UCL CENTRE FOR ADVANCED SPATIAL ANALYSIS

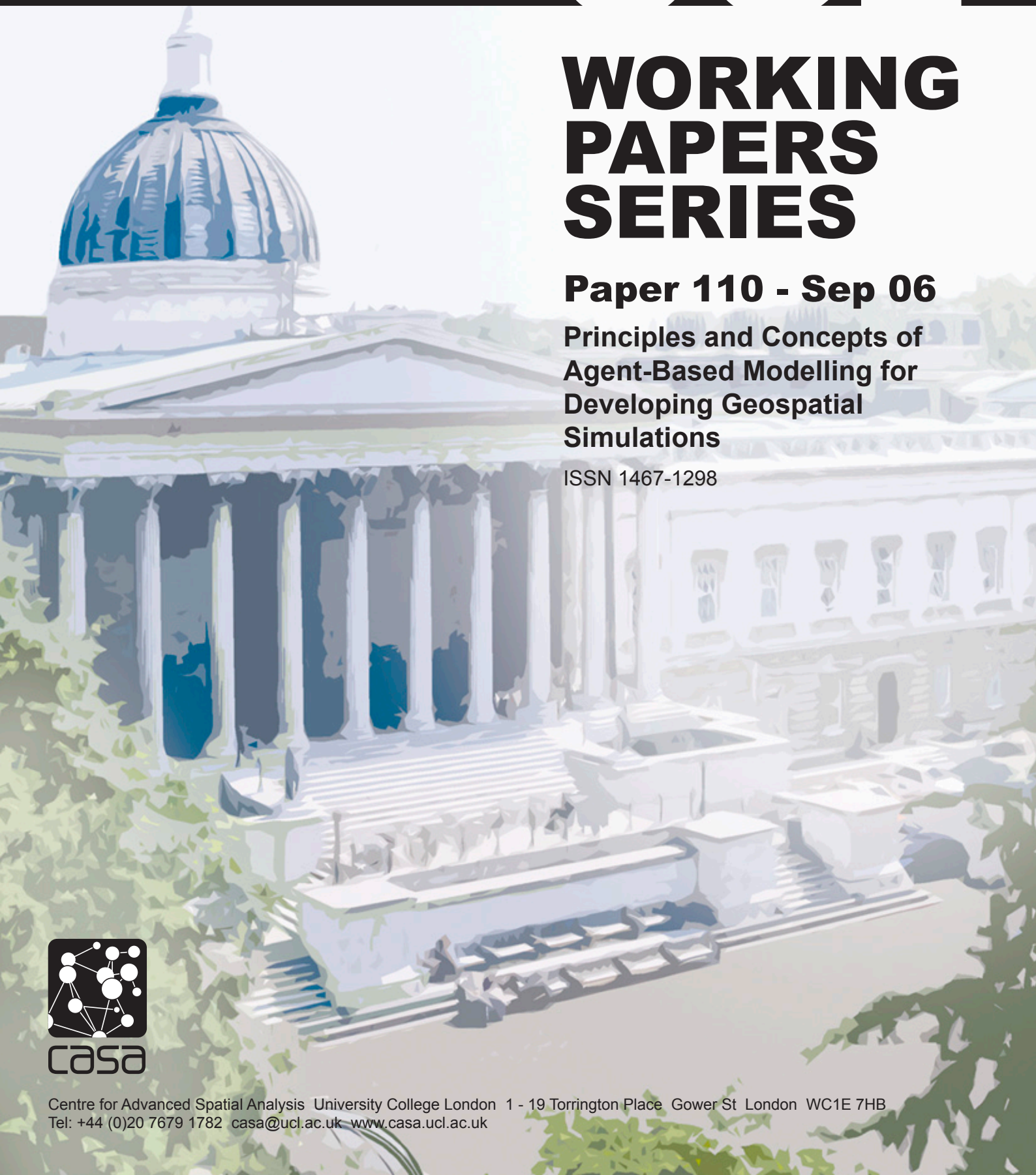
UCL

WORKING PAPERS SERIES

Paper 110 - Sep 06

Principles and Concepts of Agent-Based Modelling for Developing Geospatial Simulations

ISSN 1467-1298



casa

Centre for Advanced Spatial Analysis University College London 1 - 19 Torrington Place Gower St London WC1E 7HB
Tel: +44 (0)20 7679 1782 casa@ucl.ac.uk www.casa.ucl.ac.uk

Principles and Concepts of Agent-Based Modelling for Developing Geospatial Simulations

By

Christian J. E. Castle

E-mail: c.castle@ucl.ac.uk · Web: http://www.casa.ucl.ac.uk/cjec/phd

and

Andrew T. Crooks

E-mail: andrew.crooks@ucl.ac.uk · Web: http://virgil.casa.ucl.ac.uk/andrew

Weblog: http://www.gisagents.blogspot.com

Centre for Advanced Spatial Analysis

University College London, 1-19 Torrington Place, London, WC1E 6BT, UK

Tel. +44 (0)207 679 1782 · Fax +44 (0)207 813 2843 · http://www.casa.ucl.ac.uk

Abstract

The aim of this paper is to outline fundamental concepts and principles of the Agent-Based Modelling (ABM) paradigm, with particular reference to the development of geospatial simulations. The paper begins with a brief definition of modelling, followed by a classification of model types, and a comment regarding a shift (in certain circumstances) towards modelling systems at the individual-level. In particular, automata approaches (e.g. Cellular Automata, CA, and ABM) have been particularly popular, with ABM moving to the fore. A definition of agents and agent-based models is given; identifying their advantages and disadvantages, especially in relation to geospatial modelling. The potential use of agent-based models is discussed, and how-to instructions for developing an agent-based model are provided. Types of simulation / modelling systems available for ABM are defined, supplemented with criteria to consider before choosing a particular system for a modelling endeavour. Information pertaining to a selection of simulation / modelling systems (Swarm, MASON, Repast, StarLogo, NetLogo, OBEUS, AgentSheets and AnyLogic) is provided, categorised by their licensing policy (open source, shareware / freeware and proprietary systems). The evaluation (i.e. verification, calibration, validation and analysis) of agent-based models and their output is examined, and noteworthy applications are discussed.

Geographical Information Systems (GIS) are a particularly useful medium for representing model input and output of a geospatial nature. However, GIS are not well suited to dynamic modelling (e.g. ABM). In particular, problems of representing time and change within GIS are highlighted. Consequently, this paper explores the opportunity of linking (through coupling or integration / embedding) a GIS with a simulation / modelling system purposely built, and therefore better suited to supporting the requirements of ABM. This paper concludes with a synthesis of the discussion that has proceeded.

Key Words: Agent-Based Modelling (ABM), agent-based models, geospatial / spatially explicit modelling, verification, calibration, validation, Geographical Information Systems (GIS), linkage (coupling or embedding / integration).

Table of Contents

List of Figures

List of Tables

Principles and Concepts of Agent-Based Modelling for Developing Geospatial Simulations

1: Introduction

The aim of this paper is to outline fundamental concepts and principles of the Agent-Based Modelling (ABM) paradigm, with particular reference to the development of geospatial simulations (i.e. spatially explicit geographic phenomena - where the nature of the features and movement that is represented varies over the Earth's surface). Essentially, geospatial models depend on the location of the features or phenomena being modelled, such that if one or more of those locations change, the results of the model change (Wegener, 2000). Geographical Information Systems (GIS) are a particularly useful medium for representing model input and output of a geospatial nature. However, GIS are not well suited to dynamic modelling (Goodchild, 2005; Maguire, 2005). Consequently, this paper explores the opportunity of linking (through coupling or integration / embedding) a GIS with a simulation / modelling system purposely built, and therefore better suited to supporting the requirements of ABM.

A particular goal of this paper is to present the reader with a comprehensive introduction to the development of geospatial agent-based models. A resource the authors of this paper believe is missing in literature to date. Ubiquitous notions within the literature are represented in concise fashion, unifying and (in some instances) descrambling terminology to present a clear, simple, and logical discussion for the uninitiated. This paper does not claim to introduce new concepts or develop original theory. On the contrary, its purpose is to consolidate pre-existing concepts and theory in a comprehendible manor. Bearing this in mind, the specific structure of the paper is as follows:

Section 1.2 begins with a definition of the term agent, identifying characteristics germane to agents, and the general structure of agent-based models. Subsequently, alternative terms used (often confusingly) to describe ABM are identified and explained. This is followed by advantages and disadvantages of developing agent-based models, especially in relation to geospatial modelling, as well as the potential use of agent-based models. Afterwards, how-to instructions for developing an agent-based model are given. Types of simulation / modelling

systems available for ABM are defined, supplemented with guidelines to consider before choosing a particular system for a modelling endeavour. Information pertaining to a selection of simulation / modelling systems (Swarm, MASON, Repast, StarLogo, NetLogo, OBEUS, AgentSheets and AnyLogic) is provided, categorised by their licensing policy (open source, shareware / freeware and proprietary systems). The penultimate subsection discusses the evaluation (i.e. verification, calibration, validation and analysis) of agent-based models and their output. The section concludes with the identification of notable ABM applications, with particular reference to research of a geospatial nature. Section 1.3 of this paper explores the current modelling capabilities of GIS. In particular, problems of representing time and change within GIS are highlighted. This is followed by a rationale for linking (i.e. through coupling or integration / embedding) a GIS with a simulation / modelling system more suited to the requirements of ABM. Section 1.4 draws this paper to a close, synthesizing the discussion that has proceeded.

However, at this juncture of the paper the reader is provided with a brief definition of modelling from a GIS viewpoint. This is followed by a classification of model types, and a comment regarding a shift towards modelling certain systems at the individual-level. In particular, automata approaches (e.g. Cellular Automata, CA, and ABM) have been particularly popular.

1.1 Modelling: A Definition

The term modelling can have different connotations in the GIS world, so it would be sensible to begin with an attempt to define its meaning; at least in the context of this paper. A model is a simplified representation of reality (i.e. of one or more processes that are believed to occur in the real-world; Longley and Batty, 2003), of which there are several types (see Section 1.1.1). A model can be constructed as a computer programme that uses (usually to some degree) a simplified digital representation of one or more aspects of the real-world, transforming them to create a new representation.

Models can be static, if the input and output both correspond to the same point in time, or dynamic if the output represents a later point in time than the input (Longley *et al.*, 2005). The common element in all such models is the operation of the GIS in multiple stages, whether they are used to create indicators from input layers or to represent time steps in the

operation of a dynamic process (Goodchild, 2005). In this setting, modelling can serve a number of purposes. Static models provide indexes or indicators that can provide some predictors of impacts, sensitivities, or vulnerabilities. Dynamic models go further by attempting to project quantifiable impacts into the future and are used to assess different management or development ('what if') scenarios. This experimental aspect is perhaps the most compelling justification for modelling, and a firm belief upon which this paper is based.

1.1.1: Model Types

Although rarely considered in the context of GIS, analogue models are probably the most common type. An analogue model is defined as a scale model, a representation of a real-world system in which every aspect is modelled in miniature. The success of an analogue model depends on the degree to which the system can be scaled. Alternatively, a digital or computational model conducts all operations using a computer. Data are assembled in a data model and coded using a variety of coding schemes that reduce relevant aspects of the real-world to a sequence of binary values. Goodchild and Proctor (1997) explain that unlike analogue models, digital models do not have a representative fraction, since there is no distance in the model to compare to distance in the real-world. Instead the level of geographic detail is captured in the spatial resolution, or the smallest feature represented in the database.

Temporal resolution is just as important as spatial resolution within a dynamic model, since it defines the length of each time increment. Any dynamic model operates in a discrete sequence of such time steps; a subsequent iteration calculates the model's next prediction based on the current state. Essential to any model is the use of an appropriate spatial and temporal resolution for the phenomena of interest. Both spatial and temporal resolutions affect the relationship between the real-world process and the replica computer model. The two will inevitably be different; thus a model will leave the user with some uncertainty about the real-world process because of the model's level of abstraction.

1.1.2: Individual or Aggregate

Hypothetically, it is possible to model any system using a set of rules about the behaviour of the system's constituent elements. For example, the behaviour of a crowd can be modelled through rules likely to characterise the behaviour of every individual. However, depending

on the size of the crowd and the purpose of the model, this may not be practical or even useful. Continuous-field models address this problem by replacing individual objects with continuously varying estimates of abstracted properties, for example the density of people in a crowd (Goodchild, 2005). Alternatively, individual objects can be aggregated into larger wholes, modelling the behaviour of the system through these aggregates. However, aggregate systems subsume variation and processes that fall below the implied spatial resolution of the representation. According to Longley (2004), aggregation creates scientific quicksand. The scale and configuration of the constituent elements of information (i.e. objects located in time and space), can exert critical influence on the outcomes of spatial analysis (see Openshaw, 1984, for a discussion of the Modifiable Areal Unit Problem, MAUP, and Bailey and Gatrell, 1995, for the concept of ecological fallacy). Areal data rarely has any validity independent of particular applications, if they have any validity at all. Problems associated with aggregating data are compounded in modelling when the focus is upon interaction, process or the representation of dynamics. Spatial analysis of unique individuals modelled as mobile point referenced 'events' [1] present the logical endpoint of the drive towards disaggregation (Longley, 2004).

Progress is clearly being made in the use of disaggregated data. Increased computer power and storage capacity has made individual-level modelling more practical in recent times. An example of which can clearly be seen in the evolution of pedestrian modelling (see Galea and Gwynne, 2006), where there has been a concerted movement from aggregate to individual-level modelling. Essential to the progression of individual-level modelling has been the development of automata approaches, which have been at the forefront of computer modelling research (Benenson and Torrens, 2004). An automaton is a processing mechanism with characteristics that change over time based on its internal characteristics, rules, and external input. Automata process information input to them from their surroundings, and their characteristics are altered according to rules that govern their reaction to those inputs. Two classes of automata tools, CA[2] and ABM have been particularly popular; their use has dominated the research literature. The origin of CA and ABM are well documented (see Wolfram, 2002; Parker *et al.*, 2003; Torrens, 2004), and it is not the intention of the authors

[1] Humans and their activities are depicted in GIS as mobile point-referenced 'events' (Martin, 1996).

[2] Cellular automata are arrangements of individual automata, usually in a regular tessellated space (e.g. a rectangular grid), although irregular geometries can also be used (e.g. Voronoi polygons).

to reiterate their foundations ad nauseam. However, it is important to provide a comprehensive yet concise introduction to ABM, since this concept is the foundation of this paper.

1.2: Agents and Agent-Based Models

There is no universal agreement on the precise definition of the term 'agent', although definitions tend to agree on more points than they disagree (Macal and North, 2005). Agent characteristics are difficult to extract from the literature in a consistent and concise manner, because they are applied differently within disciplines. Furthermore, the agent-based concept is a mindset more than a technology, where a system is described from the perspective of its constituent parts (Bonabeau, 2002). The concept of an agent is meant to be a tool for analysing a system, not an absolute classification where entities can be defined as agents or non-agents (Russell and Norvig, 2003). For example, some modellers consider any type of independent component (i.e. software, model, individual, etc), to be an agent. Others insist that a component's behaviour must be adaptive in order for it to be considered an agent, where the term agent is reserved for components that can, in some sense, learn from their environments and change their behaviours accordingly (see below). Nonetheless, from a pragmatic modelling standpoint, there are several features that are common to most agents (Wooldridge and Jennings, 1995 - extended and explained further by Franklin and Graesser, 1996; Epstein, 1999; Torrens, 2004; Macal and North, 2005):

- **Autonomy:** Agents are autonomous units (i.e. governed without the influence of centralised control), capable of processing information and exchanging this information with other agents in order to make independent decisions. They are free to interact with other agents, at least over a limited range of situations, and this does not (necessarily) affect their autonomy. In this respect, agents are active rather than purely passive (see below).
- **Heterogeneity:** The notion of mean-individuals is redundant; agents permit the development of autonomous individuals. Groups of agents can exist, but they are spawned from the bottom-up, amalgamations of similar autonomous individuals.
- **Active:** Agents are active because they exert independent influence in a simulation. The following active features can be identified:

- **Pro-active / goal-directed:** Agents are often deemed goal-directed, having goals to achieve (not necessarily objectives to maximise) with respect to their behaviours'. For example, agents within a geographic space can be developed to find or follow a set of spatial paths to achieve a goal within a certain constraint (e.g. time), when exiting a building during an emergency.
- **Reactive / Perceptive:** Agents can be designed to have an awareness, or sense of their surroundings. Agents can also be supplied with prior knowledge, in affect a 'mental map' of their environment, thus providing them with an awareness of other entities, obstacles, or required destinations within their environment. Extending the example above, agents could therefore be provided with knowledge of building exit locations.
- **Bounded Rationality:** Throughout the social sciences, the dominant form of modelling is based upon the rational-choice paradigm (Axelrod, in press). Rational-choice models generally assume that agents are perfectly rational optimisers with unfettered access to information, foresight, and infinite analytical ability (Parker *et al.*, 2003). These agents are therefore capable of deductively solving complex mathematical optimisation problems in order to maximise their well being; balancing long-run and short-run payoffs in the face of uncertainty. While rationale-choice models can have substantial explanatory power, some of there axiomatic foundations are contradicted by experimental evidence, leading prominent social scientist to question their empirical validity. However, agents can be configured with 'bounded' rationality (through their heterogeneity), to circumnavigate the potential limitations of these assumptions (i.e. agents can be provided with fettered access to information at the local level). In affect, the aforementioned 'perception' of agents can be constrained. Thus, rather than implementing a model containing agents with optimal solutions that can fully anticipate all future states of which they are part of, agents make inductive, discrete, and adaptive choices that move them towards achieving goals. For instance, an agent may have knowledge of all building exit locations, but agents will be unaware if all exits are accessible (e.g. some may have become blocked through congestion).
- **Interactive / Communicative:** Agents have the ability to communicate extensively. For example, agents can query other agents and / or the environment within a neighbourhood, via neighbourhoods of (potentially) varying size, searching for

specific attributes, with the ability to disregard an input which does not match a desirable threshold.

- **Mobility:** The mobility of agents is a particularly useful feature, not least for spatial simulations. Agents can roam the space within a model. Juxtaposed with agent's ability to interact and their intelligence, this permits a vast range of potential uses.
- **Adaptation / Learning:** Agents can also be designed to be adaptive, which can produce Complex Adaptive Systems (CAS; Holland, 1995). Agents can be designed to alter (limited to a given threshold if required) their state depending on their current state, permitting agents to adapt with a form of memory or learning, but not necessarily in the most efficient way possible. Agents can adapt at the individual level (e.g. learning alters the probability distribution of rules that compete for attention), or the population level (e.g. learning alters the frequency distribution of agents competing for reproduction).

Agent-based models are comprised of multiple, interacting agents situated within a model or simulation environment. A relationship between agents is specified, linking agents to other agents and / or other entities within a system. Relationships may be specified in a variety of ways, from simply reactive (i.e. agents only perform actions when triggered to do so by some external stimulus e.g. actions of another agent), to goal-directed (i.e. seeking a particular goal). The behaviour of agents can be scheduled to take place synchronously (i.e. every agent performs actions at each discrete time step), or asynchronously (i.e. agent actions are scheduled by the actions of other agents, and / or with reference to a clock).

Environments define the space in which agents operate, serving to support their interaction with the environment and other agents. Agents within an environment may be spatially explicit, meaning agents have a location in geometrical space, although the agent itself may be static. For example, within a building evacuation model agents would be required to have a specific location for them to assess their exit strategy. Conversely, agents within an environment may be spatially implicit; meaning their location within the environment is irrelevant. For instance, a model of a computer network does necessarily require each computer to know the physical location of other computers within the network.

In a modelling context, agent-based models can be used as experimental media for running and observing agent-based simulations. To this extent, they can be thought of as a miniature laboratory, where the attributes and behaviour of agents, and the environment in which they are housed, can be altered and the repercussions observed over the course of multiple simulation runs. The ability to simulate individual actions of many diverse agents and measure the resulting system behaviour and outcomes over time (e.g. changes in patterns of pedestrian emergency egress), means agent-based models can be useful tools for studying the effects on processes that operate at multiple scales and organisational levels, and their effects (Brown, 2006). In particular, the roots of ABM are within the simulation of human social behaviour and individual decision-making (Bonabeau, 2002).

The acronym ABM will be used throughout the remainder of this paper, but a caveat is required. There are various alternative terms (and their acronym's) applied in the literature to what, for all intent and purpose, is essentially ABM. Examples include: Agent-Based Computational Modelling, (ABCM), Agent-Based Social Simulation (ABSS), Agent-Based Computation Simulation, and Agent-Based Modelling and Simulation (ABMS). Multi-Agent Systems (MAS) is another very popular term which is often, confusingly, used interchangeably to describe agent-based models. The field of MAS is a well established research and applied branch of Artificial Intelligence (AI), and although ABM has strong roots in the field of AI, agent-based models are not limited to the design and understanding of artificial agents. Impetus to develop MAS was spawned from problems encountered in the implementation of tasks on distributed computational units interacting with one another and with the external environment (Distributed Artificial Intelligence, DAI). The term MAS is more commonly applied outside the social sciences, for example, by computer scientists in relation to agent-oriented software development. Therefore, the MAS field can be characterised as the study of societies of artificial autonomous agents, while the ABM field can be typified as the study of artificial societies of autonomous agents (Conte *et al.*, 1998). These two fields differ in more substantial ways than just their formalism (i.e. logic and AI based in the MAS domain, and mathematically based in the social science domain). However, this will not be considered in this paper (see Conte *et al.*, 1998 for a more detailed treatment).

More importantly, the term agent has connotations beyond ABM. For instance, agents found within agent-based models are different from mobile agent systems, which are light-weight software proxies that perform various functions for users, and to some extent can behave autonomously (Macal and North, 2005). ABM is not the same as object-oriented simulation, although the object-oriented paradigm provides a suitable medium for the development of agent-based models. For this reason, ABM systems are invariably object-oriented.

1.2.1: Advantages of Agent-Based Models

In relation to the context of this paper, there are three main claimed advantages of the agent-based approach over traditional modelling techniques, such as top-down techniques of non-linear dynamical systems in which related state variables are aggregated (e.g. though differential equations). The agent-based approach: 1) captures emergent phenomena; 2) provides a natural environment for the study of certain systems; and, 3) is flexible, particularly in relation to the development of geospatial models.

Emergence is a phenomenon, along with other surprising and unexpected behaviours unfamiliar to the classical sciences, such as self-organisation, chaos, adaptation, etc, which are characteristic of complex systems (Couclelis, 2002). Neural networks are well known examples of complex structures that are capable of organised behaviour, as a result of parallel interaction of many interconnected neurons. More specifically, the study of phenomena characterised by interactions among many distinct components is labelled 'aggregate[3] complexity' (Manson, 2001; in press). Emergent phenomena are characterised by stable macroscopic patterns arising from local interaction of individual entities (Epstein and Axtell, 1996). By definition, emergent phenomena cannot be reduced to the system's parts; the whole is more than the sum of the parts. Thus, emergent phenomena can exhibit properties that are decoupled (i.e. logically independent) from the properties of the system's parts. For example, a traffic jam often forms in the opposing lane direction to a traffic accident; a consequence of 'rubber-necking'. Studying the behaviour of collections of entities focuses

[3] Manson's taxonomy of complexity research also includes algorithmic (i.e. the complexity of a system lies in the difficulty faced in describing system characteristics), and deterministic (i.e. unpredictable dynamic behaviour of relatively simple deterministic systems, where unpredictable refers to the sensitivity of outcomes based on initial conditions) complexity. These categories refer to aspects of phenomena that are not mutually exclusive. These three major divisions afford a more coherent understanding of complexity theory, but this it is not the only possible classification (see Reitsma, 2003 and Manson, 2003 for a heated debate), though it provides a useful framework for the purpose of this paper.

attention on relationships between entities (O'Sullivan, 2004). Characteristics of emergent phenomena make them difficult to understand and predict, particularly as emergent outcomes can be counterintuitive (Epstein, 1999). In summary, the purpose of aggregate complexity is to arrive at understanding by reduction, and reassembly of a system of aggregate complexity, where the critical break with previous reductionist science is the attempt at reassembly (O'Sullivan, 2004). Aggregate complexity is of particular interest to geographers because it implies that the local spatial configuration of interactions affects outcomes at the whole system level.

Since agent-based models describe the behaviour and interactions of a system's constituent parts from the bottom up, they are the canonical approach for modelling emergent phenomena. Bonabeau (2002) has identified a non-exhaustive list of conditions where agent-based models can be useful for capturing emergent behaviour:

1) Interaction between agents is complicated, non-linear, discontinuous, or discrete (i.e. the behaviour of an agent can be altered dramatically, even discontinuously), by other agents. This can be particularly useful if describing discontinuity of individual behaviour is difficult, for example, using differential equations;
2) The ability to design a heterogeneous population of agents with an agent-based model is significant. Agents can represent any type of unit, from which intuitive collections of individual units can be formed, from the bottom up. Unlike agent-based models, aggregate differential equations tend to smooth out fluctuations. This is important because under certain conditions, fluctuations can be amplified: a system can be linearly stable but susceptible to large perturbations. Heterogeneity also allows for the specification of agents with varying degrees of rationality (see above). This offers advantages over approaches that assume perfectly rational individuals, if they consider individuals at all;
3) The topology of agent interactions is heterogeneous and complex. Aggregate flow equations usually assume global homogeneous mixing, but the topology of an interaction network can lead to significant deviations from predicted aggregate behaviour. This is particularly poignant for social processes, because physical or social networks matter (Axtell, 2000); and,
4) When agents exhibit complex behaviour, including learning and adaptation.

In many cases ABM is a natural method for describing and simulating a system composed of real-world entities. The agent-based approach is more akin to reality than other modelling approaches, rendering ABM inherently suited to simulating people and objects in very realistic ways. For example, it is arguably easier to conceptualise, and model how evacuees exit a building during an emergency, than to produce equations that govern the dynamics of evacuee densities. Nonetheless, because equations regarding evacuee density result from the behaviour of evacuees, the agent-based approach will also enable the user to study aggregate properties. In particular, the agent-based approach can be useful when it is more natural to describe the constituent units of a system under some of the following conditions (Bonabeau, 2002):

1) The behaviour of individuals cannot clearly be defined through aggregate transition rates (e.g. panic within a fleeing crowd);
2) Individual behaviour is complex. Although hypothetically any process can be explained by an equation, the complexity of differential equations increases exponentially as the complexity of behaviour increases. Describing complex individual behaviour with equations can therefore become intractable;
3) Activities are arguably a more natural way of describing a system than processes; and,
4) Agent behaviour is stochastic. Points of randomness can be applied strategically within agent-based models, opposed to arbitrarily within aggregate equations.

Finally, the agent-based approach to modelling is flexible, particularly in relation to geospatial modelling. Notably, spatial simulations benefit from the mobility that agent-based models offer. To reiterate, an agent-based model can be defined within any given system environment (e.g. a building, a city, a road network, a computer network, etc). Furthermore, agents have the ability to physically move within their environment, in different directions and at different velocities. Agent mobility makes ABM very flexible in terms of potential variables and parameters that can be specified. Neighbourhoods can also be specified using a variety of mechanisms. The implementation of agent interactions can easily be governed by space, networks, or a combination of structures. This would be far more complex to explain by mathematics, for example (Axtell, 2000). Significantly, agent-based models can regulate behaviours based on interactions at a specific distance and direction. Agent-based models

also provide a robust and flexible framework for tuning the complexity of agents (i.e. their behaviour, degree of rationality, ability to learn and evolve, and rules of interaction). Another dimension of flexibility is the ability to adjust levels of description and aggregation. It is easy to experiment with aggregate agents, sub groups of agents, and single agents, with different levels of description coexisting within a model. Thus, the agent-based approach can be used when the appropriate level of description or complexity is unknown, and finding a suitable level requires exploration.

1.2.2: Limitations of Agent-Based Models

The enthusiasm of adopting the ABM approach for geospatial modelling is curtailed by some limitations. Although common to all modelling techniques, one issue relates to the purpose of the model; a model is only as useful as the purpose for which it was constructed. A model has to be built at the right level of description for every phenomenon, judiciously using the right amount of detail for the model to serve its purpose (Couclelis, 2002). This remains an art more than a science (Axelrod, in press). The nature of the system being modelled is another consideration. For example, a system based on human beings will involve agents with potentially irrational behaviour, subjective choices, and complex psychology. These factors are difficult to quantify, calibrate, and sometimes justify, which complicates the implementation and development of a model, as well as the interpretation of the simulation outputs. However, the fundamental motivation for modelling arises from a lack of full access to data relating to a phenomenon of interest. Often, the target itself is neither well understood nor easy to access. The development of agent-based models offers a means to increase the utility of simulation models, by closely tailoring the model and subsequent analysis to the needs of end users (Parker *et al.*, 2003). In particular, the often visual communication provided by spatially explicit models, especially those coupled with GIS, can be effective at depicting formal model results to a wide range of users (Axtell, 2000). Nevertheless, a model's output must be interpreted appropriately. Varying degrees of accuracy and completeness in the model inputs determine whether the output should be used purely for qualitative insight, or accurate quantitative forecasting. The following section reviews the purpose of different ABM approaches in more detail.

By their very definition, agent-based models consider systems at a disaggregated level. This level of detail involves the description of potentially many agent attributes and behaviours,

and their interaction with an environment. The only way to treat this type of problem in agent computing is through multiple runs, systematically varying initial conditions or parameters in order to assess the robustness of results (Axtell, 2000). There is a practical upper limit to the size of the parameter space that can be checked for robustness, and this process can be extremely computationally intensive, thus time consuming. Although computing power is increasing rapidly, the high computational requirement of ABM remains a limitation when modelling large systems.

Finally, critics of complexity theory point out that the wide variety of surprising behaviour exhibited by mathematical and computational models are rarely found in the real-world. In particular, agent-based models are very sensitive to initial conditions and to small variations in interaction rules (Couclelis, 2002). Consequently, modellers of complex systems are never likely to enjoy the intellectual comfort of laws. Despite this, and the other limitations that have been highlighted, ABM is a useful tool for exploring systems that exhibit complex behaviour. Complexity theory has brought awareness of the subtle, diverse, and interconnected facets common to many phenomena, and continues to contribute many powerful concepts, modelling approaches and techniques. In this vein, the following section explores the potential use of agent-based models.

1.2.3: Purpose of Agent-Based Models

Just as there are different types of models, each with their own characteristic features, advantages, and disadvantages, there are different ways that a model can be used (Casti, 1997). Consideration of these uses allows a clearer understanding of what distinguishes a good or bad model. The following discussion separates the utility of ABM into two broad categories: explanatory or predictive[4].

The explanatory modelling approach strives to explore theory and generate hypotheses. The primary purpose is not to predict the future behaviour of a system, but rather to provide a framework in which past observations can be understood as part of an overall process. Explanatory models generally focus on a specific aspect of a system, placing emphasise on some details about a phenomenon and ignoring others, in the hope that such laboratory

[4] Also referred to as prognostic or descriptive.

explorations will lead to empirically relevant insights. These models purport to be explanatory by stating how reality should, or would be, under ideal circumstances, but they do not attempt to reproduce actual systems (Parker *et al.*, 2003). The potential drawback of this approach is the lack of analytical methods to empirically evaluate ABM results (see Section 1.2.6). Furthermore, although explanatory models can provide considerable insight about theory and thinking, it is difficult to establish whether the final model is informative about specific real-world systems and scenarios.

The purpose of an agent-based model adopting an exploratory approach could be to programme plausible agent behaviours and interactions that, when run as a simulation, produce similar trends and patterns to those observable through the analysis of real-world systems. A model of this nature would produce a 'candidate explanation' for the emergence of observed patterns (see Epstein, 1999 for a detailed overview of candidate explanations). The main challenge for such an application, after ensuring agent behaviours and interactions are plausible, is to develop and test alternative models to identify the range of agent representations that can produce given macro-representations (Brown, 2006).

The predictive modelling approach follows a fundamentally different logic to the explanatory approach. Predictive models are commonly used for extrapolation of trends, evaluation of scenarios, and the prediction of future states. More specifically, changes in initial conditions (e.g. rules governing agent behaviours and interactions, such as information available to agents, constraints upon or incentives for particular agent behaviour or movement, etc), can be used to evaluate the possible effects on the model outcome. Predictive models are designed to mimic real-world systems, and are particularly useful for scenario development and policy decisions.

Juxtaposed with the underpinnings of ABM outlined above, Parker (2003) identifies some of the key benefits of predictive modelling. In particular the author notes that by modelling at a fine scale of granularity agent-based models make very good statistical use of data at a fine resolution. In addition, because agent-based models are not constructed to meet a set of equilibrium criteria, it is possible for the model to simulate discontinuous and non-linear phenomena. Moreover, by accounting for heterogeneity and inter-dependencies, models can reflect important endogenous feedbacks between processes. However, this leads the

discussion to some limitations of predictive models. Any model with positive feedback can create system behaviour referred to as path dependence; where a path to a process can be very sensitive to both initial conditions and small variations in stochastic processes. For this reason, predictive modelling, including ABM, of a system containing positive feedbacks can be very challenging. Also, predictive models can be parameterised with too much real-world data. This can lead to an overly-fitted model (i.e. where the model's calibration is overly constrained to existing data). Predictive models of this nature can be insufficiently general to represent a large range of potential outcomes related to the system under analysis, or to analyse alternative systems.

The choice between adopting an explanatory or predictive approach to modelling is not mutually exclusive. This choice is dependant on the required precision of the model, which in turn, is directly related to the type of information and knowledge that is required. The purpose of a model, including an agent-based model, is not necessarily to faithfully capture all aspects of a system; and this complicates this decision process further. At a fundamental level, an agent-based model can be used solely to enrich understanding of a process that is present within a system through controlled computation experimentation. The aforementioned decision between the models purpose is made harder because agent-based models do not fit easily into the classic deductive / inductive approaches to modelling, familiar to scientists. Scientists use deduction to derive theorems from assumptions, and induction to find patterns in empirical data (Axelrod and Tesfatsion, 2006). For instance, the discovery of equilibrium results in game theory using rational choice axioms is a good example of deduction, whilst induction, in the social sciences, is widely used in the analysis of opinion surveys and macro-economic data (Axelrod, in press). Thence, an agent-based modeller might use a deductive approach to develop a set of assumptions regarding the behaviour and interaction of agents from a body of literature. However, in contrast to classic deduction, the simulated output of agent-based models cannot be used to prove theorems. Thus, a modeller might generate data from several different simulation runs and analyse these results with inductive methods, similar to those employed for analysis of empirical data. However, unlike typical induction, the simulated data comes from a rigorously specified set of rules rather than direct measurement of the real-world. The inability to implement ABM in either a purely deductive or inductive manner, has led Axelrod (in press) to arguably define simulation in general, and ABM in particular, as a third way of doing science.

The following section discusses specific steps required to develop of an agent-based model, focussing on how the distinction in a models final purpose (see above) impacts the design and implementation of agents and their environment.

1.2.4: Developing an Agent-Based Model

The process of building an agent-based model begins with a conceptual model, where basic questions or goals, elements of the system (e.g. agent attributes, rules of agent interaction and behaviour, the model environment, etc), and the measurable outcomes of interest are identified (Brown, 2006). It is important to 'ground' a model during the conceptualisation process (i.e. establish whether simplifications made during the design process do not seriously detract from the credibility and likelihood that the model will provide important insights; Carley, 1996). It is usual for a modeller to set forth a claim as to why the proposed model is reasonable. This claim will be enhanced if the applicability of the model is not over stated, and by defining the models limitations and scope. Grounding can be reinforced by demonstrating that other researchers have made similar or identical assumptions in their models, and by justifying how a proposed model will be of benefit in relation to pre-existing models.

Conceptualising the fundamental aspects of an agent-based model (i.e. one or more agents interacting within an environment), juxtaposed with the distinction between explanatory vs. predictive purposes of a model suggests a fourfold typology of agent and environment types (Table 1). Couclelis (2001) classifies agents and their environment as either being designed (i.e. explanatory) or analysed (i.e. predictive - empirically grounded). If designed, agents are endowed with attributes and behaviours that represent (often simplified) conditions for testing specific hypotheses about general cases. Analysed agents are intended to accurately mimic real-world entities, based on empirical data or ad hoc values that are realistic substitutes for observed processes. Similarly, the environment that agents are situated within can be designed (i.e. provided with characteristics that are simplified to focus on specific agent attributes), or analysed (i.e. represent a real-world location). The boundary between designed and analyzed is not always distinct, especially when ad hoc data are employed. Subtle but profound differences, both practical and conceptual, exist between the design or analysis approach of developing agents and their environment. A major difference in

practical terms is that designing something provides direct (partial or total) control over the outcome, whereas there can only be hope that something has been analyzed correctly (Couclelis, 2001). Table 1 provides further details to consider when developing agents and their environment; including a brief description of the model, the purpose and intent of the model (see Section 1.2.3), verification and validation strategies used to assess the model outputs (see Section 1.2.5 and 1.2.6 respectively), and appropriate software for the development of a model (see Section 1.4.2.3).

		AGENT	
		Designed	**Analysed**
ENVIRONMENT	**Designed**	Model Description - Abstract Purpose / Intent - Discovery of new relationships - Existence proof Verification & Validation Strategy - Theoretical comparison - Replication Appropriate Development Tools - Easy to implement simulation / modelling system	Model Description - Experimental Purpose / Intent - Role-playing games among stakeholders - Laboratory experiments Verification & Validation Strategy - Repetitions - Adequacy of design Appropriate Development Tools - Flexible simulation / modelling systems with well developed user interfaces
	Analysed	Model Description - Historical Purpose / Intent - Explanation Verification & Validation Strategy - Qualitative: goodness of fit Appropriate Development Tools - Advanced simulation / modelling systems linked with GIS	Model Description - Empirical Purpose / Intent - Explanation - Projection - Scenario analysis Verification & Validation Strategy - Quantitative: goodness of fit Appropriate Development Tools - Low-level programming languages

Table 1: Description, purpose / intent, verification & validation strategies, and appropriate development tools for agent-based models incorporating designed or analysed agents / environments (adapted from Berger and Parker, 2001).

Once a model has been conceptualised, it must be formalised into a specification which can be developed into a computer programme; if the model is required to be run as a computer

simulation. The process of formalisation involves being precise about what an identified theory relating to a phenomena of interest means, making sure that it is complete and coherent. There are several reasons why computer simulation is more appropriate for formalising social science theories than mathematics, which has often been used in the social sciences (Gilbert and Troitzsch, 2005). First, programming languages are more expressive and less abstract than most mathematical techniques. Second, a computer simulation can deal more easily with parallel process and processes without well defined order or actions than systems of mathematical equations. Third, a computer model can include heterogeneous agents (e.g. pedestrians with varying degrees of knowledge about a building layout), while this is usually relatively difficult using mathematics. Finally, computer programmes are (or can easily be made to be) modular, so that major changes can be made to one part of the model without requiring large changes in other parts of the programme, an ability which mathematical systems often lack.

The object-oriented paradigm provides a very suitable medium for the development of agent-based models. In particular, it provides the aforementioned modularity useful for developing a computer simulation. It is not the intention of this paper to outline the fundamental object-oriented concepts, this has been achieved by numerous others (refer to Booch (1994) for a seminal discussion, Hathaway (2003) for a non-technical discussion, and Armstrong (2006) for a useful evaluation and clarification of key object-oriented notions).

At the time of writing, there are many simulation / modelling systems available to assist the development stage of ABM. The majority of these simulation / modelling systems are programmed, and / or require the user to develop their model in an object-oriented language. The subsequent section of this paper identifies some of the simulation / modelling systems available for ABM, highlighting key questions that should be considered for a user to determine an appropriate system for their needs.

1.2.4.1: Types of Simulation / Modelling Systems for Agent-Based Modelling

In general, two types of simulation / modelling systems are available to develop agent-based models: toolkits or software[5]. Based on this dichotomy, toolkits are simulation / modelling systems that provide a conceptual framework for organising and designing agent-based models. They provide appropriate libraries[6] of software functionality that include pre-defined routines / functions specifically designed for ABM. However, the object-oriented paradigm allows the integration of additional functionality from libraries not provided by the simulation / modelling toolkit, extending the capabilities of these toolkits. Of particular interest to this paper is the integration of functionality from GIS software libraries (e.g. OpenMap, GeoTools, ESRI's ArcGIS, etc), which provide ABM toolkits with greater data management and spatial analytical capabilities required for geospatial modelling (see Section 1.3).

The development of agent-based models can be greatly facilitated by the utilisation of simulation / modelling toolkits. They provide reliable templates for the design, implementation and visualisation of agent-based models, allowing modellers to focus on research (i.e. building models), rather than building fundamental tools necessary to run a computer simulation (Tobias and Hofmann, 2004; Railsback *et al.*, in press). In particular, the use of toolkits can reduce the burden modellers face programming parts of a simulation that are not content-specific (e.g. a Graphical User Interface, GUI, data import-export, visualisation / display of the model). It also increases the reliability and efficiency of the model, because complex parts have been created and optimised by professional developers, as standardised simulation / modelling functions. Unsurprisingly, there are limitations of using simulation / modelling systems to develop agent-based models, for example: a substantial amount of effort is required to understand how to design and implement a model in some toolkits; the programming code of demonstration models or models produced by other researchers can be difficult to understand or apply to another purpose; a modeller will have to learn or already have an understanding of the programming language required to use the toolkit; and finally the desired / required functionality may not be present, although

[5] An agent-based model could be programmed completely from scratch using a low-level programming language if a modeller has sufficient programming knowledge and experience; see below for disadvantages of this approach.

[6] A collection of programming classes grouped together, termed packages (i.e. classes with similar purpose).

additional tools might be available from the user community or from other software libraries. Benenson *et al.* (2005) also note that toolkit users are accompanied by the fear of discovering that a particular function cannot be used, will conflict, or is incompatible with another part of the model late in the development process.

Probably the earliest and most prominent toolkit was SWARM, although many other toolkits now exist. At the time of writing there are more than one hundred toolkits available for ABM (see AgentLink, 2006; SwarmWiki, 2006; Multiagent Systems, 2006; Tesfatsion, 2006b for comprehensive listings). However, variation between toolkits can be considerable. For example, their purpose (some toolkits have different design objectives e.g. AI rather than social science focus, or network opposed to raster or vector model environments), level of development (e.g. some models are no longer supported or have ceased development), and modelling capabilities (e.g. the number of agents that can be modelled, degree of interaction between agents) can vary. A review of all toolkits currently available is beyond the scope of this paper. However, section 1.2.4.3 identifies a selection of noteworthy simulation / modelling toolkits (e.g. Swarm, MASON, Repast, OBEUS, AnyLogic), highlighting there purpose and capabilities, as well as resources providing further information.

In addition to toolkits, software is available for developing agent-based models, which can simplify the implementation process. For example, simulation / modelling software often negates the need to develop an agent-based model via a low-level a programming language (e.g. Java, C^{++}, Visual Basic, etc). In particular, software for ABM is useful for the rapid development of basic or prototype models. However, modellers using software are restricted to the design framework advocated by the software. For instance, some ABM software will only have limited environments (e.g. raster only) in which to model, or agent neighbourhoods might be restricted in size (e.g. von Neumann or Moore). Furthermore, a modeller will be constrained to the functionality provided by the software (unlike ABM toolkits modellers will be unable to extend or integrate additional tools), especially if the toolkit is written in its own programming language (e.g. NetLogo). Section 1.2.4.3 identifies a selection of noteworthy software for the development of agent-based models; StarLogo, its derivative NetLogo, and AgentSheets.

1.2.4.2: Guidelines for Choosing a Simulation / Modelling System

Ideally, a modeller would have comprehensive practical experience in a range of modelling / simulation systems before choosing which system to use for a modelling endeavour. Unfortunately, this is not usually feasible. For this reason several authors (Najlis et al., 2001; Serenko and Detlor, 2002; Tobias and Hofmann, 2004; Rixon *et al.*, 2005; Roberson, 2005; Dugdale, 2006) have gained practical experience and / or have surveyed several systems, identifying key criteria that should be considered before making a decision. General criteria include, but are not limited to: ease of developing the model / using the system; size of the community using the system; availability of help or support (most probably from the user community); size of the community familiar with the programming language in which the system is implemented (if a programming language is necessary to implement the model); is the system still maintained and / or updated; availability of demonstration or template models; technical and how-to documentation, etc. Criteria relating specifically to a systems modelling functionality include: number of agents that can be modelled; degree of interaction between agents; ability to represent multiple organisational / hierarchical levels of agents; variety of model environments available (network, raster, and vector); possible topological relationship between agents; management of spatial relationships between agents, and agents with their environment; mechanisms for scheduling and sequencing events, etc. These criteria will be weighted differently depending on a modeller's personal preferences and abilities (e.g. the specification of the model to be developed, programming experience / knowledge, etc).

Another important distinction separating simulation / modelling systems is there licensing policy; open source, shareware / freeware, or proprietary. Open source simulation / modelling systems constitute toolkits or software whose source code is published and made available to the public, enabling anyone to copy, modify and redistribute the system without paying royalties or fees. A key advantage of open source simulation / modelling systems relates to the transparency of their inner workings. The user can explore the source code, permitting the modification, extension and correction of the system if necessary. This is particularly useful for verifying a model (see Section 1.2.5). The predominant open source simulation / modelling systems are toolkits (e.g. MASON, Repast, Swarm, etc). The distinction between an open source simulation / modelling system and a shareware / freeware

system is subtle. There is no one accepted definition of the term shareware / freeware, but the expression is commonly used to describe a system that can be redistributed but not modified, primarily because the source code is unavailable. Consequently, shareware / freeware systems (e.g. StarLogo, NetLogo, OBEUS, etc) do not have the same flexibility, extendibility or potential for verification (in relation to access to their source code), as open source systems. Similarly, shareware / freeware systems tend to be toolkits, rather than software. Finally, proprietary simulation / modelling systems are available for developing agent-based models. Proprietary systems are mainly software, developed by an organisation who exercises control over its distribution and use; most require a licence at a financial cost to the user. These systems have the advantage of being professionally designed and built for a specific use, and are often relatively simple to use. However, they often lack the community support found with open source or shareware / freeware systems. Moreover, since access to their source code is prohibited, a model developed with proprietary software is essentially black box. A modeller will therefore, to some extent, be left unsure about the inner validity of a model constructed with a proprietary system. This situation is compounded when the output of a model is emergent or unexpected.

Striking a balance between the aforementioned criteria is difficult. Unfortunately, while identifying a suitable system for the development of an agent-based model, too much time can often be expended trying to find this balance. This balance can be perceived as a trade off between the difficulty of developing a model (e.g. in terms of time required to programme the model, understand how to develop a model with a specific system, or acquiring experience and knowledge of a programming language if required, etc), versus the modelling power provided by the simulation / modelling system (e.g. modelling capabilities and functionality, Figure 1). The key is striking a 'personal' balance between these criteria. For example, those more accustomed to programming may prefer the functionality and flexibility of a simulation / modelling toolkit. However, modellers that only wish to develop a basic or prototype model quickly and easily, possibly with little or no programming skills may prefer to use simulation / modelling software.

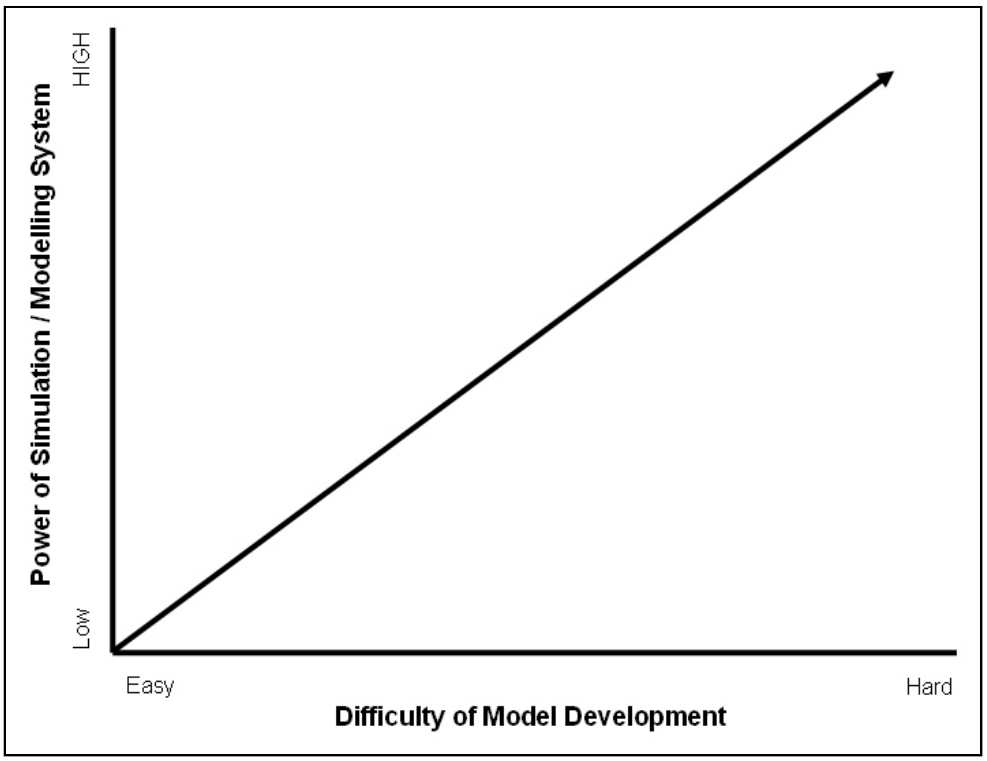


Figure 1: Balance between power versus difficulty of developing a model with a simulation / modelling system.

1.2.4.3: Simulation / Modelling Systems for Agent-Based Modelling

This section provides key criteria pertaining to a selection of simulation / modelling systems available for the development of agent-based models (the rationale for each criterion was described in Section 1.2.4.2). Although there are many systems available for developing agent-based models, this paper reviews eight, separated into three categories of licensing policy (see Section 1.2.4.2 for a definition of each type): 1) open source (Swarm, MASON and Repast); 2) shareware / freeware (StarLogo, NetLogo and OBEUS); and 3) proprietary systems (AgentSheets and AnyLogic). These systems were chosen because they fulfilled the (majority of the) following criteria, they are: maintained and still being developed; widely used and supported by a strong user community; accompanied by a variety of demonstration models and in some instances the model's programming script or source code is available; and finally they are capable of developing spatially explicit models, possibly via the integration of GIS functionality. Tables 2, 3 and 4 tabularise information of each system for comparison purposes; categorised by their licensing policy (adapted from Najlis *et al.*, 2001 and Parker, 2001). The reminder of this section provides further information about each system, identifying examples of geospatial models that have been developed with the system. A caveat must be noted at this point, the information provided within this section is accurate

at the time of publication. However, the systems reviewed are constantly being updated, thus modellers are advised to check each systems website to obtain up to date information.

Open Source Simulation / Modelling Systems			
	SWARM	**MASON**	**Repast**
Developers	Santa Fe Institute / SWARM Development Group, USA	Center for Social Complexity, George Mason University, USA	University of Chicago, Department of Social Science Research Computing, USA
Date of Inception	1996	2003	2000
Website	http://www.swarm.org	http://cs.gmu.edu/~eclab/projects/mason	http://repast.sourceforge.net
E-mail List	http://www.swarm.org/mailman/listinfo	https://listserv.gmu.edu/archives/mason-interest-l	https://lists.sourceforge.net/lists/listinfo/repast-interest
Implementation Language	Objective-C / Java	Java	Java / Python / Microsoft.Net
Operating System	Windows, UNIX, Linux, Mac OSX	Windows, UNIX, Linux, Mac OSX	Windows, UNIX, Linux, Mac OSX
Required programming experience	Strong	Strong	Strong
Integrated GIS functionality	Yes (e.g. Kenge GIS library for Raster data: http://www.gis.usu.edu/swarm)	None	Yes (e.g. OpenMap, Java Topology Suite, and GeoTools). Repast simulations can also be run within ArcGIS through an extension called Agent Analyst.
Integrated charting / graphing / statistics	Yes (e.g. R and S-plus statistical packages)	None	Yes (e.g. Colt statistical package, and basic Repast functionality for simple network statistics)
Availability of demonstration models	Yes	Yes	Yes
Source code of demonstration models	Yes	Yes	Yes
Tutorials / How-to Documentation	Yes	Yes	Yes
Additional information	Minar et al. (1996)	Luke et al. (2004)	Agent Analyst Extension (http://www.institute.redlands.edu/agentanalyst) Useful weblog: http://www.gisagents.blogspot.com

Table 2: Comparison of open source simulation / modelling systems (adapted from Najlis *et al.*, 2001 and Parker, 2001).

Swarm (Table 2) is an open source simulation / modelling system designed specifically for the development of multi-agent simulations of complex adaptive systems (Swarm, 2006);

although agent-based models can easily be develop using Swarm as well. Inspired by artificial life, Swarm was designed to study biological systems; attempting to infer mechanisms observable in biological phenomena (Minar *et al.*, 1996). In addition to modelling biological systems, Swarm has been used to develop models for anthropological, computer science, ecological, economic, geographical, and political science purposes. Useful examples of spatially explicit models include: the simulation of pedestrians in the urban centres (Schelhorn *et al.*, 1999 and Haklay *et al.*, 2001); and the examination of crowd congestion at London's Notting Hill carnival (Batty *et al.*, 2003). Najlis *et al.* (2001) identify the steep learning curve of Swarm as a significant factor to consider before choosing this system to develop an agent-based model; although this should be less of a problem for a modeller with strong programming skills.

MASON (Multi Agent Simulation Of Neighbourhood - Table 2) was developed by the Evolutionary Computation Laboratory (ECLab) and the Centre for Social Complexity at George Mason University. At present MASON does not provide functionality for dynamically charting (e.g. histograms, line graphs, pie charts, etc) model output during a simulation, or allow GIS data to be imported / exported (Luke *et al.*, 2004). However, the developers of MASON are continuing to develop further functionality, and they hope users will develop and contribute tools themselves (e.g. GIS integration). Unfortunately there is little technical documentation and a relatively small user group in comparison to some of the other systems identified within this paper. However, how-to documentation, demonstration models (e.g. the seminal heat bugs example, network models, etc), and several publications detailing the implementation and / or application of MASON are available for a prospective modeller to evaluate the system further (MASON, 2006).

Originally developed at the University of Chicago, the Recursive Porous Agent Simulation Toolkit (Repast - Table 2) is currently maintained by Argonne National Laboratory and managed by the Repast Organisation for Architecture and Development (ROAD). Repast caters for the implementation of models in three programming languages: Python (RepastPy); Java (RepastJ); and Microsoft.Net (Repast.Net). RepastPy allows basic models to be developed by modellers with limited programming experience via a 'point-and-click' GUI (Collier and North, 2005). RepastPy models can subsequently be exported / converted into Java for further development in RepastJ. Repast.Net and RepastJ allow for more advanced

models to be developed (Vos, 2005), because more complex functionality can be programmed into a model. Agent Analyst is an ABM extension that allows users to create, edit, and run Repast models from within ArcGIS (Redlands Institute, 2006). Repast has a relatively large user group and an actively supported e-mail list, as well as extensive how-to documentation and demonstration models available from the system website. Useful examples of spatially explicit models created using Repast include the studying of segregation, and residential and firm location (Crooks, 2006) and the evacuation of pedestrians from within an underground station (Castle, 2006).

Whilst still being maintained RepastJ, Repast.Net and RepastPy have now reached maturity and are no longer being developed. They have been superseded by Repast Simphony (RepastS) which provides all the core functionality of RepastJ or Repast.Net, although limited to implementation in Java. The Repast development team have provided a series of articles regarding RepastS. The architecture and core functionality are introduced by North *et al.* (2005a), and the development environment is discussed by Howe *et al.* (2006). The storage, display and behaviour / interaction of agents, as well as features for data analysis (i.e. via the integration of the R statistics package) and presentation of models within Repast S are outlined by North *et al.* (2005b). Tatara *et al.* (2006) provide a detailed discussion outlining how-to develop a "simple wolf-sheep predation" model; illustrating RepastS modelling capabilities.

StarLogo (Table 3) is an shareware / freeware modelling system developed at the Media Laboratory, Massachusetts Institute of Technology (MIT). Unlike the other six agent-based simulation / modelling systems discussed in this section, both StarLogo and NetLogo models are programmed procedurally, opposed to an object-oriented nature. Thus, models developed with StarLogo do not benefit from the similarity in abstraction shared between the agent-based and object-oriented paradigms. Furthermore, StarLogo lacks the same flexibility offered by open source systems, since modellers are constrained to functionality provided by the system. However, the StarLogo website indicates that an open source version of the system (OpenStarLogo) has been developed and will be available in the near future. Despite these limitations, StarLogo is very easy to use, notably for people with very little programming experience. Dynamic charting functionality of model output during a simulation is provided. In addition, a number of demonstration models and detailed how-to

documentation relating to these models is supplied with StarLogo, and many more are available to download from the World Wide Web (WWW). Batty et al. (1998) have used StarLogo to examine visitor movement within London's British Tate Gallery, specifically how changes in room configuration can affect movement between exhibits.

Shareware / Freeware Simulation / Modelling Systems			
System Name	**StarLogo**	**NetLogo**	**OBEUS**
Developers	Media Laboratory, Massachusetts Institute of Technology, USA	Centre for Connected Learning and Computer-Based Modelling, Northwestern University, USA	Environmental Simulation Laboratory, Tel Aviv University, Israel
Date of Inception	Early 1990s, Java based version 2000	1999	Early 2000s
Website	http://education.mit.edu/starlogo	http://ccl.northwestern.edu/netlogo	http://eslab.tau.ac.il/Research/default.aspx
E-mail List	http://education.mit.edu/pipermail/starlogo-users	None	None
Implementation Language	Proprietary scripting	Proprietary scripting	Microsoft.Net
Operating System	Windows, UNIX, Linux, Mac OSX	Windows, UNIX, Linux, Mac OSX	Windows
Required programming experience	Basic	Basic	Moderate-Strong
Integrated GIS functionality	None	None	Yes
Integrated charting / graphing / statistics	Yes	Yes	Unknown
Availability of demonstration models	Yes	Yes	Unknown
Source code of demonstration models	Yes	Yes	Unknown
Tutorials / How-to Documentation	Yes	Yes	Yes
Additional information	OpenStarLogo website: http://education.mit.edu/openstarlogo/	http://groups.yahoo.com/group/netlogo-users	User manual: http://eslab.tau.ac.il/OBEUS/OBEUSManual.pdf Geosimulation website: http://www.geosimulation.org

Table 3: Comparison of shareware / freeware simulation / modelling systems (adapted from Najlis *et al.*, 2001 and Parker, 2001).

NetLogo (originally named StarLogoT - Table 3) is a variant of StarLogo, originally developed at the Centre for Connected Learning and Computer-Based Modelling at

Northwestern University, to allow StarLogo models to be developed on computers using the Macintosh operating system. It is now possible to create StarLogo models on a computer using a Macintosh operating system, thus the critically distinction between the two simulation / modelling systems is that NetLogo is specifically designed for the deployment of models over the internet (NetLogo, 2006). Both NetLogo (Figure 2) and StarLogo provide functionality to import image files, which can be used to define the environment agents are located within, thus facilitating the development of spatial models. NetLogo has been used to develop applications in disciplines varying from biology and physics to the social sciences. Extensive how-to documentation / tutorials and demonstration models are available from the system website, and functionality can be extended through Application Programming Interfaces (APIs), although the source code for the system is currently unavailable.

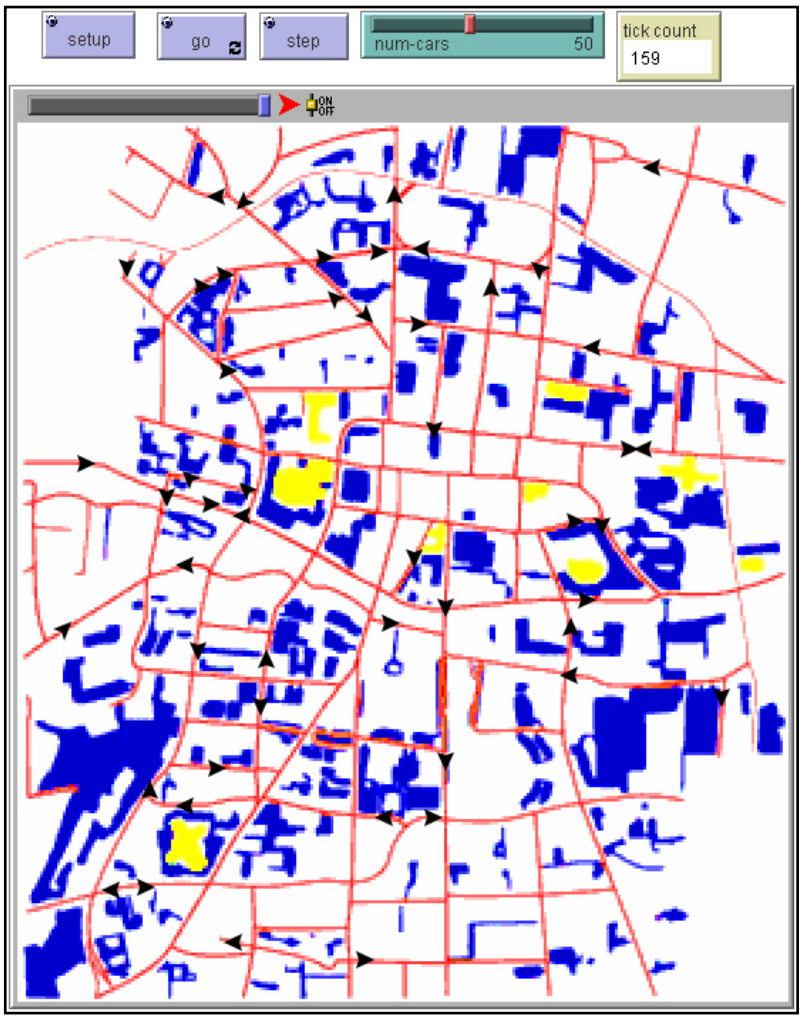


Figure 2: Example of GIS integration in NetLogo, the Cruising Model (NetLogo, 2006).

OBEUS (Object-Based Environment for Urban Simulation - Table 3) was developed at Tel Aviv University, Israel. Based on the theory of Geographic Automata Systems (GAS; Benenson and Torrens, 2004), it is designed for the simulation of urban phenomena; specifically in a geospatial context. GAS considers an urban system as consisting of objects that are either fixed (e.g. houses or roads) or non-fixed (e.g. people or cars; Benenson *et al.*, 2005). OBEUS is implemented in the Microsoft.NET framework, but relies on several third-party components (Microsoft.NET Framework, Borland C# compiler, etc), which must be installed in order to operate the system. OBEUS provides a GUI to develop the structure of a model, although the behaviour and interaction rules of agents must be programmed using one of the Micorsoft.NET languages (e.g. C#, C++, or Visual Basic, etc). Consequently, moderate to strong programming skills are required. OBEUS has been used to develop a number of spatially explicit models, including: the simulation of ethnic residential distributions within the Yaffo area of Tel Aviv between 1955-1995 (Benenson *et al.*, 2002, Benenson *et al.*, 2005), and street parking and urban sprawl (Benenson *et al.*, 2004).

Proprietary Simulation / Modelling Systems		
	AgentSheets	**AnyLogic**
Developers	AgentSheets Inc., USA	XJ Technologies, Russia
Date of Inception	1991	Unknown
Website	http://www.agentsheets.com	http://www.xjtek.com
E-mail List	None	None
Implementation Language	Proprietary scripting	Proprietary scripting
Operating System	Windows, UNIX, Linux, Mac OSX	Windows, UNIX, Linux, Mac OSX
Required programming experience	None - Basic	Moderate
Integrated GIS functionality	None	None
Integrated charting / graphing / statistics	None	Yes
Availability of demonstration models	Yes	Yes
Source code of demonstration models	N /A	N /A
Tutorials / How-to Documentation	Yes	Yes
Additional information	Carvalho, 2000and Repenning *et al.*, 2000	http://www.xjtek.com/support/forums/general

Table 4: Comparison of proprietary simulation / modelling systems (adapted from Najlis *et al.*, 2001 and Parker, 2001).

AgentSheets (Table 4) is a proprietary simulation / modelling system that allows modellers with limited programming experience to develop an agent-based model, because models are developed through a GUI (Repenning *et al.*, 2000). A number of demonstration models are available from the system website. For example, Sustainopolis is a simulation analogous to the computer game SimCity; exploring pollution dispersal within a city (Figure 3). Carvalho (2000) has used AgentSheets extensively to teach undergraduate students, the author comments that it is easy to use the system to develop models quickly, providing students with hands-on experience of ABM without the need to learn a programming language. However, the author notes models created with AgentSheets are limited in their sophistication (e.g. the complexity of agent behaviour and interaction). Furthermore, the system lacks functionality to dynamically chart simulation output, and agents are limited to movement within a two-dimensional cell-based environment.

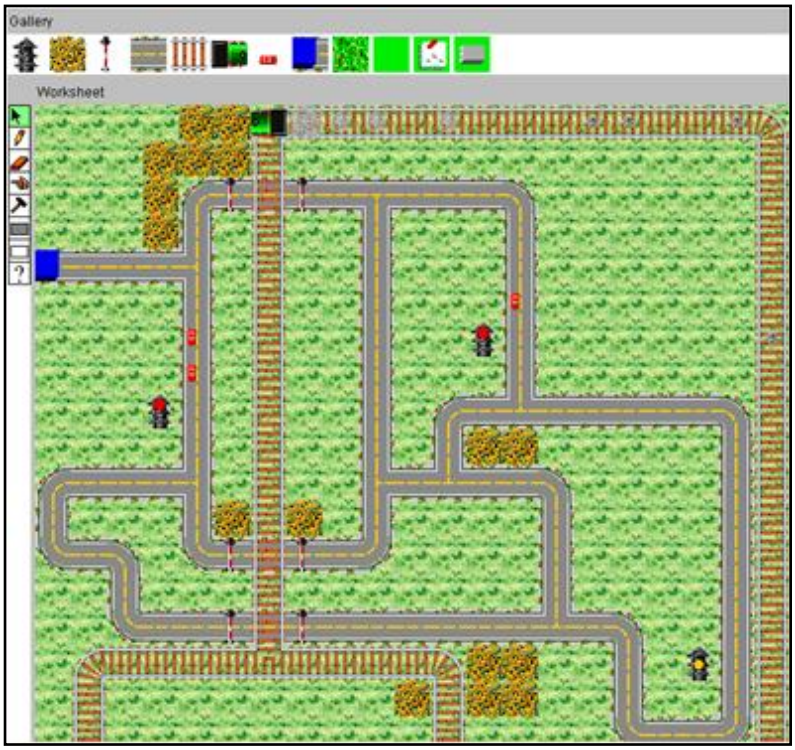


Figure 3: The Sustainopolis model developed in AgentSheets (2006).

AnyLogic (Table 4) incorporates a range of functionality for the development of agent-based models. For example, models can dynamically read and write data to spreadsheets or databases during a simulation run, as well as dynamically chart model output. Furthermore, external programmes can be initiated from within an AnyLogic model for dynamic

communication of information, and vice versa. However, AnyLogic models can only be created on Microsoft operating systems, although a simulation can be run on any Java-enabled operating system once compiled (e.g. a Macintosh operating system). The AnyLogic website notes that models have been developed for a diverse range of applications including: the study of social, urban (Figure 4) and ecosystem dynamics (e.g. a predator-prey system); planning of healthcare schemes (e.g. the impact of safe syringe usage on HIV diffusion); computer and telecommunication networks (e.g. the placement of cellular phone base stations); and the location of emergency services and call centres. However, the source code of these examples and / or documentation of these models are unavailable.

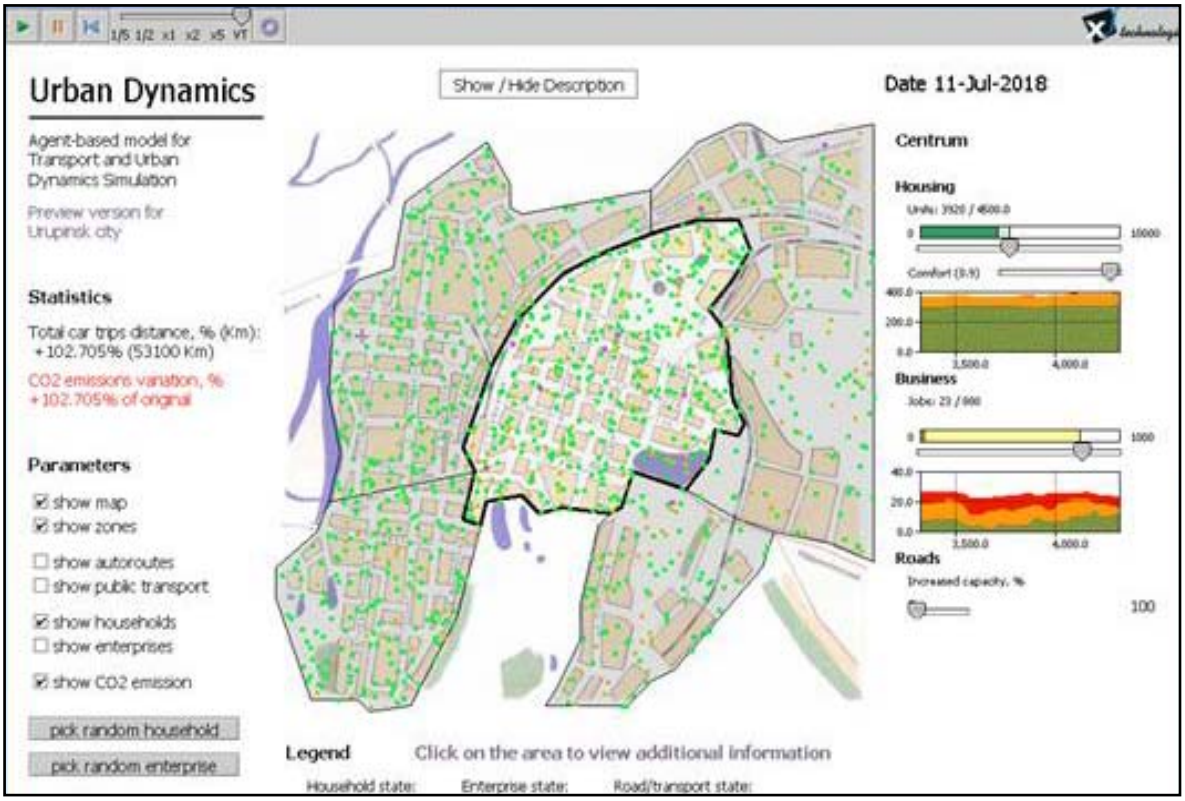


Figure 4: An urban and transport dynamics model developed in AnyLogic (2006).

Each system discussed within this section can be positioned within the continuum illustrated in Figure 1 (power versus difficulty of developing a model with a simulation / modelling system). However, the exact location of each system is very subjective (i.e. dependant upon a modeller's knowledge and experience of ABM in general, and each simulation / modelling system in particular). The information presented within this section has provided the reader with a selection of useful criteria to assess the eight simulation / modelling systems

presented, allowing each system to be (approximately) located within this continuum based on the readers own knowledge and experience. Once a candidate system(s) has been identified the reader will need to investigate the potential suitable of the system(s) further.

1.2.5: Verification and Calibration of Agent-Based Models

Once developed, the computer programme must be verified by checking the model behaves as expected; often referred to as 'inner validity' (Brown, 2006; Axelrod, in press). Whether the model itself is an accurate representation of the real-world is a different type of validity (see below). Achieving inner validity is harder than it might seem. For instance, hypotheses about the model's output can be tested under a range of parameter input settings. Perhaps the model can be examined under an extreme situation where the outcome is easily predictable. These hypotheses should yield expected results from the model because they are based on the conceptual model design. However, it is difficult for a programmer to know whether unexpected outcomes are a reflection of a mistake in the computer programme (a 'bug'), logical errors of the model, or a surprising consequence of the model itself (Gilbert and Terna, 1999). This predicament is conflated because complex systems can often produce emergent and counterintuitive results. A modeller can guard against the former of these problems by adopting 'unit tests' while programming their model. Unit testing involves the execution of the computer programme after each modification of the code to check that a bug has not been introduced (Gilbert and Troitzsch, 2005). Nevertheless, a modeller must still determine if unexpected results are due to an error in the model logic, or just a feature of the system being modelled. These difficulties of verification are compounded by the fact that most simulations are dependent on random number generators[7] to simulate the effects of unmeasured variables and random choices. Thus, repeated runs can be expected to produce different outcomes. Fortunately, one of the main advantages of ABM is that they provide a natural method for describing and simulating a real-world system, which helps simplify the model logic (Batty, 2001). However, it is not uncommon to spend more time confirming that a model has been programmed correctly than programming the model itself.

The most thorough way of verifying a model is to re-implement the model using a different programming language and ideally a different ABM toolkit; a process sometimes referred to

[7] Computers are likely to generate pseudo-random numbers, not actually random numbers. This facet is important when simulations are sensitive to subtle differences in the random numbers used.

as ‘docking’ or ‘alignment’ (Axtell *et al.*, 1996). Although this method will never attain the status of a proof, it helps the modeller become more confident as to the veracity of the model results (Hales *et al.*, 2003). Docking is not always practical (i.e. in relation to time) or feasible (i.e. for a modeller to learn a new programming language or how to develop their model in an alternative system). However, replication is one of the hallmarks of cumulative science, and is an important facet needed to confirm whether claimed results of a simulation are reliable (i.e. can be reproduced by somebody else starting from scratch). Axelrod (in press) notes that without this confirmation, it is possible that published results could be due to programming errors, misrepresentation of the system being modelled, or errors analysing the simulation results. Furthermore, replication is required to determine whether a new model can subsume a previous model. If it is not practical or feasible for a modeller to dock their model, it is critical that a thorough description of the model is provided for another to attempt replication. Unfortunately, the later stipulation has rarely been accomplished to date; a noteworthy exception includes Railsback (in press). Documentation should include information regarding the source code for running the model, how to run the programme, and how to understand the output. Fortunately, the use of standardised ABM toolkits and programming languages facilitates replication. Unfortunately, it is often difficult for a modeller to provide a complete description of their model within the word limit of publications (except digital media), especially when addressing an interdisciplinary audience. Carley (1996) stresses that computer models and their output should be described and presented as separate publications.

After a modelling endeavour has been verified the final stages are to calibrate and validate the model. Calibration entails setting the model structure and parameter values in ways that accurately reflect a real-world system. Calibration typically requires data on the micro-level processes that the agent-based model is based upon. These data can be acquired through various means, such as surveys, statistical analysis of empirical data, experiments designed to obtain decision-making strategies and factors, etc. Calibration occurs in stages, usually repeated iteratively, until the outcomes of the model fit (within a reasonable tolerance) the real-world data collected. Therefore, calibration is useful for assessing the feasibility of the model to simulate the real-world system (i.e. showing that the model can generate results that match the real-world data). If the model output cannot be fitted to the real-world data, it may be necessary for the modeller to re-programme aspects of the model (e.g. rules dictating agent

behaviour and interaction, etc). Thus, calibration is also helpful in the verification process of the model. It should not be forgotten that the level of correspondence between the model and the real-world data is partly dependant on the purpose of the model. In which case, the modeller must have confidence in the accuracy of the real-world data (e.g. ensuring that the data does not represent extreme situations). The impulse to calibrate a model can lead to the model being overly fitted. In this situation, a model fits the real-world data but is insufficiently general to represent a diverse range of system outcomes, or be applied to alternative systems.

It is often argued that a model with sufficient parameters can always be tweaked until the real-world data is matched (Carley, 1996). To this extent, modellers should be wary that calibration does not guarantee the validity of a model. However, for many models this criticism is less appropriate. In particular, within agent-based models that represent processes with rules (e.g. the interaction and behaviour of agents) rather than parameterised equations, there are often few if any parameters. Conversely, there is no guarantee that a model with a large number of rules dictating the interaction and behaviour of agents can be configured to generate the observed data.

1.2.6: Validation and Analysis of Agent-Based Model Outputs

A model is valid to the extent that it adequately represents the system being modelled (Casti, 1997). However, the validity of a model should not be thought of as binary (i.e. a model cannot simply be classified as valid or invalid); a model has a certain degree of validity (Law and Kelton, 1991). Validity can be ascertained by comparing the output of the model with comparable data collected from a real-world system. For example, to understand the output of an agent-based model it is often necessary to evaluate the details of a specific simulation 'history'. There are at least three ways in which history can be described (Axelrod, in press):

1) History can be reported as a selection of key events in chronological order. For instance, the simulation of a train station evacuation could be described at the point in time when the emergency alarm sounds, when strategic confines of the station have been evacuated (e.g. platform, escalator / stairs, ticket hall, etc), until the station is fully vacated. Whilst informative, this method provides little explanatory power about the model itself.

2) Alternatively, the history of one agent can be documented. For example, the location of a pedestrian with the station when the emergency alarm sounds, the time taken for the pedestrian to reach strategic locations within the station thereafter (e.g. platform, escalator / stairs, ticket hall, building exit), and a summary of the route traversed by the agent. This is often the easiest type of history to understand, and can be very revealing about the way the model works (i.e. how the logic of the model effects agents over time).
3) Finally, the history from a global viewpoint can be noted. For example, the distribution of pedestrians throughout the station, to potentially assess the use of different emergency exits. Although the global viewpoint is often regarded as the best method for observing large-scale patterns, several detailed histories are often required to explain the reason for these observed patterns.

Although the analysis of individual histories is interesting, they can be misleading; especially if the model incorporates random elements. For example, simulations often use a random number generator to imitate the decision making process of an agent (e.g. direction choices, mood preferences, etc), to randomise the order in which agents move, or to substitute an unmeasured parameter (equivalent to the modeller making a guess in the absence of more accurate information), etc. In order to determine whether the conclusion from a simulation run is typical, it is necessary to undertake a repeated number of simulations using identical parameters and initial conditions, but using different random number seeds[8]. This will help distinguish whether particular patterns observed in a single illustrative history are idiosyncratic or typical. Results from these simulation runs will need to be presented as distributions, or as means with confidence intervals. Statistical analysis will be required to assess any variation in the model output, and to determine whether inferences from the simulation histories are well founded. Gilbert and Troitzsch (2005), and Axelrod (in press) state that regression will be required for analysing quantitative changes in the output of a simulation, and analysis of variance will be required to assess the output of a simulation if the differences are qualitative[9] (e.g. if patterns or clusters have formed). As with any statistical analysis, it is necessary to determine if observed differences are statistically significant (i.e.

[8] A random number generator can be 'seeded' with an initial value. This seed can be used by the modeller to recreate an identical sequence of numbers.

[9] Observable rather than directly quantifiable.

unlikely to have occurred by chance), and if differences are substantive (i.e. the magnitude is large enough to be important).

It is usually desirable to engage in sensitivity analysis once a model (at least for a specific set of initial conditions and parameter values) appears to be valid. The aim of sensitivity analysis is to determine the extent variation in the model's assumptions yield differences in the model output. The principle behind sensitivity analysis is to vary the initial conditions and parameters of the model by a small amount and observe differences in the model outcomes. For example, a model might be run several times, varying a given parameter between 10% above and below the original value. If the impact on the output is negligible, it can be assumed the parameter is not of critical importance to the model, and its accuracy is not of major concern. However, a note of caution should be observed since complex systems can exhibit large and sudden shifts in system behaviour in response to relatively small perturbations in inputs (Manson, in press).

Sensitivity analysis is also used to investigate the robustness of the model. If the behaviour of the model is very sensitive to small differences in the value of one or more parameters, the modeller might be concerned whether these particular values are correct (Gilbert and Troitzsch, 2005). Unfortunately, even with a small number of variables, the required number of parameter combinations can become very large, and the resources required to perform a thorough analysis can become excessive. In practice, a modeller is likely to have a good intuition about which parameters are likely to have the largest impact on the model, and will therefore be more important to examine. The effect of different model versions can also be assessed by running controlled experiments with sets of simulation runs, akin to the evaluation of parameter changes. The difference in the logic of a model (e.g. changes in rules governing agent behaviour and / or interaction, etc) can be studied by systematically comparing different versions of the model. However, it is imperative initial conditions are kept identical for any comparison to be valid.

There are a few caveats that must be considered while validating and analysing the output of a model (Gilbert and Troitzsch, 2005). Firstly, both the model and the system under analysis are likely to be stochastic. Thus, comparison between the model output and data from the real-world system are unlikely to correspond on every occasion. Whether the significance of

this difference is enough to cast doubt in the model depends partly on the expected statistical distribution of the simulation output. Unfortunately, these distributions are rarely known a priori and are difficult to estimate with simulations, especially if outcomes are emergent. Another problem relates to the capability of the model to make predictions, since they will almost certainly be conditional (i.e. it is unlikely that all postulated outcomes can be produced). For instance, a model may be able to produce plausible future predictions, but may not be able to recreate known past system states. Furthermore, there is a possibility that the model is correct, but the data from the real-world system may not (i.e. inappropriate assumptions or estimates could have been obtained from the data). Finally, many simulations are path dependant (i.e. the outcome of a simulation is dependant on the exact initial setup chosen). Different runs of the same model can generate variation in outputs due to changes in initial conditions, parameters, or the stochastic behaviour / interaction of agents. Thus, the 'history' of a simulation is highly significant.

Calibration and validation are arguably the hardest two issues of ABM. Even though there may be correspondence between a model's output and a real-world system, this is not sufficient condition to conclude that the model is correct (Gilbert, 2004). There are many different processes which could yield a given outcome, and just because a model generates similar outcomes does not prove that the processes included within the model account for the real-world outcome. However, a model should be regarded as a basis for reducing uncertainty about the future, from a prior state of unawareness, to one of more limited uncertainty. A model should not be considered a failure if its predictions are not perfectly accurate or if a modeller is left unsure whether processes included within the model account for the real-world outcome. It takes time and many refinements of a model, but a modeller can gradually increase their confidence in a model by testing it against real-world data in more and more ways. Bearing this in mind, a large number of agent-based models have been developed to study various phenomena.

1.2.7: Applications of Agent-Based Models

It is impractical to comprehensively review the full range of ABM applications within this paper, and even examination of a representative sample presents a challenging exercise. Agent-based models have been developed for a diverse range of subject areas, such as: archaeological reconstruction of ancient civilisations (Axtell *et al.*, 2002; Gumerman *et al.*,

2003); understanding theories of political identity and stability (Lustick, 2002); understanding processes involving national identity and state formation (Cederman, 2001); biological models of infectious diseases (Eidelson and Lustick, 2004; Chen *et al.*, 2006); growth of bacterial colonies (Kreft *et al.*, 1998; Krzysztof *et al.*, 2005); single- (Emonet *et al.*, 2005) and multi-cellular level interaction and behaviour (Athale and Deisboeck, 2006); alliance formation of nations during the Second World War (Axelrod and Bennett, 1993); modelling economic processes as dynamic systems of interacting agents (Agent-based Computational Economics, ACE, Tesfatsion, 2006a); company size and growth rate distributions (Axtell, 1999); size-frequency distributions for traffic jams (Nagel and Rasmussen, 1994); price variations within stock-market trading (Bak *et al.*, 1999); voting behaviours in elections (Kollman *et al.*, 1992); identifying and exploring behaviour in battlefields (Ilachinski, 1997); spatial patterns of unemployment (Topa, 2001); trade networks (Epstein and Axtell, 1996); business coalitions over industry standards (Axelrod, 2006); social networks of terrorist groups (North *et al.*, 2004), to name but a few. These examples can be constructed as lying on a continuum, from minimalist academic models based upon ideal assumptions, to large scale commercial decision support systems based upon real-world data. In relation to the focus of this paper, the remainder of this section concentrates on the origin of ABM applied to social phenomena, particularly in a geographical context.

Despite the advantages of ABM as a tool for simulation (see Section 1.2.1), ABM has not been widely adopted in geospatial research; although there is no obvious reason why this is the case. Thomas Schelling is credited with developing the first social agent-based model in which agents represent people, and agent interactions represent a socially relevant process. Schelling's (1971) model demonstrated that stark segregation patterns can emerge from migratory movements among two culturally distinct, but relatively tolerant, types of households. Yet, ABM did not begin to feature prominently in the geographical literature until the mid-1990s, when Epstein and Axtell (1996) extended the notion of modelling people to growing entire artificial cities. The goal was to understand the emergence of patterns, trends, or other characteristics observable in society or geography. Epstein and Axtell's Sugarscape model demonstrated that agents could emerge with a variety of characteristics and behaviours suggestive of a rudimentary society (e.g. death, disease, trade, health, culture, conflict, war, etc).

On a technical note, until recently the majority of ABM research has still involved the population of regular lattices, similar to CA, with agents that could migrate between cells. Alternatively, CA were just re-interpreted as agent-based models by attributing anthropomorphic state variables to cells (Torrens and Benenson, 2005). Moreover, the movement of agents has been dependent solely on the attributes of their immediate neighbouring cells and their inhabitants, and their environments have not been based on real-world geographic features. Gimblett *et al.* (2002) were amongst the first to use real-world geographic features. Their agent-based model was developed to evaluate the recreational use of Broken Arrow Canyon, Arizona. Specifically, current hiking, bike, and off-road trail paths were mapped in a GIS and potential alternatives simulated in order to aid management decisions of environmental protection and enhance recreational user experiences of the canyon. More recently, Dibble and Feldman (2004) have developed a three-dimensional extension to the Repast ABM toolkit. The extension has enabled the authors to model the control of infectious disease transmission, dynamics of civil violence, and coordination of social networks within three-dimensional landscape terrains, and social and spatial networks.

The examples identified within this section demonstrate the vast array of subjects and disciplines that researchers have investigated through ABM. In particular, the use of ABM for experimenting and exploring with geographical phenomena is still in its infancy. Consequently, the future is set to be a very exciting for agent-based modellers. Especially since the use of ABM to investigate a phenomenon is limited only by the researcher's imagination.

1.3: Modelling within GIS: Current Capabilities

It can be difficult to comprehend how GIS technology, built essentially for handling maps and "map-related ideas", can be adapted to the needs of dynamic simulation modelling; especially when it is not even perceived as an optimal platform for modelling (Goodchild, 2005). Particular criticisms of GIS with respect to modelling is their ability to handle time (Langran, 1992; Peuquet, 2005 – see Section 1.3.1), the representation of continuous variation (Longley *et al.*, 2005), and most have only rudimentary modelling capabilities (Maguire, 2005). Nevertheless, there are several good reasons to justify why the use, or linkage of GIS with simulation / modelling systems (see Section 1.3.2), is an effective means of modelling when spatial and temporal analysis is necessary.

Current commercial and public domain GIS software systems all contain numerous tools for acquiring, pre-processing, and transforming data. Their use in modelling includes data management, format conversion, projection change, re-sampling, raster-vector conversion, etc. GIS also include excellent tools for visualisation / mapping, rendering, querying, and analysing model results, as well as assessing the accuracies and uncertainties associated with inputs and outputs.

Typically, all of the capabilities described above are accessible via end-user graphical and command line interfaces. However, these capabilities have recently become accessible through APIs, via software libraries. The exposure of APIs was a significant recent improvement in terms of GIS and spatial modelling, as external programmers now have access to the underlying software components upon which GIS software vendors base their end-user versions of systems. This is perhaps the most pertinent enhancement, as many of the techniques used in GIS analysis are potentially far more robust if they can be linked with an extensive toolkit of methods for simulation; an issue which is addressed at greater length later in Section 1.3.2. GIS vendors have invited this situation as it allows GIS to be extended and customised for use in new application areas, thus expanding the market potential of their systems.

Alternatively, a model can be expressed as a sequence of GIS commands executed by a script (Maguire, 2005). Recently in GIS there has been a move to use industry-standard low-level programming languages (e.g. Java, C^{++}, and Visual Basic), and scripting languages (e.g. Python, VBScript, and Jscript), rather than proprietary, home grown scripting languages (e.g. ESRI's Arc Macro Language, AML, or Avenue). Interoperability standards such as the Microsoft.Net framework facilitate this process by allowing compliant packages to be called from the same script.

In addition to scripts, graphical flowcharts can be used to express sequences of operations that define a model. Longley *et al.* (2005) note that one of the first graphic platforms for conceptualising and implementing spatial models was probably the ERDAS IMAGINE software, which allows the user to build complex modelling sequences from primitive operations. ESRI is another GIS vendor that provides an environment that allows models to

be authored and executed in a graphical environment: ModelBuilder within ArcGIS 9.x, which superseded Spatial Modeller within ArcView 3.x.

In principle, graphic-model building can be used for dynamic modelling via an iterative process, where the output of one time step becomes the input for the next. However, this method posses two dilemmas: 1) the GIS will not have been designed for an iterative process, requiring the user to re-enter the data at the beginning of each time step, and; 2) the time required to run a model could be considerable. The former of these problems can be overcome with scripting languages; both can potentially be overcome by integrating the GIS with a simulation / modelling system better equipped for the task at hand. Before exploring the possibilities of linking GIS and simulation / modelling systems (Section 1.3.2), the following section of this paper evaluates the capability of GIS to handle space-time information, which computer simulations generate in volume, and has always been a limitation.

1.3.1: Representing Time and Change within GIS

The subject of time within GIS has received a considerable amount of attention. Heywood (2006) comments that ideally, GIS would be able to represent temporal change using methods that explicitly represent spatial change, as well as different states through time. Furthermore, methods allowing direct manipulation and comparison of simulated or observational data in a temporal and spatial dimensions should be catered for. In reality, two main challenges for the integration of time within GIS exist: 1) continuous data over a period of time are rarely available for an entity or system of interests; 2) data models and structures able to record, store, and visualise information about an object in different temporal states are still in their infancy (Heywood *et al.*, 2006). In the context of this paper, the former challenge is less of a constraint since an agent-based computer simulation is capable of generating an abundance of data over a continuous period of time, while much progress has been made on the later issue. The following discussion outlines issues related to the representation of time and change, as well as approaches for incorporating space-time information within GIS.

The basic objective of any temporal database is to record change over time, where change can be thought of as an event or collection of events. An event might be a change in state of one or more locations, entities, or both. Changes that might affect an event can be distinguished

in terms of their temporal pattern; Peuquet (2005) has suggested four types: 1) continuous - events occurring throughout some period of time; 2) majorative - events occurring most of the time; 3) sporadic - events occurring some of the time, and; 4) unique - events that only occur once. The distribution of events within these temporal patterns can also be very complex (e.g. chaotic, cyclic, or steady state), complicated further as change, to some extent, is always occurring at various rates as well (e.g. from sudden to gradual). Hence, duration and frequency are important descriptive characteristics within this taxonomy of temporal patterns.

There are three approaches for capturing space-time information within a GIS: 1) location-based; 2) time-based, and; 3) entity-based. The only method of viewing a data model within existing GIS, as a space-time representation, is as a temporal series of spatially-registered 'snapshots' (Peuquet, 2005). Invariably this approach employs a raster data model, although vector has also been used, with only a single information type stored (e.g. elevation, density, precipitation, etc) for each cell at any one point in time. Information for the entire layer is stored for each time step, regardless of whether change has occurred since the previous step. There are several criticisms of this approach. Firstly, the data volume increases enormously, because redundant data is stored in consecutive snapshots. The state of a spatial entity can only be retrieved by querying cells of adjacent snapshots, because information is stored implicitly between each time step. Finally, the exact point when change has occurred cannot be determined. Langran (1992) has proposed a modification of this approach. The temporal-raster (or grid) approach allows multiple values to be stored for each pixel. A new value, and the time at which change occurred for each pixel is stored, which can result in a variable number of records for each cell. Recording the time at which change has occurred allows for values to be sorted by time. The most recent value for each cell can therefore be retrieved, which represents the present state of the system. The obvious advantage to this approach is the reduction of redundant data stored for each cell.

Peuquet and Duan (1995) have proposed a time-based approach to storing space-time information within a GIS, where change is stored as a sequence of events through time. Time is stored in increasing order from an initial point, with the temporal interval correlating to successive events. An event is recorded at the time when the amount of accumulated change is considered significant, or by another domain-specific rule. This type of representation has

the advantage of facilitating time-based queries, and the addition of a new event is straight forward as it can simply be added to the end of the timeline. Furthermore, in terms of modelling an important capacity of any model is the ability to represent alternative versions of the same reality. The concept of representing multiple realities over time is called branching. Branching allows various model simulation runs to be compared, or simulation results to be compared to observed data. The time-based approach facilitates the branching of time in order to represent alternative or parallel sequences of events resulting from specific scenarios, because it is strictly an ordinal timeline.

Finally, several entity-based space-time models have been proposed. Conceptually these models extend the topological vector approach (e.g. coverage model); tracking changes in the geometry of entities incrementally through time. The amendment vector model was the first of this type, and extended frameworks have been proposed subsequently. Besides maintaining the integrity of entities and their changing topology, these approaches are able to represent asynchronous changes to entity geometries. However, the space-time topology of these vectors becomes increasingly complex as amendments accumulate through time. In addition, aspatial entity attributes can change over time. To record aspatial changes, a separate relational database is often used. However, if change occurs at a different rate between the spatial and aspatial aspects of an entity, maintaining the identity of individual entities becomes difficult, especially when entities split or merge.

Object-oriented data models have transformed the entity-based storage of space-time information within GIS (Zeiler, 1999), and have become mainstream within commercial GIS (e.g. the geodatabase structure with ArcGIS). They have grown increasingly more sophisticated, catering for a powerful modelling environment. The object-oriented data model approach provides a cohesive representation that allows the identity of objects, as well as complex interrelationships to be maintained through time. Specifically, temporal and location behaviour can be assigned as an attribute of features rather than the space itself, which has the distinct advantage of allowing objects to be updated asynchronously. Despite the advantages of the object-oriented data model, Reitsma and Albrecht (2006) observe that, to date, no data model or data structure allows the representation of processes (i.e. recording a process that has changed the state of an object within a model). Consequently, queries about where a process is occurring at an instant of time cannot be expressed with these current

approaches. Notwithstanding, object-oriented data models are the canonical approach to the storage of space-time data generated by agent-based models, and their visualisation within GIS, given their complementarities. Nevertheless, the visualisation of agent-based models within GIS is still limited to a temporal series of snapshots.

1.3.2: Linkage - Coupling versus Integration / Embedding

Models implemented as direct extensions of an underlying GIS, through either graphic model-building or scripts, generally make two assumptions: 1) all operations required by the model are available in the GIS (or in another system called by the model); and, 2) the GIS provides sufficient performance to handle the execution of the model (Longley *et al.*, 2005). In reality, a GIS will often fail to provide adequate performance, especially with very large datasets and a large number of iterations, because it has not been designed as a simulation / modelling engine. This one-size-fits-all approach inherent in GIS provides limited applicability, and attention has subsequently been devoted to linking, either through coupling or integration / embedding, GIS with simulation / modelling systems more directly suited to users needs. General classifications have been produced by numerous authors (e.g. Maguire, 1995; Bernard and Krüger, 2000; Westervelt, 2002; Goodchild, 2005; Longley *et al.*, 2005; Maguire, 2005). Several of their definitions now overlap as technological advance has blurred the boundaries of their classifications, whist some definitions are convoluted because terminology has been used interchangeably or sometimes inappropriately (e.g. coupling, linkage or integration). Nevertheless, categorisation of these techniques is possible, and a brief description of each is developed below, in an attempt to clarify the situation. This is followed by a critique of these different approaches, with a view to identifying an appropriate method for developing geospatial agent-based models.

In situations where GIS and simulation / modelling systems already exist (e.g. as commercial products), or the cost of rebuilding the functionality of one system into another is too great, the systems can be coupled together (Maguire, 2005). Coupling can therefore be broadly defined as the linkage of two stand-alone systems by data transfer. Three types of coupling are distinguishable, although these are only a subset of the much larger fields of enterprise application integration (Linthicum, 2000) and software interoperability (Sondheim *et al.*, 2005). The attributes of each approach cascaded along the coupling continuum, from loose to tight / close (Table 5 summaries the competing objectives of the different coupling

approaches; greyed boxes are considered more desirable characteristics - adapted from Westervelt, 2002):

1) **Loose Coupling**. A loose connection usually involves the asynchronous operation of functions within each system, with data exchanged between systems in the form of files. For example, the GIS might be used to prepare inputs, which are then passed to the simulation / modelling system, where after execution the results of the model are returned to the GIS for display and analysis. This approach requires the GIS and simulation / modelling system to understand the same data format; if no common format is available an additional piece of software will be required to convert formats in both directions. Occasionally, specific new programmes must be developed to perform format modifications;

2) **Moderate Coupling**. Essentially this category encapsulates techniques between loose and tight / close coupling. For example, Westervelt (2002) advocates remote procedure calls and shared database access links between the GIS and simulation / modelling system, allowing indirect communication between the systems. Inevitably, this reduces the execution speed of the integrated system, and decreases the ability to simultaneously execute components belonging to the different software; and,

3) **Tight or Close Coupling**. This type of linkage is characterised by the simultaneous operation of systems allowing direct inter-system communication during the programme execution. For example, standards such as Microsoft's COM and .NET allow a single script to invoke commands from both systems (Ungerer and Goodchild, 2002). A variant of this approach allows inter-system communication by different processes that may be run on one of more networked computers (i.e. distributed processing).

Coupling has often been the preferred approach for linking GIS and simulation / modelling systems. However, this has tended to result in very specialised and isolated solutions, which have prevented the standardisation of general and generic linkage. An alternative to coupling is to embed or to integrate the required functionality of either the GIS or simulation / modelling system within the dominant system using its underlying programming language (Maguire, 2005). The final system is either referred to as GIS-centric or modelling-centric depending on which system is dominant. In both instances, the GIS tools or modelling

capabilities can be executed by calling functions from the dominant system, usually through a GUI. Compared to coupling, an embedded or integrated system will appear seamless to a user (Maguire, 1995). However, in the past integration has been based on existing closed and monolithic GIS and simulation systems, which poses a risk of designing systems that are also closed, monolithic, and therefore costly (Fedra, 1996).

Objective and Explanation	Loose	Moderate	Close / Tight
Integration Speed: The programmer time involved in linking the programmes.	Fast	Medium	Slow
Programmer Expertise: Required level of software development expertise.	Low	High	Medium
Multiple Authorship Avoidance: In some instances it might be necessary for the programmer to modify the original software product. Any alteration reduces the ownership responsibility. Major alterations could totally sever this link, resulting in limited or no support by the original author(s).	High	Medium	Low
Execution Speed: How rapidly does the integrated software execute?	Slow	Medium	Fast
Simultaneous Execution: Can components of the system run simultaneously and communicate with one another? Can the components operate on separate platforms?	Low	Low	High
Debugging: How difficult is it to locate execution errors in the linked system?	Easy	Moderate	Hard

Table 5: Comparison of coupling approaches (adapted from Westervelt, 2002).

Interest in modelling-centric systems has increased considerably over recent years, predominately due to the development of simulation / modelling toolkits with scripting capabilities that do not require advanced computer programming skills (Gilbert and Bankes, 2002). Often the simulation / modelling toolkit can access GIS functions, such as data management and visualisation capabilities, from a GIS software library. For example, the Repast toolkit exploits functions from GeoTools (a Java GIS software library) for importing and exporting data, Java Topology Suite (JTS) for data manipulation, and OpenMap for visualisation. The toolkit itself maintains the agents and environment (i.e. their attributes), using identity relationships for communication between the different systems. Functions available from GIS software libraries reduce the development time of a model, and are likely to be more efficient because they have been developed over many years with attention to efficiency. Additionally, the use of standard GIS tools for spatial analysis improves functional transparency of a model, as it makes use of well known and understood algorithms. Alternatively, spatial data management and analysis functions can be developed

within the modelling toolkit, although this strategy imposes huge costs, in terms of time to programme the model, and time required to frequently update spatial data or use spatial analysis functions within the model.

Conversely, the GIS-centric approach is an attractive alternative; not least because the large user-base of some GIS expands the potential user-base for the final model. Analogous to the modelling-centric approach, GIS-centric integration can be carried out using software libraries of simulation / modelling functions accessed through the GIS interface. There are many examples of simulation / modelling systems integrated within commercial GIS, including: the Consequences Assessment Tool Set (CATS) system, designed for emergency response planning; the Hazard Prediction and Assessment Capability (HPAC) system, for predicting the effect of hazardous material releases into the atmosphere; the NatureServe Vista system, for land use and conservation planners.

Brown *et al.* (2005) propose an alternative approach which straddles both the GIS-centric and modelling-centric frameworks. Rather than providing functionality within one system, the middleware-based approach manages connections between systems, allowing a model to make use of the functionality available within the GIS or the simulation / modelling toolkit most appropriate for a given task. Thus, the middleware approach allows the simulation / modelling toolkit to handle the identity and relationship of, and between agents and their environment. Conversely, the GIS would manage spatial features, as well as temporal and topological relationships of the model. Essentially, the simulation / modelling toolkit handles what it is designed for (i.e. implementing the model), while the GIS can be used to run the model, and visualise the output. An example of this approach is the ABM extension within ArcGIS (referred to as Agent Analysts), which allows users to create, edit, and run RepastPy models from within ArcGIS (Redlands Institute, 2006). However, it is the opinion of the authors that only a dichotomy of integration classifications exists. A GIS is either integrated into a simulation / modelling toolkit, or vice versa. The definition of the middleware approach is essentially tight coupling (see above). The review within this section suggests that agent-based modellers interested in developing a geospatial model involving many (possibly tens of thousands) interacting agents with complex behaviours and interactions between themselves, and their environment should consider either GIS-centric or modelling-centric integration.

1.4: Conclusion

This paper identified advantages of developing agent-based models to simulate systems (for certain circumstances) at the individual-level. In particular, agent-based models are useful for capturing emergent phenomena, they provided a natural environment for the study of systems composed of real-world entities, and are flexible, particularly in relation to the development of geospatial models. In light of these benefits, fundamental concepts and principles of the ABM paradigm, principally in relation to geospatial modelling, were defined. A comprehensive introduction into the development and evaluation of agent-based models and their output followed. GIS were recognized as particularly useful media for representing model input / output of a geospatial nature. However, despite current advance it was noted there are still only rudimentary modelling capabilities available within GIS. The discussion highlighted that, depending on the modelling endeavour, the linkage of a simulation / modelling system with a GIS could be useful. In particular, users interested in developing a geospatial model consisting of many (possibly tens of thousands) agents, with potentially complicated behaviour and interactions between themselves and their environment might consider either GIS-centric or modelling-centric integration. These linkage approaches offer a number of mutually reinforcing advantages. The simulation / modelling toolkit will benefit from the data acquisition, pre-processing, transformation, and visualisation tools provided by the GIS. While the GIS will benefit from the management of identities and relationships of, and between agents and their environment supported by the simulation / modelling toolkit. Thus, guidelines for choosing, and key criteria pertaining to a selection of simulation / modelling systems where provided to facilitate the reader's identification of a suitable system for the development of a geospatial agent-based model.

Acknowledgments

We gratefully acknowledge the financial support of the Economic and Social Research Council (ESRC, http://www.esrc.ac.uk) for our Collaborative Award in Science and Engineering (CASE) studentship's (PTA-033-2004-00034 and PTA-033-2003-00008), Camden Primary Care Trust (PCT, http://www.camdenpct.nhs.uk), and the Greater London Authority (GLA) Economics Unit (http://www.london.gov.uk/mayor/economic_unit). A special note of gratitude is also extended to our supervisors Professor Mike Batty, Professor Paul Longley, and Doctor Muki Haklay.

Bibliography

AgentLink (2006), Software, Available at: http://eprints.agentlink.org/view/type/software.html [Accessed on August 24th, 2006].

AgentSheets (2006), AgentSheets: The Sustainopolis model, Available at: http://www.agentsheets.com/Applets/sustainopolis/ [Accessed on August 24th, 2006].

AnyLogic (2006), AnyLogic: Urban Dynamics Agent Based Model, Available at: http://www.xjtek.com/models/applet.xml?archive=social_dynamics/urban_dynamics_agent_based/model.jar,business_graphics_library.jar,xjanylogic5engine.jar&root=urban_dynamics_agent_based.Model$Simulation&width=1010&height=765 [Accessed on August 24th, 2006].

Armstrong, D.J. (2006), 'The Quarks of Object-Oriented Development', Communication of the ACM, 49(2): 123-128.

Athale, C.A. and Deisboeck, T.S. (2006), 'The Effects of EGF-Receptor Density on Multiscale Tumour Growth Patterns', Journal of Theoretical Biology, 238: 771-779.

Axelrod, R. (2006), 'Agent-Based Modelling as a Bridge Between Disciplines', in Tesfatsion, L. and Judd, K.L. (eds.), Handbook of Computational Economics: Agent-Based Computational Economics, North-Holland Publishing, Amsterdam, The Netherlands.

Axelrod, R. (in press), 'Advancing the Art of Simulation in the Social Sciences', in Rennard, J.-P. (ed.) Handbook of Research on Nature Inspired Computing for Economy and Management, Idea Group, Hersey, USA.

Axelrod, R. and Bennett, S.D. (1993), 'A Landscape Theory of Aggregation', British Journal of Political Science, 23(2): 211-233.

Axelrod, R. and Tesfatsion, L. (2006), 'A Guide for Newcomers to Agent-Based Modelling in the Social Sciences', in Tesfatsion, L. and Judd, K.L. (eds.), Handbook of Computational Economics: Agent-Based Computational Economics, North-Holland Publishing, Amsterdam, The Netherlands.

Axtell, R., Axelrod, R., Epstein, J.M. and Cohen, D. (1996), 'Aligning Simulation Models: A Case Study and Results', Computational and Mathematical Organization Theory, 1: 123-141.

Axtell, R.L. (1999), The Emergence of Firms in a Population of Agents: Local Increasing Returns, Unstable Nash Equilibria, and Power Law Size Distributions, Center on Social and Economic Dynamics: Working Paper No. 3, Washington, USA.

Axtell, R.L. (2000), Why Agents? On the Varied Motivations for Agent Computing in the Social Sciences, Center on Social and Economic Dynamics (The Brookings Institute): Working Paper 17, Washington, D.C.

Axtell, R.L., Epstein, J.M., Dean, J.S., Gumerman, G.J., Swedlund, A.C., Harburger, J., Chakravarty, S., Hammond, R., Parker, J. and Parker, M. (2002), 'Population Growth and

Collapse in a Multiagent Model of the Kayenta Anasazi in Long House Valley', Proceedings of the National Academy of Sciences of the United States of America (PNAS), 99(3): 7275-7279.

Bailey, T.C. and Gatrell, A.C. (1995), Interactive Spatial Data Analysis (3nd Edition), Longman, UK.

Bak, P., Paczuski, M. and Shubik, M. (1999), Price Variations in a Stock Market with Many Agents, Cowles Foundation: Discussion Paper 1132, Available at: http://cowles.econ.yale.edu/P/cd/d11a/d1132.pdf.

Batty, M. (2001), 'Agent-Based Pedestrian Modelling', Environment and Planning B, 28(3): 321-326.

Batty, M., Conroy, R., Hillier, B., Jiang, B., Desyllas, J., Mottram, C., Penn, A., Smith, A. and Turner, A. (1998), The Virtual Tate, Centre for Advanced Spatial Analysis (University College London): Working Paper 5, London, Available at: http://www.casa.ucl.ac.uk/tate.pdf.

Batty, M., Desyllas, J. and Duxbury, E. (2003), 'Safety in Numbers? Modelling Crowds and Designing Control for the Notting Hill Carnival', Urban Studies, 40(8): 1573-1590.

Benenson, I., Aronovich, S. and Noam, S. (2005), 'Let's talk objects: generic methodology for urban high-resolution simulation', Computers, Environment and Urban Systems, 29: 425–453.

Benenson, I., Birfur, S. and Kharbash, V. (2004), 'OBEUS (Object-Based Environment for Urban Simulation) Shareware Version', Association Geographic Information Laboratories Europe (AGILE), Crete.

Benenson, I., Omer, I. and Hatna, E. (2002), 'Entity-Based Modelling of Urban Residential Dynamics: The Case of Yaffo, Tel Aviv', Environment and Planning B: Planning and Design, 29: 491-512.

Benenson, I. and Torrens, P. (2004), Geosimulation: Automata-Based Modelling of Urban Phenomena, John Wiley & Sons, London.

Berger, T. and Parker, D.C. (2001), 'Examples of Specific Research', in Parker, D.C., Berger, T. and Manson, S.M. (eds.), Meeting the Challenge of Complexity: Proceedings of a Special Workshop on Land-Use/Land-Cover Change, Irvine, California.

Bernard, L. and Krüger, T. (2000), 'Integration of GIS and Spatio-Temporal Simulation Models: Interoperable Components for Different Simulation Strategies', Transactions in GIS, 4(3): 197-215.

Bonabeau, E. (2002), 'Agent-Based Modelling: Methods and Techniques for Simulating Human Systems', Proceedings of the National Academy of Sciences of the United States of America (PNAS), 99(3): 7280-7287.

Booch, G. (1994), Object-Oriented Analysis and Design with Applications, Benjamin/Cummings, Redwood City, USA.

Brown, D.G. (2006), 'Agent-Based Models', in Geist, H. (ed.) The Earth's Changing Land: An Encyclopedia of Land-Use and Land-Cover Change, Greenwood Publishing Group, Westport, pp. 7-13.

Brown, D.G., Riolo, R., Robinson, D.T., North, M. and Rand, W. (2005), 'Spatial Process and Data Models: Toward Integration of Agent-Based Models and GIS', Journal of Geographical Systems, 7(1): 25-47.

Carley, K.M. (1996), Validating Computational Models, Carnegie Mellon University: Working Paper, Pittsburgh, USA.

Carvalho, J. (2000), 'Using AgentSheets to Teach Simulation to Undergraduate Students', Journal of Artificial Societies and Social Simulation, 3(3).

Casti, J.L. (1997), Would-Be-Worlds: How Simulation is Changing the Frontiers of Science, John Wiley & Sons, New York, USA.

Castle, C.E. (2006), 'Using Repast to Develop a Prototype Agent-Based Pedestrian Evacuation Model', Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects, Chicago, USA.

Cederman, L.-E. (2001), 'Agent-Based Modelling in Political Science', The Political Methodologist, 10(1): 16-22.

Chen, L.-C., Carley, K.M., Fridsma, D., Kaminsky, B. and Yahja, A. (2006), 'Model Alignment of Anthrax Attack Simulations', Decision Support Systems, 41: 654-668.

Collier, N. and North, M.J. (2005), 'Repast for Python Scripting', Annual Conference of the North American Association for Computational Social and Organizational Science (NAACSOS), Notre Dame, Indiana, USA.

Conte, R., Gilbert, N. and Sichman, J.S. (1998), 'MAS and Social Simulation: A Suitable Commitment', in Sichman, J.S., Conte, R. and Gilbert, N. (eds.), Multi-Agent Systems and Agent-Based Simulation, Springer, New York.

Couclelis, H. (2001), 'Why I No Longer Work with Agents: A Challenge for ABMs of Human-Environment Interactions', in Parker, D.C., Berger, T. and Manson, S.M. (eds.), Meeting the Challenge of Complexity: Proceedings of a Special Workshop on Land-Use/Land-Cover Change, Irvine, California.

Couclelis, H. (2002), 'Modelling Frameworks, Paradigms, and Approaches', in Clarke, K.C., Parks, B.E. and Crane, M.P. (eds.), Geographic Information Systems and Environmental Modelling, Prentice Hall, London.

Crooks, A.T. (2006), 'Exploring Cities using Agent Based Models and GIS', Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects, Chicago, USA.

Dibble, C. and Feldman, P.G. (2004), 'The GeoGraph 3D Computational Laboratory: Network and Terrain Landscapes for Repast', Journal of Artificial Societies and Social Simulation, 7(1).

Dugdale, J. (2006), An Evaluation of Seven Software Simulation Tools for Use in the Social Sciences, Available at: http://www.irit.fr/COSI/training/evaluationoftools/Evaluation-Of-Simulation-Tools.htm [Accessed on August 22nd, 2006].

Eidelson, B.M. and Lustick, I. (2004), 'VIR-POX: An Agent-Based Analysis of Smallpox Preparedness and Response Policy', Journal of Artificial Societies and Social Simulation, 7(3).

Emonet, T., Macal, C.M., North, M.J., Wickersham, C.E. and Cluzel, P. (2005), 'AgentCell: A Digital Single-Cell Assay for Bacterial Chemotaxis', Bioinformatics, 21(11): 2714-2721.

Epstein, J.M. (1999), 'Agent-Based Computational Models and Generative Social Science', Complexity, 4(5): 41-60.

Epstein, J.M. and Axtell, R. (1996), Growing Artificial Societies: Social Science from the Bottom Up, MIT Press, Cambridge, USA.

Fedra, K. (1996), 'Distributed Models and Embedded GIS: Integration Strategies and Case Studies', in Goodchild, M.F., Steyaert, L.T. and Parks, B.O. (eds.), GIS and Environmental Modelling: Progress and Research Issues, GIS World Books, Fort Collins, pp. 413-8.

Franklin, S. and Graesser, A. (1996), 'Is it an Agent, or just a Program? A Taxonomy for Autonomous Agents', Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag.

Galea, E.R. and Gwynne, S. (2006), Principles and Practice of Evacuation Modelling (7th Edition), Fire Safety Engineering Group - University of Greenwich, London.

Gilbert, N. (2004), Agent-Based Social Simulation: Dealing with Complexity, University of Surrey, Guildford, UK, Available at: http://www.complexityscience.org/NoE/ABSS-dealing%20with%20complexity-1-1.pdf.

Gilbert, N. and Bankes, S. (2002), 'Platforms and Methods for Agent-Based Modeling ', Proceeding of the National Academy of Sciences of the USA, 99(3): 7197-7198.

Gilbert, N. and Terna, P. (1999), 'How to Build and Use Agent-Base models in Social Science', Mind & Society, 1: 57-72.

Gilbert, N. and Troitzsch, K.G. (2005), Simulation for the Social Scientist, Open University Press, UK.

Gimblett, H.R., Richards, M.T. and Itami, R.M. (2002), 'Simulating Wildland Recreation Use and Conflicting Spatial Interactions using Rule-Driven Intelligent Agents', in Gimblett, H.R. (ed.) Integrating Geographic Information Systems and Agent-Based Modeling Techniques for Simulating Social and Ecological Processes, Oxford University Press, Oxford, pp. 211-243.

Goodchild, M.F. (2005), 'GIS, Spatial Analysis, and Modelling Overview', in Maguire, D.J., Batty, M. and Goodchild M, F. (eds.), GIS, Spatial Analysis and Modelling, ESRI Press, Redlands, California.

Goodchild, M.F. and Proctor, J. (1997), 'Scale in a Digital Geographic World', Geographical and Environmental Modelling, 1: 5-23.

Gumerman, G.J., Swedlund, A.C., Dean, J.S. and Epstein, J.M. (2003), 'The Evolution of Social Behavior in the Prehistoric American Southwest', Artificial Life, 9(4): 435-444.

Haklay, M., O'Sullivan, D., Thurstain-Goodwin, M. and Schelhorn, T. (2001), '"So Go Downtown": Simulating Pedestrian Movement in Town Centres', Environment and Planning B: Planning and Design, 28(3): 343-359.

Hales, D., Rouchier, J. and Edmounds, B. (2003), 'Model-to-Model Analysis', Journal of Artificial Societies and Social Simulation, 6(4).

Hathaway, R.J. (2003), Basics of Object-Orientation, Available at: http://www.objectfaq.com/oofaq2/index.html [Accessed on August 20th, 2006].

Heywood, I., Cornelius, S. and Carver, S. (2006), An Introduction to Geographical Information Systems (3rd Edition), Pearson Education, UK.

Holland, J.H. (1995), Hidden Order: How Adaptation Builds Complexity, Addison-Wesley, Reading.

Howe, T.R., Collier, N.T., North, M.J., Parker, M.T. and Vos, J.R. (2006), 'Containing Agents: Contexts, Projections, and Agents', Proceedings of Agent 2006 Conference on Social Agents: Results and Prospects, Chicago, USA.

Ilachinski, A. (1997), Irreducible Semi-Autonomous Adaptive Combat (ISAAC): An Artificial-Life Approach to Land Combat, Center for Naval Analyses, Virginia, USA.

Kollman, K., Miller, J.H. and Page, S.E. (1992), 'Adaptive Parties in Spatial Elections', The American Political Science Review, 86(4): 929-937.

Kreft, J.-U., Booth, G. and Wimpenny, W.T. (1998), 'BacSim, a Simulator for Individual Based Modelling of Bacterial Colony Growth', Microbiology, 144: 3275-3287.

Krzysztof, K., Dzwinel, W. and Yuen, D.A. (2005), 'Nonlinear Development of Bacterial Colony Modelled with Cellular Automata and Agent Objects', International Journal of Modern Physics C, 14(10): 1385-1404.

Langran, G. (1992), Time in Geographic Information Systems, Taylor and Francis, London.

Law, A.M. and Kelton, D. (1991), Simulation Modelling and Analysis (2nd Edition), McGraw-Hill, New York, USA.

Linthicum, D.S. (2000), Enterprise Application Integration, Addison-Wesley, Boston.

Longley, P.A. (2004), 'Geographical Information Systems: On Modelling and Representation', Progress in Human Geography, 28(1): 108-116.

Longley, P.A. and Batty, M. (2003), 'Prologue: Advanced Spatial Analysis: Extending GIS', in Longley, P.A. and Batty, M. (eds.), Advanced Spatial Analysis: The CASA Book of GIS, ERSI Press, Redlands, California.

Longley, P.A., Goodchild, M.F., Maguire, D.J. and Rhind, D.W. (2005), Geographical Information Systems and Science, John Wiley and Sons, USA.

Luke, S., Cioffi-Revilla, C., Panait, L. and Sullivan, K. (2004), 'MASON: A New Multi-Agent Simulation Toolkit', SwarmFest 2004, Eighth Annual Swarm Users / Researchers Conference, University of Michigan, Ann Arbor, Michigan USA.

Lustick, I. (2002), 'PS-I: A User-Friendly Agent-Based Modelling Platform for Testing Theories of Political Identity and Political Stability', Journal of Artificial Societies and Social Simulation, 5(3).

Macal, C.M. and North, M.J. (2005), 'Tutorial on Agent-Based Modelling and Simulation', in Euhl, M.E., Steiger, N.M., Armstrong, F.B. and Joines, J.A. (eds.), Proceedings of the 2005 Winter Simulation Conference.

Maguire, D.J. (1995), 'Implementing Spatial Analysis and GIS Applications for Business and Service Planning', in Longley, P.A. and Batty, M. (eds.), GIS for Business and Service Planning, GeoInformation International, Cambridge, UK.

Maguire, D.J. (2005), 'Towards a GIS Platform for Spatial Analysis and Modelling', in Maguire, D.J., Batty, M. and Goodchild M, F. (eds.), GIS, Spatial Analysis and Modelling, ESRI Press, Redlands, California, USA.

Manson, S.M. (2001), 'Simplifying Complexity: A Review of Complexity Theory', Geoforum, 32(3): 405-414.

Manson, S.M. (2003), 'Epistemological Possibilities and Imperatives of Complexity Research: A Reply to Reitsma', Geoforum, 34(1): 17-20.

Manson, S.M. (in press), 'Challenges to Evaluating Models of Geographic Complexity', Environment and Planning B: Planning and Design.

Martin, D.J. (1996), Geographic Information Systems: Socio Economic Applications, Routledge, London.

MASON (2006), Multi Agent Simulation Of Neighbourhood, Available at: http://cs.gmu.edu/~eclab/projects/mason [Accessed on October 3rd, 2006].

Minar, N., Burkhart, R., Langton, C. and Askenazi, M. (1996), The Swarm Simulation System: A Toolkit for Building Multi-agent Simulations, Available at: http://www.santafe.edu/projects/swarm/overview/overview.html [Accessed on August 8th, 2006].

Multiagent Systems (2006), Tools, Available at: http://www.multiagent.com [Accessed on August 24th, 2006].

Najlis, R., Janssen, M.A. and Parker, D.C. (2001), 'Software Tools and Communication Issues', in Parker, D.C., Berger, T. and Manson, S.M. (eds.), Meeting the Challenge of Complexity: Proceedings of a Special Workshop on Land-Use/Land-Cover Change, Irvine, California.

NetLogo (2006), NetLogo: Multi-Agent Programmable Modeling Environment, Available at: http://ccl.northwestern.edu/netlogo/ [Accessed on August 8th, 2006].

North, M., Macal, C.M. and Vos, J.R. (2004), 'Terrorist Organization Modelling', North American Association for Computational Social and Organizational Science Conference 2004, Pittsburgh, USA.

North, M.J., Howe, T.R., Collier, N.T. and Vos, J.R. (2005a), 'The Repast Simphony Development Environment', Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms, Chicago, USA.

North, M.J., Howe, T.R., Collier, N.T. and Vos, J.R. (2005b), 'The Repast Simphony Runtime System', Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms, Chicago, USA.

O'Sullivan, D. (2004), 'Complexity Science and Human Geography', Transactions of the Institute of British Geographers, 29(3): 282-295.

Openshaw, S. (1984), The Modifiable Areal Unit Problem (Concepts and Techniques in Modern Geography: 38), Geo-Books, Norwich, England.

Parker, D.C. (2001), Object-Oriented Packages for Agent-Based Modelling, Available at: http://mason.gmu.edu/~dparker3/spat_abm/lectures/lecture2_tables.pdf [Accessed on October 27th, 2005].

Parker, D.C., Manson, S.M., Janssen, M.A., Hoffmann, M.J. and Deadman, P. (2003), 'Multi-Agent Systems for the Simulation of Land-Use and Land-Cover Change: A Review', Annals of the Association of American Geographers, 93(2): 314-337.

Peuquet, D.J. (2005), 'Time in GIS and Geographical Databases', in Longley, P.A., Goodchild, M.F., Maguire, D.J. and Rhind, D.W. (eds.), Geographical Information Systems: Principles, Techniques, Management and Applications (Abridged Edition), John Wiley and Sons, USA.

Peuquet, D.J. and Duan, N. (1995), 'An Event-Based Spatio-Temporal Data Model (ESTDM) for Temporal Analysis of Geographic Data', International Journal of Geographical Information Systems, 9(1): 2-24.

Railsback, S.F., Lytinen, S.L. and Jackson, S.K. (in press), 'Agent-Based Simulation Platforms: Review and Development Recommendations', Simulation.

Redlands Institute (2006), What is Agent Analyst?, Available at: http://www.institute.redlands.edu/agentanalyst/AgentAnalyst.html [Accessed on May 31st, 2006].

Reitsma, F. (2003), 'A Response to Simplifying Complexity', Geoforum, 34(1): 13-16.

Reitsma, F. and Albrecht, J. (2006), 'A Process-Oriented Data Model', in Drummond, J., Billen, R., Forrest, D. and João, E. (eds.), Dynamic & Mobile GIS: Investigating Change in Space and Time, Taylor & Francis, London.

Repenning, A., Ioannidou, A. and Zola, J. (2000), 'AgentSheets: End-User Programmable Simulations', Journal of Artificial Societies and Social Simulation, 3(3).

Rixon, A., Moglia, M. and Burn, S. (2005), 'Bottom-Up Approaches to Building Agent-Based Models: Discussing the Need for a Platform', Proceedings of the Joint Conference on Multi-Agent Modelling for Environmental Management, Bourg-St-Maurice, France.

Russell, S. and Norvig, P. (2003), Artificial Intelligence: A Modern Approach, Prentice Hall, USA.

Schelhorn, T., O'Sullivan, D., Hakley, M. and Thurstain-Goodwin, M. (1999), STREETS: An Agent-Based Pedestrian Model, Centre for Advanced Spatial Analysis (University College London): Working Paper 9, London.

Schelling, T.C. (1971), 'Dynamic Models of Segregation Journal of Mathematical Sociology', Journal of Mathematical Sociology 1: 143-186.

Serenko, A. and Detlor, B. (2002), Agent Toolkits: A General Overview of the Market and an Assessment of Instructor Satisfaction with Utilizing Toolkits in the Classroom, McMaster University (School of Business), Onario, USA.

Sondheim, M., Gardels, K. and Buehler, K. (2005), 'GIS Interoperability', in Longley, P.A., Maguire, D.J., Goodchild, M.F. and Rhind, D. (eds.), Geographical Information Systems: Principles, Techniques, Applications, and Management (Abridged Edition), John Wiley & Sons, New York.

Swarm (2006), Swarm: A Platform for Agent-Based Models, Available at: http://www.swarm.org/ [Accessed on August 9th, 2006].

SwarmWiki (2006), Tools for Agent-Based Modelling, Available at: http://www.swarm.org/wiki/Tools_for_Agent-Based_Modelling [Accessed on August 24th, 2006].

Tatara, E., North, M.J., Howe, T.R., Collier, N.T. and Vos, J.R. (2006), 'An Introduction to Repast Simphony Modeling Using a Simple Predator-Prey Example', Proceedings of the Agent 2006 Conferenceon Social Agents: Results and Prospects, Chicago, USA.

Tesfatsion, L. (2006a), 'Agent-Based Computational Economics: A Constructive Approach to Economic Theory', in Tesfatsion, L. and Kenneth, L. (eds.), Handbook of Computational Economics, North-Holland Publishing, Amsterdam, the Netherlands.

Tesfatsion, L. (2006b), General Software and Toolkits, Available at: http://www.econ.iastate.edu/tesfatsi/acecode.htm [Accessed on August 24th, 2006].

Tobias, R. and Hofmann, C. (2004), 'Evaluation of Free Java-libraries for Social-Scientific Agent Based Simulation', Journal of Artificial Societies and Social Simulation, 7(1).

Topa, G. (2001), 'Social Interactions, Local Spillovers and Unemployment', Review of Economic Studies, 68: 261-295.

Torrens, P.M. (2004), Simulating Sprawl: A Dynamic Entity-Based Approach to Modelling North American Suburban Sprawl Using Cellular Automata and Multi-Agent Systems, Ph.D. Thesis, University College London, London.

Torrens, P.M. and Benenson, I. (2005), 'Geographic Automata Systems', International Journal of Geographical Information Science, 19(4): 385-412.

Ungerer, M.J. and Goodchild, M.F. (2002), 'Integrating Spatial Data Analysis and GIS: A New Implementation using Component Object Model (COM)', International Journal of Geographic Information Science, 16(1): 41-53.

Vos, J.R. (2005), 'Repast .NET: The Repast Framework Implemented in the .NET', Annual Conference of the North American Association for Computational Social and Organizational Science (NAACSOS), Notre Dame, Indiana, USA.

Wegener, M. (2000), 'Spatial Models and GIS', in Fotheringham, A.S. and Wegener, M. (eds.), Spatial Models and GIS: New Potential and New Methods, Taylor and Francis, London.

Westervelt, J.D. (2002), 'Geographic Information Systems and Agent-based Modelling', in Gimblett, H.R. (ed.) Integrating Geographic Information Systems and Agent-Based Modelling Techniques for Simulating Social and Ecological Processes, Oxford University Press, Oxford, pp. 83-104.

Wolfram, S. (2002), A Knew Kind of Science, Wolfram Media, USA.

Wooldridge, M. and Jennings, N.R. (1995), 'Intelligent Agents: Theory and Practice', Knowledge Engineering Review, 10(2): 115-152.

Zeiler, M. (1999), Modelling our World: The ESRI guide to Geodatabase Design, ESRI Press, Redlands, California.