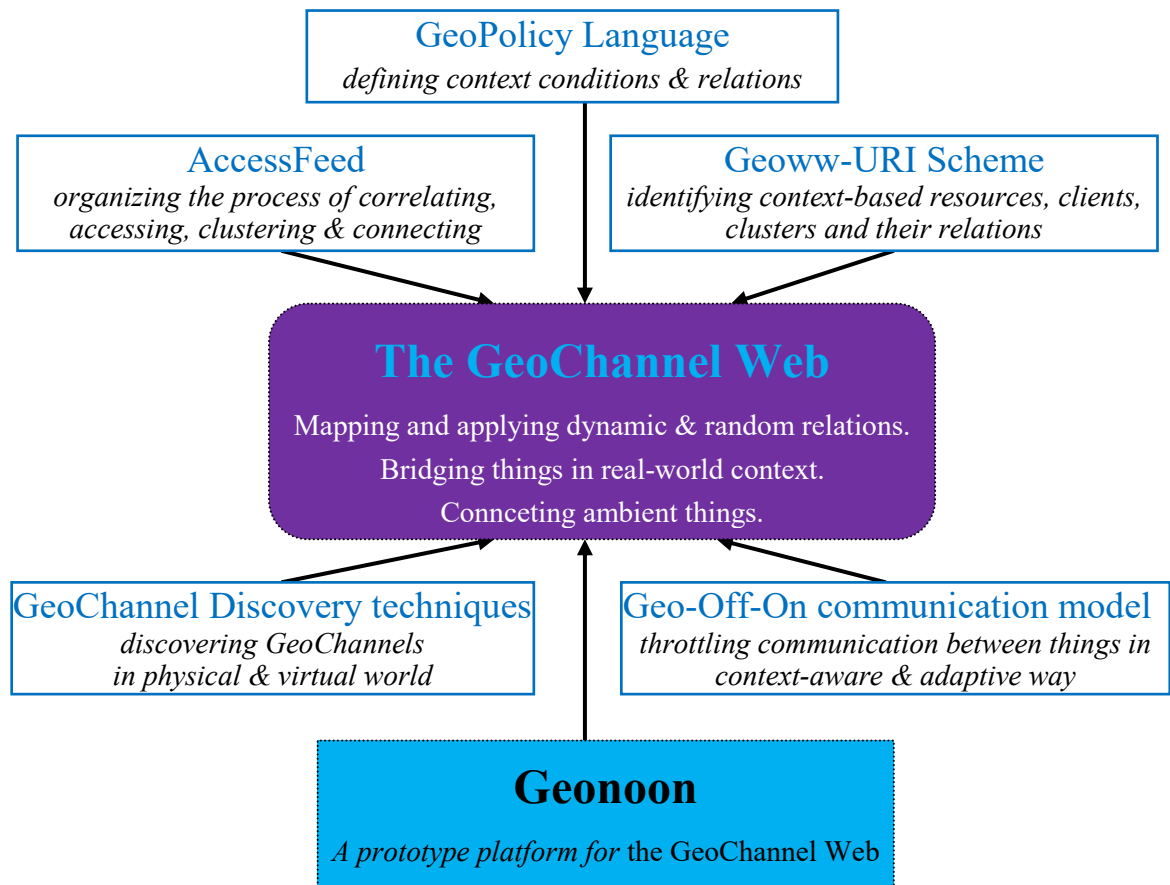


Figure 10-1: Research contributions: a set of new techniques and facilities for the GeoChannel Web

The following sections will summarize these individual components of techniques and facilities.

### 10.1.2 GeoPolicy Language

This research has developed a general-purpose language with expressive capability for representing context conditions and relations. This GeoContext policy-expressing facility, named GeoPolicy language, fills in current absence of a condition language for real-world context.

The GeoPolicy Language features some capabilities:

**Expressive enough:** This language model supplies a set of fully expressive language elements for defining GeoPolicy, i.e. expressing context attribute conditions between things. It can express various context attributes, such as space/time or other detectable attributes by current

sensors;

**Extensible:** It is extensible to support future sensors that can detect more kinds of context attributes, and to support defining domain-specific context conditions.

**Embeddable:** It can seamlessly fuse into other languages (i.e. host languages), for example other data-representing language such as GML, KML, ARML (OGC 2014), etc., to make them become conditional resources or services. Host languages and the GeoPolicy language can mix up according to both grammar. This feature can make the GeoPolicy Language widely applicable with various application scenarios where corresponding host languages exist.

**Encoding-neutral:** This language itself is format/encoding-independent. It can be represented by different encoding formats that host environments/languages are using.

**Normative:** It has normative and unambiguous model for its syntax and semantics. For example, Appendix 1 codifies an extended-KML schema for the GeoPolicy language.

The GeoPolicy language is not limited to the use by the GeoChannel architecture, but can function as a general-purpose context condition language for the general Web. The GeoPolicy language has richer control-condition-expressing capability and continuously-dynamic evaluating mechanism that overcome some related solutions, for example, the Geo-tag, Geo-filter or (Geo)XACML techniques as reviewed in Section 2.5.

The GeoPolicy Language can work as a general language tool for pervasive services, as it can define and trigger rich, complex and fine-granular context events to enable smart pervasive (e.g. location-based) services and applications.

### **10.1.3 Geo-Off-On Communication Model**

This research has designed an adaptive and cooperative communication mechanism, termed Geo-Off-On Communication Model, to cater for the GeoChannel Web which weaves real-world things into a network based on their dynamic context relations, for saving resources and improving scalability on network, computation, battery, etc.

The Geo-Off-On Communication Model takes the relation of GeoContexts (e.g. physical location, map viewport, time point or range, or some sensor attributes.) as a switch for controlling communication patterns and behaviours between interactive objects (e.g. resource-client, resource-resource, or client-client).

It is an adaptive and cooperative communication model. Here the “adaptive” indicates it can automatically adjust communication behaviour based on latest context relations between interactive objects, for throttling communication so as to save network bandwidth, reduce traffic cost, slow battery consumption of mobile devices and mitigate the overload of computation on server and client side. And the “cooperative” refers to multiple patterns (e.g. pull/push) and multiple participatory components working jointly and complementarily for communicating information;

The objective of the Geo-off-on communication model is for improving the efficiency of networking and computing, and for facilitating performance scalability of interactive objects. The Geo-off-on communication is controlled by GeoPolicy conditions that define different Geo-Client-Zones (such as GeoListenZone, GeoObserveZone and GeoBlindZone), which reflect the different degree of urgency and necessity or demand for communication interaction. Delicate working mechanisms with parameters configuration for the Pull and Push communication patterns and their shift have been designed.

This research work has mainly used the Geo-off-on communication model for delivering AccessFeed updates between AccessFeed Update Services and clients. It features the GeoContext-aware and self-adaptive capability for controlling and adjusting the communication for delivering AccessFeed updates in a resource-efficient, on-demand and personalized way.

This Geo-Off-On Communication Model featuring timeliness, efficiency, adaptability and scalability can also act as a general communication paradigm applicable to a broad of context-involved application scenarios other than being used in the GeoAccessFeed framework.



#### **10.1.4 GeoAccessFeed**

The GeoAccessFeed technology involves a set of techniques such as GeoPolicy Language, Geo-Off-On Communication Model, AccessFeed, Geoww-URI, etc.

The central part of the GeoAccessFeed technology is AccessFeed, which makes use of GeoPolicy language and Geo-Off-On communication model techniques, for controlling the correlation of things and for throttling communication between things. So AccessFeed can dynamically bridge resource with client, cluster clients and federate resources in a context-aware and automatic way. AccessFeed has self-updating mechanism, so it can cater for dynamic environment where things' contexts may change randomly.

Conditionally organizing dynamic access for a client/resource to a resource can imply controlling the dynamic relation between the involved parties. This capability can be generally used for control-accessing pervasive services (e.g. LBS), for gathering users in similar context for participative interaction (e.g. social networking), for implementing large-scale real-time maps with massive dynamics, and for an extensive range of applications or services that feature dynamic context conditions.

So AccessFeed just meets the demand of the GeoChannel Web for mapping and applying dynamic and random relations between things in real-world context.

#### **10.1.5 Geoww-URI Scheme**

This research has invented a new URI model, named Geoww-URI scheme, for identifying contextualized resources, clients, clusters or their relations.

Beyond traditional Http-URI scheme, the Geoww-URI scheme introduces context and access policy into URIs, making resources (or clients) function and used in specific contexts and constrained conditions. So, by analogy to Http-URIs for the general web, the Geoww-URI model aims to become a native URI technique for the GeoChannel Web which weaves things in real-world contexts, and can also act a native URI technique for the GeoWeb that has space/time attributes for objects.

The Geoww-URI scheme is built on and works with the AccessFeed technique. So in addition to the functionality of identifying things, Geoww-URIs can seamlessly associates with AccessFeeds to control correlating and clustering of things. Some typical functionality and usages apply to this Geoww-URI technique:

- Identify, bookmark and recommend contextualized resources, users, clusters or their relations.
- Organize (e.g. select, group, federate) resources on demand.
- Facilitate discovery and access-control of pervasive services. Geoww-URI is particularly useful for identifying, discovering and controlling access to pervasive services with their context conditions.

#### **10.1.6 GeoChannel Discovery technology**

GeoChannel Discovery technology designed in this research returns Geoww-URIs via virtual-world or physical-world approaches. GeoChannel Catalogue Services return query results containing Geoww-URIs. More significantly, the Geoww-URI technique can supply a smart approach for discovering GeoChannel resources directly from the physical world. A Geoww-URI information can be encoded and conveyed by QR code, RFID or Eddystone beacon frame signals to be captured for our physical environment. Once discovered, a Geoww-URI can automatically initiate corresponding AccessFeed(s) for controlling access to a GeoChannel.

This research has designed a series of normative models for information, interfaces and workflow processes for GeoChannel Discovery. These includes: the metadata model of GeoChannel catalogues; an equivalent implementation of OGC CSW (Catalogue Service for the Web) interfaces; various message models for interaction with GeoChannel Catalogue Services, for example, GeoChannel metadata feeds model, the extended OpenSearch service description document format and request format, and the extended-KML-encoded schema for responses of GeoChannel search operations, the workflow process models for discovering of GeoChannel resources using physical/virtual-world approaches, etc.

### **10.1.7 The architecture of the GeoChannel Web**

This research has framed some core concepts and models. Firstly, the concept of GeoChannel can generalize and characterize real-world things with dynamic relations based on their context attributes. A GeoChannel refers to a contextualized resource and its clients that correlate with each other based on latest context conditions. So a GeoChannel is virtually a unit or group of dynamic relations of resource-client and client-client. Multiple GeoChannels may make up a GeoChannel network, which comprise context-based relations between things. This research (in Chapter 8) designed some mechanisms for operating GeoChannel networks, such as GeoCluster, GeoBridge and GeoFederation, etc.

The GeoChannel Web is a brand-new concept, architecture model and infrastructure originally proposed and designed in this research. It is a new type of Web for mapping and applying dynamic and random relations between real-world things based on their contexts. It will be made up from extensive GeoChannel networks on a globe scale. From a certain perspective, the GeoChannel Web is an architectural model and a relations system platform.

The new techniques and facilities as summarized in prior sections are just for building the architecture of the GeoChannel Web. These enabling technical components can work for defining, identifying, discovering, communicating, correlating and accessing things with context conditions in a context-controlled topology network of the GeoChannel Web.

### **10.1.8 Geonoon: a prototype platform**

A prototype platform named Geonoon has been developed as a prototype implementation of the concepts, models and techniques proposed and designed in this research. Currently the Geonoon platform mainly involves some backend and frontend facility components, which have supported the developing and testing of the use case projects. It is noted that the software development of this prototype platform has taken up a considerable part of time and efforts of the whole research work. The Geonoon expects to become a productive platform to support extensive applications and services for the GeoChannel Web.

## 10.2 Addressing Research Questions

The technological aim of this research is to achieve context-based dynamic correlation between real-world things in a context-aware, on-demand, automatic and scalable way, so as to fulfil functional objective towards a GeoChannel Web for mapping and applying dynamic and random relations between real-world things to support various context-based applications and services, for example, comprehensively real-time dynamics, pervasively location-based services and GeoClustered users interaction.

With the techniques and facilities in place (as summarized in Section 10.1), now it is time to reconsider the research questions, checking how they have been addressed, and to verify that the research objectives have been achieved.

The main research question formulated (in Section 1.2) for this research work is:

What architectural system can conduct dynamic correlation of real-world things based on their context conditions, so as to improve the current Web for mapping and applying real-world dynamic and random relations to support various context-based application and services?

This general research question, in terms of its constituent technical components as identified and stated in Section 2.4, has been divided into some sub-questions, which are now reviewed as following with their answers by using the technologies summarized in Section 10.1.

### 10.2.1 Answering the research sub-question 1

Table 10-1: Answering the research sub-question 1

Question 1	What concept model with enabling components framework can work for mapping and applying dynamic and random relations between real-world things?
Answered in	Chapter 2, Chapter 8
Techniques / facilities involved	Concepts and models; GeoChannel; GeoChannel networks; the GeoChannel Web

This research has answered this question by having framed some main concepts and models.

Firstly, the concept of GeoChannel can generalize and characterize real-world things with dynamic relations in terms of their representation and identification. Generally a GeoChannel refers to a contextualized resource and its clients that correlate with each other based on context conditions. A GeoChannel generally involves three constituents, i.e. target resource, context condition, and condition-matched clients. So a GeoChannel is virtually a unit or group of dynamic relations of resource-client and client-client, based on their context conditions. Multiple GeoChannels may make up a GeoChannel network, on which resource-resource may correlate with each other in case they meet specific context conditions.

On a globe scale, extensive GeoChannel networks can form a new type of Web, termed the GeoChannel Web, for mapping and applying dynamic and context-based relations between real-world things. From a certain perspective, the GeoChannel Web is an architectural model and a relations system platform with enabling technical components for discovery, connection and interaction of things involved in this context-aware network topology.

### 10.2.2 Answering the research sub-question 2

Table 10-2: Answering the research sub-question 2

Question 1	What language can be generically competent for representing context conditions between real-world things?
Answered in	Chapter 3
Techniques / facilities involved	the GeoPolicy Language

The GeoPolicy Language can answer this question. Section 10.1.2 summarizes its capability for representing context conditions between real-world things.

### 10.2.3 Answering the research sub-question 3

Table 10-3: Answering the research sub-question 3

Question 1	What model can achieve adaptive communication between mutually interested things based on their latest context relations?
Answered in	Chapter 4
Techniques / facilities	the Geo-Off-On Communication Model

involved	
----------	--

As summarized in Section 10.1.3, the Geo-Off-On Communication Model can provide adaptive mechanism for throttling communication and interaction between things based on their latest context relations.

#### 10.2.4 Answering the research sub-question 4

Table 10-4: Answering the research sub-question 4

Question 1	How can real-world things be dynamically correlated based on their latest context?
Answered in	Chapter 5
Techniques / facilities involved	GeoAccessFeed

The question can be answered by the GeoAccessFeed technology as summarized in Section 10.1.4.

#### 10.2.5 Answering the research sub-question 5

Table 10-5: Answering the research sub-question 5

Question 1	What URL model can work for identifying things and relations with their latest context conditions?
Answered in	Chapter 6
Techniques / facilities involved	Geoww-URI scheme

The Geoww-URI scheme, as summarized in Section 10.1.5, is just the answer to this question.

#### 10.2.6 Answering the research sub-question 6

Table 10-6: Answering the research sub-question 6

Question 1	How can contextualized things be discovered in a context-aware way?
Answered in	Chapter 7
Techniques / facilities involved	GeoChannel Catalogue Service, physical-world approaches (e.g. Eddystone beacon, QR-code, RFID, etc., with Geoww-URI)

GeoChannel Discovery technology, as summarized in Section 10.1.6, provides both virtual-world and physical-world approaches for discovering GeoChannel in a context-aware way.

### 10.2.7 Answer to the general question

With the answers ready to those sub-questions (as reviewed in the previous sections), the general question, which takes the sub-questions as constituent components technically, can now be addressed.

Table 10-7: Answering the general research question

Question	<b>What architectural system can conduct dynamic correlation of real-world things based on their context conditions</b> , so as to improve the current Web for mapping and applying real-world dynamic and random relations to support various context-based application and services?
Answered in	this thesis
Techniques / facilities involved	GeoChannel (concept and model); GeoPolicy Language; Geo-Off-On Communication Model; GeoAccessFeed, AccessFeed; Geoww-URI scheme; GeoChannel Discovery technology; GeoChannel networks, the GeoChannel Web (concept, model, architecture); Geonoon platform (a prototype)

As listed in Table 10-7, virtually all the new techniques and facilities devised in this research are used for answering the general research question. In Figure 10-2, a brief diagram illuminates how these components work together for conducting dynamic correlation of real-world things based on their context conditions. In this diagram, several GeoChannels are depicted, each of which has its resource and clients. The GeoAccessFeed technology involves its technique components such as GeoPolicy language, Geo-Off-On communication model, AccessFeed, Geoww-URI, etc. GeoChannel Discovery technology can obtain Geoww-URIs, which identify GeoChannels respectively. Geoww-URIs can fetch AccessFeeds, which dynamically control the correlation between GeoChannel resources with their clients based on latest context conditions.

GeoChannel networks can be formed inside a GeoChannel with dynamic connections between a resources and its clients, or between clients and clients; and can also formed across GeoChannels with connections between cross-GeoChannel resources and clients. The GeoChannel Web is a more generalized network which involves all GeoChannel networks on a globe scale.

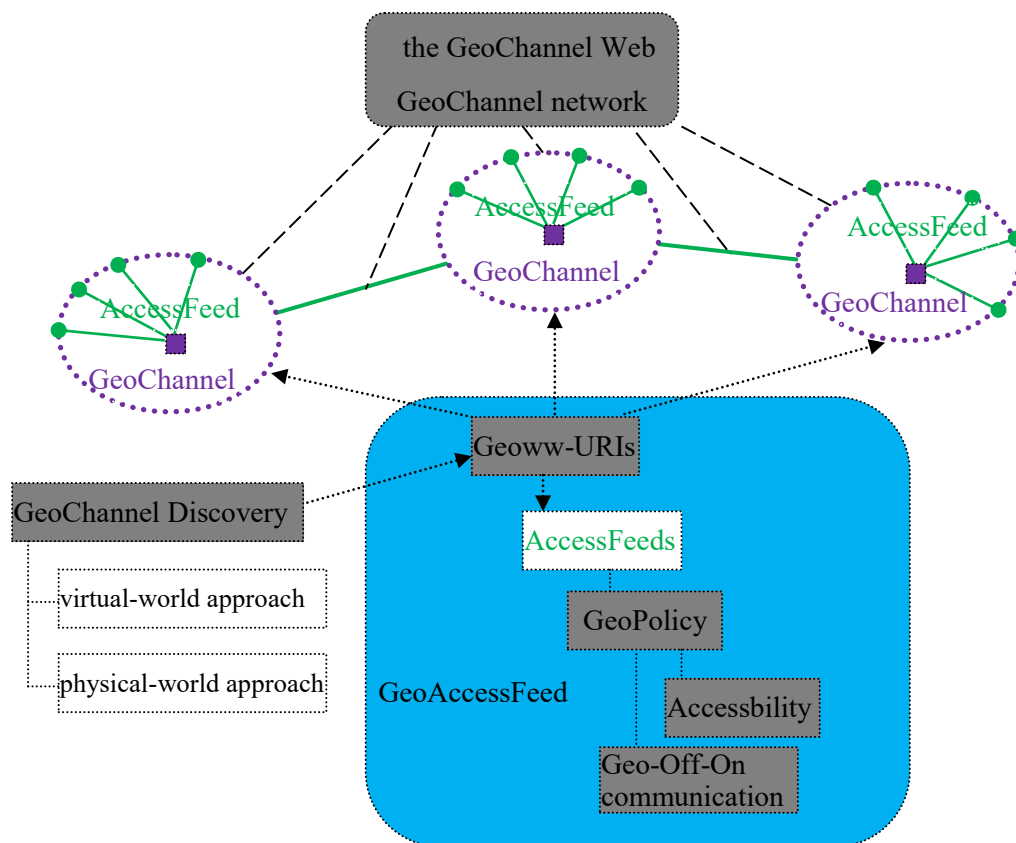


Figure 10-2: Answer to the research question: an GeoChannel architecture for correlating things dynamically

All these new techniques and facilities can work jointly and systematically for supporting a new architecture model, i.e. the GeoChannel Web, for addressing the general research question.

**To conclude**, the technological aim and functional objective of this research have been achieved. This architectural technology can conduct context-based dynamic correlation between real-world things in a context-aware, on-demand, automatic and scalable way. As a result, the functional objective towards a GeoChannel Web is also fulfilled. The GeoChannel Web works for mapping and applying dynamic and random relations between real-world things to support various context-based applications and services, for example, comprehensively real-time



dynamics, pervasively location-based services and context-clustered users interaction.

### **10.3 Evaluation & Discussion**

The GeoChannel Web is a brand-new concept, architecture model and infrastructure with its enabling techniques originally proposed and designed in this research. There have not been seen such similar facilities or constituent techniques in literature. So it is not applicable to directly compare the new technology invented in this research with other existing technologies. However, it is still meaningful to make some analogies or comparison between the new creations with related technology or practice.

The sub-sections below firstly illuminates the peculiarity of the GeoChannel Web, then identifies the limitations of this research work.

#### **10.3.1 The GeoChannel Web and other types of Web**

There is actually only one general Web exists, but with various focuses and usages that forms some types of Web (as reviewed in Chapter 1), which feature their peculiarities with enabling techniques and serving purposes respectively. The GeoChannel Web proposed in this research differs from existing types of Web in some aspects.

##### **10.3.1.1 Diversity of Functionality, Usages & Purposes**

This section is to answer two questions:

##### **1) *How the GeoChannel Web differs from existing types of Web?***

The general Web and its diversified types of Web, all reflect network relations between things, just as the term “Web” implies. This is the similarity of all types of Web.

The GeoChannel Web, as a new type of Web, mainly works for mapping and applying dynamic, random and context-based relations between real-world things. This is the main difference from other existing types of Web, which largely deal with relatively stable or static relations between things, or do not have a common architectural facility for applying dynamically-changed

relations based-on context conditions. The peculiarity and diversity of some types of Web are clarified in Figure 10-3 (that has ever shown in Section 1.1.2.2).

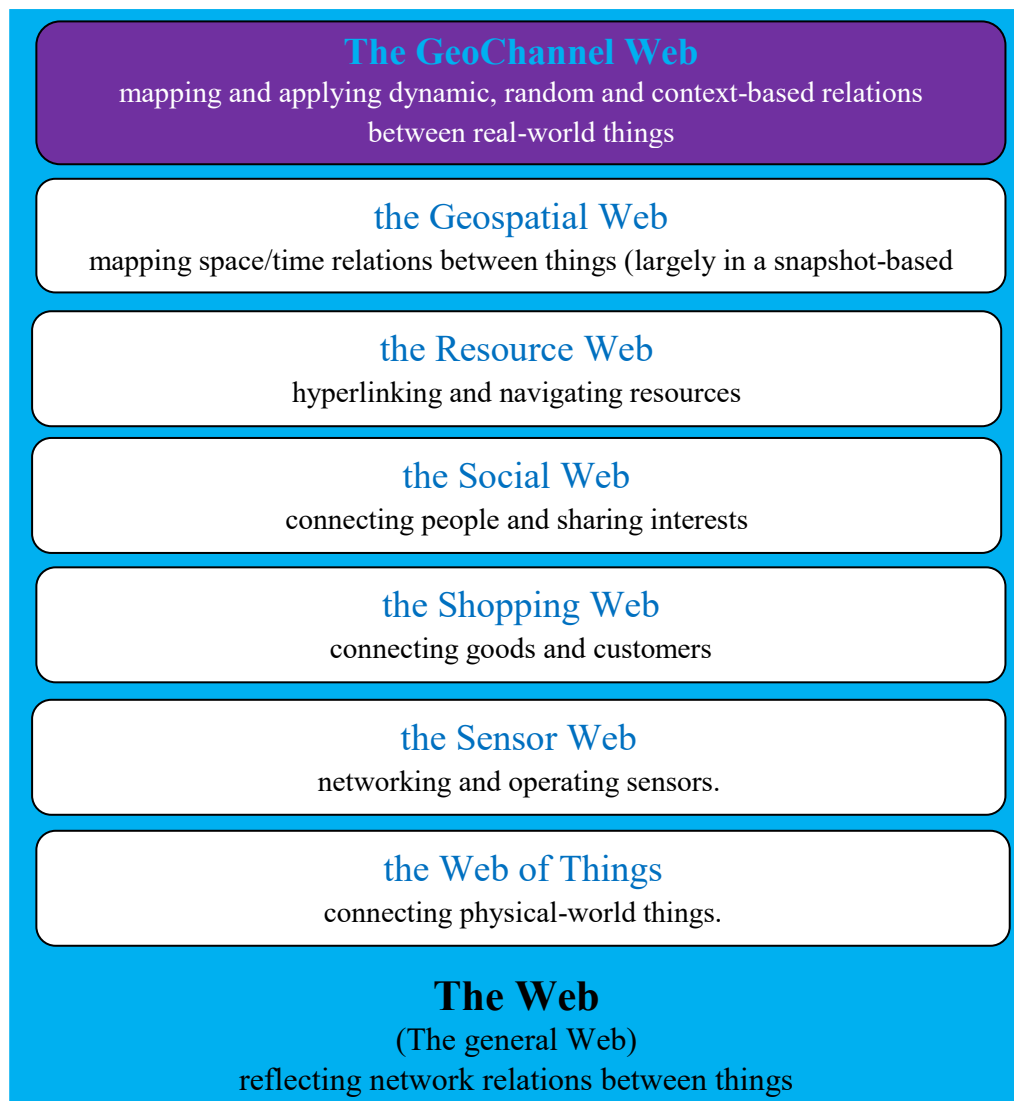


Figure 10-3: The GeoChannel Web differs from othe types of Web

## ***2) How the GeoChannel Web technology fits into the current Web architecture?***

In a wider perspective, the GeoChannel Web proposed essentially adds a Context-Organize Layer into the architectural stack of the general Web, making the latter and all types of Web be able to connect things in a context-aware way. The native enabling technique and facility components developed for the GeoChannel Web can also work for the general Web and its diversified types of Web. Figure 10-4 illuminates the relations between the GeoChannel Web and other types of Web.

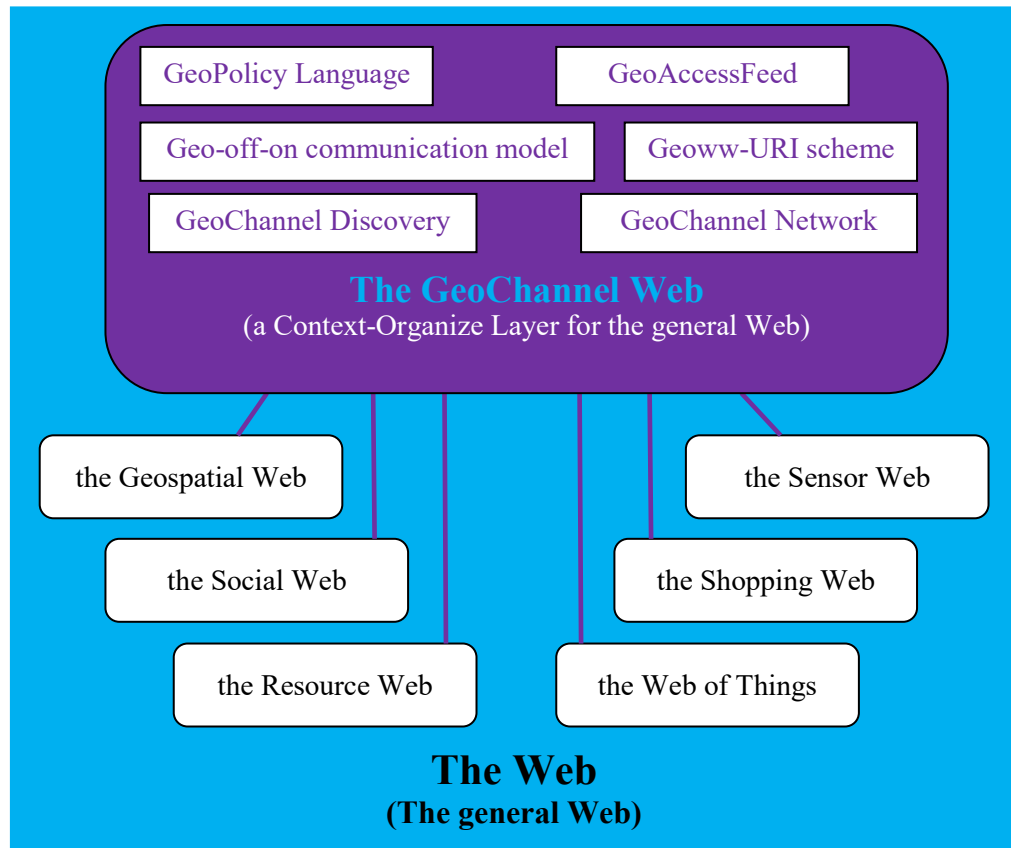


Figure 10-4: The GeoChannel Web adds a Context-Organize Layer for the general Web

### 10.3.1.2 Analogy between functionality components

For further clarifying the new creations of this research, here some analogies are making on related technical components between the GeoChannel Web and the general Web.

Table 10-8: Analogy on related technical components between the GeoChannel Web and the general Web

Components	The general Web	The GeoChannel Web
Language	HTML	GeoPolicy language
URI	HTTP-URI	Geoww-URI scheme
Communication	HTTP, etc.	Geo-Off-On model
Discovery	Search Service	Search (Catalogue) Service & physical-world approach
Networking	Hyperlink relations	Dynamical context relations

As enumerated in Table 10-8, this research has devised a set of substantial techniques for

enabling a new type of Web, i.e. the GeoChannel Web. Although not exactly used for equal functionality, each analogy on corresponding techniques can make sense for highlighting the significance of the new technique components. For example, HTML language is used for designing web pages or user interfaces, while the GeoPolicy language functions for defining context conditions. HTTP-URI is for identifying general web resources, while Geoww-URI as a novel scheme can identify contextualized things (e.g. resources, clients or clusters, such as pervasive services and users with context conditions), and can dynamically control the process of correlating, accessing, clustering and connecting these things. HTTP protocol does not involve controlling conditions, while Geo-Off-On model can automatically adapt communication behaviour based on latest context conditions of both sides (e.g. resource and client). The GeoChannel Web features highly dynamic network topology changing based on changed context attributes of things involved in GeoChannel networks, which can just reflect the real world with dynamic nature.

### **10.3.2 Review on the Research Output**

This has been presented in Section 10.1 and Section 10.2, which have summarized the research contributions and answered the research questions. This section here does not repeat this content, but just acts as a placeholder for reference purpose.

### **10.3.3 Potentials of Development**

#### **10.3.3.1 Infancy of this Research Field**

It was really a big work to have conceived, developed and tested the architectural system of the proposed GeoChannel Web, which has now involved a series of new concepts, models, techniques, implementation and application cases. Actually this thesis has just presented a part of my research work, due to the university rule that controls words number of a thesis.

Although this big work has achieved a set of original contributions, it was virtually in the early phase of a big research field opened by my research work so far. The GeoChannel Web as a brand-new concept, system architectural model and infrastructure has an extensive domain for further research on its theory and practice. I have recognized that my efforts so far just have

touched some fundamental or elementary aspects in this big field. When progressing in my work, I was clearly aware that, more new thoughts materialized and techniques devised, more novel ideas and research opportunities would be surfacing.

So in both breadth and depth, it is needed to extend and improve my work. This is left for future research. Some tasks will be identified in Section 10.4.

#### **10.3.3.2      Practicability & Productivity**

This research work has devoted main efforts on the theoretical and technical models for the GeoChannel Web and its constituent technical facilities. Some realistic application scenarios have been exemplified. The use cases in Chapter 9 are also based on real-world data and have demonstrated practical value with usefulness, representativeness and creativity. However, great potentials have not been fully explored.

The practicability of these new technical components need to be further improved in more real-world projects in the future, which in turn can help further extend the GeoChannel Web technology.

Productivity will be a further advanced demand on this new type of Web based on its practicability. The GeoChannel Web expects mapping and applying dynamic and random relations between real-world things based on their context on a globe-wide scale. The fulfilment of this grand vision and mission will rest with the productivity of the GeoChannel Web.

To make the GeoChannel Web technology become more practicable and productive, there is a need to further materialize all the concepts, models and techniques, and to further develop the prototype platform, Geonoon, for supporting more extensive real-world applications and services.

### **10.4 Recommendation for Future Research**

The GeoChannel architecture as a new model opens a door into a new domain with a broad scope for further research work theoretically and practically.

This PhD project, from a systematic view, has laid out an overall architecture framework with

many substantial designs and implementations conducted and completed already. However, more work having been carried out with more creative thoughts taken, more further interesting and significant tasks are surfacing and are worthy of exploration. The fundamental reason for the increasingly emerging work is due to this newly-opened research domain with promising application potentials.

The subsections below try to identify and give prominence to some of these future tasks, with identified rationale, basic requirements and methods for doing these tasks.

#### **10.4.1 Further implementing GeoChannel facilities**

The recommended future work here is to develop some information or interface models and to design a visual AccessFeed editor utility for the GeoChannel architecture.

##### **10.4.1.1 Interface descriptor & common APIs model for AccessFeeds**

AccessFeeds at runtime are mashable and interoperable program components dynamically plugged on the AccessFeed Bus in client-side environment such as the GeoChannel Explorer. An AccessFeed can provide “client-side web services”, that is, an AccessFeed, which connects to and accesses its GeoChannel node to provide services and resources, can expose its functions to be invoked by other GeoChannels’ AccessFeeds plugged on the AccessFeed Bus.

There is a demand for a formal information model for describing the interfaces of GeoChannel AccessFeeds, to facilitate identifying their functions and usage methods by developers that program interoperable AccessFeeds. A normative description language model is expected for AccessFeed interface descriptors. For designing this desired AccessFeed interface description language, some ideas can be inspired from similar or related techniques such as WSDL (Web Services Description Language (W3C 2007)), WADL (Web Application Description Language OpenAjax-Alliance 2010 3447).

In addition, there is also a need to model some common APIs (Application Programming Interfaces) for GeoChannel AccessFeeds. The GeoChannel Web supports cooperative GeoChannel resources (e.g. pervasive location-based services), which converge in some GeoContexts for cooperation via the interaction between their AccessFeeds. To facilitate automatically discovering and coupling functions of interest between GeoChannel services, it is advisable to implement some common interfaces for AccessFeeds. With the common APIs,

which may involve compulsory and optional interfaces, an AccessFeed can be able to automatically query the capability and functions of other AccessFeeds that are currently plugged in the same AccessFeed Bus, and may selectively couple with some of them for interaction. This is significant for achieving GeoContext-aware cooperation between pervasive services smartly.

As illuminated above, the normative interface descriptor language and the common APIs model for AccessFeeds can improve interoperability between GeoChannels when federating and integrating resources, functions and tasks for cooperative pervasive GeoChannel services.

#### **10.4.1.2 Visual editor for GeoPolicy Language and AccessFeeds**

AccessFeeds are key information components for contextualizing and utilizing GeoChannel resources, so producing and maintaining AccessFeeds is a commonly and frequently conducted work. The GeoAccessFeed technology has introduced new information models involving GeoPolicy language, Geo-Off-On communication model and AccessFeed schema. In order to lower the technical threshold, reduce errors of syntax and semantics, and improve efficiency when creating and updating AccessFeeds, it will be meaningful to design a visual editor for the use by the authors of AccessFeeds. An easy-to-use AccessFeed editing tool can foster crowd-sourcing AccessFeeds for various GeoChannel resources, so as to facilitate democratizing the GeoChannel architecture.

This desired editor for AccessFeeds is meant to cater for both nontechnical users who do not have much knowledge with the GeoAccessFeed syntax rules, and for those advanced users that can edit AccessFeeds with scalable complexity. Its function will involve the process of designing, creating, validating, deploying and updating AccessFeeds. Basically it is expected to have (but not be limited to) the following features:

- visual graphic user interface with WYSIWYG (What You See Is What You Get) editing capability.

It can help easily edit GeoPolicy context conditions and AccessFeeds in a visual way.

- function for auto-completion and smart predicting/sensing/suggesting.

A context-aware environment is supplied to aid smartly editing AccessFeeds content.

- syntax and semantics validator.

It can validate AccessFeeds syntax to get rid of errors based on the GeoAccessFeed information models, such as the schemas codified in Appendix 1 ~ Appendix 5.

- deploy and update AccessFeeds.

It can deploy AccessFeeds to the Web, and can work with AccessFeed Update Services to publish updates of AccessFeeds.

- capability for customizing and extending functionality.

It may provide an open programmable framework to support plug-in or add-on components for customizing and extending its functionality. For example, a specific add-on extension may be specialized in dealing with social-networking-domain AccessFeeds.

#### **10.4.2 Theory & Practice for the GeoChannel Web**

This research has illuminated the principles and explored some practical techniques of the GeoChannel Web. For example, the networking capability of GeoChannels can work for propagating dynamics, relaying services, super-Geo-Clustering interaction; and can bring new ways for discovering and navigating resources and users, Geo-diffuse social networking, and integrating and interoperating geo-compatible resources across GeoChannels, etc.

However, there is still considerable scope for further exploring this new concept and technology of the GeoChannel Web. Many practical techniques are worthy of further exploration and development. Here are (but not limited to) some aspects or points.

##### **10.4.2.1 Extending graph theory with added context attributes**

The GeoChannel Web, which expects mapping and applying dynamic and random relations between real-world things in latest contexts, is mathematically a massive graph network that connects things based on their context relations. There is a need for theory research on graph theory (or network science and complex systems) in association with dynamic and random relations of real-world things, try to contribute some conceptual or algorithm components into the general graph theory for supporting the practice of the GeoChannel Web. For example, dynamic community detection (Fortunato 2010), dynamic network analysis (Carley 2003).

So exploration is to be conduct for incorporating the technical components of the GeoChannel Web into general graph technology. For example, to fuse the GeoPolicy language, Geo-off-on



communication model, AccessFeed, Geoww-URL scheme, GeoChannel Discovery, etc., into some graph technological products or services. These new technical components should be able to customize and further improve general graph technologies to support the GeoChannel Web applications and services.

#### **10.4.2.2 Algorithm models for ordering search results of GeoChannel resources**

This research (in Chapter 7) has designed the information and process models for GeoChannel Catalogue Services. There will be a further demand on appropriately ordering query results of returned GeoChannel resources metadata to facilitate users' identifying of most relevant GeoChannel resources of interest.

Order sorting of search results is a commonly crucial functionality of search engines. Current general web search engines just demonstrate typical paradigms. Current web search engines crawl the Web to retrieve web pages to servers, then index and rank these web contents, and order search results. These search engines employ some algorithm models (Sharma & Sharma 2010) with various metrics for ordering search results. Taking Google's PageRank (Sergey & Lawrence 1998) (Page et al. 1998) algorithm for example, it performs link analysis for ranking web pages to assign a numerical weighting to each hyperlinked document for "measuring" its relative importance within a set of web documents.

By analogy to the theories and techniques about web (pages) search involving indexing, ranking, recommending, etc., there will be research scope about applicable algorithm models for ordering search results of GeoChannel resources for enhancing GeoChannel Catalogue Services, or for ordering search results of GeoClustered users for social networking.

The algorithm models desired will work beyond using the metadata information provided in GeoChannel catalogue repositories, but will exploit the features of GeoChannel Web networks, whose full topology view is built and updated by the Geonoon's GeoChannel Web Engine component, which can maintain a real-time and may keep historical information about a comprehensive network graph of global-range GeoChannel resources with their access users.

The algorithm models may involve (but not be limited to) the following aspects or requirements:

***Relevant factors with quantitative metrics:*** The models will quantitatively parameterize relevant factors that should be considered to take part in calculating the ranking (e.g. importance or relevance) of GeoChannel resources. This is to model a set of factors, with their respective

numerical index for measurement and statistics. For example, among many, the basic factor, which can be named popularity, may be represented by real-time or historical number of users that participates/participated in a GeoChannel (resource), and may take into account of the statistics about using/attention time that users spent on GeoChannel resources. And a personalized factor may involve the relative proximity or similarity of current GeoContexts of the searcher/querist and each of the searched GeoChannel resources.

**Weighting system:** This to assign various percentage values about the importance/relevance for each of relevant factors. A weighting value reflects the impact or contribution of a relevant factor when taking part in calculating the final ranking value of a GeoChannel resource in the search results. For example, the “popularity” factor as exemplified above basically may be assigned a relatively higher weighting value.

**Calculating models:** They refer to the comprehensive algorithm formula/procedure for calculating the general ranking value of individual GeoChannel resource entries within the result set of a search query on a GeoChannel Catalogue Service.

**Large-scale processing capability:** To deal with the global-range real-time and dynamic graph network about GeoChannel resources, users and access/participation relationships, the Geonoon’s GeoChannel Web Engine needs optimal structure and algorithms involving parallel and distributed software system design, for achieving decent performance on resource consumption, responding time and result quality when sorting order of search results and personalizing recommendation from GeoChannel Catalogue Services.

#### **10.4.2.3 System framework for coordinating geo-compatible pervasive services**

This GeoChannel architecture has an advanced feature about enabling cooperative pervasive services, which are underpinned by the GeoAccessFeed technology that can achieve dynamic resource-client, client-client and resource-resource correlation in a context-aware and sensitive way. Whenever and wherever a client meets the context conditions of multiple pervasive services, these service resources can interoperate and cooperate with each other. This feature can be exploited for composing new complex services.

This technology can be further generalized and expanded for achieving more pervasive and cooperative geospatial intelligence. There is a potential to develop a new and complete theory and model about an Earth-GeoContext-Service system. This system can be inspired by and can be analogized with current event-coordinated software systems. The event-driven paradigm

can induce the interoperation between software modules within a software program when a user's interaction induces various events. Now we can consider the GeoChannel Web as a comprehensive software system that comprises various pervasive service resources as software modules, some of which may be geo-compatible, i.e. their accessing conditions can be simultaneously met by a same user. These geo-compatible resources modules are those inter-neighbouring resource nodes depicted in the network graph of the GeoChannel Web.

The expected theory and model about this Earth-GeoContext-Service system is to develop a methodological and technical framework, via the AccessFeeds common interface model (that is also proposed as a future work in Section 10.4.1.1) and the GeoChannel Web topology graph supplied by the Geonoon platform, for automatically discovering, matching and connecting geo-compatible pervasive services converged in a user's GeoContext and bridged via AccessFeed Bus. This will achieve a pervasive environment for federating and interoperating GeoChannel resources such as location-based and pervasive services or applications.

#### **10.4.3 Normalization & Standardization**

This GeoChannel architecture expects to become a widely-adopted technology by being seamlessly fused and integrated into the general Web fabric. The openness and normality of the GeoChannel technological components is crucial for interoperability between different implementations of the GeoChannel architectural facilities.

This research work has developed some normative information or working models for these technical components, such as GeoPolicy language, Geo-Off-On communication model, AccessFeed structure, Geoww-URI scheme, GeoChannel Discovery metadata model, etc. There are potentials to further improve the normality of these models to meet more generalized or advanced application environment and tasks. Actually, the proposed future work in Section 10.3.1 also involves normatively modelling common APIs and interface descriptor for GeoChannel Web applications and services.

In terms of standardizing GeoChannel architectural components, the mostly compelling and promising work is to promote the GeoAccessFeed technology (involving GeoPolicy language, Geo-Off-On communication model, AccessFeed and Geoww-URI scheme), making it or its components become industrial specifications, for example, to become OGC (Open Geospatial

Consortium) standards.

The novelty of this GeoAccessFeed technology is its capability and mechanism for dynamically and automatically correlating, controlling-access, clustering and connecting real-virtual-world things, and throttling their communication and interaction, based on their latest contexts. This is a highly desired functionality for context-aware organizing access to resources for pervasive services, clustering clients for interaction and cooperation as a GeoWeb (or Sensor Web) -native social networking capability, and federating services that can cooperate in some contexts for composite geospatial intelligence.

This GeoAccessFeed technology with its information and working models developed so far has already possessed some basic features for becoming an industrial standard. Firstly, it can fill current absence of such a kind of technological facility with a broad application potential for enabling a more dynamic, context-aware and pervasive-service Web. Secondly, it has components completeness, integrity and models normality. Currently its constituents include the GeoPolicy Language, Geo-off-on communication model, AccessFeed information model, and Geoww-URI scheme, etc., all of which have been designed in normative encoding models, as codified in Appendix 1, Appendix 2, Appendix 3, Appendix 4, Appendix5, Appendix 6 and Figure 6 1. Especially, some encoding schemas are modelled by extending OGC's standard KML model.

To develop and promote the GeoAccessFeed technology into an industrial standard needs to comply with officially-designated procedures. Taking OGC's standardization process (OGC 2013) for example, this work may involve Standards Program, Interoperability Program, Compliance Program, and Marketing and Communications Program.

#### **10.4.4 Commercialization: Potentials & Opportunities**

The Geonoon, as an implementation and a practical facility of the GeoChannel architectural technology, expects to become a public and commercial platform for the GeoChannel Web. It is significant to further develop the Geonoon platform for providing practical services, and to explore its commercial models for generating revenues. So this proposed future work is to

improve and monetize the Geonoon platform.

#### **10.4.4.1 Providing practical functions and services**

The Geonoon platform mainly features the following platform roles and services:

- GeoChannel Web Engine: GeoChannel Networks Graph Service

The Networks Graph Service, which can automatically map out and reflect dynamic connection or context relationships among real-world things on a global scale, can facilitate discovering, federating and interacting with resources/clients of interest and in specific contexts.

- Social Networking Platform: GeoCluster Service

A new social-networking paradigm, i.e. GeoCluster (or GeoCommunity) as social media, featuring the nature of locality and anonymity, is enabled by the GeoChannel technology. The Geonoon provides a common social-networking engine named GeoCluster Service for all GeoChannel resources by maintaining their dynamic GeoClustered communities identified by respective Geoww-URIs. This GeoCluster Service can function as a GeoChannel Web native facility for connecting people for organizing their activities and interaction, to facilitate building and running participative and cooperative context-based applications and services.

- Pervasive Service Platform

The Geonoon, underpinned by the GeoAccessFeed which is a novel technology for contextualizing and access-controlling web resources, becomes a universal platform for supporting pervasive (such as location based) services. The Geoww-URI technique facilitates discovering pervasive service resources from physical or virtual world.

To make the Geonoon a practical platform with practical usability, decent performance and robustness, further work is needed to improve and enhance its constituents, involving backend components and frontend GeoChannel Explorer program.

#### **10.4.4.2 Exploring applicable business models**

This Geonoon platform will benefit providers, developers and users of GeoChannel resources, which may be general web users, merchants, application developers and other individuals or organizations. Apart from open services (e.g. acting as local news, social and service media that are available for free uses by the general public) which deliver social value, commercial

functionality should be an essential component for generating revenues to better support and improve open services. Exploring applicable business models is a future work that will design and apply value creation, delivery, and capture mechanisms.

Actually the Geonoon platform technologically possesses promising opportunities and potentials for generating commercial value. Probably the most exploitable asset of the GeoChannel technology is the locality with explicitness of GeoContexts of both GeoChannel resources and users. Various business patterns can grow around this kind of GeoContext locality.

Local advertising for precision-target marketing is definitely a profit-making approach. The GeoAccessFeed technology forms various dynamic geo-communities whose participants have certain interest in specific resources and are located in specific space/time contexts, which provide the sellers/suppliers of products or services with explicit clues for delivering marketing information to right audience in right locations at right times. Now business marketing such as precisely-targeted advertising can be more easily achieved with the knowledge about the GeoContext, and content and interest of a GeoChannel (geo-community) and its clients. Technically the Geonoon platform structure is adept at delivering GeoContext-based advertisements, which can be injected and communicated via the Geonoon components such as GeoChannel brokering nodes, GeoChannel Explorer, or client/server-side library for GeoChannel resources, etc., to reach target audience of interest. As the Geonoon platform mostly work for open services for local news/events, social media and pervasive services, this free-of-charge strategy can reach a big and solid user base that can in turn increase and consolidate the influence of the Geonoon platform. Big and active audience base can insure commercial impact of advertisements.

Various LBS-related commercial applications/services can reside on this profitable Geonoon platform. They can benefit from the unique capability of GeoAccessFeed for dynamically correlating resources with clients, resources with resources, and clients with clients. With the advancement of precise and combinatorial positioning techniques such as indoor iBeacon (Cox 2014) with outdoor GPS, more accurate and detailed GeoContexts acquired can foster GeoAccessFeed-based services. For example, Geoww-URIs for identification and AccessFeeds for correlation can automatically connect attendees to conference rooms or exhibition booths, and can connect wandering customers with in-store shelves of retail goods, for delivering proximity-based mobile information/services or marketing promotion.

The Geonoon platform can also build and sustain a new ecosystem of value chain that all

stakeholders can involve in. For example, an App-Store-like utility can live on the Geonoon that provides a platform for discovering, trading and running of free or paid, crowd-sourced or professionally-developed GeoChannel application programs. This marketplace utility can link developers, operators and users of GeoChannel resources/applications. This kind of commercial patterns and experience can be inspired and borrowed from application marketplaces existed in current mobile application industries.

Future work will further explore, develop and implement more applicable business models, which are supported by the innovative GeoChannel technology, to make the Geonoon a productive platform for the GeoChannel Web to create and deliver both social and commercial value.

## References

- Abargues, C., Beltran, A. & Granell, C. 2010, 'MIMEXT: a KML extension for georeferencing and easy share MIME type resources', *Geospatial Thinking*, pp. 315-334.
- Abowd, G. D. & Mynatt, E. D. 2000, 'Charting past, present, and future research in ubiquitous computing', *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 7, no. 1, pp. 29-58.
- AGI. , *UK GEMINI 2.1--Specification for discovery metadata for geospatial data resources*, [Online], Available from: <[http://www.agi.org.uk/storage/standards/uk-gemini/GEMINI2\\_1\\_published.pdf](http://www.agi.org.uk/storage/standards/uk-gemini/GEMINI2_1_published.pdf)>. [Accessed 18<sup>th</sup> June 2013]
- Ahlqvist, T., Bäk, A., Halonen, M. & Heinonen, S. 2008, 'social media roadmaps-- exploring the futures triggered by social media', *VTT Tiedotteita –Valtion Teknillinen Tutkimuskeskus (2454): 13*.
- Annoni, D. A., Craglia, D. M., Ehlers, M., Georgiadou, Y., Giacomelli, A., Konecny, M., Ostlaender, N., Remetey-Fulopp, G., Rhind, D., Smits, P. & others. 2011, 'A European perspective on Digital Earth', *International Journal of Digital Earth*, vol. 4, no. 4, pp. 271-284.
- Amplifon. , *The Sounds of Street View*, [Online], Available from: <<http://www.amplifon.com/web/uk/i-wear-a-hearing-aid/sounds-of-streetview>>. [Accessed 16<sup>th</sup> Sep 2014]
- Apache. , *Apache HBase*, [Online], Available from: <<http://hbase.apache.org/>>. [Accessed 19<sup>th</sup> Aug 2016]
- Apache. , *Apache Hadoop*, [Online], Available from: <<http://hadoop.apache.org/>>. [Accessed 19<sup>th</sup> Aug 2016]
- Apache. , *Apache Kafka*, [Online], Available from: <<http://kafka.apache.org/>>. [Accessed 19<sup>th</sup> Aug 2016]
- Apache. , *Apache Spark*, [Online], Available from: <<http://spark.apache.org/>>. [Accessed 19<sup>th</sup> Aug 2016]
- Apache. , *Apache ZooKeeper*, [Online], Available from: <<http://https://zookeeper.apache.org/>>. [Accessed 19<sup>th</sup> Aug 2016]
- Apache. , *Giraph--Apache graph processing system*, [Online], Available from: <<http://giraph.apache.org/>>. [Accessed 18<sup>th</sup> June 2013]
- Apache. , *TinkerPop*, [Online], Available from: <<http://tinkerpop.apache.org/>>. [Accessed 19<sup>th</sup> Aug 2016]
- Apple. , *iBeacon for Developers*, [Online], Available from: <<http://https://developer.apple.com/ibeacon/>>. [Accessed 16<sup>th</sup> Sep 2014]
- Axiomatics. , *100% pure XACML*, [Online], Available from: <<http://www.axiomatics.com/pure-xacml.html>>. [Accessed 18<sup>th</sup> June 2013]
- Axiomatics. , *Attribute Based Access Control (ABAC)*, [Online], Available from: <<http://www.axiomatics.com/attribute-based-access-control.html>>. [Accessed 18<sup>th</sup> June 2013]



- Axiomatics. , *XACML Architecture*, [Online], Available from: <<http://www.axiomatics.com>>. [Accessed 18<sup>th</sup> June 2013]
- Batty, M., Hudson-Smitha, A., Miltona, R. & Crooksb, A. 2010, 'Map mashups, Web 2.0 and the GIS revolution', *Annals of GIS*, vol. 16, no. 1, pp. 1-13.
- Bellavista, P., Kupper, A. & Helal, S. 2008, 'Location-based services: Back to the future', *Pervasive Computing, IEEE*, vol. 7, no. 2, pp. 85-89.
- Belshe, M. , *Hypertext Transfer Protocol Version 2 (HTTP/2)*, [Online], Internet Engineering Task Force (IETF), Available from: <<http://https://tools.ietf.org/html/rfc7540>>. [Accessed 16<sup>th</sup> Sep 2014]
- Berners-Lee, T. , *The WorldWideWeb (W3) project*, [Online], Available from: <<http://info.cern.ch/hypertext/WWW/TheProject.html>>. [Accessed 16<sup>th</sup> Sep 2014]
- Berners-Lee, T. , *"First mention of HTML Tags on the www-talk mailing list"*, [Online], World Wide Web Consortium, Available from: <<http://lists.w3.org/Archives/Public/www-talk/1991SepOct/0003.html>>. [Accessed 16<sup>th</sup> Sep 2014]
- Berners-Lee, T. , *The Original HTTP as defined in 1991b*, [Online], World Wide Web Consortium, Available from: <<http://https://www.w3.org/Protocols/HTTP/AsImplemented.html>>. [Accessed 16<sup>th</sup> Sep 2014]
- Berners-Lee, T. & Cailliau, R. , *WorldWideWeb: Proposal for a HyperText Project*, [Online], Available from: <<http://info.cern.ch/hypertext/WWW/Proposal.html>>. [Accessed 16<sup>th</sup> Sep 2014]
- Berners-Lee. , *Uniform Resource Locators (URL)-- A Syntax for the Expression of Access Information of Objects on the Network*, [Online], Available from: <<http://https://www.w3.org/Addressing/URL/url-spec.txt>>. [Accessed 16<sup>th</sup> Sep 2014]
- Bertino, E. & Kirkpatrick, M. S. 2011, 'Location-based access control systems for mobile users: concepts and research directions', in *Proceedings of the 4th ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS*, ACM, pp. 49-52.
- Bing, L. 2011, *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data* , Springer Science & Business Media.
- Bondi, A. B. 2000, 'Characteristics of scalability and their impact on performance', in *Proceedings of the 2nd international workshop on Software and performance*, ACM, pp. 195-203.
- Bosch, A., Bogers, T. & Kunder, M. 2016, 'Estimating search engine index size variability: a 9-year longitudinal study', *Scientometrics*, vol. 107, no. 2, pp. 839-856.
- Botts, M., Percivall, G., Reed, C. & Davidson, J. 2007, 'OGC's sensor web enablement: Overview and high level architecture', *GeoSensor Networks*, pp. 175-190.

- Boulos, M. N., Resch, B., Crowley, D. N., Breslin, J. G., Sohn, G., Burtner, R., Pike, W. A., Jezierski, E. & Chuang, K. Y. 2011, 'Crowdsourcing, citizen sensing and sensor web technologies for public and environmental health surveillance and crisis management: trends, OGC standards and application examples', *International journal of health geographics*, vol. 10, no. 1, p. 67.
- Broch, J., Maltz, D. A., Johnson, D. B., Hu, Y.-C. & Jetcheva, J. 1998, 'A performance comparison of multi-hop wireless ad hoc network routing protocols', in *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking*, ACM, pp. 85-97.
- Broring, A., Echterhoff, J., Jirka, S., Simonis, I., Everding, T., Stasch, C., Liang, S. & Lemmens, R. 2011, 'New generation sensor web enablement', *Sensors*, vol. 11, no. 3, pp. 2652-2699.
- Brown, C., Nicosia, V., Scellato, S., Noulas, A. & Mascolo, C. 2012, 'Where Online Friends Meet: Social Communities in Location-Based Networks', in *ICWSM*.
- Buettner & Ricardo. 2015, 'Towards a New Personal Information Technology Acceptance Model: Conceptualization and Empirical Evidence from a Bring Your Own Device Dataset', in *21st Americas Conference on Information Systems (AMCIS)*.
- Butler, D. 2006a, 'Mashups mix data into global service', *Nature*, vol. 439, no. 7072, pp. 6-7.
- Butler, D. 2006b, 'Virtual globes: The web-wide world', *Nature*, vol. 439, pp. 776-778.
- Butler, H., Daly, M., Doyle, A., Gillies, S., Schaub, T. & Schmidt, C. , *The GeoJSON Format Specification, revision 1.0*, [Online], Available from: <<http://www.geojson.org/geojson-spec.html>>. [Accessed 18<sup>th</sup> June 2013]
- Carley, K. M. 2003, *Dynamic network analysis*, Citeseer.
- CASA-UCL. , *MapTube--a place to put maps*, [Online], Available from: <<http://www.maptube.org/>>. [Accessed 18<sup>th</sup> June 2013]
- Cho, E., Myers, S. A. & Leskovec, J. 2011, 'Friendship and mobility: user movement in location-based social networks', in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, pp. 1082-1090.
- Clinton, D. , *OpenSearch 1.1 Draft 5*, *OpenSearch.org*, [Online], Available from: <[http://www.opensearch.org/Specifications/OpenSearch/1.1/Draft\\_5](http://www.opensearch.org/Specifications/OpenSearch/1.1/Draft_5)>. [Accessed 18<sup>th</sup> June 2013]
- Connolly, D. & Sperberg-McQueen, C. M., eds. . , *Web addresses in HTML 5*, [Online], World Wide Web Consortium. , Available from: <<http://https://www.w3.org/html/wg/href/draft#url>>. [Accessed 16<sup>th</sup> Sep 2014]
- Cox, J. 2014, 'Apple's iBeacon turns location sensing inside out', *Network World*.
- Craglia, M., de Bie, K., Jackson, D., Pesaresi, M., Remetey-Fulopp, G., Wang, C., Annoni, A., Bian, L., Campbell, F., Ehlers, M. & others. 2012, 'Digital Earth 2020: towards the vision for the next decade', *International Journal of Digital Earth*, vol. 5, no. 1, pp. 4-21.

- Craglia, M., Goodchild, M. F., Annoni, A., Camara, G., Gould, M., Kuhn, W., Mark, D., Masser, I., Maguire, D., Liang, S. & Parsons, E. 2008, 'Next-Generation Digital Earth--A position paper from the Vespucci Initiative for the Advancement of Geographic Information Science', *International Journal of Spatial Data Infrastructures Research*, vol. 3.
- Cramer, H., Rost, M. & Holmquist, L. E. 2011, 'Performing a check-in: emerging practices, norms and conflicts' in location-sharing using foursquare', in *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, ACM, pp. 57-66.
- Cranshaw, J., Schwartz, R., Hong, J. I. & Sadeh, N. 2012, 'The livelihoods project: Utilizing social media to understand the dynamics of a city', *Association for the Advancement of Artificial Intelligence*.
- Damsma, T. , *Animation of the measured winds during Katrina, and the associated water level set up*, [Online], Available from: <<http://dtvirt5.deltares.nl/kml/Katrina/hurricane.kmz>>. [Accessed 18<sup>th</sup> June 2013]
- Damsma, T. , *Animation of water movement during a tidal cycle in an estuary*, [Online], Available from: <[http://dtvirt5.deltares.nl/kml/Westerschelde/westerschelde\\_arrows.kmz](http://dtvirt5.deltares.nl/kml/Westerschelde/westerschelde_arrows.kmz)>. [Accessed 18<sup>th</sup> June 2013]
- DCMI. , *Dublin Core Metadata Element Set, Version 1.1*, [Online], Available from: <<http://dublincore.org/documents/dces/>>. [Accessed 18<sup>th</sup> June 2013]
- De Longueville, B. 2010, 'Community-based geoportals: The next generation? Concepts and methods for the geospatial Web 2.0', *Computers, Environment and Urban Systems*, vol. 34, no. 4, pp. 299-308.
- DENSO. , *QR Code Essentials*, [Online], Available from: <<http://www.nacs.org/LinkClick.aspx?fileticket=D1FpVAvvJuo%3D&tabid=1426&mid=4802>>. [Accessed 18<sup>th</sup> June 2013]
- Echterhoff, J. , *OGC's OWS-7 Event Architecture Engineering Report*, [Online], OGC, Available from: <[http://portal.opengeospatial.org/files/?artifact\\_id=39509](http://portal.opengeospatial.org/files/?artifact_id=39509)>. [Accessed 18<sup>th</sup> June 2013]
- Echterhoff, J. & Everding, T. , *OGC's (OpenGIS) Sensor Event Service Interface Specification*, [Online], Available from: <[http://portal.opengeospatial.org/files/?artifact\\_id=29576](http://portal.opengeospatial.org/files/?artifact_id=29576)>. [Accessed 18<sup>th</sup> June 2013]
- Echterhoff, J. & Everding, T. , *OGC's Event Service - Review and Current State, Reference number: OGC 11-088r1*, [Online], Available from: <[https://portal.opengeospatial.org/files/?artifact\\_id=45850](https://portal.opengeospatial.org/files/?artifact_id=45850)>. [Accessed 18<sup>th</sup> June 2013]
- Economist. , *Everywhere and nowhere*, [Online], Available from: <[http://www.economist.com/business/displaystory.cfm?story\\_id=10880936](http://www.economist.com/business/displaystory.cfm?story_id=10880936)>. [Accessed 18<sup>th</sup> June 2013]
- Egenhofer, M. J. 1992, 'Why not SQL!', *International Journal of Geographical Information Systems*, vol. 6, no. 2, pp. 71-85.

- Egenhofer, M. J. 1994, 'Spatial SQL: A query and presentation language', *Knowledge and Data Engineering, IEEE Transactions on*, vol. 6, no. 1, pp. 86-95.
- Elastic. , *ElasticSearch*, [Online], Available from: <<http://https://www.elastic.co/>>. [Accessed 19<sup>th</sup> Aug 2016]
- Elwood, S. 2008, 'Volunteered geographic information: future research directions motivated by critical, participatory, and feminist GIS', *GeoJournal*, vol. 72, no. 3, pp. 173-183.
- Elwood, S., Goodchild, M. F. & Sui, D. Z. 2012, 'Researching volunteered geographic information: Spatial data, geographic research, and new social practice', *Annals of the Association of American Geographers*, vol. 102, no. 3, pp. 571-590.
- EuropeanCommission. , *Directive 2007/2/EC of the European Parliament and of the Council of 14 March 2007 establishing an Infrastructure for Spatial Information in the European Community (INSPIRE)* , [Online], Available from: <<http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32007L0002:EN:>>>. [Accessed 18<sup>th</sup> June 2013]
- Everding, T. & Echterhoff, J. , *OGC's OWS-6 SWE Event Architecture Engineering Report*, [Online], Available from: <[http://portal.openeospatial.org/files?artifact\\_id=33347](http://portal.openeospatial.org/files?artifact_id=33347)>. [Accessed 18<sup>th</sup> June 2013]
- Facebook. , *facebook*, [Online], Available from: <<https://www.facebook.com/>>. [Accessed 18<sup>th</sup> June 2013]
- Faulkner, S., Eicholz, A., Leithead, T. & Danilo, A. , *HTML 5*, [Online], World Wide Web Consortium (W3C), Available from: <<http://https://www.w3.org/TR/html/>>. [Accessed 16<sup>th</sup> Sep 2015]
- Fielding, R. T. 2000, *PhD Thesis--Architectural styles and the design of network-based software architectures*, Doctoralthesis, University of California, Irvine.
- Fielding, R. T., Gettys, J., Mogul, J. C., Nielsen, H. F., Masinter, L., Leach, P. & Berners-Lee, T. , *Hypertext Transfer Protocol —HTTP/1.1. Internet RFC 2616*, [Online], Available from: <<http://www.w3.org/Protocols/rfc2616/rfc2616.html>>. [Accessed 18<sup>th</sup> June 2013]
- Fitzpatrick, B., Slatkin, B. & Atkins, M. , *PubSubHubbub Core 0.3 -- Working Draft*, [Online], Available from: <<http://pubsubhubbub.googlecode.com/svn/trunk/pubsubhubbub-core-0.3.html>>. [Accessed 18<sup>th</sup> June 2013]
- Foresman, T. W. 2008, 'Evolution and implementation of the Digital Earth vision, technology and society', *International journal of digital earth*, vol. 1, no. 1, pp. 4-16.
- Fortunato, S. 2010, 'Community detection in graphs', *Physics reports*, vol. 486, no. 3, pp. 75-174.
- Foursquare. , *foursquare*, [Online], Available from: <<http://www.foursquare.com>>. [Accessed 18<sup>th</sup> June 2013]

- Freed, N. & Borenstein, N. , *Multipurpose internet mail extensions (mime) part one: Format of internet message bodies, RFC 2045*. , [Online], Available from: <<http://tools.ietf.org/html/rfc2045>>. [Accessed 18<sup>th</sup> June 2013]
- Gao, H., Barbier, G. & Goolsby, R. 2011, 'Harnessing the crowdsourcing power of social media for disaster relief', *Intelligent Systems, IEEE*, vol. 26, no. 3, pp. 10-14.
- GeoRSS. , *GeoRSS*, [Online], Available from: <<http://georss.org/>>. [Accessed 18<sup>th</sup> June 2013]
- Golden, K. , *Google Earth: Organizing the world's information geographically*, [Online], Available from: <[http://www.mbari.org/seminars/2006/winter2006/google\\_golden.htm](http://www.mbari.org/seminars/2006/winter2006/google_golden.htm)>. [Accessed 18<sup>th</sup> June 2013]
- Goodchild, M. F. 2007a, 'Citizens as Voluntary Sensors: Spatial Data Infrastructure in the World of Web 2.0', *International Journal of Spatial Data Infrastructures Research*, vol. 2.
- Goodchild, M. F. 2007b, 'Citizens as sensors: the world of volunteered geography', *GeoJournal*, vol. 69, no. 4, pp. 211-221.
- Goodchild, M. F. 2007c, 'Citizens as sensors: web 2.0 and the volunteering of geographic information', *Geofocus*, vol. 7, pp. 8-10.
- Goodchild, M. F. 2008, 'The use cases of digital earth', *International Journal of Digital Earth*, vol. 1, no. 1, pp. 31- 42.
- Goodchild, M. F., Guo, H., Annoni, A., Bian, L., de Bie, K., Campbell, F., Craglia, M., Ehlers, M., van Genderen, J., Jackson, D. & others. 2012, 'Next-generation Digital Earth', *Proceedings of the National Academy of Sciences*, vol. 109, no. 28, pp. 11088-11094.
- Google. , *Bigtable: A Distributed Storage System for Structured Data* , [Online], Available from: <<http://https://research.google.com/archive/bigtable.html>>. [Accessed 19<sup>th</sup> Aug 2016]
- Google. , *British Museum in Google Street View Maps*, [Online], Available from: <<http://https://www.google.com/culturalinstitute/beta/partner/the-british-museum>>. [Accessed 16<sup>th</sup> Sep 2015]
- Google. , *Eddystone beacon message format protocol*, [Online], Available from: <<http://https://github.com/google/eddystone>>. [Accessed 16<sup>th</sup> Sep 2015]
- Google. , *KML Reference--Keyhole Markup Language*, [Online], Available from: <<https://developers.google.com/kml/documentation/kmlreference>>. [Accessed 18<sup>th</sup> June 2013]
- Google. , *KML Reference*, [Online], Available from: <<https://developers.google.com/kml/documentation/kmlreference>>. [Accessed 18<sup>th</sup> June 2013]
- Google. , *PubSubHubbub Project*, [Online], Available from: <<http://code.google.com/p/pubsubhubbub/>>. [Accessed 18<sup>th</sup> June 2013]
- Google. , *'The Museum of the World' microsite for the British Museum*, [Online], Available from: <<http://https://britishmuseum.withgoogle.com/>>. [Accessed 16<sup>th</sup> Sep 2014]

Google. , *The Physical Web*, [Online], Available from: <<http://https://google.github.io/physical-web/>>. [Accessed 16<sup>th</sup> Sep 2015]

Google. , *UriBeacon Specification*, [Online], Available from: <<http://https://github.com/google/uribeacon>>. [Accessed 16<sup>th</sup> Sep 2015]

Google. , *Waze--outsmarting traffic together*, [Online], Available from: <<http://www.waze.com/>>. [Accessed 18<sup>th</sup> June 2013]

Gore, A. 1998, 'The digital earth: Understanding our planet in the 21st century', *The Australian Surveyor*, vol. 43, no. 2, pp. 89-91.

Gregorio, J. & hOra, B. d. , *The Atom Publishing Protocol--RFC 5023*, [Online], Available from: <<http://tools.ietf.org/html/rfc5023>>. [Accessed 18<sup>th</sup> June 2013]

Grosky, W. I., Kansal, A., Nath, S., Liu, J. & Zhao, F. 2007, 'Senseweb: An infrastructure for shared sensing', *Multimedia, IEEE*, vol. 14, no. 4, pp. 8-13.

Grossner, K. E. & Clarke, K. 2006, 'Is Google Earth, "igital Earth?"?Defining a Vision', *University Consortium of Geographic Information Science, Summer Assembly, Vancouver, WA*.

Grossner, K. E., Goodchild, M. F. & Clarke, K. C. 2008, 'Defining a Digital Earth System', *Transactions in GIS*, vol. 12, no. 1, pp. 145-160.

GSDI. , *GLOBAL SPATIAL DATA INFRASTRUCTURE ASSOCIATION*, [Online], Available from: <<http://www.gsdi.org/>>. [Accessed 18<sup>th</sup> June 2013]

Guinard, D., Trifa, V., Mattern, F. & Wilde, E. 2011, 'From the Internet of Things to the Web of Things: Resource Oriented Architecture and Best Practices', in *Architecting the Internet of Things*, Springer.

Guo, H. D., Liu, Z. & Zhu, L. W. 2010, 'Digital Earth: decadal experiences and some thoughts', *International Journal of Digital Earth*, vol. 3, no. 1, pp. 31-46.

Haklay, M., Singleton, A. & Parker, C. 2008, 'Web mapping 2.0: the Neogeography of the Geoweb', *Geography Compass*, vol. 2, no. 6, pp. 2011-2039.

Henricksen, K., Indulska, J. & Rakotonirainy, A. 2002, 'Modeling context information in pervasive computing systems', in *Pervasive Computing*, Springer, pp. 167-180.

Herrmann, J. 2011, 'Administration of (Geo) XACML policies for spatial data infrastructures', in *Proceedings of the 4th ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS*, ACM, pp. 53-59.

Hirsch, F., Kemp, J. & Ilkka, J. 2006, *Mobile Web Services: Architecture and Implementation*, Wiley-Blackwell.

Howe, J. 2006, 'The rise of crowdsourcing', *Wired*, vol. 14, no. 6 (June).

- Huang, C. M., Lan, K. & Tsai, C. Z. 2008, 'A survey of opportunistic networks', in *Advanced Information Networking and Applications-Workshops, 2008. AINAW 2008. 22nd International Conference on*, IEEE, pp. 1672-1677.
- Hudson-Smith, A., Batty, M., Crooks, A. & Milton, R. 2009, 'Mapping for the Masses: Accessing Web 2.0 Through Crowdsourcing', *Social Science Computer Review*.
- IANA. , *Official list of MIME types assigned by the IANA (Internet Assigned Number Authority)*, [Online], Available from: <<http://www.iana.org/assignments/media-types>>. [Accessed 18<sup>th</sup> June 2013]
- IETF. , *Media Type Specifications and Registration Procedures, RFC6838*, [Online], Available from: <<http://www.rfc-editor.org/rfc/rfc6838.txt>>. [Accessed 18<sup>th</sup> June 2013]
- IETF. , *RFC 5870 - A Uniform Resource Identifier for Geographic Locations (geo URI)*, [Online], Available from: <<http://tools.ietf.org/html/rfc5870>>. [Accessed 16<sup>th</sup> Sep 2015]
- IETF. , *RFC 6350 - vCard Format Specification*, [Online], Available from: <<http://tools.ietf.org/html/rfc6350>>. [Accessed 16<sup>th</sup> Sep 2015]
- INSPIRE. , *INSPIRE Metadata Implementing Rules--Technical Guidelines based on EN ISO 19115 and EN ISO 19119*, [Online], Available from: <[http://inspire.jrc.ec.europa.eu/documents/Metadata/INSPIRE\\_MD\\_IR\\_and\\_ISO\\_v1\\_2\\_20100616.pdf](http://inspire.jrc.ec.europa.eu/documents/Metadata/INSPIRE_MD_IR_and_ISO_v1_2_20100616.pdf)>. [Accessed 18<sup>th</sup> June 2013]
- INSPIRE. (2011/11/07), *Technical Guidance for the implementation of INSPIRE Discovery Services\_v3.1*, [Online], Available from: <[http://inspire.jrc.ec.europa.eu/documents/Network\\_Services/TechnicalGuidance\\_DiscoveryServices\\_v3.1.pdf](http://inspire.jrc.ec.europa.eu/documents/Network_Services/TechnicalGuidance_DiscoveryServices_v3.1.pdf)>. [Accessed 18<sup>th</sup> June 2013]
- Inversini, A., Eynard, D., Marchiori, E. & Gentile, L. 2012, 'Destinations Similarity Based on User Generated PicturesTags', in *Information and Communication Technologies in Tourism 2012: Proceedings of the International Conference in Helsingborg, Sweden, January 24-27, 2012*, Springer, p. 483.
- ISO. , *ISO 19115-2003, Geographic information --Metadata*, [Online], Available from: <[http://www.iso.org/iso/catalogue\\_detail.htm?csnumber=26020](http://www.iso.org/iso/catalogue_detail.htm?csnumber=26020)>. [Accessed 16<sup>th</sup> Sep 2015]
- ISO. , *ISO 19119:2005, Geographic information -- Services*, [Online], Available from: <[http://www.iso.org/iso/catalogue\\_detail.htm?csnumber=39890](http://www.iso.org/iso/catalogue_detail.htm?csnumber=39890)>. [Accessed 16<sup>th</sup> Sep 2015]
- ISO. , *ISO 19115/Cor.1:2006, Geographic information--Metadata, Technical Corrigendum 1*, [Online], Available from: <[http://www.iso.org/iso/home/store/catalogue\\_tc/catalogue\\_detail.htm?csnumber=44361](http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=44361)>. [Accessed 16<sup>th</sup> Sep 2015]
- ISO. , *ISO/TS 19139:2007, Geographic information - Metadata XML Schema Implementation*, [Online], Available from: <[http://www.iso.org/iso/catalogue\\_detail.htm?csnumber=32557](http://www.iso.org/iso/catalogue_detail.htm?csnumber=32557)>. [Accessed 16<sup>th</sup> Sep 2015]

- ISO. , *ISO 19119:2005/Amd 1:2008, Extensions of the service metadata model*, [Online], Available from: <[http://www.iso.org/iso/home/store/catalogue\\_tc/catalogue\\_detail.htm?csnumber=44268](http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=44268)>. [Accessed 16<sup>th</sup> Sep 2015]
- ITU. 2005, 'The Internet of Things', *ITU--Internet Reports 2005*.
- Jin, X., Krishnan, R. & Sandhu, R. 2012, 'A unified attribute-based access control model covering DAC, MAC and RBAC', *Data and Applications Security and Privacy XXVI*, pp. 41-55.
- Jirka, S., Bröing, A. & Nüt, D. , *OGC's Sensor Observable Registry (SOR) Discussion Paper*, [Online], Available from: <[http://portal.opengeospatial.org/files/?artifact\\_id=40571](http://portal.opengeospatial.org/files/?artifact_id=40571)>. [Accessed 18<sup>th</sup> June 2013]
- Jones, M. T. 2007, 'Google's Geospatial Organizing Principle', *IEEE Computer Graphics and Applications*, pp. 8-13.
- Jurens, E. H., Broring, A. & Jirka, S. 2009, 'A human sensor web for water availability monitoring', *Proceedings of OneSpace--2nd International Workshop on Blending Physical and Digital Spaces on the Internet, Berlin, Germany, September 2009*.
- Kietzmann, J. H., Hermkens, K., McCarthy, I. P. & Silvestre, B. S. 2011, 'Social media? Get serious! Understanding the functional building blocks of social media', *Business Horizons*.
- Kling, F. & Pozdnoukhov, A. 2012, 'When a City Tells a Story: Urban Topic Analysis', *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*.
- Kobialka, T., Buyya, R., Deng, P., Kulik, L. & Palaniswami, M. 2010, 'Sensor web: Integration of sensor networks with web and cyber infrastructure', in *Handbook of Research on Developments and Trends in Wireless Sensor Networks: From Principle to Practice*, eds. H. Jin & Jiang, W., CiteSeer.
- Kraak, M.-J. 2001, 'Settings and needs for web cartography', *Web Cartography: Developments and Prospects. London and New York: Taylor Francis*.
- Kunder, M. d. , *The size of the World Wide Web*, [Online], Available from: <<http://www.worldwidewebsize.com/>>. [Accessed 16<sup>th</sup> Sep 2015]
- Kupper, A. 2005, *Location-Based Services : Fundamentals and Operation*, John Wiley & Sons.
- Lake, R. & Farley, J. 2007, 'Infrastructure for the Geospatial Web', in *The Geospatial Web-How Geobrowsers, Social Software and the Web 2.0 are Shaping the Network Society*, Springer, pp. 15-26.
- Landt, J. , *Shrouds of Time: The history of RFID*, [Online], Available from: <[http://www.transcore.com/pdf/AIM%20shrouds\\_of\\_time.pdf](http://www.transcore.com/pdf/AIM%20shrouds_of_time.pdf)>. [Accessed 18<sup>th</sup> June 2013]
- Leggetter, P. , *What are WebHooks and How Do They Enable a Real-time Web?*, [Online], Available from: <<http://blog.programmableweb.com/2012/01/30/webhooks-realtime-web/>>. [Accessed 18<sup>th</sup> June 2013]



- Li, N. & Chen, G. 2010, 'Sharing location in online social networks', *Network, IEEE*, vol. 24, no. 5, pp. 20-25.
- Linux-Foundation. , *JanusGraph Distributed Graph Database*, [Online], Available from: <<http://janusgraph.org/>>. [Accessed 19<sup>th</sup> Aug 2016]
- Longueville, B. D., Annoni, A., Schade, S., Ostlaender, N. & Whitmore, C. 2010, 'Digital Earth's Nervous System for crisis events: real-time Sensor Web Enablement of Volunteered Geographic Information', *International Journal of Digital Earth*, vol. 3, no. 3, pp. 242-259.
- Luo, J., Yu, J., Joshi, D. & Hao, W. 2008, 'Event recognition: viewing the world with a third eye', in *Proceedings of the 16th ACM international conference on Multimedia*, ACM, pp. 1071-1080.
- Maguire, D. 2005, 'GeoWeb 2.0: implications for ESDI', *Proceedings of the 12th EC-GIGIS Workshop*.
- Malewicz, G., Austern, M. H., Bik, A. J., Dehnert, J. C., Horn, I., Leiser, N. & Czajkowski, G. 2010, 'Pregel: a system for large-scale graph processing', in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, ACM, pp. 135-146.
- Malmi, E., Do, T. M. & Gatica-Perez, D. 2012, 'Checking in or checked in: comparing large-scale manual and automatic location disclosure patterns', in *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia*, ACM, p. 26.
- Melia, S. e. 2012, 'Activity duration analysis for context-aware services using foursquare check-ins', in *Proceedings of the 2012 international workshop on Self-aware internet of things*, ACM, pp. 13-18.
- Milojicic, D., Messer, A., Bernadat, P., Greenberg, I., Spinczyk, O., Beuche, D. & Schroder-Preikschat, W. 2001, 'Pervasive Services Infrastructure', *Technologies for E-Services*, pp. 187-199.
- MIT. , *HubCab -- Exploring New York City taxi trails and sharing our way to a more sustainable urban future*, [Online], Available from: <<http://hubcab.org/>>. [Accessed 18<sup>th</sup> May 2016]
- N. Freed, N. B. , *Multipurpose internet mail extensions (MIME) part two: Media types, RFC 2046*, [Online], Available from: <<http://tools.ietf.org/html/rfc2046>>. [Accessed 18<sup>th</sup> June 2013]
- Nath, S., Liu, J. & Zhao, F. 2007, 'Sensormap for wide-area sensor webs', *Computer*, vol. 40, no. 7, pp. 90-93.
- Nebert, D. D. , *Developing spatial data infrastructures: the SDI cookbook*, [Online], Available from: <<http://www.gsdi.org/docs2004/Cookbook/cookbookV2.0.pdf>>. [Accessed 18<sup>th</sup> June 2013]
- Neirotti, P., De Marco, A., Cagliano, A. C., Mangano, G. & Scorrano, F. 2014, 'Current trends in Smart City initiatives: Some stylised facts', *Cities*, vol. 38, pp. 25-36.
- Nittel, S. 2009, 'A survey of geosensor networks: Advances in dynamic environmental monitoring', *Sensors*, vol. 9, no. 7, pp. 5664-5678.
- Nottingham & Sayre, R. , *The Atom Syndication Format--RFC 4287*, [Online], Available from: <<http://tools.ietf.org/html/rfc4287>>. [Accessed 18<sup>th</sup> June 2013]

Noulas, A., Scellato, S., Lambiotte, R., Pontil, M. & Mascolo, C. 2012, 'A tale of many cities: universal patterns in human urban mobility', *PloS one*, vol. 7, no. 5, p. e37027.

NYC & TLC. , *TLC Trip Record Data*, [Online], Available from:

<[http://www.nyc.gov/html/tlc/html/about/trip\\_record\\_data.shtml](http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml)>. [Accessed 16<sup>th</sup> May 2016]

O'eilly, T. , *What is web 2.0*, [Online], Available from:

<<http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html>>. [Accessed 18<sup>th</sup> June 2013]

OASIS. , *OASIS Web Services Notification*, [Online], Available from:

<[https://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=wsn](https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsn)>. [Accessed 18<sup>th</sup> June 2013]

OASIS. , *eXtensible Access Control Markup Language (XACML) Version 3.0. 26 September 2012.*

*Candidate OASIS Standard 01.*, [Online], Available from:

<<http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-cos01-en.pdf>>. [Accessed 18<sup>th</sup> June 2013]

OASIS. , *searchRetrieve: Part 0. Overview Version 1.0, OASIS Standard, 30 January 2013a*, [Online], Available from:

<<http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/searchRetrieve-v1.0-part0-overview.html>>. [Accessed 18<sup>th</sup> June 2013]

OASIS. , *searchRetrieve: Part 1. Abstract Protocol Definition Version 1.0, OASIS Standard, 30 January 2013b*, [Online], Available from:

<<http://docs.oasis-open.org/search-ws/searchRetrieve/v1.0/os/part1-apd/searchRetrieve-v1.0-os-part1-apd.html>>. [Accessed 18<sup>th</sup> June 2013]

OGC. , *Augmented Reality Markup Language*, [Online], OGC, Available from:

<<http://www.opengeospatial.org/standards/arml>>. [Accessed 16th Sep 2015]

OGC. , *Sensor Web Enablement (SWE)*, [Online], Available from:

<<http://www.opengeospatial.org/projects/groups/sensorwebdwg>>. [Accessed 18<sup>th</sup> June 2013]

OGC. , *OpenGIS®Implementation Specification for Geographic information - Simple feature access - Part 1: Common architecture, Version: 1.2.0, Date: 2006-10-05*, [Online], Available from:

<[http://portal.opengeospatial.org/files/?artifact\\_id=18241](http://portal.opengeospatial.org/files/?artifact_id=18241)>. [Accessed 18<sup>th</sup> June 2013]

OGC. 2007, *OpenGIS Catalogue Services Specification 2.0.2 - ISO Metadata Application Profile, Version 1.0.0, OGC 07-045*, [Online], Available from:

<[http://portal.opengeospatial.org/files/?artifact\\_id=21460](http://portal.opengeospatial.org/files/?artifact_id=21460)>. [Accessed 18th June 2013]

OGC. , *OpenGIS's Catalogue Services Specification, Version 2.0.2*, [Online], Available from:

<[http://portal.opengeospatial.org/files/?artifact\\_id=20555](http://portal.opengeospatial.org/files/?artifact_id=20555)>. [Accessed 18<sup>th</sup> June 2013]

OGC. , *KML Encoding Standard, Version: 2.2.0, Reference number: OGC 07-147r2*, [Online], Available from: <<http://www.opengeospatial.org/standards/kml>>. [Accessed 18<sup>th</sup> June 2013]

- OGC. , *OpenGIS Web Map Tile Service Implementation Standard*, [Online], Available from:  
<[http://portal.opengeospatial.org/files/?artifact\\_id=35326](http://portal.opengeospatial.org/files/?artifact_id=35326)>. [Accessed 16<sup>th</sup> Sep 2015]
- OGC. , *OpenGIS's Filter Encoding 2.0 Encoding Standard*, [Online], Available from:  
<[http://portal.opengeospatial.org/files/?artifact\\_id=39968](http://portal.opengeospatial.org/files/?artifact_id=39968)>. [Accessed 18<sup>th</sup> June 2013]
- OGC. , *Open GeoSMS Standard*, [Online], Available from:  
<<http://www.opengeospatial.org/standards/opengeosms>>. [Accessed 16<sup>th</sup> Sep 2015]
- OGC. , *Geospatial eXtensible Access Control Markup Language (GeoXACML), Version: 1.0.1, Reference number: OGC 11-017*, [Online], Available from:  
<<http://www.opengeospatial.org/standards/geoxacml>>. [Accessed 18<sup>th</sup> June 2013]
- OGC. , *OGC's OpenSearch GeoSpatial and Temporal Extensions, OGC 10-032r3*, [Online], Available from: <[http://wiki.services.eoportal.org/tiki-download\\_wiki\\_attachment.php?attId=1447](http://wiki.services.eoportal.org/tiki-download_wiki_attachment.php?attId=1447)>. [Accessed 18<sup>th</sup> June 2013]
- OGC. , *OGC's Sensor Observation Service Interface Standard, Version 2.0*, [Online], Available from:  
<<http://www.opengeospatial.org/standards/sos>>. [Accessed 18<sup>th</sup> June 2013]
- OGC. , *OGC\_City\_Geography\_Markup\_Language\_CityGML\_Encoding\_Standard-Version 2.0.0*, [Online], Available from: <[https://portal.opengeospatial.org/files/?artifact\\_id=47842](https://portal.opengeospatial.org/files/?artifact_id=47842)>. [Accessed 18<sup>th</sup> June 2013]
- OGC. , *OGC's Programs*, [Online], Available from: <<http://www.opengeospatial.org/ogc/programs>>. [Accessed 18<sup>th</sup> June 2013]
- OpenAjax-Alliance. , *OpenAjax Metadata 1.0 Specification*, [Online], Available from:  
<[http://www.openajax.org/member/wiki/OpenAjax\\_Metadata\\_1.0\\_Specification](http://www.openajax.org/member/wiki/OpenAjax_Metadata_1.0_Specification)>. [Accessed 18<sup>th</sup> June 2013]
- O'Reilly, T. , *Web 2.0: Compact Definition?*, [Online], Available from:  
<<http://radar.oreilly.com/2005/10/web-20-compact-definition.html>>. [Accessed 18<sup>th</sup> June 2013]
- O'Reilly, T. , *What Is Web 2.0--Design Patterns and Business Models for the Next Generation of Software*, [Online], O'Reilly Media, Available from: <<http://oreilly.com/web2/archive/what-is-web-20.html>>. [Accessed 18<sup>th</sup> June 2013]
- O'reilly, T. 2007, 'What is Web 2.0: Design patterns and business models for the next generation of software', *Communications strategies*, no. 1, p. 17.
- O'Reilly, T. & Battelle, J. , *Web Squared: Web 2.0 Five Years On*, [Online], Available from:  
<<http://www.web2summit.com/web2009/public/schedule/detail/10194>>. [Accessed 18<sup>th</sup> June 2013]
- Page, L., Brin, S., Motwani, R. & Winograd, T. 1998, 'The PageRank citation ranking: bringing order to the web', .

- Pelusi, L., Passarella, A. & Conti, M. 2006, 'Opportunistic networking: data forwarding in disconnected mobile ad hoc networks', *Communications Magazine, IEEE*, vol. 44, no. 11, pp. 134-141.
- Perkins, C. E. & others. 2001, *Ad hoc networking*, Addison-wesley Reading.
- Pietroniro, E. & Fichter, D. 2006, 'Map mashups and the rise of amateur cartographers and mapmakers', *BULLETIN-ASSOCIATION OF CANADIAN MAP LIBRARIES AND ARCHIVES*, vol. 127, p. 26.
- Pinterest. , *Pinterest*, [Online], Available from: <<https://www.pinterest.com/>>. [Accessed 18<sup>th</sup> June 2013]
- Pontes, T., Vasconcelos, M. A., Almeida, J. M., Kumaraguru, P. & Almeida, V. 2012, 'We know where you live: privacy characterization of foursquare behavior', in *UbiComp*, pp. 898-905.
- Portele , C. , *OGC Geography Markup Language (GML) — Extended schemas and encoding rules, Version 3.3.0, OGC Document 10-129r1*, [Online], Available from: <[http://portal.openeospatial.org/files/?artifact\\_id=46568](http://portal.openeospatial.org/files/?artifact_id=46568)>. [Accessed 16<sup>th</sup> Sep 2015]
- Quercia, D., Lathia, N., Calabrese, F., Di Lorenzo, G. & Crowcroft, J. 2010, 'Recommending social events from mobile phone location data', in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, IEEE, pp. 971-976.
- Reclus, F. & Drouard, K. 2009, 'Geofencing for fleet freight management', in *Intelligent Transport Systems Telecommunications,(ITST), 2009 9th International Conference on*, IEEE, pp. 353-356.
- Reed, C. , *OGC Newsletter: OGC standards and the Geospatial Web (CTO'S MESSAGE )*, [Online], Available from: <<http://www.openeospatial.org/pressroom/newsletters/200701>>. [Accessed 18<sup>th</sup> June 2013]
- Reed, C., Singh, R., Lake, R., Lieberman, J. & Maron, M. 2006, 'An introduction to georss: A standards based approach for geo-enabling rss feeds', *White Paper OGC 06-050r3, Open Geospatial Consortium*.
- Resch, B., Blaschke, T. & Mittlboeck, M. 2011, 'Live Geography: Interoperable Geo-Sensor Webs Facilitating the Vision of Digital Earth', *International Journal on Advances in Networks and Services*, vol. 3, no. 3 and 4, pp. 323-332.
- RFC3986. , *Uniform Resource Identifier (URI): Generic Syntax*, [Online], Available from: <<https://tools.ietf.org/html/rfc3986#section-2.4>>. [Accessed 18<sup>th</sup> June 2013]
- Roick, O. & Heuser, S. 2013, 'Location Based Social Networks--Definition, Current State of the Art and Research Agenda', *Transactions in GIS*.
- RSS. , *RSS 2.0 Specification*, [Online], Available from: <<http://www.rssboard.org/rss-specification>>. [Accessed 18<sup>th</sup> June 2013]
- Saha, D. & Mukherjee, A. 2003, 'Pervasive computing: a paradigm for the 21st century', *Computer*, vol. 36, no. 3, pp. 25-31.

- Sakr, S. , *Processing large-scale graph data: A guide to current technology*, [Online], Available from: <<http://www.ibm.com/developerworks/library/os-giraph/os-giraph-pdf.pdf>>. [Accessed 18<sup>th</sup> June 2013]
- Sandhu, R. S. 1993, 'Lattice-based access control models', *Computer*, vol. 26, no. 11, pp. 9-19.
- Sandhu, R. S., Coyne, E. J., Feinstein, H. L. & Youman, C. E. 1996, 'Role-based access control models', *Computer*, vol. 29, no. 2, pp. 38-47.
- Sani, A. & Rinner, C. 2011, 'A scalable geoweb tool for argumentation mapping', *Geomatica*, vol. 65, no. 2, pp. 145-156.
- Santanche, A., Nath, S., Liu, J., Priyantha, B. & Zhao, F. 2006, 'Sensor Web: Browsing the physical world in real-time', *Demo Abstract, ACM/IEEE IPSN06, Nashville, TN*.
- Satyanarayanan, M. 2001, 'Pervasive computing: Vision and challenges', *Personal Communications, IEEE*, vol. 8, no. 4, pp. 10-17.
- Scellato, S., Noulas, A., Lambiotte, R. & Mascolo, C. 2011a, 'Socio-Spatial Properties of Online Location-Based Social Networks', *ICWSM*, vol. 11, pp. 329-336.
- Scellato, S., Noulas, A., Lambiotte, R. & Mascolo, C. 2011b, 'Socio-spatial properties of online location-based social networks', *Proceedings of ICWSM*, vol. 11, pp. 329-336.
- Schade, S., Diaz, L., Ostermann, F., Spinsanti, L., Luraschi, G., Cox, S., Nunez, M. & Longueville, B. D. 2011, 'Citizen-based sensing of crisis events: sensor web enablement for volunteered geographic information', *Applied Geomatics*, pp. 1-16.
- Scharl, A. 2007, 'Towards the geospatial web: Media platforms for managing geotagged knowledge repositories', in *The Geospatial Web-How Geobrowsers, Social Software and the Web 2.0 are Shaping the Network Society*, Springer, pp. 3-14.
- Scharl, A. & Tochtermann, K. 2007, *The geospatial web: how geobrowsers, social software and the Web 2.0 are shaping the network society*, Springer.
- Schneider, G., Dreher, B. & Seidel, O. 2008, 'Using GeoFencing as a means to support flexible real time applications for delivery services', in *5th International Workshop on Ubiquitous Computing (IWUC-2008). Barcelona, Spain*.
- Schneider, T. W. , *Analyzing 1.1 Billion NYC Taxi and Uber Trips, with a Vengeance*, [Online], Available from: <<http://toddschneider.com/posts/analyzing-1-1-billion-nyc-taxi-and-uber-trips-with-a-vengeance/>>. [Accessed 16<sup>th</sup> Sep 2015]
- Sergey, B. & Lawrence, P. 1998, 'The anatomy of a large-scale hypertextual web search engine', *Computer Networks and ISDN Systems*, vol. 30, no. 1-7, pp. 107-117.

- Sharma, D. K. & Sharma, A. K. 2010, 'A Comparative Analysis of Web Page Ranking Algorithms', *International Journal on Computer Science and Engineering*, vol. 2, no. 08, pp. 2670-2676.
- Sheth, A. 2009, 'Citizen sensing, social signals, and enriching human experience', *Internet Computing, IEEE*, vol. 13, no. 4, pp. 87-92.
- Simon Jirka, D. N. , *OGC's Sensor Instance Registry Discussion Paper*, [Online], Available from: <[http://portal.opengeospatial.org/files/?artifact\\_id=40609](http://portal.opengeospatial.org/files/?artifact_id=40609)>. [Accessed 18<sup>th</sup> June 2013]
- Simonis, I. , *OGC's Sensor Alert Service Candidate Implementation Specification*, [Online], Available from: <<http://www.citeulike.org/user/ehjuerrens/article/5279752>>. [Accessed 18<sup>th</sup> June 2013]
- Simonis, I. & Echterhoff, J. , *OGC's Sensor Planning Service Implementation Standard*, [Online], Available from: <[http://portal.opengeospatial.org/files/?artifact\\_id=38478](http://portal.opengeospatial.org/files/?artifact_id=38478)>. [Accessed 18<sup>th</sup> June 2013]
- Smith, E. , *PubSubHubbub Hullabaloo*, [Online], Available from: <<http://thelimberlambda.com/2009/08/10/pubsubhubbub-hullabaloo/>>. [Accessed 18<sup>th</sup> June 2013]
- Smith, G. (2007/04/04), *Social software building blocks*, [Online], Available from: <<http://nform.com/publications/social-software-building-block>>. [Accessed 18<sup>th</sup> June 2013]
- Stephenson, N. 1992, *Snow Crash*, London: Penguin Books.
- Strimpakou, M. A., Roussaki, I. G. & Anagnostou, M. E. 2006, 'A context ontology for pervasive service provision', in *Advanced Information Networking and Applications, 2006. AINA 2006. 20th International Conference on*, IEEE, pp. 5-pp.
- Sun. , *Sun's XACML Implementation*, [Online], Available from: <<http://sunxacml.sourceforge.net/>>. [Accessed 18<sup>th</sup> June 2013]
- Tobler, W. R. 1970, 'A computer movie simulating urban growth in the Detroit region', *Economic geography*, vol. 46, pp. 234-240.
- Turner, A. 2006, *Introduction to neogeography*, O'Reilly Media, Inc..
- Turner, A. & Forrest, B. 2008, *Where 2.0: The State of the Geospatial Web*, O'Reilly Media, Inc..
- Tyler, M. 2005, *Web Mapping Illustrated*, O'Reilly, Sebastopol USA.
- Vervoort, J. M., Kok, K., van Lammeren, R. & Veldkamp, T. 2010, 'Stepping into futures: Exploring the potential of interactive media for participatory scenarios on social-ecological systems', *Futures*, vol. 42, no. 6, pp. 604-616.
- W3C. , *Web Services Description Language (WSDL) 1.1*, [Online], Available from: <<http://www.w3.org/TR/wsdl>>. [Accessed 18<sup>th</sup> June 2013]
- W3C. , *Web Services Description Language (WSDL) Version 2.0 Part 0: Primer*, [Online], Available from: <<http://www.w3.org/TR/wsdl20-primer/>>. [Accessed 18<sup>th</sup> June 2013]

- W3C. , *Web Application Description Language (WADL)*, [Online], Available from:  
<<http://www.w3.org/Submission/wadl/>>. [Accessed 18<sup>th</sup> June 2013]
- W3C. , *A Standards-based, Open and Privacy-aware Social Web, W3C Incubator Group Report 6th December 2010*, [Online], Available from:  
<<http://www.w3.org/2005/Incubator/socialweb/XGR-socialweb-20101206/>>. [Accessed 18<sup>th</sup> June 2013]
- W3C. , *W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures*, [Online], Available from:  
<<http://www.w3.org/TR/xmlschema11-1/>>. [Accessed 18<sup>th</sup> June 2013]
- Wave, D. , *QR Code features*, [Online], Available from: <<http://www.qrcode.com/en/>>. [Accessed 18<sup>th</sup> June 2013]
- Weiser, M. 1991, 'The computer for the 21st century', *Scientific American*, vol. 265, no. 3, pp. 94-104.
- Weiser, M. 1993, 'Some computer science issues in ubiquitous computing', *Communications of the ACM*, vol. 36, no. 7, pp. 75-84.
- Wellington, B. , *How Software in Half of NYC Cabs Generates \$5.2 Million a Year in Extra Tips*, [Online], Available from:  
<<http://iquantny.tumblr.com/post/107245431809/how-software-in-half-of-nyc-cabs-generates-52>>. [Accessed 16<sup>th</sup> Sep 2015]
- Wellington, B. , *How to Fix NYC' No-Cabs-At-4PM Problem*, [Online], Available from:  
<<http://iquantny.tumblr.com/post/115096016059/how-to-fix-nycs-no-cabs-at-4pm-problem>>. [Accessed 16<sup>th</sup> Sep 2015]
- WHATWG. , *URL standard*, [Online], Web Hypertext Application Technology Working Group, Available from: <<http://https://url.spec.whatwg.org/>>. [Accessed 16<sup>th</sup> Sep 2015]
- Whong, C. , *NYC Taxis: A Day in the Life*, [Online], Available from: <<http://nyctaxi.herokuapp.com/>>. [Accessed 16<sup>th</sup> Sep 2015]
- Wikipedia. , *Scalability*, [Online], Available from: <<http://en.wikipedia.org/wiki/Scalability>>. [Accessed 18<sup>th</sup> June 2013]
- Yuan, E. & Tong, J. 2005, 'Attributed based access control (ABAC) for web services', in *Web Services, 2005. ICWS 2005. Proceedings. 2005 IEEE International Conference on*, IEEE.
- Zheng, Y. & Zhou, X. 2011, *Computing with spatial trajectories*, Springer.
- Zuckerberg, M. , *Building the Social Web Together*, [Online], Available from:  
<<http://www.facebook.com/blog/blog.php?post=383404517130>>. [Accessed 18<sup>th</sup> June 2013]
- Zyl, T. L., Simonis, I. & McFerren, G. 2009, 'The Sensor Web: systems of sensor systems', *International Journal of Digital Earth*, vol. 2, no. 1, pp. 16-30.

## Appendix 1 – GeoPolicy Language: extended-KML Schema

*(Part of the GeoChannel Markup Language in Appendix 5)*

<!-- Notes:

The GeoPolicy language is a language facility with expressive capability for representing context condition/policy for the GeoChannel Web. It has a wide range of usability. For example, it can be used to define conditions for access control, for controlling communication, for correlating resources and clients, for clustering clients, or for organizing (e.g. Geo-Federating) resources, etc.

It was originally designed as a constituent component of the GeoAccessFeed system in this research for resource access-organize task. Now it is modularized in this separate model schema. This modularization provides flexibility for independent evolution of this GeoPolicy language, and for it to be able to function in more extensive application domains other than being used in the GeoAccessFeed framework.

This schema uses extended-KML for encoding this language. However, this GeoPolicy Language model (as designed in Table 3-1) is essentially encoding-neutral, i.e. it can also be encoded into other encoding formats, such as CityGML, ARML (Augmented Reality Markup Language), GeoJSON, etc., to make it usable in other data format environments.

This condition-expressing language is extensible. Further component parts can be added into this model.

----->

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:geopolicy="http://www.geoww.net/geopolicy" xmlns:kml="http://www.opengis.net/kml/2.2"
targetNamespace="http://www.geoww.net/geopolicy" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="20121018A001">
  <annotation>
    <documentation> This XML schema describes a proposed extension to current KML standard to
support the GeoPolicy Language </documentation>
  </annotation>
  <!-- Import the schema of current KML standard, OGC KML 2.2 -->
```



```

    <import namespace="http://www.opengis.net/kml/2.2"
schemaLocation="http://schemas.opengis.net/kml/2.2.0/ogckml22.xsd"/>
    <!-- GeoPolicy is the top-level element -->
    <element name="GeoPolicy">
        <complexType>
            <complexContent>
                <extension base="kml:AbstractObjectType">
                    <sequence>
                        <element ref="kml:description" minOccurs="0"/>
                        <group ref="geopolicy:GeoPolicyGroup" minOccurs="0"
maxOccurs="unbounded"/>
                    </sequence>
                    <attribute name="policy_ID" type="string"/>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <group name="GeoPolicyGroup">
        <choice>
            <element ref="kml:AbstractTimePrimitiveGroup"/>
            <element ref="kml:Region"/>
            <element ref="geopolicy:AbstractGeoPolicyItem"/>
        </choice>
    </group>
    <element name="AbstractGeometryConstructGroup" type="geopolicy:AbstractGeometryConstructType"
abstract="true"/>
    <complexType name="AbstractGeometryConstructType" abstract="true">
        <complexContent>
            <extension base="kml:AbstractGeometryType"/>
        </complexContent>
    </complexType>
    <element name="AbstractGeoPolicyItem" type="geopolicy:AbstractGeoPolicyItemType"
abstract="true"/>
    <complexType name="AbstractGeoPolicyItemType">
        <complexContent>
            <extension base="kml:AbstractObjectType"/>
        </complexContent>
    </complexType>
    <element name="AbstractNumericObjectGroup" type="geopolicy:AbstractNumericObjectType"
abstract="true"/>
    <complexType name="AbstractNumericObjectType" abstract="true">
        <complexContent>
            <extension base="kml:AbstractObjectType"/>
        </complexContent>
    </complexType>

```

```

</complexType>
<element name="accuracy" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractNumericObjectType"/>
    </complexContent>
  </complexType>
</element>
<element name="Add" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractNumericObjectType">
        <sequence>
          <group ref="geopolicy:NumericGroup" minOccurs="2"
maxOccurs="unbounded"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</element>
<element name="Altitude" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractNumericObjectType">
        <sequence>
          <element ref="kml:Point"/>
          <element name="altitudeMode">
            <simpleType>
              <restriction base="string">
                <enumeration value="clampToGround"/>
                <enumeration value="relativeToGround"/>
                <enumeration value="absolute"/>
              </restriction>
            </simpleType>
          </element>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</element>
<element name="altitudeAccuracy" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractNumericObjectType"/>

```

```

        </complexContent>
    </complexType>
</element>
<element name="AND" substitutionGroup="geopolicy:AbstractGeoPolicyItem">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractGeoPolicyItemType">
                <sequence>
                    <group ref="geopolicy:GeoPolicyGroup" minOccurs="2"
maxOccurs="unbounded"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<element name="Area" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractNumericObjectType">
                <sequence>
                    <group ref="geopolicy:GeometryGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<element name="Boundary" substitutionGroup="geopolicy:AbstractGeometryConstructGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractGeometryConstructType">
                <group ref="geopolicy:GeometryGroup"/>
            </extension>
        </complexContent>
    </complexType>
</element>
<element name="BoundingBox" substitutionGroup="geopolicy:AbstractPolygonObjectGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractPolygonObjectType">
                <sequence>
                    <group ref="geopolicy:GeometryGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>

```

```

        </complexType>
    </element>
    <element name="Buffer" substitutionGroup="geopolicy:AbstractGeometryConstructGroup">
        <complexType>
            <complexContent>
                <extension base="geopolicy:AbstractGeometryConstructType">
                    <sequence>
                        <group ref="geopolicy:GeometryGroup"/>
                        <element name="distance" type="double"/>
                    </sequence>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <element name="Centroid" substitutionGroup="geopolicy:AbstractGeometryConstructGroup">
        <complexType>
            <complexContent>
                <extension base="geopolicy:AbstractGeometryConstructType">
                    <group ref="geopolicy:GeometryGroup"/>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <element name="Contains" type="geopolicy:TopologyType"
substitutionGroup="geopolicy:AbstractGeoPolicyItem"/>
    <element name="ConvexHull" substitutionGroup="geopolicy:AbstractGeometryConstructGroup">
        <complexType>
            <complexContent>
                <extension base="geopolicy:AbstractGeometryConstructType">
                    <group ref="geopolicy:GeometryGroup"/>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <element name="Crosses" type="geopolicy:TopologyType"
substitutionGroup="geopolicy:AbstractGeoPolicyItem"/>
    <element name="DecodePath" substitutionGroup="geopolicy:AbstractPointObjectGroup">
        <complexType>
            <complexContent>
                <extension base="geopolicy:AbstractPointObjectType">
                    <sequence>
                        <group ref="geopolicy:LineStringGroup"/>
                    </sequence>
                </extension>
            </complexContent>
        </complexType>
    </element>

```

```

        </complexContent>
    </complexType>
</element>
<element name="Difference" substitutionGroup="geopolicy:AbstractGeometryConstructGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractGeometryConstructType">
                <sequence>
                    <group ref="geopolicy:GeometryGroup"/>
                    <group ref="geopolicy:GeometryGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<element name="Disjoint" type="geopolicy:TopologyType"
substitutionGroup="geopolicy:AbstractGeoPolicyItem"/>
<element name="Distance" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractNumericObjectType">
                <sequence>
                    <group ref="geopolicy:GeometryGroup"/>
                    <group ref="geopolicy:GeometryGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<element name="Divide" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractNumericObjectType">
                <sequence>
                    <group ref="geopolicy:NumericGroup"/>
                    <group ref="geopolicy:NumericGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<element name="EncodePath" substitutionGroup="geopolicy:AbstractLineStringObjectGroup">
    <complexType>
        <complexContent>

```

```

        <extension base="geopolicy:AbstractLineStringObjectType">
            <sequence>
                <group ref="geopolicy:PointGroup" minOccurs="2"
maxOccurs="unbounded"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
</element>
<element name="EncodePoint" substitutionGroup="geopolicy:AbstractPointObjectGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractPointObjectType">
                <sequence>
                    <element name="latitude">
                        <complexType>
                            <group ref="geopolicy:NumericGroup"/>
                        </complexType>
                    </element>
                    <element name="longitude">
                        <complexType>
                            <group ref="geopolicy:NumericGroup"/>
                        </complexType>
                    </element>
                    <element name="altitude">
                        <complexType>
                            <group ref="geopolicy:NumericGroup"/>
                        </complexType>
                    </element>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<element name="EncodePolygon" substitutionGroup="geopolicy:AbstractPolygonObjectGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractPolygonObjectType">
                <sequence>
                    <group ref="geopolicy:LineStringGroup" maxOccurs="unbounded"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>

```

```

</element>
<element name="Equals" type="geopolicy:TopologyType"
substitutionGroup="geopolicy:AbstractGeoPolicyItem"/>
<element name="Floor" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractNumericObjectType">
        <sequence>
          <group ref="geopolicy:NumericGroup"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</element>
<group name="GeometryGroup">
  <choice>
    <element ref="kml:AbstractGeometryGroup"/>
    <element ref="geopolicy:AbstractGeometryConstructGroup"/>
    <element ref="geopolicy:AbstractPointObjectGroup"/>
    <element ref="geopolicy:AbstractLineStringObjectGroup"/>
    <element ref="geopolicy:AbstractPolygonObjectGroup"/>
  </choice>
</group>
<element name="GetNorthEast" substitutionGroup="geopolicy:AbstractPointObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractPointObjectType">
        <sequence>
          <element ref="geopolicy:BoundingBox"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</element>
<element name="GetPaths" substitutionGroup="geopolicy:AbstractLineStringObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractLineStringObjectType">
        <sequence>
          <group ref="geopolicy:PolygonGroup"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>

```

```

</element>
<element name="GetSouthWest" substitutionGroup="geopolicy:AbstractPointObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractPointObjectType">
        <sequence>
          <element ref="geopolicy:BoundingBox"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</element>
<element name="GreaterThan" type="geopolicy:NumericComparisonType"
substitutionGroup="geopolicy:AbstractGeoPolicyItem"/>
<element name="GreaterThanOrEqual" type="geopolicy:NumericComparisonType"
substitutionGroup="geopolicy:AbstractGeoPolicyItem"/>
<element name="GroundAltitude" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractNumericObjectType">
        <sequence>
          <element name="latitude" type="double"/>
          <element name="longitude" type="double"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</element>
<element name="Heading" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractNumericObjectType">
        <sequence>
          <element ref="kml:Point"/>
          <element ref="kml:Point"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</element>
<element name="Interpolate" substitutionGroup="geopolicy:AbstractPointObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractPointObjectType">

```



```

        <sequence>
            <group ref="geopolicy:PointGroup"/>
            <group ref="geopolicy:PointGroup"/>
            <element name="fraction"/>
        </sequence>
    </extension>
</complexContent>
</complexType>
</element>
<element name="Intersection" substitutionGroup="geopolicy:AbstractGeometryConstructGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractGeometryConstructType">
                <sequence>
                    <group ref="geopolicy:GeometryGroup"/>
                    <group ref="geopolicy:GeometryGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<element name="Intersects" type="geopolicy:TopologyType"
substitutionGroup="geopolicy:AbstractGeoPolicyItem"/>
<element name="IsWithinDistance" substitutionGroup="geopolicy:AbstractGeoPolicyItem">
    <complexType>
        <complexContent>
            <extension base="geopolicy:TopologyType">
                <sequence>
                    <element name="distance" type="double"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<element name="Latitude" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractNumericObjectType">
                <sequence>
                    <element ref="kml:Point"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>

```

```

</element>
<element name="LatLngBounds" substitutionGroup="geopolicy:AbstractPolygonObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractPolygonObjectType">
        <sequence>
          <element name="SouthWest">
            <complexType>
              <group ref="geopolicy:PointGroup"/>
            </complexType>
          </element>
          <element name="NorthEast">
            <complexType>
              <group ref="geopolicy:PointGroup"/>
            </complexType>
          </element>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</element>
<element name="Length" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractNumericObjectType">
        <sequence>
          <group ref="geopolicy:GeometryGroup"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</element>
<element name="LessThan" type="geopolicy:NumericComparisonType"
substitutionGroup="geopolicy:AbstractGeoPolicyItem"/>
<element name="LessThanOrEqual" type="geopolicy:NumericComparisonType"
substitutionGroup="geopolicy:AbstractGeoPolicyItem"/>
<element name="Longitude" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractNumericObjectType">
        <sequence>
          <element ref="kml:Point"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>

```

```

        </complexContent>
    </complexType>
</element>
<group name="NumericGroup">
    <choice>
        <element name="numeric" type="decimal"/>
        <element ref="geopolicy:AbstractNumericObjectGroup"/>
    </choice>
</group>
<element name="Mod" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractNumericObjectType">
                <sequence>
                    <element name="a1" type="integer"/>
                    <element name="a2" type="integer"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<element name="Abs" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractNumericObjectType">
                <sequence>
                    <group ref="geopolicy:NumericGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<element name="Multiply" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractNumericObjectType">
                <sequence>
                    <group ref="geopolicy:NumericGroup" minOccurs="2"
maxOccurs="unbounded"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>

```

```

<element name="NOT" substitutionGroup="geopolicy:AbstractGeoPolicyItem">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractGeoPolicyItemType">
        <sequence>
          <element ref="geopolicy:AbstractGeoPolicyItem"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</element>
<complexType name="NumericComparisonType">
  <complexContent>
    <extension base="geopolicy:AbstractGeoPolicyItemType">
      <sequence>
        <group ref="geopolicy:NumericGroup"/>
        <group ref="geopolicy:NumericGroup"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<element name="Offset" substitutionGroup="geopolicy:AbstractPointObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractPointObjectType">
        <sequence>
          <element name="From">
            <complexType>
              <group ref="geopolicy:PointGroup"/>
            </complexType>
          </element>
          <element name="distance" type="double"/>
          <element name="heading" type="double"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</element>
<element name="OffsetOrigin" substitutionGroup="geopolicy:AbstractPointObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractPointObjectType">
        <sequence>
          <element name="To">

```

```

        <complexType>
            <group ref="geopolicy:PointGroup"/>
        </complexType>
    </element>
    <element name="distance" type="double"/>
    <element name="heading" type="double"/>
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="OR" substitutionGroup="geopolicy:AbstractGeoPolicyItem">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractGeoPolicyItemType">
                <sequence>
                    <group ref="geopolicy:GeoPolicyGroup" minOccurs="2"
maxOccurs="unbounded"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<element name="Overlaps" type="geopolicy:TopologyType"
substitutionGroup="geopolicy:AbstractGeoPolicyItem"/>
<element name="Round" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractNumericObjectType">
                <sequence>
                    <group ref="geopolicy:NumericGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<element name="Subtract" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractNumericObjectType">
                <sequence>
                    <group ref="geopolicy:NumericGroup"/>
                    <group ref="geopolicy:NumericGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>

```

```

        </extension>
      </complexContent>
    </complexType>
  </element>
  <element name="SymDifference" substitutionGroup="geopolicy:AbstractGeometryConstructGroup">
    <complexType>
      <complexContent>
        <extension base="geopolicy:AbstractGeometryConstructType">
          <sequence>
            <group ref="geopolicy:GeometryGroup"/>
            <group ref="geopolicy:GeometryGroup"/>
          </sequence>
        </extension>
      </complexContent>
    </complexType>
  </element>
  <complexType name="TopologyType">
    <complexContent>
      <extension base="geopolicy:AbstractGeoPolicyItemType">
        <sequence>
          <group ref="geopolicy:GeometryGroup"/>
          <group ref="geopolicy:GeometryGroup"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <element name="Touches" type="geopolicy:TopologyType"
substitutionGroup="geopolicy:AbstractGeoPolicyItem"/>
  <element name="Union" substitutionGroup="geopolicy:AbstractGeometryConstructGroup">
    <complexType>
      <complexContent>
        <extension base="geopolicy:AbstractGeometryConstructType">
          <sequence>
            <group ref="geopolicy:GeometryGroup"/>
            <group ref="geopolicy:GeometryGroup"/>
          </sequence>
        </extension>
      </complexContent>
    </complexType>
  </element>
  <group name="PointGroup">
    <choice>
      <element ref="kml:Point"/>
      <element ref="geopolicy:AbstractPointObjectGroup"/>
    </choice>
  </group>

```

```

        </choice>
    </group>
    <element name="AbstractPointObjectGroup" type="geopolicy:AbstractPointObjectType"/>
    <complexType name="AbstractPointObjectType">
        <complexContent>
            <extension base="kml:AbstractGeometryType"/>
        </complexContent>
    </complexType>
    <group name="LineStringGroup">
        <choice>
            <element ref="kml:LineString"/>
            <element ref="geopolicy:AbstractLineStringObjectGroup"/>
        </choice>
    </group>
    <element name="AbstractLineStringObjectGroup" type="geopolicy:AbstractLineStringObjectType"/>
    <complexType name="AbstractLineStringObjectType">
        <complexContent>
            <extension base="kml:AbstractGeometryType"/>
        </complexContent>
    </complexType>
    <group name="PolygonGroup">
        <choice>
            <element ref="kml:Polygon"/>
            <element ref="geopolicy:AbstractPolygonObjectGroup"/>
        </choice>
    </group>
    <element name="AbstractPolygonObjectGroup" type="geopolicy:AbstractPolygonObjectType"/>
    <complexType name="AbstractPolygonObjectType">
        <complexContent>
            <extension base="kml:AbstractGeometryType"/>
        </complexContent>
    </complexType>
    <element name="clientLocation" substitutionGroup="geopolicy:AbstractPointObjectGroup">
        <complexType>
            <complexContent>
                <extension base="geopolicy:AbstractPointObjectType"/>
            </complexContent>
        </complexType>
    </element>
    <element name="clientAltitude" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
        <complexType>
            <complexContent>
                <extension base="geopolicy:AbstractNumericObjectType"/>
            </complexContent>
        </complexType>
    </element>

```

```

        </complexType>
    </element>
    <element name="clientHeading" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
        <complexType>
            <complexContent>
                <extension base="geopolicy:AbstractNumericObjectType"/>
            </complexContent>
        </complexType>
    </element>
    <element name="clientLatitude" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
        <complexType>
            <complexContent>
                <extension base="geopolicy:AbstractNumericObjectType"/>
            </complexContent>
        </complexType>
    </element>
    <element name="clientLongitude" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
        <complexType>
            <complexContent>
                <extension base="geopolicy:AbstractNumericObjectType"/>
            </complexContent>
        </complexType>
    </element>
    <element name="clientSpeed" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
        <complexType>
            <complexContent>
                <extension base="geopolicy:AbstractNumericObjectType"/>
            </complexContent>
        </complexType>
    </element>
    <element name="viewCenter" substitutionGroup="geopolicy:AbstractPointObjectGroup">
        <complexType>
            <complexContent>
                <extension base="geopolicy:AbstractPointObjectType"/>
            </complexContent>
        </complexType>
    </element>
    <element name="viewHeading" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
        <complexType>
            <complexContent>
                <extension base="geopolicy:AbstractNumericObjectType"/>
            </complexContent>
        </complexType>
    </element>

```



```

<element name="viewPoint" substitutionGroup="geopolicy:AbstractPointObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractPointObjectType"/>
    </complexContent>
  </complexType>
</element>
<element name="viewportCornerBounds"
substitutionGroup="geopolicy:AbstractGeometryConstructGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractGeometryConstructType">
        <sequence>
          <element name="HitTestMode">
            <simpleType>
              <restriction base="string">
                <enumeration value="globe"/>
                <enumeration value="terrain"/>
                <enumeration value="buildings"/>
              </restriction>
            </simpleType>
          </element>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</element>
<element name="viewportGlobeBounds"
substitutionGroup="geopolicy:AbstractGeometryConstructGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractGeometryConstructType"/>
    </complexContent>
  </complexType>
</element>
<element name="viewRoll" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
  <complexType>
    <complexContent>
      <extension base="geopolicy:AbstractNumericObjectType"/>
    </complexContent>
  </complexType>
</element>
<element name="viewTilt" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
  <complexType>

```

```

        <complexContent>
            <extension base="geopolicy:AbstractNumericObjectType"/>
        </complexContent>
    </complexType>
</element>
<element name="viewZoom" substitutionGroup="geopolicy:AbstractNumericObjectGroup">
    <complexType>
        <complexContent>
            <extension base="geopolicy:AbstractNumericObjectType"/>
        </complexContent>
    </complexType>
</element>
<element name="Within" type="geopolicy:TopologyType"
substitutionGroup="geopolicy:AbstractGeoPolicyItem"/>
</schema>

```

## Appendix 2 – Geo-off-on Communication Model: extended-KML

### Schema

*(Part of the GeoChannel Markup Language in Appendix 5)*

<!-- Notes:

The name Geo-off-on implies context switches that use context conditions for controlling connection/disconnection status or interaction patterns in a communication system. It uses the GeoPolicy Language (which is modelled in a separate schema in Appendix 1) for expressing context conditions for controlling communication behaviours.

This Geo-off-on communication model was originally designed as a constituent component of the GeoAccessFeed system in this research for communicating AccessFeed updates. Now it is modularized in this separate model schema. This modularization provides flexibility for independent evolution of this communication model, and for it to be able to function in more extensive application domains other than being used in the GeoAccessFeed framework.

This schema uses extended-KML for encoding this Geo-off-on communication model, which, however, is essentially encoding-neutral. That is, it can also be encoded into other encoding formats, such as CityGML, ARML (Augmented Reality Markup Language), GeoJSON, etc., to make it usable in other data format environments.

----->

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" xmlns:geoffon="http://www.geoww.net/geoffon"
xmlns:kml="http://www.opengis.net/kml/2.2" xmlns:geopolicy="http://www.geoww.net/geopolicy"
targetNamespace="http://www.geoww.net/geoffon" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="20121018A001">
  <annotation>
    <documentation> This XML schema describes a proposed extension to current KML standard to
support the Geo-off-on Communication Model </documentation>
  </annotation>
  <!-- Import the schema of current KML standard, OGC KML 2.2 -->
  <import namespace="http://www.opengis.net/kml/2.2"
schemaLocation="http://schemas.opengis.net/kml/2.2.0/ogckml22.xsd"/>
  <!-- Import the schema of the GeoPolicy Language - extended-KML Schema -->
```

```

    <import namespace="http://www.geoww.net/geopolicy"
schemaLocation="http://www.geoww.net/schemas/GeoPolicy_Language--extended-KML_Schema.xsd"/>
    <element name="locationRefreshMode" type="geoffon:locationRefreshModeEnumType"/>
    <simpleType name="locationRefreshModeEnumType">
        <restriction base="string">
            <enumeration value="never"/>
            <enumeration value="onChange"/>
        </restriction>
    </simpleType>
    <element name="Pull" type="geoffon:PullType"/>
    <complexType name="PullType">
        <complexContent>
            <extension base="kml:AbstractObjectType">
                <sequence>
                    <element ref="geopolicy:GeoPolicy" minOccurs="0"/>
                    <element ref="kml:href"/>
                    <element ref="geoffon:refreshMode" minOccurs="0"/>
                    <element ref="kml:refreshInterval" minOccurs="0"/>
                    <element ref="kml:TimeSpan" minOccurs="0"/>
                    <element ref="geoffon:viewRefreshMode" minOccurs="0"/>
                    <element ref="kml:viewRefreshTime" minOccurs="0"/>
                    <element ref="geoffon:locationRefreshMode" minOccurs="0"/>
                    <element ref="kml:viewBoundScale" minOccurs="0"/>
                    <element ref="kml:viewFormat" minOccurs="0"/>
                    <element ref="kml:httpQuery" minOccurs="0"/>
                    <element name="clientContext" type="string" minOccurs="0"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>

```

Example:Channel=[channelID],[controlVersion];Location=[geoLocation];Time=[Time] -->

```

    </sequence>
    </extension>
    </complexContent>
</complexType>
<element name="Push" type="geoffon:PushType"/>
<complexType name="PushType">
    <complexContent>
        <extension base="kml:AbstractObjectType">
            <sequence>
                <element ref="geopolicy:GeoPolicy" minOccurs="0"/>
                <element name="pushPattern">
                    <simpleType>
                        <restriction base="string">
                            <enumeration value="WebSockets"/>
                            <enumeration value="ServerSentEvents"/>
                        </restriction>
                    </simpleType>
                </element>
            </sequence>
        </extension>
    </complexContent>

```

```

        </simpleType>
    </element>
    <element ref="kml:href"/>
    <element ref="kml:refreshMode" minOccurs="0"/>
    <element ref="kml:refreshInterval" minOccurs="0"/>
    <element ref="geoffon:viewRefreshMode" minOccurs="0"/>
    <element ref="kml:viewRefreshTime" minOccurs="0"/>
    <element ref="geoffon:locationRefreshMode" minOccurs="0"/>
    <element ref="kml:viewBoundScale" minOccurs="0"/>
    <element ref="kml:viewFormat" minOccurs="0"/>
    <element name="Query" type="string" minOccurs="0"/>
    <element name="clientContext" type="string" minOccurs="0"/>
    <!--

```

Example:Channel=[channelID],[controlVersion];Location=[geoLocation];Time=[Time] -->

```

    </sequence>
</extension>
</complexContent>
</complexType>
<element name="adhocRefreshInterval" type="double" default="60.0"/>
<element name="refreshMode" type="geoffon:refreshModeEnumType"/>
<simpleType name="refreshModeEnumType">
    <restriction base="string">
        <enumeration value="onChange"/>
        <enumeration value="onInterval"/>
        <enumeration value="onIntervalTimeSpan"/>
        <enumeration value="onExpire"/>
    </restriction>
</simpleType>
<element name="viewRefreshMode" type="geoffon:viewRefreshModeEnumType"/>
<simpleType name="viewRefreshModeEnumType">
    <restriction base="string">
        <enumeration value="never"/>
        <enumeration value="onStop"/>
        <enumeration value="onRequest"/>
        <enumeration value="onPolicyToTrue"/>
    </restriction>
</simpleType>
</schema>

```

## Appendix 3 – AccessFeed: extended-KML Schema

*(Part of the GeoChannel Markup Language in Appendix 5)*

<!-- Notes:

The AccessFeed model includes some components. This schema here just involves its main information structure framework. Some constituent modules such as the GeoPolicy Language component and the Geo-off-on communication model component are modelled in separate schemas (see Appendix 1 and Appendix 2). An AccessFeed can be contained within a GeoChannel element whose model is codified in the GeoChannel schema (see Appendix 4), or can be directly embedded into a general KML file. This modularized design provides flexibility for independent evolution of these related components.

This schema uses extended-KML for encoding the AccessFeed information model, which, however, is essentially encoding-neutral. That is, AccessFeeds can also be encoded into other encoding formats, such as CityGML, ARML (Augmented Reality Markup Language), GeoJSON, etc., to make AccessFeeds usable in other data format environments.

----->

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" xmlns:geofeed="http://www.geoww.net/geofeed"
xmlns:kml="http://www.opengis.net/kml/2.2" xmlns:geopolicy="http://www.geoww.net/geopolicy"
xmlns:geoffon="http://www.geoww.net/geoffon" targetNamespace="http://www.geoww.net/geofeed"
elementFormDefault="qualified" attributeFormDefault="unqualified" version="20121018A001">
  <annotation>
    <documentation>The XML schema describes a proposed extension to current KML standard to
support the GeoAccessFeed</documentation>
  </annotation>
  <!-- Import the schema of current KML standard, OGC KML 2.2-->
  <import namespace="http://www.opengis.net/kml/2.2"
schemaLocation="http://schemas.opengis.net/kml/2.2.0/ogckml22.xsd"/>
  <!-- Import the schema of the GeoPolicy Language - extended-KML Schema -->
  <import namespace="http://www.geoww.net/geopolicy"
schemaLocation="http://www.geoww.net/schemas/GeoPolicy_Language--extended-KML_Schema.xsd"/>
  <!-- Import the schema of the Geoffon Communication Model - extended-KML Schema -->
  <import namespace="http://www.geoww.net/geoffon"
schemaLocation="http://www.geoww.net/schemas/Geoffon_Communication_Model--extended-KML_Schema.
xsd"/>
  <!-- AccessFeed is the top-level element -->
```

```

    <element name="AccessFeed" type="geofeed:AccessFeedType"
substitutionGroup="kml:AbstractFeatureGroup"/>
    <complexType name="AccessFeedType">
        <complexContent>
            <extension base="kml:AbstractFeatureType">
                <sequence>
                    <element ref="kml:AbstractGeometryGroup" minOccurs="0"/>
                    <element ref="geoffon:adhocRefreshInterval" minOccurs="0"/>
                    <element ref="geopolicy:GeoPolicy" minOccurs="0"/>
                    <element ref="geoffon:Pull" minOccurs="0"/>
                    <element ref="geoffon:Push" minOccurs="0"/>
                    <element ref="geofeed:Accessibility" minOccurs="0"/>
                </sequence>
                <attribute name="src" type="anyURI" use="required"/>
                <attribute name="version" type="dateTime"/>
                <attribute name="channel_ID">
                    <simpleType>
                        <restriction base="string">
                            <minLength value="1"/>
                            <pattern value=".*[^\s].*"/>
                        </restriction>
                    </simpleType>
                </attribute>
                <attribute name="feed_ID">
                    <simpleType>
                        <restriction base="string">
                            <minLength value="1"/>
                            <pattern value=".*[^\s].*"/>
                        </restriction>
                    </simpleType>
                </attribute>
                <attribute ref="geofeed:canCrossDomain"/>
            </extension>
        </complexContent>
    </complexType>
    <attribute name="canCrossDomain" type="boolean" default="false">
        <annotation>
            <documentation>

```

This "canCrossDomain" attribute specifies whether or not the GeoPolicy conditions must enforce the accessibility of the GeoChannel resources via the Accessor(s) in this AccessFeed. This attribute can be used at the AccessFeed level or/and at any Accessor level.

The default value of this attribute is false, implying the user of the AccessFeed or the Accessor cannot access this GeoChannel resource identified by all Accessors or this Accessor, in case this user does not

meet corresponding GeoPolicy conditions. This user will be automatically connected or disconnected to a GeoChannel resource once becoming matching or unmatching of the GeoPolicy conditions.

If this attribute value is set to be true, the user can, via client-side user interface, opt to have access to the GeoChannel resource(s) even if this user does not match the GeoPolicy conditions; or this user's access is just automatically controlled by the GeoPolicy conditions, as equivalent to the situation that the "canCrossDomain" attribute has a false value.

```

        </documentation>
    </annotation>
</attribute>
<element name="AccessorSelector">
    <!-- this element can be nested in GeoChannel element in a KML file for choosing which Accessor(s)
to run rather than running all the Accessors within an AccessFeed. When this AccessorSelector is absent, all
Accessors will run in the case they meet respective GeoPolicy conditions.-->
    <complexType>
        <sequence>
            <element name="accessorIdentifier" type="string" minOccurs="0"
maxOccurs="unbounded"/>
        </sequence>
    </complexType>
</element>
<element name="UpdateAccessFeed" type="geofeed:UpdateAccessFeedType"
substitutionGroup="kml:AbstractFeatureGroup"/>
<complexType name="UpdateAccessFeedType">
    <complexContent>
        <extension base="kml:AbstractFeatureType">
            <choice>
                <!-- Two patterns apply to this update process. One is to replace an old version
with a new version of the whole AccessFeed; another is to incrementally modify the old version, by employing
the method of Update element that is originally defined as a child of NetworkLinkControl in the standard KML
language.-->
                <element ref="geofeed:AccessFeed"/>
            </choice>
            <sequence>
                <element name="targetVersionID" type="string"/>
                <choice maxOccurs="unbounded">
                    <element name="Create">
                        <complexType>
                            <sequence>
                                <element ref="kml:AbstractObjectGroup"
minOccurs="0" maxOccurs="unbounded"/>
                            </sequence>
                        </complexType>
                    </element>
                    <element name="insideNodeID" type="string"
use="required"/>
                    <element name="beforeNodeID" type="string"/>
                    <element name="afterNodeID" type="string"/>
                </choice>
            </sequence>
        </extension>
    </complexContent>
</complexType>
</element>

```



```

        </complexType>
      </element>
      <element name="Delete">
        <complexType>
          <sequence>
            <element ref="kml:AbstractObjectGroup"
minOccurs="0" maxOccurs="unbounded"/>
          </sequence>
        </complexType>
      </element>
      <element name="Change">
        <complexType>
          <sequence>
            <element ref="kml:AbstractObjectGroup"
minOccurs="0" maxOccurs="unbounded"/>
          </sequence>
        </complexType>
      </element>
    </choice>
  </sequence>
</choice>
  <attribute name="channel_ID" type="string" use="required"/>
  <attribute name="newVersionID" type="string" use="required"/>
  <attribute name="src" type="anyURI" use="required"/>
  <attribute name="client_specific" type="boolean" use="required"/>
  <attribute name="timeSource" type="dateTime"/>
</extension>
</complexContent>
</complexType>
<element name="Accessibility" type="geofeed:AccessibilityType"/>
<complexType name="AccessibilityType">
  <complexContent>
    <extension base="kml:AbstractContainerType">
      <sequence>
        <element ref="geopolicy:GeoPolicy" minOccurs="0"/>
        <element ref="geofeed:Accessor" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="client_specific" type="boolean"/>
    </extension>
  </complexContent>
</complexType>
<element name="Accessor" type="geofeed:AccessorType"/>
<complexType name="AccessorType">
  <complexContent>

```

```

<extension base="kml:AbstractFeatureType">
  <sequence>
    <element ref="geopolicy:GeoPolicy" minOccurs="0"/>
    <element ref="kml:href" minOccurs="0"/>
    <element ref="kml:AbstractGeometryGroup" minOccurs="0"/>
    <element ref="kml:AbstractOverlayGroup" minOccurs="0"/>
    <element ref="geofeed:ContentType" minOccurs="0"/>
    <!-- the nested geofeed:Accessor element(s) may appear in the case that the nesting
Accessor element uses the nested Accessor element(s) as dependency for interoperation via GeoChannel
Accessor Bus.-->
    <element ref="geofeed:Accessor" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
  <attribute name="accessor_ID" use="required">
    <simpleType>
      <restriction base="string">
        <minLength value="1"/>
        <pattern value=".*[^\s].*" />
      </restriction>
    </simpleType>
  </attribute>
  <attribute name="access" type="boolean"/>
  <attribute ref="geofeed:canCrossDomain"/>
</extension>
</complexContent>
</complexType>
<element name="ContentType" type="geofeed:ContentTypeType"/>
<complexType name="ContentTypeType">
  <complexContent>
    <extension base="kml:AbstractObjectType">
      <sequence>
        <element name="MediaType" type="geofeed:MediaTypeEnumType"/>
        <element name="MediaSubType" type="geofeed:MediaSubTypeEnumType"/>
        <element name="MediaFileExtension"
type="geofeed:MediaFileExtensionEnumType"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<simpleType name="MediaFileExtensionEnumType">
  <restriction base="string">
    <enumeration value="html"/>
    <enumeration value="js"/>
    <enumeration value="kml"/>
    <enumeration value="kmz"/>

```

```

    </restriction>
</simpleType>
<simpleType name="MediaSubTypeEnumType">
  <restriction base="string">
    <enumeration value="vnd.GeoChannel-Accessor.accessFeed"/>
    <enumeration value="vnd.GeoChannel-Accessor.html"/>
    <enumeration value="vnd.GeoChannel-Accessor.javascript-webbrowser"/>
    <enumeration value="vnd.GeoChannel-Accessor.javascript-webworker"/>
    <enumeration value="vnd.google-earth.kml+xml"/>
    <enumeration value="vnd.google-earth.kmz"/>
  </restriction>
</simpleType>
<simpleType name="MediaTypeEnumType">
  <restriction base="string">
    <enumeration value="application"/>
  </restriction>
</simpleType>
</schema>

```

## Appendix 4 – GeoChannel & Discovery: extended-KML & -XML

### Schema

*(Part of the GeoChannel Markup Language in Appendix 5)*

<!-- Notes:

This schema uses extended-KML for encoding GeoChannel element information, which, however, is essentially encoding-neutral, i.e. it can also be encoded into other encoding formats such as CityGML, ARML (Augmented Reality Markup Language), GeoJSON, etc., to make it usable in other data format environments.

A GeoChannel element/object can contain some components, such as GeoChannel Discovery metadata, and/or GeoChannel Access-Organize component, i.e. AccessFeed which is separately modelled in another schema (see Appendix 3).

This schema codifies the information models:

extended-KML encoding to support GeoChannel element (involving its sub-elements) in KML files;

extended-KML encoding for returning OpenSearch results of GeoChannel Catalogue Services;

extended-XML encoding for describing OpenSearch services (i.e..OpenSearchDescription) to support GeoChannel information;

extended-XML feed (e.g. Atom, RSS) encoding to support PubSubHubBub protocol and GeoChannel information for real-time harvesting GeoChannel metadata for GeoChannel Catalogue Services.

----->

<?xml version="1.0" encoding="UTF-8"?>

<schema xmlns="http://www.w3.org/2001/XMLSchema"

xmlns:geochannel="http://www.geowww.net/geochannel" xmlns:geofeed="http://www.geowww.net/geofeed"

```

xmlns:kml="http://www.opengis.net/kml/2.2" xmlns:atom="http://www.w3.org/2005/Atom"
xmlns:georss="http://www.georss.org/georss/11"
xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
xmlns:geo="http://a9.com/-/opensearch/extensions/geo/1.0/"
xmlns:time="http://a9.com/-/opensearch/extensions/time/1.0/"
targetNamespace="http://www.geoww.net/geochannel" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="20121018A001">
  <annotation>
    <documentation>This schema codifies the information models:
      extended-KML encoding to support GeoChannel element (involving its sub-elements) in KML files;
      extended-KML encoding for returning OpenSearch results of GeoChannel Catalogue Services;
      extended-XML encoding for describing OpenSearch services (i.e..OpenSearchDescription) to support
      GeoChannel information;
      extended-XML feed (e.g. Atom, RSS) encoding to support PubSubHubBub protocol and GeoChannel
      information for real-time harvesting GeoChannel metadata for GeoChannel Catalogue Services.
    </documentation>
  </annotation>
  <!-- Import the schema of current KML standard, OGC KML 2.2, to be extended to support GeoChannel
  information. -->
  <import namespace="http://www.opengis.net/kml/2.2"
  schemaLocation="http://schemas.opengis.net/kml/2.2.0/ogckml22.xsd"/>
  <!-- import atom feed schema. -->
  <import namespace="http://www.w3.org/2005/Atom"
  schemaLocation="http://www.kbcafe.com/rss/atom.xsd.xml"/>
  <!-- import atom GeoRSS schema. -->
  <import namespace="http://www.georss.org/georss/11"
  schemaLocation="http://www.geoww.net/schemas/georss11.xsd"/>
  <!-- Import the schema of Opensearch, which was originally an old online version and has been
  revised/corrected by Xuelin He, according to the latest Opensearch specification 1.1 draft 5.
  The old online version is on:
  http://lastfmapi.svn.sourceforge.net/viewvc/lastfmapi/lastfmlib/trunk/lastfmlib/src/jaxme/OpenSearch.xsd
  ?view=markup
  -->
  <import namespace="http://a9.com/-/spec/opensearch/1.1/"
  schemaLocation="http://www.geoww.net/schemas/OpenSearch.xsd"/>
  <import namespace="http://a9.com/-/opensearch/extensions/geo/1.0/"
  schemaLocation="http://www.geoww.net/schemas/OpenSearch-Geo-Schema.xsd"/>
  <import namespace="http://a9.com/-/opensearch/extensions/time/1.0/"
  schemaLocation="http://www.geoww.net/schemas/OpenSearch-Time-Schema.xsd"/>
  <!-- Import the schema of the GeoAccessFeed - extended-KML Schema -->
  <import namespace="http://www.geoww.net/geofeed"
  schemaLocation="http://www.geoww.net/schemas/GeoAccessFeed--extended-KML_Schema.xsd"/>

```

```

    <element name="GeoChannel" type="geochannel:GeoChannelType"
substitutionGroup="kml:AbstractFeatureGroup"/>
    <element name="GeoChannelSearch" type="geochannel:GeoChannelSearchType"
substitutionGroup="kml:AbstractContainerGroup"/>
    <complexType name="GeoChannelSearchType">
        <complexContent>
            <extension base="kml:AbstractContainerType">
                <sequence>
                    <element ref="opensearch:totalResults" minOccurs="0"/>
                    <element ref="opensearch:startIndex" minOccurs="0"/>
                    <element ref="opensearch:itemsPerPage" minOccurs="0"/>
                    <element ref="opensearch:Query" minOccurs="0" maxOccurs="unbounded"/>
                    <element name="SearchLinks">
                        <complexType>
                            <sequence>
                                <element ref="atom:link" minOccurs="0"
maxOccurs="unbounded"/>
                            </sequence>
                        </complexType>
                    </element>
                    <element ref="geochannel:GeoChannel" minOccurs="0"
maxOccurs="unbounded"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <complexType name="GeoChannelType">
        <complexContent>
            <extension base="kml:AbstractFeatureType">
                <sequence>
                    <element ref="geochannel:Metadata" minOccurs="0"/>
                    <element ref="geofeed:AccessorSelector" minOccurs="0"/>
                    <element ref="geofeed:AccessFeed" minOccurs="0"/>
                </sequence>
                <attribute name="channel_ID" use="required">
                    <simpleType>
                        <restriction base="string">
                            <minLength value="1"/>
                            <pattern value=".*[^\s].*" />
                        </restriction>
                    </simpleType>
                </attribute>
            </extension>
        </complexContent>
    </complexType>

```

```

</complexType>
<element name="Metadata" type="geochannel:MetadataType"/>
<element name="MetadataList" type="geochannel:MetadataListType"/>
<complexType name="MetadataType">
  <complexContent>
    <extension base="kml:AbstractObjectType">
      <sequence>
        <element name="title" type="string"/>
        <element ref="kml:AbstractGeometryGroup" minOccurs="0"
maxOccurs="unbounded"/>
        <element ref="geochannel:MetadataList"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<complexType name="MetadataListType">
  <complexContent>
    <extension base="kml:AbstractObjectType">
      <sequence>
        <element name="GeowwURI">
          <complexType>
            <sequence>
              <element name="note" type="string"/>
              <element name="geowwURI" type="anyURI"/>
            </sequence>
          </complexType>
        </element>
        <element ref="kml:Data" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<element name="CatalogMetadata">
  <annotation>
    <documentation> These sub-elements below describe the metadata information model of a
GeoChannel for the discovery (search) purpose. They may be included in the feeds that are published by
a GeoChannel and harvested by a GeoChannel catalogue service. These metadata information can be
returned (included) in the search results of a GeoChannel Catalogue Service, for example a search result
is encoded in the extended-KML format as codified to the MetadataList element in this schema.
    </documentation>
  </annotation>
  <complexType>
    <sequence>
      <!-- the title element below is a mandatory element-->

```

```

<element name="title" type="string"/>
<element name="alternativeTitle" type="string" minOccurs="0"/>
<element name="serviceLanguage" type="string" minOccurs="0"/>
<element name="abstract" type="string" minOccurs="0"/>
<element name="topicCategory" type="string" minOccurs="0"/>
<!-- the geoChannelIdentifier element below is a mandatory element-->
<element name="geoChannelIdentifier" type="string"/>
<element name="keyword" type="string" minOccurs="0"/>
<element name="resourceType" type="string" minOccurs="0"/>
<!--This value of the spatialFeature element may be encoded in the WKT (i.e.
Well-Known Text) format. -->
<element name="spatialFeature" type="string" minOccurs="0"/>
<element name="spatialMetrics" type="decimal" minOccurs="0"/>
<!--This value of the boundingBox element is defined by "west, south, east, north"
coordinates of longitude, latitude, For example "120,10,134,14"-->
<element name="boundingBox" type="string" minOccurs="0"/>
<!-- the spatialExtent element below refers to the spatial coverage of this GeoChannel
service. A detailed geometry, e.g. a polygon or multi-polygon, can be employed to outline the spatial area.
BoundingBox (i.e. the element boundingBox above )is another simple method for this purpose. This element's
value may be encoded in the WKT (i.e. Well-Known Text) format. -->
<element name="spatialExtent" type="string" minOccurs="0"/>
<element name="verticalExtent" type="string" minOccurs="0"/>
<element name="spatialReferenceSystem" type="string" default="WGS 84"
minOccurs="0"/>
<element name="spatialResolution" type="decimal" minOccurs="0"/>
<element name="TemporalExtent" minOccurs="0">
  <complexType>
    <sequence>
      <element ref="geochannel:TimeSpan" maxOccurs="unbounded"/>
    </sequence>
  </complexType>
</element>
<element name="publicationDate" type="date" minOccurs="0"/>
<!-- the geowwURI element below is a mandatory element-->
<element name="geowwURI" type="anyURI"/>
<element name="dataFormat" type="string" minOccurs="0"/>
<element name="responsibleOrganisation" type="string" minOccurs="0"/>
<element name="frequencyOfUpdate" type="string" minOccurs="0"/>
<element name="limitationOnPublicAccess" type="string" minOccurs="0"/>
<element name="useConstraints" type="string" minOccurs="0"/>
<element name="coupledGeoChannels" type="string" minOccurs="0"/>
<element name="additionalInformation" type="string" minOccurs="0"/>
<element name="metadataDateTime" type="dateTime"/>
<element name="metadataLanguage" type="string" minOccurs="0"/>

```



```

        <element name="metadataContact" type="string" minOccurs="0"/>
    </sequence>
</complexType>
</element>
<element name="TimeSpan">
    <complexType>
        <sequence>
            <element name="start" type="dateTime" minOccurs="0"/>
            <element name="end" type="dateTime" minOccurs="0"/>
        </sequence>
    </complexType>
</element>
<element name="SearchCoverage">
    <annotation>
        <documentation>
            This element is to be used to extend an OpenSearchDescription document, indicating spatial
            and/or temporal coverage that a GeoChannel Catalogue (Discovery) Service can involve.
        </documentation>
    </annotation>
    <complexType>
        <sequence>
            <element name="SpaceExtent" minOccurs="0">
                <complexType>
                    <choice minOccurs="1" maxOccurs="unbounded">
                        <element ref="georss:box"/>
                        <element ref="georss:polygon"/>
                        <element ref="georss:circle"/>
                    </choice>
                </complexType>
            </element>
            <element name="TimeExtent" minOccurs="0">
                <complexType>
                    <sequence>
                        <element ref="geochannel:TimeSpan" maxOccurs="unbounded"/>
                    </sequence>
                </complexType>
            </element>
        </sequence>
    </complexType>
</element>
<!--The attributes listed below are to be used for extending the OpenSearchDescription's Url element's
parameters or Query element's attributes. These are all queryable metadata of GeoChannel catalogue.-->
<attribute name="title" type="string"/>
<attribute name="alternativeTitle" type="string"/>

```

```

<attribute name="serviceLanguage" type="string"/>
<attribute name="abstract" type="string"/>
<attribute name="topicCategory" type="string"/>
<attribute name="geoChannelIdentifier" type="string"/>
<attribute name="keyword" type="string"/>
<attribute name="resourceType" type="string"/>
<!-- This value of the spatialFeature attribute may be encoded in the WKT (i.e. Well-Known Text) format.
-->
<attribute name="spatialFeature" type="string"/>
<attribute name="spatialMetrics" type="decimal"/>
<!-- This value of the boundingBox attribute is defined by "west, south, east, north" coordinates of
longitude, latitude, For example "120,10,134,14". -->
<attribute name="boundingBox" type="string"/>
<!-- the spatialExtent attribute below refers to the spatial coverage of a GeoChannel resource A detailed
geometry, e.g. a polygon or multi-polygon, can be employed to outline the spatial area. BoundingBox (i.e. the
attribute boundingBox above )is another simple method for this purpose. This attribute's value may be
encoded in the WKT (i.e. Well-Known Text) format. -->
<attribute name="spatialExtent" type="string"/>
<attribute name="verticalExtent" type="string"/>
<attribute name="spatialReferenceSystem" type="string" default="WGS 84"/>
<attribute name="spatialResolution" type="decimal"/>
<attribute name="TemporalExtent" type="string"/>
<attribute name="publicationDate" type="date"/>
<attribute name="dataFormat" type="string"/>
<attribute name="responsibleOrganisation" type="string"/>
<attribute name="frequencyOfUpdate" type="string"/>
<attribute name="limitationOnPublicAccess" type="string"/>
<attribute name="coupledGeoChannels" type="string"/>
<attribute name="additionalInformation" type="string"/>
<attribute name="metadataDateTime" type="dateTime"/>
<attribute name="metadataLanguage" type="string"/>
<attribute name="metadataContact" type="string"/>
</schema>

```

## Appendix 5 – GeoChannel Markup Language

*(Incorporating the components in previous Appendixes)*

<!-- Notes:

The **GeoChannel Markup Language** here is an umbrella term which uses this single term to represent a common category that covers a set of information models for representing GeoChannel-related information.

In a broad sense, the GeoChannel Markup Language may cover/involve some XML markup languages that are industrial standards already and were not originally created for the GeoChannel technology, e.g. KML, CityGML, ARML, etc., as they can be employed to represent GeoChannel-related information. However, most specifically the term GeoChannel Markup Language refers to the content parts that extend, derive, or customize these existing markup languages, or completely independent parts irrelevant to other languages, all of which specially work for GeoChannel purpose.

The scope of the GeoChannel Markup Language is extensible. With future advancement of the research on the GeoChannel technology, more extended or independent components may be added; And with the further adoption of the GeoChannel technology to broader application domains, more other existing domain-specific markup languages may be incorporated. For example, the domains of social media, sensor networks, etc., which have the potential to employ the GeoChannel technology.

So the GeoChannel Markup Language is a general term representing current and future coverage of various components of information models. This term can also inspire promising directions and scopes to advance and apply GeoChannel technology.

This XML schema here is just an aggregator/wrapper that collects some (but not exhaustive) parts of information models and language components, which are defined or exist separately somewhere, to form this general camp of this GeoChannel Markup language. The scope of the involved content components can be extended, customized or tailored anytime in an on-demand way. This general schema can be referenced in GeoChannel applications which can also sometime only reference to those on-demand components separately defined in specific schemas respectively.

This schema here currently includes the information models: GeoChannel & Discovery information model (in Appendix 4), and GeoChannel AccessFeed model (in Appendix 3) which involve GeoPolicy language model (in Appendix 1) and Geo-off-on communication model (in Appendix 2). All these models here employ extended-KML encoding.

----->

<?xml version="1.0" encoding="UTF-8"?>

<schema xmlns="http://www.w3.org/2001/XMLSchema"

xmlns:geoww="http://www.geoww.net/geochannel-language"

```

xmlns:geochannel="http://www.geoww.net/geochannel"
xmlns:geofeed="http://www.geoww.net/geofeed"
xmlns:geopolicy="http://www.geoww.net/geopolicy"
xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
xmlns:geo="http://a9.com/-/opensearch/extensions/geo/1.0/"
xmlns:time="http://a9.com/-/opensearch/extensions/time/1.0/"
xmlns:geoffon="http://www.geoww.net/geoffon"
xmlns:kml="http://www.opengis.net/kml/2.2"
xmlns:atom="http://www.w3.org/2005/Atom"
xmlns:georss="http://www.georss.org/georss/11"
targetNamespace="http://www.geoww.net/geochannel-language"
elementFormDefault="qualified" attributeFormDefault="unqualified" version="2012-12-18-A001">

```

```
<annotation>
```

```
<documentation>
```

Here the GeoChannel Markup Language is an umbrella term which uses this single term to represent a common category that covers a set of information models for representing GeoChannel-related information.

This schema here currently includes the information models:

GeoChannel Discovery information model, and GeoChannel AccessFeed model which involve GeoPolicy language model and Geo-off-on communication model. All these models here employs extended-KML encoding.

This GeoChannel Markup Language schema is extensible. Further component parts may be added into this schema document.

```
</documentation>
```

```
</annotation>
```

```
<!-- Import the schema of current KML standard, OGC KML 2.2, to be extended by some
GeoChannel-related information models. -->
```

```
<import namespace="http://www.opengis.net/kml/2.2"
```

```
schemaLocation="http://schemas.opengis.net/kml/2.2.0/ogckml22.xsd"/>
```

```
<!-- Import the schema of KML-extended encoding model for GeoChannel & Discovery information -->
```

```
<import namespace="http://www.geoww.net/geochannel"
```

```
schemaLocation="http://www.geoww.net/schemas/GeoChannel_and_Discovery--extended-KML_and_-XML_Schema.xsd"/>
```

```
<!-- Import the schema of KML-extended encoding model for AccessFeeds -->
```

```
<import namespace="http://www.geoww.net/geofeed"
```

```
schemaLocation="http://www.geoww.net/schemas/GeoAccessFeed--extended-KML_Schema.xsd"/>
```

```
<!-- Import the schema of KML-extended encoding model for the GeoPolicy Language -->
```

```
<import namespace="http://www.geoww.net/geopolicy"
```

```
schemaLocation="http://www.geoww.net/schemas/GeoPolicy_Language--extended-KML_Schema.xsd"/>
```

```
<!-- Import the schema of KML-extended encoding model for the Geoffon Communication Model -->
```

```
<import namespace="http://www.geoww.net/geoffon"
```

```
schemaLocation="http://www.geoww.net/schemas/Geoffon_Communication_Model--extended-KML_Schema.xsd"/>
```

```
</schema>
```

## Appendix 6 – Metadata Information Model for GeoChannel Catalogue Services

Name	Definition	Data Type	Queryable (Q)/ Returnable (R)	Obligation (Present): Mandatory (M)/ Conditional (C)/ Optional (O)	Mapping to OGC CSW ISO Application Profile Metadata
Title	a name given to this GeoChannel resource	CharacterString	Q/R	M	Title
AlternativeTitle	short name, other name, acronym or alternative language title for this GeoChannel resource	CharacterString	Q/R	O	AlternateTitle
ServiceLanguage	language used in the GeoChannel resource	CharacterString	Q/R	C	Language
Abstract	brief narrative summary of the content of this GeoChannel resource	CharacterString	Q/R	M	Abstract
TopicCategory	main theme(s) of the involvement of this GeoChannel resource.	CharacterString	Q/R	O	TopicCategory
GeoChannelIdentifier	a value uniquely identifying this GeoChannel resource. Usually a GUID (Globally Unique Identifier) is used.	CharacterString Identifier	Q/R	M	ResourceIdentifier
ResourceIdentifiers	a list of identifiers for the resources accessible via this AccessFeed. A resource identifier is usually made of concatenated URLs of this AccessFeed and its hierarchically contained Accessors.	CharacterString	Q/R	M	-(not supported)
Keyword	topic(s) of the content of this GeoChannel resource	CharacterString	Q/R	O	Subject
ResourceType	brief classification of the functions/purposes of this GeoChannel, e.g. the generic types may involve one or some of the work for delivering/providing geospatial dynamics, location-based services, social interaction, remote control, etc.	CharacterString	Q/R	C	Type
SpatialFeature	geometric representation/depiction for the spatial property and characteristic of this GeoChannel resource	Geometry	Q/R	C	-(not supported)
SpatialMetrics	the approximate measurement about the spatial features of a GeoChannel, such as its spatial coverage/involvement. Some	numeric Real	Q/R	O	-(not supported)

	indicators may be used, e.g. area, length, volume, etc., for the query/filter of the matched GeoChannels.				
BoundingBox	using a BoundingBox to indicate the spatial coverage of this GeoChannel service. This is a simple/brief method for representing the SpatialExtent—another metadata about this GeoChannel.	Geometry: BoundingBox	Q/R	C	BoundingBox
SpatialExtent	the spatial coverage of this GeoChannel service. A detailed geometry, e.g. a polygon or multi-polygon, can be employed to outline the spatial area. BoundingBox is another simple method for this purpose.	Geometry: Polygon, MultiGeometry	Q/R	C	-(not supported)
VerticalExtent	the vertical dimension for outline the spatial extent if it is necessary to outline the vertical extent of this GeoChannel service's function domain.	min value:Real max value:Real CRS: Identifier	Q/R	O	-(not supported)
SpatialReferenceSystem	identifier of the system of spatial referencing, whether by coordinates or geographic identifiers, used in this GeoChannel resource	CharacterString Identifier	Q/R	M	CRS
SpatialResolution	measure of the granularity of the data produced by this GeoChannel. It usually refers to a suggested scale of client-side map view where this GeoChannel produces the visual output. Spatial resolution may also be expressed as distance (in metres).	numeric	Q/R	O	SpatialResolution
TemporalExtent	temporal range for the usage of this GeoChannel service, which will be available during this time period.	DateTime, DateTime	Q/R	C	TemporalExtent
PublicationDate	the date when this GeoChannel service was firstly published/made for use.	Date	Q/R	O	PublicationDate
GeowwURI	the Geoww-URI of the target GeoChannel resource. The Geoww-URI schema model (invented in this research) is used for encoding Geoww-URIs.	CharacterString: URL	R	M	Linkage
DataFormat	format in which the digital data can be provided by this GeoChannel. It usually uses some well-known formats such as KML, GeoJSON, GeorSS. ARML (Augmented Reality Markup Language) will also be a candidate format for representing geospatial dynamics.	CharacterString	Q/R	O	Format

ResponsibleOrganisation	details of the organisation(s) responsible for the establishment, management, maintenance and distribution of this GeoChannel resource. Contact information is needed such as email address.	CharacterString	Q/R	O	OrganisationName
FrequencyOfUpdate	frequency with which modifications are made to the GeoChannel metadata or its service resource.	CharacterString		C	-(not supported)
LimitationOnPublicAccess	indicate whether or not the openness of access by public users, or is only limited to a specific community such as internal access by a organization. In the latter case, usually credentials (e.g. user name and password) are needed to authenticate the access.	CharacterString	Q/R	C	Right
UseConstraints	restrictions on using this GeoChannel resource. Usually spatio-temporal conditions can be enforced to authorize the access to a GeoChannel resource. In case of the reason for security or confidentiality, the real conditions applying to access and use may not be disclosed here, but only abstract indication of access restriction is stated.	CharacterString	R	C	useLimiiation
CoupledGeoChannels	the identifiers of some other GeoChannels (if any) as the dependency to this GeoChannel. A GeoChannel may need to work with (depend on) other GeoChannels jointly, all of whose Accessors are plugged in a client-side GeoChannel Accessor Bus (Hub).	CharacterString: List of GeoChannel URIs	Q/R	C	-(not supported)
AdditionalInformation	other descriptive information about this GeoChannel resource. Any references (e.g. a URL) to external information that are considered useful may be recorded.	CharacterString	Q/R	O	supplementalInformation
MetadataDateTime	date on which the metadata was last updated, or was confirmed as being up-to-date, or if not updated, then the date it was created.	DateTime	Q/R	M	dateStamp
MetadataLanguage	language used for documenting the metadata	CharacterString	Q/R	C	Language
MetadataContact	party responsible for the creation and maintenance of the metadata. Contact information is included such as email address.	CharacterString	Q/R	O	OrganisationName

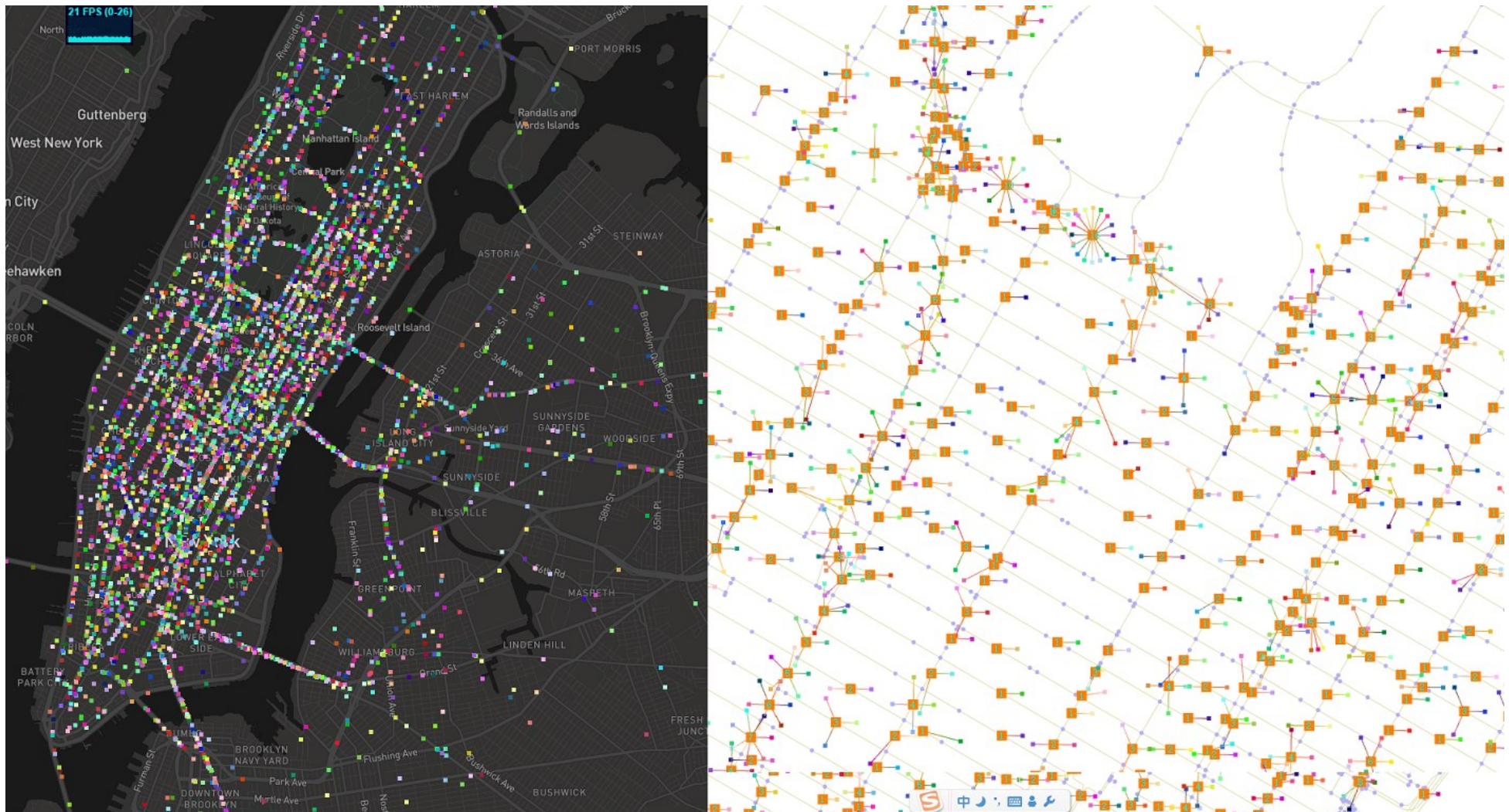


## Appendix 7 – Demo Project: Mass Dynamics of real-time Maps & Context Relations for Taxi Trips in New York City

Latest version online: [www.geonoon.net/demos](http://www.geonoon.net/demos), presented in Chapter 9.

*Demonstrating:*

Real-time map with large-scale dataset; the GeoChannel Web for mapping and applying dynamic & random relations; .....





## Appendix 8 – Demo Project: Virtual-Real-World Services & Interaction in British Museum

Latest version online: [www.geonoon.net/demos](http://www.geonoon.net/demos), presented in Chapter 9.

*Demonstrating:*

GeoAccessFeed technolog for pervasive services in physical and virtual world;  
the GeoChannel Web for bridging physical-virtual-world intersection; .....

