

Software Requirements Engineering for Transparency

Thesis submitted for the degree of

Doctor of Philosophy

at the University of Leicester

By

Baraa Zieni

Department of Informatics

University of Leicester

January 2021



UNIVERSITY OF
LEICESTER

Abstract

The last decade has seen a period of rapid development for Software Engineering, with ever larger and more complex systems making it harder for end-users to get insights into their data, operations, and functionality. Users often are not aware of the underlying processes of complex system and thus implicitly have to trust them. Trust is vital when it comes to the users' relationship with the software. Losing the trust of external or internal stakeholders in a product can have a significant negative impact on a company's success and reputation.


A major factor of users' trust is the transparency of a system. Transparency is the appropriate amount of information that the user needs from the system to use it successfully (achieve their goals) and exercise informed trust judgment. However, it is vital that transparency is applied correctly and to the right areas in order to enhance users' trust.

This research develops a methodology to help requirements engineers capture and formulate transparency requirements in order to enhance users' trust by providing them with the relevant and appropriate information. To substantiate this approach, the relationship between trust and transparency is investigated in a literature study. Then, the Transparency Engineering Methodology (TEM) is developed and evaluated using a case study. TEM has been shown to create greater regard for transparency concerns in the development cycle, leading to the user being informed about how their data used and who has access to it. This degree of transparency is extended to backend and indirect collection processes where informing the user is often overlooked. The positive effect is illustrated by the quality of the output and comments from the developers, which showed that TEM can generate holistic user stories with transparency at the centre.

Statement of Originality

The work presented in this thesis for the degree of Doctor of Philosophy (PhD) with the title “Software Requirements Engineering for Transparency” was carried out by the author in Department of Informatics at the University of Leicester.

In this thesis, the work recorded is original except where acknowledged or referenced. None of the work has been submitted and is not presently submitted, for another degree at this or any other university.

Signed 

Date29/01/2021....

Baraa Zieni

Department of Informatics,

University of Leicester, University Road,

LE1 7RH

Acknowledgements

This is addressed to those who have made the most decisive contributions to my research and academic development over the past few years.

My sincere gratitude goes to my supervisor, Prof. Reiko Heckel, for being a great support, and for his immense knowledge and guidance. Reiko is one of the most intelligent people I have met, and I have always admired his passion for life and science. Considering the challenges in this open-ended area of software engineering, Reiko never hesitates to fully support this research and me, for that, I am eternally grateful. Reiko's meticulous observation and patience were invaluable throughout the time of bringing this thesis into maturity.

I am also very thankful to the researchers in Computer Science Department of the University of Leicester, Michelle Pryce, Dr. Artur Boronat, Dr. Richard Craggs, Dr. Irek Ulidowski, Dr. Karim Mualla, Prof. Effie, Dr. Emilio Tuosto, Dr. Mattias Heintz, and my second supervisor Prof. Thomas Erlebach. It was an honour to have worked with them all. All the great discussions with them, especially Dr. Fer-Jan, have had a major role in shaping my critical awareness. I also want to thank Dr. Arianna Rossi, Dr. Shola Oyedeji, Dr. Dayana Spagnuolo for the great discussions on transparency and trust research.

I also would like to thank Prof. Raian Ali, who supervised me during my visiting researcher placement at the University of Bournemouth; thank you for your motivation and immense knowledge. His passion for research and guidance was of great value in developing this thesis.

Finally, I would like to thank Dr. Ruzanna Chitchyan for being a great inspiration, and it was an honour to have worked with her during her time at the University of Leicester. Her guidance and support throughout the first year of her supervision were invaluable in developing the thesis. This project was funded by the University of Leicester, UK. I was honoured to be the recipient of a Doctoral Scholarship, which provided me with necessary support while my research was underway.

Dedication

I would like to dedicate this thesis to:

Fawaz Al-Azki, whose passion for science and the earth constantly gives me the infinite energy to keep on going. Fawaz used to say, and I still hear it “if you don’t give back to earth, you are not part of it”. I promise you Fawaz that I am going to continue what we have started. Tomorrow is always a brighter day. You always wanted to see me in this moment when I finish my PhD thesis, and here I am, and you are in my soul forever and a day more.

Friends and family have always been a great source of inspiration, pleasure, and relief, and I am grateful to all of them. Very special gratitude goes to those who filled my soul with unconditional love and support. I have been fortunate to have the best and purest energy from these stars during my life’s journey and beyond: Sarah Nizam (my angel), Ioanna Maria Pateli (Babynini), Matthias Heintz (Barbini), Bushra Canaan, Dima Mhrez, Rami Mahfoud, Samaneh Soleimani, Katya Alkhateeb, Virginia Lopez Valera, Karam Al-Ahmed, Riman, Laura, Somar Shaaban, Mohamed Abdelwahab, and Tarek Alabbassi.

The Zieni and Fadel families, my grandmothers and fathers Fadel and Zieni, my beautiful and intelligent sisters, Rasha and Hiba, my dedicated mother and father, professors Hiam and Adib, your passion to science and continuous hard work have brought me to where I am now. My other family Obree and Harvey the ones that supported me with an open heart...and Eddie 😊

I finally dedicate this research to all of the humans and our beautiful universe.

Contents

1. Introduction.....	10
1.1 Overview	10
1.2 Problem Statement and Research Gap.....	13
1.3 Research Questions, Aim and Objectives.....	14
1.4 Research Methodology Summary.....	15
1.5 Main Contributions	15
1.6 Thesis Outline	16
1.7 Publications.....	17
2. Background and Definitions	18
2.1 Trust.....	18
2.1.1 Definitions of Trust.....	18
2.1.2 Initial Trust and Ongoing Trust	20
2.1.3 How is Trust Measured?.....	21
2.1.4 Review of Trust	22
2.1.5 Trust Factors	24
2.1.6 Rationale for Selecting Trust as an Overarching Goal	26
2.1.7 Delimitations of Trust in this Research	27
2.1.8 The Notion of Trust in Software Engineering	28
2.2 Transparency and Trust	28
2.2.1 Transparency.....	28
2.2.2 Transparency and Trust Outside the Field.....	32
2.2.3 Transparency and Trust Inside the Field.....	35
2.2.4 Trust and Transparency Challenges.....	36

2.3 Requirements Engineering.....	36
3. Research Methodology	39
3.1 Introduction.....	39
3.1.1 Qualitative Research Design (in this thesis) and Research Methods.....	40
3.1.2 Systematic Literature Review	41
3.1.3 Case Study Design	42
3.2 Sampling Strategy and Size	44
3.3 Ethical Considerations	44
4. Systematic Literature Review	46
4.1 Overview.....	46
4.2 Identification and Clustering of the Literature	46
4.2.1 Definition of Search Criteria.....	46
4.2.2 Filter Criteria.....	47
4.3 Synthesizing using Concept Map and Clusters.....	47
4.3.1 Concept Map.....	48
4.3.2 The Process of Clustering and Results	50
4.3.3 Trust and Transparency Groups.....	61
4.3.4 Findings and Discussion	65
4.3.5 Limitations	68
4.3.6 Working Definition of Transparency.....	69
5. Transparency Engineering Methodology.....	71
5.1 Aim and Vision.....	71
5.2 Scope.....	72
5.3 Overview of Transparency Requirements Patterns	73
5.3.1 Methodology Steps Outline	75

5.3.2	Transparency Requirements Pattern Classification	76
5.3.3	Requirements Pattern Template	77
5.4	Basic Concepts	78
5.4.1	Conditions	78
5.4.2	Identification of Data Interests	79
5.4.3	Domain Driven Pattern Prompt List	80
5.4.4	Content and Presentation	81
5.5	Prerequisites of the Methodology	83
5.6	Overview of the Methodology	84
5.7	Methodology Steps	85
5.8	Transparency Requirements Patterns	89
5.8.1	Data Transparency	89
5.8.2	GDPR Transparency	92
5.8.3	Use Case Transparency	96
5.8.4	Process Transparency	100
6.	Evaluation	105
6.1	Evaluation of the Transparency Engineering Methodology	105
6.1.1	Research Question and Approach to Evaluation	105
6.1.2	Case Study Setup	106
6.1.3	Results	110
6.1.4	Limitations of TEM	122
6.1.5	Interpretations of the Analysis	124
6.1.6	Threats to Validity	127
6.1.7	Discussion (summary)	128
6.2	Related Work	129

6.3 Extended Discussion on TEM	133
6.3.1 Transparency Patterns by Motivation Types	133
6.3.2 Event-Based Transparency	135
7. Conclusions and Future Work	137
7.1 Conclusions.....	137
7.2 Future Work.....	139
7.2.1 Standardisation.....	139
7.2.2 Transparency for Non-functional Requirements	140
7.2.3 Transparency Requirements Elicitation Process.....	141
7.2.4 Refinement Mechanisms.....	142
7.2.5 Trust Measurement	143
7.2.6 Sustainability	143
References.....	144
8. Appendix.....	155
8.1 Prequestionnaires	155
8.2 Postquestionnaires	164
Postquestionnaire1- Developers	164
Postquestionnaire 2 - Developers	167
Postquestionnaire 2 - Domain Experts.....	169
Postquestionnaire 2- Domain Experts.....	171
8.3 TEM Implementation on SpiritHub.....	174

Chapter 1

1. Introduction

1.1 Overview

Software systems are ubiquitous and increasingly opaque. The recent decade has been a period of rapid development for software, especially with the widespread adoption of agile development methods like SCRUM, which help to streamline the development of increasingly complex system but, in doing so, introduced new obstacles when making development decisions [1]. Software needs to be built with the aim of being transparent [2]. “Good transparency [...] enables people to make informed decisions” [3]. Transparency is also a fundamental principle for data processing under General Data Protection Regulation [4]. Transparency about the use of personal data is considered an enabler to user’s trust [5].

The current system’s complexity has, therefore, become important for the user to build the right trust with systems based on well-informed decisions, especially when it comes to their data. The right to be informed becomes essential as the users’ personal and behavioural data is used without them having much knowledge of why or how it is being used [6, 7]. This can cause them to lose trust or build false trust with a system [8]. It is important to address this in the system’s development since the software is everywhere in society and hugely impacts people’s daily life.

Furthermore, as can be seen from several industry examples, losing the trust of external or internal stakeholders in a product can have a significant negative impact [9, 10]. For instance, when Samsung Galaxy Note 7 smartphone and several consumers reported cases where their new phones exploded, users lost their trust in the product and thus in the company [9]. To avoid this, enabling trust judgement is essential, especially when it comes to making decisions on changes to software systems [9, 10]. Companies who fail to maintain stakeholders’ trust will eventually fail to deliver successful products. To maintain trust, companies need to find a way to ensure trust through their development activities [11].

One of the qualities that have been examined in enhancing trust is transparency. However, the number of projects addressing transparency in the development process is very limited. A study stated that transparency is fundamental to trust [12]. Another study mentioned that “Transparency has been often associated with positive properties such as increasing trust” [13]. This research argues that transparency needs to be applied correctly in order to increase user trust (see Section 4.3.4) therefore, further investigation on their relationship is needed.

Trust is a complex concept, and there are very few guidelines to help the requirements specialist and developers to build trustworthy systems [14]. Trust also has been identified as a “crucial requirement in the implementation process of information systems” [15]. Trust is a relationship between trustor and trustee [16]. The main focus of attention is on the end user’s relationship with a software system. For a user to have a trust relationship with a system, it is pertinent that the user knows about the system’s qualities, functionalities and data which would help them to make an informed decision in achieving their goals with that system. Trust related to the end-user can be divided into two categories: the initial trust and the ongoing trust [17]. This research supports both categories of trust. To address the ongoing trust enabling *user’s trust judgment* while interacting with the system is vital.

Previous research has stated that trust can be engineered in a system systematically during the development process [18]. Researchers have suggested methods to develop user trust in software systems, such as software patterns [14] and factor analysis [19]. However, in both studies the focus was on high level goals of the system without providing details on how the system can achieve such goals. Moreover, the work considered the static requirements of trust while the dynamic nature of the concept was not covered. Additionally, methods for engineering transparency requirements in software have not been studied in this context. In fact, transparency requirements are often underestimated during software development. This can be due to stakeholders’ lack of understanding of the trust -transparency relation and worrying that users with a detailed understanding of how their data is processed may be less willing to provide it or consent to its use [20].

Transparency can be seen as a quality to enhance trust by helping users achieve their goals. Transparency has been defined as a “user-centric principle” as it refers to openness and

disclosure of information to the user in data protection law, business and governance [7]. Transparency can be described as the appropriate amount of information that the user requires for a better experience with the system as well as a function that enable their trust judgment. Providing the appropriate amount of information can be done by hiding as well as showing information see (Section 4.3.2). Human Computer Interaction (HCI) is another area of research that examine transparency and its effects on users, we refer to the literature on qualifiers of transparency [21] and user-friendly representations of privacy-related information [22]. The vast majority of these research are mainly concern about how the information is best presented to the user and can be limited to what the user sees when interacting with a system, which encompasses what that system is transparent about. It is therefore necessary to understand the relevant and appropriate amount of information to expose the user to. Common instances where user trust is affected due to inadequate transparency include:

- When users' expectations are not met due to a bad purchasing decision after they have spent some time researching about the product or service before they paid for it [23-25].
- When users fear they have lost control over their personal information or that it is being used without their knowledge or consent [5, 23].

Transparency can either encourage users to trust a system or discourage them from doing so [24] depending on various factors. The violation of the expectations of users with respect to system behaviour seems to be a 'critical moderator' of the impact that transparency has on users' trust. A transparent system must provide information contextually relevant to the users' concerns actions and personal data [5]. When it is, it has been shown to elicit a high level of trust and enhance the stakeholder-system relationship [5, 24, 26].

This research, thus, investigates the relationship between transparency and trust. Then, after that relationship is established, transparency engineering methodology has been developed to address transparency requirements. This study looks at transparency, to identify where a software system is lacking in transparency based on its given user goals. For instance, informing the user in a systematic way about their data, why it is being taken, and what processes are occurring. This will keep the user informed during their interaction with the

system and will help them make more informed decisions. The study starts from defining the complex quality of trust in the context of information systems. Then reviewed the literature on trust and transparency, as a result a list of clusters has also been identified describing the possible connections between trust and transparency.

1.2 Problem Statement and Research Gap

This research is in requirements engineering, focusing on a system's transparency in order to support the users' trust in the system. Trust and transparency relationships have been examined in various research areas, such as business [27] , public services and governance [28, 29], education [24], art [25] etc. Most of these research areas have focused on studying this relationship or reporting on it as a whole without mentioning or fully investigating the specific transparency properties that affect trust.

There is a strong potential for transparency to inspire trust in a software system in the following ways:

- 1- Building initial trust by providing information about the functionality and limitations of the system [17, 30]. And maintaining ongoing trust by helping users to perform tasks more effectively while interacting with the system [16, 17].
- 2- Improving trust judgment, leading to more informed decisions, and managing risks while using the system [25, 31, 32]

It, therefore, becomes crucial to understand the in-depth relationship between these two concepts. However, the research in this area has not yet answered the question as to whether there is a clear relationship in software engineering.

There is no clear definition of trust in software engineering, especially for ongoing trust. Without correct transparency users would miss relevant information. The user's lack of exposure to relevant information which usually leads to users having to lose right or gaining false trust with the software system. This is a major issue specially regarding user data protection. Transparency is neither addressed in the system development nor evaluated in a real-life scenario.

1.3 Research Questions, Aim and Objectives

The research questions, aim and objectives pertaining to this study are outlined below.

Research questions

The general questions to be addressed through this research are:

1. How is trust defined in the software engineering literature?
2. What is the connection between trust and transparency?
3. How can transparency in a software system be engineered to enable users' trust judgment?

Research Aim and Objectives

This research aims to help requirements engineers to manage transparency in order to enhance user trust and provide them with the right amount of information needed to enable their users trust judgment.

In line with the research questions, a literature review is conducted considering the following objectives. The main objective of this research is to provide the fundamental concepts showing the relationship between trust and transparency in order to engineer transparency that aims to inspire user trust with the software.

To achieve the aim of this research, the following objectives will be satisfied:

- To ascertain the meaning of trust and transparency and how it relates to software engineering through a robust review of relevant literature.
- To determine the relationship between user trust and transparency and to identify all factors that have an influence on this relationship.
- To develop a transparency engineering methodology to be applied by requirements engineers and developers to guide them in their system designs.
- To test this developed methodology using a case study investigating and evaluating its applicability, benefits, and limitations in a real-life situation in order to seek areas for future development.

1.4 Research Methodology Summary

To attain the research objectives, a robust qualitative research methodology will be employed throughout this thesis. The methods to be used include a systematic literature review (using a concept map [33] and clusters [34]), a conceptual framework (transparency engineering methodology), and a case study design (using semi-structured interviews, prequestionnaire, and focus groups).

Figure 3.1: Qualitative research design in this thesis [NOP= number of participants] provides a pictorial representation of the research methods employed throughout this study. The research methodology and all methods adopted throughout this research is address in detail in (see Chapter 3) of this thesis.

1.5 Main Contributions

This research aims to help requirements engineers to manage transparency in order to enhance users' trust and provide them with the right amount of information to enable their trust judgment. The main contributions to the filed are outlined below.

- We introduce a definition of trust in software systems.
- We develop a set of clusters using a concept map in order to investigate the relationship between trust and transparency in software systems. This is done using a systematic literature review see [4].
- Through the reviewed literature, the factors that influence trust and transparency are grouped and studied in relation to transparency and trust properties to understand how transparency can enable user's trust judgment. This leads us to develop a novel transparency engineering methodology, to guide requirements engineers on how best to engineer transparency within a software system.
- In order to evaluate the effectiveness of this transparency engineering methodology in a real-life situation, a case study is designed using a software project from a health care company which leads to examining the capabilities and identifying the limitations of the developed methodology. We find that the transparency engineering methodology encourages the effective implementation of different

functional requirements and provides the relevant information to the user to enable their trust-judgment.

1.6 Thesis Outline

Chapter one introduces the research, outlining the existing problem and motivation for the study. It also highlights the research questions along with the research aim and objectives. A summary of the research methodology employed as well as the main contributions in the thesis are stated.

Chapter two describes trust within and outside the field of software systems and explains the various factors that influence trust along with the rationale for choosing trust as an overarching goal for this research. Transparency is also defined and its connections to trust within and outside the field are explored. Next, the chapter elaborates on the requirements engineering area of study along with its practices.

Chapter three explains the research methodology that has been utilized throughout the research process. It also outlines the ethical considerations applied throughout the research during data collection.

Chapter four comprises of a systematic literature review examining the relationship between trust and transparency. Using a concept map, the current state-of-the-art is derived from the literature review and provides a summary of the relationship between trust and transparency in software systems from different domains and contexts using a set of clusters.

Chapter five explains the developed transparency engineering methodology. It highlights the vision, aim and scope of the methodology and gives a summary of the main concepts and terminology used. Transparency patterns are then introduced which formalize the requirements generation process.

Chapter six describes the case study design and aggregate the results of the evaluation, showing the weaknesses and strengths of the transparency engineering methodology. The related work section highlights the connection and compares the current state of the art with the research contributions.

Chapter seven concludes the research showing the main findings and the contributions as well as the recommendations and future work.

1.7 Publications

- 1- B. Zieni, R. Chitchyan, and R. Heckel, "Trust as a sustainability requirement," in Proceedings of the 6th International Workshop on IEEE Requirements Engineering for Sustainable Systems (RE4SuSy 2017), Lisbon, Portugal, 2017, vol. 5

The main focus of this paper is to highlight the essential role of the trust as sustainability requirement. The main contribution is to address the nature of trust and summaries a working definition in software engineering (see Section 2.1.8). This paper has also been cited throughout the thesis.

- 2- B. Zieni, R. Heckel, "TEM: A Transparency Engineering Methodology Enabling Users' Trust Judgement " in International IEEE Requirements Engineering Conference (RE'21), South Bend, USA (forthcoming).

Part of this publication is presented in Chapter 5, Chapter 6 and Chapter 7.

- 3- B. Zieni, D. Spagnuolo, R. Heckel, "Transparency by default: GDPR Patterns for Agile Development " in the 10th International Conference on Electronic Government and the Information Systems Perspective (EGOVIS2021), Linz, Austria (forthcoming).

Part of this publication is presented in Chapter 5, and Chapter 6.

- 4- B. Zieni, A. Rossi R. Heckel, "What is the Relationship between Trust and Transparency? A Review of the Literature on Software Systems" submitted to STAST2021 11th International Workshop on Socio-Technical Aspects in Security, Darmstadt, Germany.

Part of this publications is presented in Chapter 2 and Chapter 4.

Chapter 2

2. Background and Definitions

This chapter first gives a brief overview and definitions of trust, transparency and requirements engineering as discussed in the literature, after which it addresses the relation of trust and transparency, in software systems and more broadly.

2.1 Trust

2.1.1 Definitions of Trust

One common synonym of trust is the word “certainty”. This, however, does not allude to another common definition of trust which states that in any trust relationship there is a risk to be taken, which carries uncertainty into the trust relationship. If the result of an action is certain then no trust is needed [35].

As a starting point for this research, the following general definition of trust will be adopted:

“Trust, in a broad sense, is the belief that other people will react in predictable ways” [36]. The broad notion of trust can be described as “the willingness of a party to be vulnerable to the actions of another party based on the expectation that the other will perform a particular action important to the trustor, irrespective of the ability to monitor or control that other party (p.712)” [37]. Trust is a relationship between trustor and trustee [16]. When the system becomes part of the trust relationship the trust cannot be built as explained [38]. However, social science has defined impersonal trust the trust between the human and a system or institution. Impersonal trust is not based on personal but system factors [39].

Trust can be built through small details and actions; Brené Brown stated “trust is built in very small moments” [40]. Therefore, continuous feedback is necessary for tracking the current trust level [41]. In the context of this research, trust is defined by the imperfect judgement of the likelihood that individual users’ needs, and expectations will be met [42].

Trust is not modular.

While most software engineering methods are modular, understanding trust requires a holistic view of the system [14, 43]. The aggregation of these small interactions needs to be considered holistically for the entire system, because trust in one subsystem can be influenced by the outcomes of interactions in a different part of the system [16].

Trust is based on a stakeholder's imperfect assessment of risk to the satisfaction of their requirements.

One way to shape and focus the perspective on trust is by looking at the decisions that currently need to be made, which again can influence each other [44]. However, there is a lot of uncertainty as many decisions are potentially influenced by factors from various sources such as the environment, market, end-users, etc., of these factors are drawing from overlapping sources and detecting the same factor can be done from different perspectives. The information gathered on these factors will therefore be incomplete and might not be entirely accurate. Therefore, assessing a risk based on this deficient information will always lead to an uncertainty in the (expected) outcomes [45]. Decisions are made upon rational factors and other irrational factors. Although some scholars [46, 47] see humans completely as rational decision makers, the idea of bounded rationality possesses a view stating that the rationality is limited. Some limiting constructs include the amount of information and the intellectual restrictions of human brains as well as the time frame in which the decision must be made. One way of initially measuring the trust can be by assessing the risk that decision makers are willing to take even with all these uncertainties present.

Trust is a vital software quality.

Trust is a vital software quality as it can increase customer loyalty and can raise the amount of money spent on digital purchases [42, 48]. To ensure this quality is present in the desired level, it would be beneficial to be able to engineer trust into a software system and constantly monitor it throughout different changes and adaptations. The way to influence the outcome of software system development is through requirements engineering.

Trust and users' trust judgment.

Enhancing user trust is achieved by enabling users to be better informed when using a system. Trust judgment is defined as the user's ability to assess the system property of concern, which is based on their needs and expectations. It includes user assessment of trust degree and trust decision to trust or distrust the trustee [49]. This judgment occurs during validation, as well as throughout the user's interaction with the system. Thus, from the user perspective the proposed approach aims to provide better understanding of the system by receiving the right amount of information in an understandable way.

Even though better trust judgement can lead to a decrease as well as an increase of trust, either way the system itself will be a better one from the users' perspective (e.g. being able to judge that the system is not right for the user since they cannot trust it to fulfil their needs or they know that this will not be the case because of the improved trust judgement) [50]. This then will prevent them from investing time and effort in a system, which is unsuitable for them [51].

2.1.2 Initial Trust and Ongoing Trust

Trust related to customer relationship management can be divided into two categories: the initial trust and the ongoing trust [17]. Initial trust requires the willingness of the other party to trust without any previous experience or knowledge of its background [17]. Initial trust has been well examined in existing research literature and is reflected in the existing definitions of trust. Moghaddam et al. present trust model that contrasts the commonly existing static trust models in their research with a dynamic one [30]. This shows that it is not enough to only consider the initial trust (mainly based on the static factors) but also important to address the dynamic nature of trust which manifests in the ongoing trust [16].

Ongoing trust is dynamic and relies on actual experiences and interactions between two parties [17]. Ongoing trust is mostly overlooked or skimmed over in existing trust research. Therefore, this study will contribute an extended definition of trust, reflecting the different nature of ongoing compared to initial trust.

The initial trust is mostly stated as a stage where the trust relationship is established between the end user and the software system. This stage mainly demonstrates the level of

the adoption with the new system and studies the level of risk and uncertainty that the end user might take whenever they start using a new system [52].

Although initial trust is widely covered, for example [31] [53], it is evident that there is a shortage of research and publications on the ongoing stage of trust [16]. The ongoing trust has a different nature than the initial one. The initial trust in a system is created and influenced mostly by static factors leading to a first 'trust impression' built during or sometimes even before the first encounter of the system whereas the ongoing trust starts out based on the initial trust level and is built over time. It is not only affected and changed by the trust factors but also their interactions. Every single interaction can affect the trust between the participating entities of the trust relationship and even influence uninvolved entities. Firstly, in order to comprehend and analyse the interactions of interest, factors which commonly and repetitively show up together in the literature are identified and it is taken into consideration, how frequently they show up together as well [16].

2.1.3 How is Trust Measured?

One of the main goals when performing the literature review on trust is to identify the research methods that have been used to study and evaluate trust. To measure trust, a commonly used instrument is the Technology Acceptance Model (TAM) [54, 55], which resulted in the TAM questionnaire [56], measuring perceived ease of use (PEOU) and perceived usefulness (PU). PEOU and PU are two properties which have been proven to be correlated to initial trust [55]. However, [57] argues that TAM might not be the best way to measure trust or identify relevant factors. Instead, they use and propose Confirmatory Factor Analysis (CFA) using Partial Least Square (PLS). CFA allows for testing hypotheses about a particular factor and, the assessment of its influence on the evaluated construct, for example by checking a variety of hypotheses. CFA is the measuring part of Structural Equation Modelling (SEM), PLS is a SEM technique [53]. SEM is used to prove the correctness of a model. The way this is done, for example when applying SEM-PLS [53], is to use SEM to get expected results from the model. These expected results are then compared with the real-world outcomes, e.g., gathered using a questionnaire. If both match, this is taken as an indication that the model adequately reflects the conditions and events in

the real world. To support the application of the PLS method of proving models, SmartPLS has been developed, which is a software that implements PLS.

Factor analysis is widely used [55, 57]. Factor analysis is a method to statistically explain observed differences between correlated variables which are measured. Factors (unobserved variables) are used to come up with the explanations. Examples of factors are perceived risk, individual propensity to trust, cultural factors, cognitive factors, reputation. Other than using one form of factor analysis or another there is no commonality with which to judge the results in the literature review, therefore, inspecting and understanding the used methods is necessary. As different studies have analysed the trust using different methods, with the different nature of these methods, one cannot choose any of them without taking into consideration the context in which they are specific factors for each requirement applied. Based on the factor analysis from the literature review, the specific factors for each requirement can be derived. How well trust is supported in the system can then be derived from the requirements specification documents.

2.1.4 Review of Trust

Trust is a diverse topic and a broad notion that has been studied in several scientific disciplines and overlapping areas [42].

Trust has been examined from different perspectives. The variety of areas and foci from which trust can be studied is also reflected in the scientific literature, ranging from psychological papers examining trust as a feeling of humans [38] to technical papers on trust between agents in game theory [58]. Based on the premise to have the end-user as primary focus of interest regarding the human part of the trust relationship, this study focused on trust from the customer relationship point of view on software as a starting point. What makes software different from human is their design to be consistent, concrete, i.e., quantified, ability to collect evidence of operation, such as traceability, interactivity where people can query the system in real-time, memorability, automated decision making, predictability, etc.

Literature shows a shortage on the research of the ongoing stage on trust. As the ongoing has different nature than the initial one. The ongoing trust starts after the initial and builds over time. It gets changed by the trust factors and their interactions [16, 53]. Every single

interaction can affect the trust entities in the trust relationship. Also, as a first step, it is pertinent to understand these interactions by analysing the commonly repetitive factors (how frequently they show up together).

Furthermore, many studies, for example [57] and [1], have illustrated the factors that play a role in the initial stage and the adoption of innovative technology in general and software systems in particular. By studying its factors, the researchers tried to understand the nature of the trust at the same time. The literature review revealed that these factors vary in different software system domains as they can be prioritised based on the nature and the services that the software system is meant to provide. They also give a better understanding to the nature of the trust relationship and measure the trust level over time.

There are two fundamentally diverse ways of studying trust. Some research focus on the trust as the main topic and analyse the factors that influence trust [55, 59-63]. Other research study its role as one of the factors that affect aspects of the software development process and its resulting artefacts [31, 52-54, 64, 65].

Trust as a topic

As trust is a broad notion and overlapping, many researchers are interested in understanding it by breaking down trust and analysing the factors that influence the trust. The entities that are involved in trust relationship in the reviewed literature are humans – humans [59] or humans – systems [55, 60-62]. It is worth mentioning that the focus of the literature review is trust between humans and systems, because information technology field is mainly interested in the relation where the software systems are involved. Information systems are examined as a mediator or trustee in the trust relationship [14, 41, 66], this study focuses on the latter case.

Trust as a factor

It important to understand trust notion to be able to engineer (measure, operate, enhance) it. But as trust interacts with other system components, to a get more holistic view, it is necessary to examine trust as a factor. In Requirement Engineering, all the system components need to be addressed as a part of the system and its outcomes. Some of these studies have examined more closely trust influence in the adoption with new system or

technology [65], intentions of the customer transactions [54], decision making [31], the growth with the e-commerce [57].

2.1.5 Trust Factors

Previous literature suggested that trust can be very difficult to address, therefore, to examine trust the best way is to investigate on the factors that influences trust [18]. Trust has been studied as a formative construct as well as reflective construct in existing literature. To contribute to the discussion, if trust should be considered a formative or a reflective construct it is therefore necessary to elicit the factors that influence trust.

Trust can be studied as a formative construct if the factors that affect the trust are directly observable. It can also be studied as a reflective construct if the factors that affect it are latent (hidden, not directly observable).

Some software systems need to address the trust using the *security* (for example: Banking system), some others under *privacy* (social media), therefore, it is important to analyze the factors behind the trust and categorize them.

Factors that are not actually influencing trust

One might think that a trust in a bank would have a positive impact on the trust in their online banking software system. But as [53] showed, the former is not actually influencing the latter.

Also, approval from certification agency has no measurable influence on the trust regarding the approved software. Kim et al. [31] showed that with their research on third party seal.

Besides properties of the software systems, characteristics of the person might have an effect. However, for the characteristic ‘individual customer trust propensity’. [55] showed no influence.

When looking at these examples of factors found not to having an influence on trust, it is important to keep in mind that they always should be viewed or interpreted with regards to the software domain. Nevertheless, existing research can serve as a starting point of addressing trust in the software domain which is of interest to this research. Yet, the factors

that have been identified as affecting trust are of an even higher relevance and will therefore be highlighted in the next subsection.

Factors that are influencing trust

Shneiderman [60] presented navigational structure and visual appeal as two integral properties of the software application that have an impact on trust. Regarding characteristics of people culture, Li et al [67] report that the initial trust is affected by the “perceived social pressure” which forces one to perform or not in accordance with some subjective ideals, the cognitive reputation, calculative, and organizational situational normality based factors. They also observe that individual’s personality or the technology did not substantially affect the trusting beliefs.

From the robust review of literature on trust, trust has been examined in a myriad of ways. The factors that affect trust were studied based on trust categories. The factors can be grouped into several “trust categories”[16] which include:

- Personality-based trust;
- Cognition-based trust;
- Institutional-based trust;
- Information technology, including factors, for instance, security, privacy, and general online experiences;
- Social factors, such as national culture;
- Diffusion of innovation: “as users initially receive some information about an innovation and its advantages and disadvantages, this forms their initial attitude toward the innovation and influences subsequent adoption decisions”.

As regards to information trust-based, [68] shows in their survey paper the factors that influence trust-based information system, security (see above) and privacy were identified as the primary concerns of the end user when dealing with software. Also, the perceived information quality, perceived usefulness, and usability, etc, are taken into account when examining user trust with the software. Likewise, [69] mentioned security control for initial trust, and mentioned that security is a vital factor when it comes to trust.

2.1.6 Rationale for Selecting Trust as an Overarching Goal

Why Trust?

Trust is a vital component in the decision-making process of not just customers and end-users but also the decision makers in the development team. Methods of risk analysis are commonly used to predict the consequences of strategic decisions, but they tend to focus on objective features. Trust is subjective concept and a complex relationship between multiple stakeholders, influenced by perceived risk, but changing over time as the result of ongoing interactions [53].

This research will also contribute to the social dimension of sustainability, specifically on its trust requirements. As noted by Goodland [70], “Social sustainability [...] create[s] the basic framework for society. It lowers the cost of working together and facilitates cooperation: trust lowers transaction costs.”

Besides lowering transactions cost, another software development challenge where trust can play an important role is in decision making [31]. As the reliance on software systems increases in areas such as health, military and in government, as does the significances of failures. Therefore, user trust with these systems becomes crucial aspect in judging their successful deployment of systems [43].

Decision making and end user empowerment

The recent decade has been a period of rapid development for software systems in many senses. Especially the recent advent and spreading of agile development methods like SCRUM [1] which introduced new obstacles when making development decisions [71]. [31] stated that costumers’ trust and perceived risk have a high influence on their purchasing decisions with of website.

Trust therefore is now becoming a more crucial issue especially when it comes to stakeholders making decisions with the software. In this research, we are interested in the user trust and the influence of their decisions with the software.

When it comes to software systems, trust plays a role at several stages and for several stakeholders. On one hand there is the developer and business owner side, on the other hand there are the customers or end-users. The developers responsible for the software

implementation foremost should trust in their own software product to convince their customers that it is trustworthy and kind of 'transfer' their own trust to the end-user. Although it would be interesting to research how this trust is built and what could influence it, there are many elements/parties that need to be considered. For this research, it will be assumed that this trust is present as a pre-requisite for the building and maintaining of end-user trust. Trust from the end-user perspective will be the focus of this research as one of the goals of this research is to bring the end user perspective in the software development process. Most users agree on the need for privacy, however, when signing up to a service they might agree to any regulation if it gives them access to the service. This is mainly caused by the end users' lack of knowledge. Therefore, empowering the end user is important to increase their knowledge about their rights and role in the development process. The end users' empowerment is essential, especially when it comes to enhancing the software qualities like trust. The end user is the main source of the data as they represent the society and its variety.

Kim et al. [31] mentioned that in case of decision support systems security, privacy and trust are all essential factors when it comes to decision making.

2.1.7 Delimitations of Trust in this Research

Firstly, the focus will be on the security - customer relationship as they are well studied, for building good bases and expanding the research.

Trust and security

Several researchers have investigated the notion of trust in software systems as directly associated with security or privacy. Some authors concentrated on security of the system as an artefact, while others emphasized on the human aspects of security. For instance, Elahi [72] cited in [16] suggests that the more trust there is between the end users of the software system and other stakeholders, the less security they need (e.g., if there is no perceived risk or loss - i.e., mistrust - there is no perceived need for protection). The concept of security of a software system is a mere prerequisite for the initiation and maintenance of trust towards it [16]. Security does not guaranty trust towards a software system, but it is important to make it trustworthy [16].

Trust and the customer relationship

Trust is one of the key foundations that enables relationships. Brown [41] suggested that trust in relationships is built over time through very small actions. Therefore, when considering changes of trust over time, it is crucial to distinguish between the initial trust and ongoing trust to clearly highlight the evolving and changing nature of the notion of trust. Trust plays a crucial role in reducing users' uncertainty. Damian-Reyes et al suggest that it is one of the factors that can affect user confidence in a software system [73]. Previous studies have reported trust as a behavioural intention which can affect vulnerability and uncertainty [74]. Similarly, Ruohomaa et. al [75] discussed trust as a key instrument needed for end users to cope with the uncertainty of making a decision.

2.1.8 The Notion of Trust in Software Engineering

To operationalise the notion of trust into software requirements to design trusted software systems, first the scope of trust needs to be defined. From research it is observed that: Trust is a relationship. Although some key characteristics, such as security, are essential for the initiation of the relationship and is formed by the interactions between the involved entities. Trust is dynamic just as humans' relationships therefore trust relationship is subject to continuous change. The change is guided by feedback from the interactions whose results are evaluated by the participants and, where considered relevant, contribute to developing or eroding the relationship. Trust is cumulative; one single result from an interaction may not have significant impact on the trust relationship, whereas, a repetition of similar results are most likely to have a cumulative effect [16].

2.2 Transparency and Trust

2.2.1 Transparency

The notion transparency appears in many areas with a variety of implications. Transparency has been defined as a user-centric principle as it refers to openness and disclosure of information to the user in data protection law, business and governance [7]. However, transparency also refers to hiding processes from the user in case of distributed systems or making action or computational process unnoticeable from the user in computer software

[76]. This is to reduce the complexity of the system from the user in both cases (see Table 2.1: Transparency types by area of application).

In software engineering, transparency can mean to make the development processes of the software visible to the stakeholders, for example, through SCRUM and agile processes [77]. This wide spectrum of definitions has made it vital for researchers to specify the transparency they are about to study in their domain context as a first step. This research looked into the definitions of transparency in software systems and the potential benefits from the different implications of this term.

Turilli and Floridi [78] describe transparency as the possibility of information, intentions or behaviours to be accessed through a process of disclosure. Likewise, it is considered an important concept that can support users in the decision-making process.

Transparency is said to be dependent on the context it is used and can thus be referred to as a quality-in-use. According to Bevan and Azuma (1997) [79], quality-in-use in ISO/IEC DIS 14598 can be measured in terms of how effectively and efficiently users can attain their required goals in a specific environment.

Transparency, in the field of system engineering, has been defined by Leite and Capelli [80] using an NFRs graph. This graph was made up of 33 soft goals which were grouped into three levels with transparency at the top. The second level consisted of five soft goals which are reported to influence transparency directly: accessibility, usability, informativeness, understandability and auditability [81]. Similarly, Tu [2016] [77] stated that transparency in software systems should have three attributes: accessibility, understandability, and relevance as they affect the abilities of stakeholders to view relevant information needed to achieve their goals.

Hosseini et al [26] in their own research divide transparency into four parts to aid its comprehension and inclusion in information systems engineering: stakeholders, meaningfulness, usefulness, and information quality. Meaningfulness here encompasses data transparency, policy transparency and process transparency which explain how stakeholders comprehend information and what actions and explanations led to that.

Leite [82] explains that transparency is very important as Mylopoulos stated it helps to connect requirements models to software. Likewise, Cysneiros et al [83] consider transparency as an important requirement needed for new technology and more robust systems to be adopted by users. They also state that NFRs like trust and privacy are equally important.

It is thus pertinent for requirements engineers to develop a framework, motivated by the need for transparency, which would enable forward and backward traceability [82]. The i^* framework has been recommended by Leite for developing a framework targeted at engineering transparency [82]. Similarly, Yu [84] reports that the i^* can be used to model trust “without creating any new concept carrying special semantics to represent trust.”

The broad definition is based on and composed of several existing transparency definitions in the literature (see Table 2.1: Transparency types by area of application). Schnackenberg and Tomlinson [85] have collected an overview of transparency definitions in information system domain, with one of them highlighting that transparency is created by revealing information, “the disclosure of timely and accurate information”. Another one states that transparency is: “The degree of visibility and accessibility of information”. The part of this study’s transparency definition related to revealing information to a stakeholder is taken from the transparency explanation by Evers and Cramer [25]: “Transparency aims to increase understanding and entails offering the user insight in how a system works, for example by offering explanations for system behaviour”. Schnackenberg and Tomlinson [85] also states that transparency increases the trustworthiness of that organisation, and this trustworthiness leads to more stakeholders’ trust in said organisation. Moreover, transparency is one of the main principles in personal data processing, in GDPR transparency “requires that any information and communication relating to the processing of those personal data be easily accessible and easy to understand ” *ibid.* (39)

To scope and develop a working definition of transparency in the context of this research, first it is important to understand the relationship between trust and transparency in the software systems. This is addressed in detail in the literature review in Chapter 4.

Table 2.1: Transparency types by area of application

Type of transparency	Definition	The area of application	Refs
Collaboration transparency	<p>“People often work with others. With the increasing importance of computers in our work and everyday lives, it is natural to expect computers to play an important role in facilitating collaborative work”</p> <p>“Flexible collaboration transparency: supporting worker independence in replicated application-sharing systems”” These systems support synchronous collaboration, where coworkers’ interactions are simultaneous or separated by short periods of time, rather than asynchronous, where interactions are separated by relatively long periods [Grudin 1994].”</p>	Information systems	[86]
Computational transparency	<p>“refers to the degree to which the computational effort of a code sequence written in a programming language is obvious to the developer. The closer a sequence of commands is to the underlying machine, the more transparent that sequence is”</p>	Software systems	[87]
Transparency of a system	<p>“makes the user of a network unaware of the fact that they are interacting with a network.”</p> <p>“In computer software, an action is transparent if it takes place without any visible effect. Transparency is usually considered to be a good characteristic of a system because it shields the user from the system's complexity.”</p>	Software systems	[88]
Operational definition of transparency	operational definition of transparency with three attributes: accessibility, understandability, and relevance:	Requirements specification Use case models Communication Software development	[77]
Transparency in software engineering	“Making a development process visible to observers”	SCRAM, Agile	[89]
Transparency GDPR	“Transparency is a user-centric principle proposed to empower users to hold data processors accountable for the usage and the processing of the user’s personal data”	Within the General Data Protection Regulation	[7]
Distribution transparency	"Transparency implies the concealment from the user and the application programmer of the complexities of distributed systems. "	Distributed systems.	

	<p><i>Access transparency</i>: “Same api for local or remote access”</p> <p><i>Location transparency</i>: “No knowledge of location”</p> <p><i>Replication transparency</i>: “Multiple copies – all kept consistent and referred to as a thing, rather than an instance”</p> <p><i>Concurrency transparency</i>: “Concurrent processes share without interference, while making no special arrangements.”</p>		[90]
Automation transparency	“refers to the extent to which automation provides the information needed for a human to make accurate predictions of its behavior in order to make this decision”	Intelligent systems	[91]
Algorithmic transparency	“is the principle that the factors that influence the decisions made by algorithms should be visible, or transparent, to the people who use, regulate, and are affected by systems that employ those algorithms.” The phrase was invented in 2016 by Nicholas Diakopoulos and Michael Koliska about the role of algorithms in deciding the content of digital journalism services.	Software systems	[92]
Technical transparency	“revealing the source code, inputs, and outputs of the algorithm which can build trust in many situations.”	Information systems	[93]
Operational Transparency	“Make your processes visible to customers and your customers visible to employees”	Software systems	[94]
Transparency Enhancing Tools (TETs), and Usable Transparency	“Can help individuals to exercise their right for transparency, and subsequently for intervenability, by technological means. TETs can be defined as tools providing insight into how the users’ data are being collected and processed and visualize related consequences in an accurate and comprehensible way.” “Besides, the GDPR emphasizes that transparency should be provided in a concise, intelligible and easily accessible, that is usable, form”	Information systems	[95]

2.2.2 Transparency and Trust Outside the Field

Trust and transparency have been studied in various areas. To examine their relationship, previous research inside and outside the field of information systems has to be examined. The areas outside information systems mainly included social science, governance, business, ethics, and health areas etc. Also, several studies targeting trust and transparency from psychological and sociological perspective and not necessarily dealing with digital

systems have also been scanned for background information. However, topics from the information system field was the focus.

Useful transparency can only be successful “when it enables stakeholders to make decisions based on the provided information and acts upon them” [26]. For instance, in the sociological and psychological sense, transparency can be described as garnering information and knowledge about the environment to prepare actions and decisions [96].

Both concepts trust and transparency have been realized in the human societies very long time ago. To transfer them into the relationship between human and piece of software required a deep understanding of the factors that have relationships with both qualities. The following pool of reviewed papers from the non-information systems selected where they studied the relationship between the concepts, where the software was the moderator of the relationship between the user and the institution and where they examined the concepts of trust and transparency as a property of that relationship.

Table 2.2: List of references outside the field

Field of study	References
Non-information system	[97] [85, 96, 98-109]

The reported factors of trust and transparency were only the common factors between literatures, where the reason is to see the commonality across disciplines. [103] [102] stated that transparency is known for its role in increasing and generating trust with public institutions. However, this is not always the case, as it is dependent on the variables moderating the effect between trust and transparency. In their case study they have proved that the frequency of accessing the information through the usage of social media has an influence on the relationship between trust and transparency. The higher the frequency of use the lower the impact of transparency on trust in the institution.

Two other related papers in management-stakeholders’ literature [85] [108] were selected, both examine in-depth the influence between trust and transparency in the relationship from stakeholder-to-organization. These papers were chosen as both examine information

transparency and its effect on trust. They reported that information disclosure has an impact on trust and transparency, and the factors of transparency, e.g., “openness”. In both [85, 108], found that transparency leads to trust. Further, [108] shows how transparency can prevent trust-crisis between the citizen and the organization where the information provided increases transparency in the system. This relationship has also been examined in other contexts (e.g., governance [29]) and in relationship to other software qualities like privacy and security.

Privacy

As example on privacy, Moreno and Molina [105] analysed the effectiveness of the transparency strategy of a Spanish university. They examine the impact transparency policy has on trust in using the school’s educational services. They explain that when users are exposed to too much information, about public institutions, they tend to harbour negative perception as to their competence.

Security

As Jahansoozi [108] has shown in the case of “organization-stakeholder relationship between oil and gas operators and community members”, “after a crisis, transparency is crucial for rebuilding trust”. When referring to this paper it is important to keep in mind that it deals with stakeholders within themselves not the relationship between stakeholder and a system. The importance of transparency in the organization-stakeholder relationship is also emphasized by Schnackenberg and Tomlinson [85] who maintain that transparency is essential to the trust stakeholders place in organizations.

Between Governments and citizens

According to Cleary and Stokes [110], although trust has not been fully defined from an institutional level, public trust in the government or public institutions can be demonstrated by how much confidence they have in the public institutions to operate in their best interests. Kim et al. [102] in their research considered e-participation in web-based applications provided by the government. E-participation refers to “citizens’ *voluntary* participation and involvement in public administration affairs and public decision making through the use of web-based applications provided by the government”. Trechsel et al.

[111] stated that diverse e-participation applications can serve to enhance transparency of political and administrative processes. Creating new platforms of information, can also enhance the public's direct involvement and the quality of opinions.

Perceived risk [99] and culture [109] have also been identified as factors that influence the relationship between trust and transparency.

Some research has defined the concepts in their context while some others have not. For instance [103, 109] both define trust and transparency in their context of examination. Both papers reported on the moderating variables effects the relationship in their context. In contrast, [100, 104] defined neither concepts before studying them in their context domain. [100], however, mentioned that transparency is essential to explain the Blackbox models for decision making from an ethical perspective.

2.2.3 Transparency and Trust Inside the Field

Transparency has been explored in the context of its impact on trust, however, research has not shown a concrete answer on that relationship. Yu, L [84] have examined trust and transparency as non-functional requirements and the likelihood of them influencing each other. However Cysneiros et al. [83] mentions that transparency needs to be managed to ensure trust. Another research done by [112] demonstrates the role of software in fostering trust and transparency. Furthermore, they advise that trust and transparency design should be included in the information systems in e-government.

Users tend to subconsciously trust software based on assumptions, in some cases, and this has an impact on transparency. Trust can be misleading without transparency, for instance, in the case of licenses and agreements of the software and data collection, users tend to trust that the system is going to act in a way that serves them best. In this case trust can cause a false sense of transparency [113].

Table 2.3: List of references inside the field

Field of study	References
Information system: Software system	[24, 25, 27, 32, 80, 82, 83, 112, 114-124]
Information system: AI based Software system (including recommendation and decision support systems)	[91] [25, 125-131]

2.2.4 Trust and Transparency Challenges

As discussed above transparency and trust can be intermingled. There is not enough established research on the relationship between transparency and trust and whether more transparency would increase or decrease trust.

Trust and transparency relationship have been examined in various research areas. Most of these research areas have mainly focused on studying this relationship or reporting on it as a whole without mentioning or investigating further which aspects and properties of these concepts have effect on this relationship. The implementation of the relationship is still very limited. Thus, in this research, a rigorous literature review is needed to describe the relationship in a more systematic way together with its possible features and expected benefits and pitfalls, (see Chapter 4).

2.3 Requirements Engineering

Requirements engineering (RE) frequently determines the success of a project [132]. It is the phase of software engineering where the scope of the system is defined, and the stakeholders' needs, and concerns are iteratively arranged and presented. RE must bring the informal needs of the stakeholders into the formal software behaviour. The process of requirements engineering includes defining and maintaining the features of the software [133] to outline what the system should do rather than how it should be done. This process consists of the following interleaved and iterative core activities: Requirements elicitation, Requirements specification, Requirements verification and validation, Requirements management [134].

Requirements are the alphabetic foundation of software products [135]. They can be functional and non-functional requirements. Functional requirements define the

specifications and features of a system by describing what a system is supposed to do. The non-functional requirements determine the behaviour of a system in a specific condition by describing how the system will respond [136]. These requirements are the consistent trace of the software product. Existing research examined transparency as a non-functional requirement (NFR) in its interrelations with other NFRs and goals [26, 82]

According to Christof Ebert [137] many surveys in industries show that software products do not always deliver their actual commitments. This is caused by multiple reasons as stated in their study. However, most of the reasons are related to requirements engineering (RE), which seems to be the cornerstone where projects either succeed or fail. RE is about finding out and defining what the system should do and specify how the system can be implemented. Klaus Pohl defines “requirements engineering as the process of eliciting individual stakeholder requirements and needs and developing them into detailed, agreed requirements documented and specified in such a way that they can serve as the basis for all other system development activities” [135].

The sudden change of requirements is identified as the key reason why projects are likely to fail. Besides not managing to keep up with changing requirements, the inability to identify the relevant requirements, is another key reason for software project failures [137]. Moreover, since it is in requirements that the core focus, functions, and constraints of the software system-to-be are defined, Requirements Engineering also has a key role to play in developing software that would foster sustainability [36].

Requirements engineering is a process of defining, documenting, and maintaining the features of a project. The process of documenting these features help to define the project for both short and long-term development [138]. There are many activities contributing to this, such as requirement elicitation, evaluation, specification, analysis and evolution of the objectives, functionalities, qualities and constraints to be attained by a software-intensive system within some organisational or physical environment [139]. Using requirements engineering, the software can adequately address issues for which it was designed, which entails requirements engineers properly comprehending and defining the issues [139].

Requirement elicitation provides methods and techniques to gain domain knowledge about the system, so its inputs used in the development stages [140]. Requirement specification

produces the software requirements including functional and non-functional requirements aspects and constrains. One main issue is that some of RE activities are not often picked up by companies because of the time required to get a deep understanding of the system aspects and their constrains and that can be difficult in a fast decision environment [138].

Formal requirements are presented using informal statements such as use cases, user stories. Use cases are used to speak for the functional behaviour of the system, and even though it is not particularly precise, it still is a useful method for representing the functional requirement of the system [140]. User stories on the other hand are used mostly in Agile methods and where it expresses the requirements in a scenario rather than a use case in this form “As a <actor> I want <something> so that <benefit>” [141].

Requirements Engineering in Agile Processes

Recently, many projects relied on agile development, like Scrum, to deliver high quality software rapidly. Unlike the traditional methods in software engineering, using agile requires iterative development [142].

Eberlein and Leite [143] stated that requirements are inadequately tackled in agile development and recommend the adoption of requirement practices such as identifying non-functional requirement and change management. Requirements engineers should therefore identify techniques to apply these practices in their development process.

Yunyun Zhu [142] explains that “Requirements are not detailed enough” in agile development. Not having enough details can be misleading to the developers as to the extent the requirements can be regarded as ready to be implemented. Requirements engineering is been avoided since requires substantial documentation [144]. Research in RE Agile shows that there is a lack of user perspective in the current practices, mainly because of the nature of agile development, where it is about speeding the iterations results [71]. Transparency is seen as a meta-requirement [13, 79]. i.e., it addresses “how requirements can be fulfilled” [13].

Chapter 3

3. Research Methodology

3.1 Introduction

The previous chapters have constituted a robust literature review pertaining to trust and transparency, as these concepts relate to users and software engineering. The relationship between trust and transparency has also been identified along with factors influencing or not influencing it.

This chapter will consider the research methodology and all the research methods to be used throughout this study to address the research questions and achieve the research aim.

To achieve the first objective, a robust review was conducted to ascertain the meaning of trust and transparency and how they relate in the field of software engineering and outside the field. To address the second objective, a systematic literature review was conducted using a concept map and clustering of papers to find the relationship between user trust and transparency, and to identify all factors that have an influence on this relationship. For the third objective, a transparency engineering methodology was developed, which will be applied by requirements engineers and developers to guide them during their system development process. This will help improve user trust by enabling users' trust judgment. Finally, to fulfil the fourth objective, the transparency engineering methodology will be evaluated using a case study to investigate its benefits and limitations in a real-life situation and seek areas for future development.

The research methodology employed throughout this work is qualitative. The methods to be used include a systematic literature review (using concept map and clusters) and a case study design (using semi-structured interviews, pre and postquestionnaire, and focus groups).

3.1.1 Qualitative Research Design (in this thesis) and Research Methods

According to Chandra and Hareendan [145], qualitative research is said to be inductive, whereby an in-depth analysis of case studies or events is studied to investigate a situation or phenomenon. Qualitative research has advantages as well as disadvantages as shown in table below.

Table 3.1: Advantages and disadvantages of qualitative research [145].

Advantages	Disadvantages
Provides detailed perspectives of a few people	Has limited generalizability
Captures the voices of participants	Gives only soft data (not hard data, such as numbers)
Allows the participant's experience to be comprehended in context	Studies few people
Is centred on the participant's views, not of the researchers	Is highly subjective
Appeals to people's enjoyment of stories	Reduces the use of the researcher's capability due to reliance on participants

The research design employed in this study follows an interpretivist's paradigm and uses a qualitative research methodology. Qualitative research methods utilized in this study include systematic literature review, case study, focus groups, interviews, etc.

Figure 3.1: Qualitative research design, provides a pictorial representation of the research methods employed throughout this study.

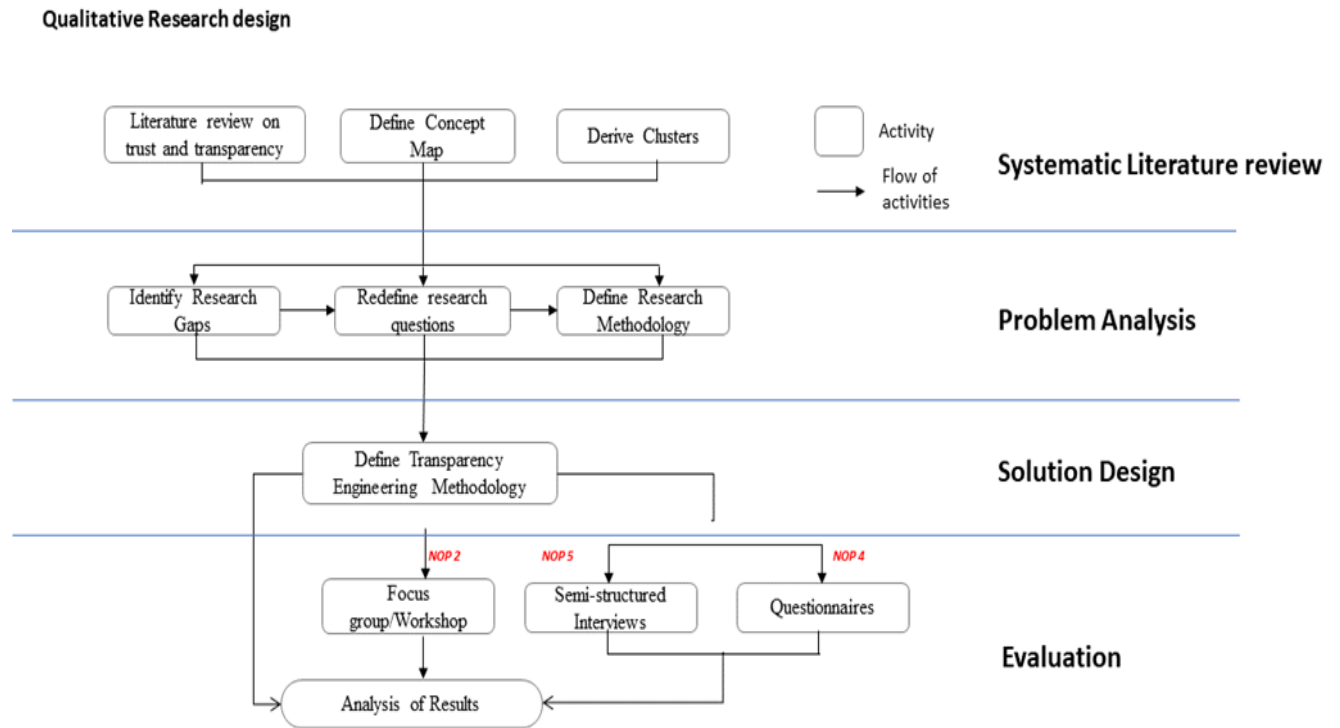


Figure 3.1: Qualitative research design in this thesis [NOP= number of participants]

3.1.2 Systematic Literature Review

A systematic literature review was conducted using a concept map and clusters to find the relationship between user trust and transparency and to identify the factors that have an influence on this relationship.

A classification table (see Table 4.1: Cluster) was also used to group authors in the field who have been reviewed for this study. Papers that presented a concrete contribution to their research area were focused on instead of the generic discussion keywords search.

In order to find relevant literature, Google Scholar was used as the main search engine for literature published at a variety of venues and outlets. Search words which were used in google.scholar.com were: Software, Trust, Transparency, Model, Information system. Additionally, the ACM portal was used for papers in informatics. “Trust and transparency” were added to the search terms used on portal.acm.org. papers and surveys.

Concept Map and Clusters

To structure and represent the fundamental knowledge derived from variety of sources a concept map can be created [33]. The concept maps are used to facilitate the learning research, evaluation, data analysis, and identify new research pathways. It also constructs ideas into a hierarchical structure of concepts [146]. Concepts are the key areas of knowledge in a research area combined to create a plan or identify new pathways.

In this study, synthesis techniques were utilised to help in summarizing and structuring the reviewed papers, resulting in content-related clustering as the guiding principle [147]. This clustering, and the data extraction process it involves, helped in summarizing the content from the results coming from the literature and go beyond a subjective review of the literature to assists in building a conceptual model [34].

The systematic review started with clustering initial papers. These example papers helped to scope the research area as reflected on later in the literature review. To structure and represent the fundamental knowledge as well as summarize and synthesize the information obtained from a variety of sources, a concept map is constructed. A concept map is a useful tool in recognizing and understanding the theories and concepts, along with the connections and relationships between them [33].

This helped to develop the transparency engineering methodology, which has been explained in detail in (see Chapter 5) of this thesis.

3.1.3 Case Study Design

A case study is an in-depth investigation into a case e.g. a person, place, an organization, a situation, etc. [148]. Similarly, Kumar [149] explains that using a case study aids in an in-depth investigation and enables the researcher to collect more detailed information.

A case study was used to evaluate the transparency engineering methodology (TEM), that was developed in the course of this Ph.D. project. A case study with Spirit Healthcare was conducted. This was done by running the TEM on one of their software projects and applying examples from other applications. Spirit Healthcare is a health organisation with a small technology branch that handles sensitive data. The participants were all team members, and include domain experts, senior developers, and developers. They were all

informed that their participation in the evaluation was voluntary. The company was chosen through the following process: an email had been sent out to several companies in health domain which is a of particular interest due to the critical nature of its data collection. Spirit Healthcare is a very successful company, currently in the top 50 fastest growing companies in the United Kingdom, UK. The evaluation of the TEM uses qualitative approaches, including the following instruments: pre- and post-questionnaire, focus groups, and semi-structured interviews.

Pre-questionnaire

A questionnaire can be described as a compilation of written questions used for gathering information. The questionnaire was distributed online due to the current COVID-19 pandemic to avoid face to face interactions. Questionnaires have the advantage of been less time consuming [150].

A prequestionnaire with open-ended questions was utilized during the evaluation of the TEM in order to elicit in-depth responses from the participants. Open-ended questions proved effective as they do not put constraints on the participants and allows them to freely give their opinions. These responses then helped to design and structure some questions for the semi-structured interview used afterwards.

Semi-Structured Interview

Interviews are a qualitative data collection method, involving cross examination using specific questions [145]. They enable the researcher to get more accurate and in-depth data from participants. Interviews however have the disadvantage of being time consuming.

When planning or conducting interviews, there are certain advantages and disadvantages to consider. The adaptability of the questions and the amount of in-depth or rich response are considered as advantages of this method. On the other hand, interviews require time and there is also the possibility of biased responses from the interviewees [150].

According to Kumar [149] there are two main types of interviews: structured and unstructured. The unstructured is evolutionary in its structure, has flexible contents and involves open-ended questions where participants have no constraint over the information they give. The structured interview, in contrast, has a pre-determined structure, usually

rigid with closed-ended questions to guide both the researcher and participants. This research however employed the use of semi-structured interviews as they adopt both open-ended and theoretical driven questions in order to garner in-depth data reflecting the participant's experience whilst drawing them fully into the topic that is being studied [151]. They have some structure in their contents, wording and order of questions while also giving the participants some flexibility and leeway in their responses. It also reduces the amount of time the interview would take.

Focus Group

Stopher describes focus groups as a qualitative data collection method used to gather in-depth information from a carefully chosen group of participants to discuss a specific topic. It can also be used to assist the design and evaluation of surveys [152].

Focus group was employed in this study throughout online sessions. During these sessions further discussions were set to elaborate on the issues that were raised during the interviews or filling out the questionnaires. Additionally, these sessions allowed a free space for brainstorming on the problem area as well as the developed methodology with the domain experts team.

3.2 Sampling Strategy and Size

The sampling strategy employed for evaluating the TEM was purposive sampling. The Spirit Hub application is used as the case study, to test the TEM in a real-life situation. The TEM was applied to other company applications to demonstrate how TEM can be used.

The group of software engineering experts of the company was made up of developers and domain experts, who were chosen to participate in the focus group and interviews. In total, there were 5 participants: 1 domain expert, 1 domain expert/project manager and three developers.

3.3 Ethical Considerations

There are ethical considerations used in the research, as the evaluation involved data collection from the participants. An ethical approval was issued for that purpose. Participants were notified prior to the study that their responses will be recorded. Data

collected will be codified and the participants' identity will remain anonymous unless they explicitly consent for it to be used in any reports from this study. Furthermore, participants will be notified that audio will be recorded. Recordings will be deleted once they have been transcribed and unless experts have consented to their identity to be used, transcripts will be anonymised. Prior to any online interview, interviewees are provided with a copy of the information sheet and will have adequate time to decide whether they wish to or have the capacity to take part. Moreover, each participant is given the opportunity to give their consent or not for the study without the intervention of any form of coercion or undue influence on the subject's decision. Interviews will focus on developers and requirement specialist and domain expert perspectives on giving feedback the Transparency Engineering Methodology not expected to cover any contentious or sensitive personal issues.

Chapter 4

4. Systematic Literature Review

4.1 Overview

This chapter reviews the literature on trust and transparency to help engineer transparency requirements that aim to enhance trust in a software. Trust and transparency have been examined in software systems in a variety of contexts [125, 128] (see also Section 2.2.3). As highlighted in previous chapters, the relationship between trust, transparency is still not fully understood in the research community. For this research, to achieve its aim of enhancing trust through transparency, it is important to understand their relationship. Therefore, an in-depth literature review will be conducted in this chapter to further delineate the matter. A total of forty-five articles were chosen and reviewed. Using a concept map and clusters, the current state-of-the-art is derived from the literature and provides a summary of the relationship between trust and transparency in software systems from different domains and areas.

4.2 Identification and Clustering of the Literature

4.2.1 Definition of Search Criteria

In this research, a literature review procedure was adopted to select the primary articles. The major databases used were IEEE Explorer, Springer Link, ACM digital library, Science Direct.

Keywords Search

In order to find relevant literature, Google Scholar was used as the primary search engine for literature published at the related venues. Search words which were used in google.scholar.com include: Software, Trust, Transparency, Information system; combined with AND/OR or both e.g., “trust AND transparency OR software” “trust AND transparency OR information system”. “User” was removed from the search terms as the user can be a patient or a customer, then searching for only “user” would be restricting. Additionally, the ACM portal was used for papers in informatics field. “Trust and

transparency” were added to the search terms used on portal.acm.org. Fifty-two papers were reviewed and the selection process for the clusters is explained in this Section.

Backward and Forward Chaining

Backward and forward chaining was applied by following the references in the articles identified via the initial search. Thus 8 papers were added, for a total of 60.

Definition of Selection Criteria

As trust is a wide area of research, covering trust of humans in other humans and systems, the search focus was narrowed down to the related research of interest. Starting out with scanning where trust and transparency have been studied, participants of trust relations were identified. Trust and transparency have been studied in various areas (Chapter 2, Sections 2.2.2 and 2.2.3). Given the Informatics background of this project the trust relationship we are referring to is between human and the software. Content scanning was necessary to identify the relevance of the papers found. The Titles, Abstracts, and conclusions, in this order, were read. The full title was used to identify the research area of the publication and as first criterion to filter out papers not matching the research.

4.2.2 Filter Criteria

The number of the papers reported in the clusters was reduced to forty-five based on the following Inclusion and Exclusion Criteria.

1. Relevance: Papers relating to the research question of this thesis, What is the relationship between trust and transparency? (See Section 1.3).
2. Recency: Papers published in the period 2005-2020.
3. Full Access: To include the paper, the contents of the paper have to be accessible in full text.
4. Duplicated papers: Repeated papers have been excluded which have been published in an extended or complete version and considered the more elaborate version.
5. Language and peer review: Selection of papers was restricted to those that were written in English and published in peer-reviewed journals and conferences.

4.3 Synthesizing using Concept Map and Clusters

To conduct this literature review, a concept map and clustering method were used as stated in the Research Methodology (see Section 3.1.2). In addition, before finalizing the text-

based clustering, a concept map was created based on the themes identified and used as concepts which grew with the addition of each paper. The concept map focuses on the relationship between trust and transparency in the identified literature [33]. These are research methods that help to summarize and conceptually organize the reviewed papers [95].

4.3.1 Concept Map

The goal of the concept map is to visually represent the relationship between concepts, and properties that have been mentioned in the literature on trust and transparency. The concept map helps to recognize the most important aspects that influence the relationship between trust and transparency. This was a useful reference point when constructing the clusters.

The concept map started out from the two main concepts that were of interest to this research in the centre of the map: Trust and Transparency. From these, concepts were added as soon as they were identified in any of the selected papers. For this step, note taking was of importance while reading the papers, as it helped to keep track of the main concepts and shaped the major findings over time and during the process. Initial nodes developed into clusters when more papers added to the concepts. As a result, some of the nodes were translated to clusters, some were also grouped together.

The concept map, shown in Figure 4.1: Concept Map, consists of the concepts and their relationships with each other. The bubbles are colour coded: The green bubbles are the main concepts; the purple bubbles are the perspectives, and the light green bubbles are the clusters. Concepts are the main key words which are found in the literature. Clusters and perspectives are explained below (see Section 4.3.2). The Figure 4.1: Concept Map shows a visual depiction of how the different concepts relate to each other. Some concepts are tightly bound to one another whilst some are directly derived from others. The map depicts this complex relationship and allow for clear visualisation of the common concepts.

4.3.2 The Process of Clustering and Results

The concept map was used to define and edit the clusters iteratively. After classifying the main themes from different perspectives, a cluster table was created to easily derive and analyse information, adding content to the visual depiction of the concept map. The perspectives take on various values across the reviewed papers, because different results may be obtained. Papers with the same value form a cluster. Thus, the name of the clusters is used to refer to the values of the perspectives. For instance, under perspective 5, there are the values " Trust and acceptance of the system", " Increasing user's privacy-awareness" etc. Papers presenting the same value " Trust and acceptance of the system " are clustered together. The number of papers does not imply minor or major importance of a cluster, in fact it only indicates if that value has been discussed in the literature.

When constructing the clusters, the review process started out with four initial papers. These papers show different type of the relationship. The papers draw the first list of the trust-transparency relationship forming the initial cases of this research, which will be reflected on later in this literature review. These four papers are [27] relationship unclear, [32] transparency leading to trust, [128] transparency affects trust based on variables moderating the effect, [93] high transparency erodes trust. This constituted the groundwork that was expanded with additional themes as the review proceeded. As a result, they have been analysed in more detail to build the initial groundwork to be expanded with more papers when additional themes were identified.

The perspectives show a strong association of the trust transparency relationship presented in the papers (from a particular viewpoint). However, there are various values (i.e., where the papers examine same topics and concepts but show different results) for these perspectives across the reviewed papers. These perspectives cannot be definitively fixed under a single value and thus the papers which have the same values align are clustered together (form a cluster). Hence, we use the clusters' name to refer to the perspectives value. The resulting clusters aim to assist in understanding the relationship between trust and transparency. The number of papers in each cluster has no indication on efficacy of the cluster, it is only to indicate if that perspectives value existing and have been discussed in the literature.

The included papers must define explicitly their view on the relationship between trust and transparency that have been experimented using a case study or have reported on a specific case study. This was done throughout a reviewing process in order to fully examine all factors influencing the relationship. Based on the main perspectives identified, the paper was then assigned to the appropriate clusters. It progressed by completely adding the concepts of one paper before moving on to reading the next one, adding its concepts.

Table 4.1: Clusters

PERSPECTIVES	CLUSTERS	KEY REFS
1. Influence of transparency on trust	Transparency leading to trust	[91, 124, 125]
	Transparency affects trust based on variables moderating the effect	[128]
	Transparency does not improve trust	[24]
	High transparency erodes trust	[93]
	Relationship unclear / not understood yet	[27, 126]
	Transparency's role of managing trust in decision making	[153]
	Transparency rebuilds trust	[28]
2.Type of system	Non-AI-based Software	[24, 25, 27, 32, 80, 82, 112, 114-124]
	AI-based software	[25, 91, 125-131]
3. Definitions and "anti-definitions"	Definitions of trust:	[24, 27, 116, 117]
	Definitions of transparency	[83, 125]
	Neither trust nor transparency are defined	[32, 114, 126, 128]
4. Application domain (areas)	Business	[27, 32]
	Education (peer assessment)	[24]
	Art	[25]
	Health	[32, 154]
	Public services and Governance	[28, 29, 80]
	Cloud Computing	[116]

5. Goal of transparency in relation to trust and other system qualities	Trust and acceptance of the system	[25]
	Increasing user's privacy-awareness	[6]
	Data protection	[22]
	Rebuild or enhance legitimacy and trust	[22, 28]
	Predictability, and expectations	[24, 91]
6. Existing Requirement Specifications Methods	i* models	[82, 84]
	Design patterns	[14, 22]
	Modelling languages	[13]
	Taxonomy of transparency requirements	[6]
7. Facets and Attributes of transparency	Level of disclosure	[25, 28]
	Level of abstraction of content description	[32]
	Accessibility, understandability, and relevance. (attributes)	[3, 25]
	Accessibility, usability, informativeness, understandability and auditability (NFR-soft goals)	[80]
	Meaningfulness, stakeholders, usefulness, and information quality (facets)	[26]
	Procedural information	[24]
8. Factors influences trust and transparency	User expectation violation	[24, 25, 32, 123, 129]
	Quality of information and information disclosure	[22, 26, 155]
	Dependency relation between the user and the system: voluntary relation, vulnerability relation	[24, 25, 27]
	Sensitivity of the decision making	[32]
	Culture, user role with the systems	[91, 156]
	User perceived risk	[25, 27]
	User confidence with the system	[91, 156]
	Assumptions	[113]

A total of eight perspectives were defined from the literature survey, then based on the process mentioned above, the retrieved information was categorized into clusters. These perspectives and their clusters are briefly represented in Table 4.1: Clusters and each perspective and their respective clusters is explained below.

Perspective 1: Influence of trust and transparency on each other

This perspective represents the direction of the relationship between transparency and trust, that can be positive (e.g., “leads to”), negative (e.g., “erodes”) or neutral (“unclear”). However, only a few papers have reported how particular variables influence such a relationship. The researcher in [128] reported that transparency has an effect on the user confidence. It can increase confidence when transparency bridges the gap between a users’ mental model of how the system works and the actual functioning system. However, transparency can also undermine confidence when users have a correct mental model of the system and transparency causes the user to be focused on the software’s flaws. Thus, transparency is reliant on the user’s perception of the software. Kizilcec [24] further reported that *expectation violation* measured by the user’s confidence with the software affects the relationship. However, other researchers have reported that the relationship is unclear and not well understood based on their findings. Mercuri [27] explains that although there is a clear relationship between trust and transparency, this relationship can be difficult to understand as there are more trust definitions than actual case studies from previous research. They have reported that in the context of security assurances, transparency and trust relationship is not well understood. They therefore recommend a “trust-centric approach” to achieving transparency in order to boost users’ confidence in a system. Moreover, another paper [25] in the art domain explains that low risks trust is not improved by transparency. However, in other situations that have a higher risk, for example in an automated system where a user cannot directly judge the underlying processes, transparency have a great influence on trust. Cramer et al. [25] also mentioned another element to consider, that the effect of transparency is based on the user expectations, i.e. transparency can be required in the first encounter and when unexpected outcomes are presented. Chien et al. [91] stated that in all cultures the lack of transparency will reduce trust because of the lack of “evidence about system operation and performance”. While another research result shows that “while medium transparency increased trust significantly, high transparency eroded it completely” [93]. Transparency’s role of managing trust in decision making so users can build appropriate trust with the system i.e. users do not over-trust or under-trust the system considering its explanations [153]. As a

result, transparency generally increases trust, however it needs to provide relevant information to users.

Perspective 2: Type of system

There are two broad software-based system types in this literature: AI and non-AI based. There is different emphasis on the factors that impact the relationship between trust and transparency, which can be noticed in the examples provided in the related Sections (see Perspective 5, 7 and 8). Due to the frequent opaqueness of AI systems, transparency can be a crucial factor in trusting their results [127]. AI systems are often a black box with a set of inputs and an output and delineating the complexity of the processes is difficult. Indeed, there is whole areas of research devoted to this objective [127]. The most important aspect of trust in an AI system relates to the quality of output, if the expected result from a system is received then it is trusted, but this may not be reflected in how the result is determined. For example, art recommendation systems are judged based on the quality of their recommendation [25]. Culture is another factor of importance when examining trust and transparency in automation systems [91].

Other non-AI systems generally tend to rely on more human interaction in progressive steps, each of which invokes a trust relationship [16]. As a result, the factors that influence trust in such a system are quite different and more tangible, with things like quality of information and contextual relevance being more important [155]. However, there are some crucial similarities, they still have to be informative, relevant, accessible and understandable.

Perspective 3: Definitions and “anti-definitions”

Some of the reviewed papers have experimented with the relationship between trust and transparency and posted their results without defining one or neither notions of trust nor transparency in their application domain [32, 114, 126, 128]. Defining these notions sets up the research’s initial focus with the corresponding attributes and facets and should act as a starting point. The notion of transparency can carry several inconsistent meanings [157] in the literature (see Section 2.2.3). It is interesting to see a counter example where an anti-definition of trust and transparency is provided. Mercuri [27] did not define trust and transparency directly but stated that, for example, trust and transparency are “not

necessarily synonymous with full exposure”. Having a definitive definition of trust and transparency has a direct impact on the paper’s relation and interpretation of trust and transparencies constituent parts, and thus have been clustered. Some papers have no definition leaving interpretation of their results and conclusions more open, and more difficult to relate with other works.

Perspective 4: Application domain (areas)

Areas of a software applications define a different trust transparency relationship, with particular factors and facets acting as the predominant influence on their relationship. Moreover, the goal of transparency in a domain can vary from one to another. The domains that have been reported on in the literature are as diverse as Health care, Education, Art, Cloud computing etc. Different types of transparency are more pertinent in particular fields, for example health care software has sensitive data with privacy concerns [32]. Ethical principles are enabled with dependence and regulation information transparency [78] see Section 4.3.3. As this reliance increase as the failure of the software becomes more significant [43]. Whereas the Art domain requires transparency which is more informative without much risk or other dependencies [25]. In this way a domain can, much like in (see Perspective 2 and 3) have an overarching impact on the relationship of trust and transparency and directly affect the relationship between the other factors and facets that make up the trust transparency relationship.

Perspective 5: Goal of transparency in relation to trust and other system qualities

Transparency goals in relation to another system quality were diverse in the literature, however, the majority of the research examined transparency in relation to privacy, and/or trust. In the case of trust goals, the literature reported how transparency can increase trust and user acceptance of the system [25] and how to rebuild or enhance legitimacy and trust [22, 28]. Other goals reported were the role of transparency in increasing user’s privacy-awareness, expectations, and user data protection. The authors in [25] shows that transparency increased perceived understanding in the correlated with perceived competence, therefore increased trust and acceptance of the system. Transparency has

frequently and effectively been used as a tool to achieve particular high-level goals of systems.

In the last few years, new data protection regimes have been put in place that reformed the notion of transparency in supporting user-centred approaches [7]. Transparency should not only care about the content of the information provided to the user but also the quality and the understandability of this information. Rossi et al. [22] stated that the data controllers must implement transparency in their data processes. They also suggested that this transparency should have a positive relationship with trust by enhancing user's trust. Thus, it has been shown transparent about data usage can be used to align with data protect regulation such as GDPR.

Perspective 6: Existing requirements specifications methods

There are few methods to generate requirements specifications for trust and/or transparency. Design patterns are one of them. Hoffmann [14] used requirements patterns to created high-level goals of trust a system should justify (e.g.,) along with a related forces to drive the domain expert into making decision with trust as a greater focus. These pattens were derived from trust antecedents and factors that are collected from systematic literature review of the research performed on trust in automation. Researchers stated that transparency/trust are non-functional requirements and therefore, i* models can manage transparency requirements [82, 84]. Meis et al. created a taxonomy of transparency, where the components of ISO/IEC 29100:2011 standard were structured in an extensible meta model to understand how the different types of transparency are related in order to address privacy goals transparency [6]. Hosseini [26] argued that i* modelling is not the definite answer to transparency, as they concluded in their research that to manage transparency “more effectively” there are other required processes and actions needed. Therefore, they used modelling languages to engineer transparency in business information systems, “enriches a goal model with additional transparency dimensions[...] transparency is information oriented” [13]. Similarly, this research agrees with Hosseini's argument that, while i* is goal oriented and focuses on what the systems goals should be, transparency is information oriented. However, to address transparency with different system aspects, this cannot be done by only addressing the flow of information in the system. Therefore, there

is a need to develop a methodology that can help the requirements specialist to address transparency.

Perspective 7: Facets and attributes of transparency

In reviewing the existing literature, different facets and attributes of transparency were stated. Leite defined transparency SIG, which is a graph of non-functional requirements consisting of 33 soft goals. The five main soft goals which have direct influence on transparency are accessibility, usability, informativeness, understandability and auditability. Another research has divided transparency into the facets of stakeholders, meaningfulness, usefulness, and information quality describing that in order for the system to be transparent the information need to be understandable by the stakeholders and the intentions behind the system actions need to be clear. It can be noticed that the main facets of transparency can be various from one application domain to another. For instance the level of information disclosed are reported as important facets in the context of government [25, 28] whilst procedural information is important in the context of algorithmic transparency [24]. Yu-Cheng definition of transparency “rests on three attributes: accessibility, understandability, and relevance”. Yu-Cheng stated that these attributes are necessary for the stakeholders to achieve their said goals with the system [3, 25]. The common attributes of transparency that can be seen in above definitions are accessibility, understandability and relevance are thus key to successful transparency.

Perspective 8: Factors that influence trust and transparency.

User Expectation Violation: Transparency can either encourage or discourage users to/from trusting a system [24]. Likewise, violation of expectation can be considered a ‘critical moderator’ of the impact transparency has on user trust. Kizilcec [24] carried out an online field experiment to determine the impact of system transparency design on algorithmic interfaces and the resulting effect on users’ trust. They tested three levels of system transparency (low, medium, and high) in a peer assessment for an online course and reported that users who had their expectations violated had less trust in the system. However, when too much information was given or high transparency design was used, their trust reduced further. When users suspect their expectation is violated, they are said to pay more attention to the information provided to discover the reason why their expectation

did not match the system output. Kizilcec [24] therefore suggests a balance in system transparency design or information given to users (medium transparency). For the methodology to be developed it can be derived from this cluster that transparency need to be designed in a way where it can provide relevant information to the user.

Similarly, Vaccaro et al. [123] reported an increase in user satisfaction when news feed control settings on social media platforms were present, whether they performed any actual algorithmic functions or not. According to Springer et al. [130], users' daily decisions, when interacting with software, have constantly been influenced by hidden algorithms in apps or software. Users trust can also be increased if they feel the system or algorithm is "intelligent" [130]. Likewise, textual or procedural explanations given to users when using a system tends to foster their trust in that system [24, 130]. Both authors stated that understanding how manipulation of algorithm can impact user expectation or satisfaction is thus important for designers in creating products that foster user trust.

Dependency Relation with the System: Some of the reviewed papers also noted the type of relationship between end-user and system which explains how much user depends on system. Cramer et al. [25] studied the voluntary relationship between the user and the system as one of the factors and found that it has an impact on transparency to trust relationship in their web-based application. In case of computer security, Mercuri et al. [27] suggested trust-centric approach ("as opposed to a vulnerability-based one") to help in achieving transparency that assures the needed confidence and reduces the perceived risk. Kizilcec et al. [24] also stated that the dependency relationship between the user and the system has a major impact on trust, as they adopted trust definition is "of risk that one's vulnerabilities will not be exploited".

User Confidence: In the case of AI software systems, several explanations have been examined in terms of transparency. According to Okamura et al. [153] explainability needs to be designed using user-centric principles, so users can build appropriate trust with the system. This means that users do not over-trust or under-trust the system as a result of its explanations. Thus, these explanations need to provoke the user's judgment on the system's processes, decisions, and recommendations. The main goal of explanations is to justify the reasons for action. The point is to make these actions logical to the user, i.e., by providing

information on “why” an action is occurring. The action can be a decision or a recommendation that has been shown to the user from the system. Recently the explanations also include “How” indeed there is a direct relationship by explain how data is going to be used also explains why. The user might need to know information about some concepts in order to understand how the system has reached their decisions. Thus, the main aim of these explanations is to ensure user confidence with the system’s decisions and recommendations.

Sensitivity of Decision Making: Trust and transparency are essential factors known to influence the user’s decisions when interacting with the software [156]. In order to make informed decisions it is necessary to elaborate the relevant and right amount of information. Transparency is seen as non-functional requirements that evaluates if the system has met this above criterion of providing the relevant information or not [32]. However, without considering the design of this information to improve user trust with the system, this transparency could potentially be implemented in a way that could cause manipulation of decision making. It is thus vital that transparency is passive in the sense that it does not influence a user to make a decision one way or another but provides the relevant information for them to autonomously make a decision.

Assumptions: Mercuri et al. [27] reported on several case studies where assumptions (assuming reasonable understanding) are the causes of a “false sense” of trust, where transparency was missing. For instance, in case of licenses and data collection, trust without transparency can be misleading. Sink et al. [113] demonstrate cases where having both trust and transparency can remove the “veil”. They explained that transparency do so by allowing users to “see behind the corporate” this can “extend them your trust, making it easier for them to trust you in return”.

Perceived Risk: One of factors that have a major role when examining the relationship between trust and transparency is the perceived risk of an action. Mercuri et al. [27] mentioned that achieving the transparency that reduces the user’s perceived risk is important. In order achieve this they have suggested trust-centric approach. Either a risk factor needs to be considered when examining the relationship or a successfully overcoming risk factor indicates the effectiveness of the relationship between trust and transparency. Cramer et al. [25] stated that in case of art domain with general low risks,

trust is not improved by transparency. However, in the situation when the system type is a recommendation system with high risk, where the results cannot be judged by the user directly, transparency has a great influence on trust.

Culture: User's culture has been examined as one of the factors that impact their trust with the system. The effects of transparency are also strongly influenced by it and it is tightly bound with user's expectation. Much of a user's expectations are derived from culture, some cultures encourage caution when using software, whilst others completely trust the technologies they are using. Study results found that systems may have to be adapted to be transportable to different cultures. Applications developed in western cultures would need to create additional mechanism, eliciting transparency, to be suitable for eastern cultures [91]. Companies themselves also have identifiable cultures and users come to expect certain standards and design practices. These must be adhered to when implementing transparency [91].

Information Quality and Information Disclosure: Information can be provided to assure the user that the system is secure, reliable, and safe. At the same time, however, it can also lead to problems like misleading, manipulating, or overwhelming the user [25]. Thus, the amount of information given is not an accurate measure on the system's trustworthiness or transparency. An abundance of information can make it difficult for the user to find the content of any interest or importance. Therefore, it all depends on the understandability and relevance of content to the user. Luiz et al. [83] also reported that even when the software is fully disclosed it might not be considered "transparent software" as shown in the video recording software TiVo 2004. According to [91], inadequate system transparency will reduce users' trust as a result of the lack of information or explanation about how a system operates. Following law and legislations can help, however these regulations are not always applied in high data sensitivity domains like health domain and even when related information is disclosed it is not necessarily disclosed in "comprehensible and verifiable manner" [155]. Information disclosure can be unfavourable for institutions as it can reveal sensitive or confidential information compromising individuals or the institution [28]. However, transparency on the personal data is an enabler for user trust [5].

Moreover, the use of the information and where it is placed in the system is an essential element in transparent systems. Information labelled as vital [126] further explains that

including clear personalization features and signals on user interface design within apps will increase user satisfaction. E. A. Lind [28], therefore, recommends that software developers should adopt a transparent approach in their designs by making system or app information available to users where they can easily access it. This will also improve their trust and perceived risks related to privacy.

4.3.3 Trust and Transparency Groups

Groups are collections of concepts around transparency which differ from the above clusters due to being more generic concepts widely discussed in the field. As a result, they cannot be clustered due to their more ubiquitous nature and that they can overlap. While reviewing the literature, the frequently examined system qualities having an impact on the relationship between trust and transparency have been grouped as follows.

System Qualities

According to Yu and Leite, trust and transparency are non-functional requirements and therefore they are “likely” to influence each other [82, 84]. This means that they are also likely to be influenced by and have an influence on other non-functional system requirements. These other requirements can be used to better understand user trust. Based on the goals of the user, they will be refined further. A system then must fulfil them to make the user trust in its capability to achieve their needs and expectations. However, the software having certain qualities alone is not enough, the user also needs to be able to recognize these qualities. This is one of the roles of transparency; by revealing the correct amount of information about qualities that are important for the user, their trust judgement is improved.

To study the relationship between trust and transparency, in this literature review the software qualities that have been highly examined in the reviewed papers have been identified. User experience, usability, reliability, and privacy are the qualities that have been reported the most frequently and are influenced by the positive relationship between trust and transparency. Security has an effect and is also affected by both positive and negative relationship. Table (4.2) shows these high impact system qualities.

Table 4.2: System qualities

User experience and Usability	[158] [14, 159]
Security	[27]
Privacy	[160, 161] [162]
Reliability	[163] [25]

User Experience and Usability: While researching transparency and trust in software systems, a relationship with usability as well as a user experience has been observed. Hoffmann et al mentioned in their findings that there is an overlap between trust patterns including transparency pattern, with usability and ease of use [14].

User experience can be affected by trust and transparency [159]. Transparency can improve user experience by improving their mental model [158]. A mental model is a psychological representation, internal conceptualization of processes or system features used to highlight the most important features used during their interaction with the system [164, 165]. It is a simplified version of the system, the idea that the user has about how the system works. It represents user’s expectations during the user interaction with the system. However, the user mental model can be inaccurate and that affects their experience with the system [164] and can also affect their trust into the system directly. An improved mental model contributes to enhancing the overall trust with the system and improves user experience [166].

Security: Security has been shown to impact the relationship between users and a system. If a system has been compromised for example being hacked, Jahansoozi [108] explains that transparency is crucial in order to reinforce organization-stakeholder relationship. This security relationship extends outside of the software field (see Section 2.2.2). Trust can be created through higher security causing lack of transparency in case of cryptography [27]. Mercuri et al. [27] stated that “transparency is deemed inversely proportional to trust”. This approach relies on the belief that transparency on security can increase the perceived complexity of the system for the user where only few people have the knowledge to truly

understand. The level of revealed information can therefore differ in transparent systems based on the targeted users perceived complexity of the that system.

Reliability: Wright et al [163] research observing human-robot interaction reports that when there were robot errors, participants recorded reduced confidence in the reliability of the robots. They, therefore, explain that system reliability can have more impact on users' perception of trust than system transparency. Thus, some reliability has to be achieved before trust can be influenced through transparency. Cramer et al [25] explains the importance for users to be able to rely on systems, in adapting their feedbacks and correctly meet their expectation.

Privacy: The concepts of transparency and privacy are becoming more and more important due to the ever-increasing human reliance on technology. Users' trust is said to be improved if they are aware of how data items are used or managed by a system [160]. Bertino et al. [160] also assert that although it is essential to provide data transparency, it should not violate user's privacy or security requirements. Zinovatna et al. [161] mentioned several cases where lack of transparency can cause a privacy violation. They closely examined how privacy and transparency are connected and positively affect each other as well as the other system qualities like trust.

Hedbom [162] sees transparency as a tool to enhance privacy, for the data subject has control on their personal data. Trust, in their research context, is the level of assurance that the data controller will behave as expected. The transparency that shall increase trust and not false trust should accurately show how the system is dealing with the user's personal data. Thus, transparency must be used as tool to provide information to the user personal data and reduce the instances of false trust.

Negative Patterns of Transparency

Negative effects of transparency on the user are reported from the reviewed literature in Table 4.3. This shows some dark patterns of transparency, for instance manipulation effects that can occur through these actions including unfavourable default, limited view, and competing elements. Manipulation, in the case of intelligent based systems, can be caused by limiting the view of relevant content thus distorting a users' mental model [24]. Further an "unfavourable default" where the system preselects an option during the user interaction

with it [167]. Similar effect, according to Chromik et al., can be used to induce unwanted data exposure such as “requiring the user to perform a certain action to access [...] certain functionality”. As well as nagging by “redirection of expected functionality that may persist over one or more interactions” [167]. Craft et al. [168] stated that transparency can do opposite effect by confusing the users with so much information that it becomes impossible for them to distinguish between the relevant information from the noise.

Table 4.3: Dark patterns of transparency

Manipulation	[24, 91, 167]
Explanation Marketing	[167]
Forced Data Exposure and nagging	[167]
Information Overload	[167, 168]

Transparency Categories (subjects of transparency)

Table 4.4: Transparency shows transparency categories (subjects of transparency). Information transparency can be “enabling” or “impairing”. The authors in [78] state that the ethical effect of transparency relies on the ethical principles of dependence and regulation. It is important to be transparent when particular information is required in order to successfully perform an action or understand the risks of that action, i.e., there is a dependency on being informed. Further the ethical effects of transparency can be affected by regulation where information is restricted or controlled for privacy or copyright, etc.

Leite et al. [82] explains that there are two categories of transparency for a software system, information transparency and process transparency. They stated that the software systems are only transparent if its internal functionalities and information are transparent. Additionally, Bannister et al. define three categories of transparency requirements process, data, and policy transparency. Each category should deal with questions and answer them [169]. Data transparency should answer “questions relating to data, content, and information” these questions answers “what information is need and who are the stakeholders”, while process transparency answers the “questions relating to processes, behaviours and interactions”. Finally, the policy transparency answer “questions relating to

intentions, policies and decision making [...] why an action is performed” [26]. These questions primarily answer how something is performed”. According to Cappelli et al. process transparency is a “challenge by itself” and become more challenging to address with more processes being implemented by the software [170]. Additionally, Hosseini et al. and Bannister et al. pointed out that these categories “requires” each other i.e. process transparency requires data transparency and policy transparency requires process and data transparency [26, 169].

Table 4.4: Transparency categories

Process transparency	[26, 80, 82, 170]
Data transparency	[26, 169]
Policy transparency	[26, 169]
Information transparency	[25, 80, 82]

The papers were also grouped based on the targeted audience of trust and transparency. Table 4.5 shows various entities who demand trust and transparency. Some research shows that different audience can require different ways of achieving transparency. Leite et al. [80] mentions that in dealing with transparency, citizens as customers require a “different sort of models and strategies than is needed for just dealing with users”.

Table 4.5: Who demands trust, transparency.

<i>Who demands transparency and/or trust?</i> <ul style="list-style-type: none"> • Customers • Citizen • Patients and doctors • Data subject 	[25, 80, 83]
	[28, 80]
	[32]
	[22]

4.3.4 Findings and Discussion

This chapter has looked at the factors, system qualities and the properties of trust and transparency in software systems. It has drawn on the literature in the area to construct a holistic view of their relationship defining where they correlate closely and discusses the

factors that influence their relationship. Moreover, these factors have been grouped into the overarching categories, software qualities, subjects of transparency, the target audiences, and negative patterns to describe more directly how transparency have an effect on trust. Much of the research describes trust and transparency as high-level goals and requirements, which relies on the discretion of the development team members to make implementation decisions. Here this delineate how and why a more systematic approach to software transparency could be formalized.

Trust and transparency have a complex relationship, where depending on the influencing factors can have both a directly and indirectly proportional relationship. The clustering shows that trust-transparency relationship can be affected by a number of facets such as understandability, relevance, informativeness as well as factors such as the sensitivity of the decision making and reliance on the system. Moreover, cluster items such as application domain area and system type, the goal of trust and transparency for the system and even the definition of trust and transparency all affect the intricate relationship. It is thus clear that any methodology that formalises trust and transparency need to be flexible and dynamic within a framework that delivers robust implementations of transparency.

It is significant to note the importance of an applications domain and system type on transparency, which can affect the level of details of required transparency, risk of dependency and the reliance on a system as well as a myriad of other factors described in the clustering. Domain and system type thus overarch a lot of the trust-transparency relationship and have a particular signature of the predominant factors that affect the relationship. However, any systematic methodology must be more tightly bound to the data and procedural in order for it to be universally applicable and be effectively transparent about the inner workings of a system. Transparency to be functional cannot be addressed over the system as a whole and must be addressed over particular system aspect (data and process). As a result, the system must be applicable to a variety of different system processes and data operations but be “aware” of its context and allow for adaptability to provide the right level of transparency for a given domain. Acting on the individual process and operations is crucial as it is this that reveals true purpose and actions of a system.

However, an individual's understanding of computer systems vary significantly, thus it is vital to induce a methodology which maintains relevance and meaningfulness.

There are a number of key factors to consider. More transparency does not directly lead to trust and the relationship more centred around whether transparency can resolve a user expectation violation without inducing a violation by itself. What would be considered a violation varies between application and relies on the software's domain, context, and culture, and will be most understood by the domain expert. The methodology must provide a mechanism to present different levels of information whereby it systematically defines a holistic transparent picture of the system, but the output is scalable to match a user perceived expectation.

Users rely on systems to different degrees and each process has a different degree of dependency upon it and require various levels of confidence. To aid in this, a methodology must elicit high quality relevant information which can only be achieved through a user focus approach. How a system is perceived by a user is important, as a lack of trust can develop when a system does not function in an expected way. By being transparent and displaying the available options open to the user at a given point in the system, many of these potential conflicts can be resolved. Further, at certain process and decisions points within a system, particular information must be provided to enable a user's trust judgement. A transparent system must reveal any dependency and regulation which relate to a process to be successful. A core component of transparency is keeping the user informed, balancing their expectations of a system with reporting critical information that they must know in order to make informed decisions. In this way, trust is generated as a user can act with confidence and certainty.

A further critical consideration when considering trust and transparency is information quality and disclosure. Information can induce confidence and a perception of reliability and safety. However, it is only a perception and can also lead to manipulation, be misleading and even overwhelm the user with information. A fully transparent system does not reflect this research definition of transparency, which provides all *relevant* information. This again denotes the importance of a user-focused approach which focuses on a user's data and the processes they directly induce. A user would be concerned about how data

relating to themselves is being used by a system and who has access to it but may not be concerned about a system's architectural design and database choices. This can only be resolved by centralising the user and their concerns in any systematic transparency methodology. This can then be delineated through the users' data and what are the relevant processes that act on that data.

Transparency and trust affect a number of important system qualities: User experience, Security, Privacy and Reliability. Each of these are crucial in maintaining trust in a system from a user's perspective and can be both undermined and enhanced by transparency. User experience can be enhanced when transparency resolves the gaps in the user's mental model of a system but can be undermined if a user is overloaded with information. Security can be enhanced by warning users of the risk of particular actions or processes, but transparency could reveal system secrets making it vulnerable to hackers. Similarly, privacy can be compromised by transparency by revealing personal information about the user and how they use the system. The effects of reliability on trust can be managed by transparency, by informing the user of any errors and available options to resolve problems but can also degrade trust by informing the user of any small mistake or fault which might not be critical to functionality but adjusts the user's perception of the system and company. A methodology as a result must fit within these system qualities and be sensitive to them. Due to the dynamic aspect of these qualities in relation to transparency a methodology must be adaptable to adjust not violate these qualities the weighting of which are defined by the applications domain and context.

4.3.5 Limitations

A list of limitations has been noted by this research. First, while the selection of the paper criteria was wide, particular key words were used to search for related literature, so some related work might have been missed. The complexity of the trust - transparency relationship and the inconsistency of particular terms, or even the lack of defining terms, lead to some assumption being made about how a concept was held within a paper. The ability to directly measure trust is limited and thus quantitatively defining the effects of altering their facets is difficult. Some areas of research are less developed than others, which means that the weights in concept map and clustering are not proportional to the

strengths of the concepts themselves. The resulting clusters can be subjective to the reviewer's understanding of the problem area from the literature as well as observing and analysing the current state of the art. However, this clustering and data extraction process goes beyond the subjective review of the literature. This way, the clusters as well as the model can be used and built upon by other researchers. Additionally, there are limitations to what the research has already discussed, and each domain seems to focus on a particular group of facets and factors when investigating transparency meaning there could be protentional factors that have not been examined.

4.3.6 Working Definition of Transparency

Most of the existing definitions of transparency are indicating that transparency is about information disclosure. Transparency is related to the information disclosure; however, it cannot be measured by how much information is being revealed (see Section 4.3.2). Transparency in the context of this research is defined analogous to correctness: software is either transparent or not. It cannot be considered over-transparent or under-transparent. Based on this literature transparency can be describe as the appropriate amount of information that the user concerns [171], and goals [5] with respect to a certain system type .It thus refers to what the user needs and expects from the system to achieve its goals successfully and exercise informed trust judgment in a specified context. For example, a user may give consent to use their personal data if they understand why, it is needed and how it is going to be used. Providing the appropriate amount of information can be done by hiding as well as disclosing information, i.e., the appropriate amount can be more or less than currently presented in a system that is not transparent. When it succeeds, transparency can elicit a high level of trust, if combined with receiving the expected benefits from software use [5], and enhance the relationship between user and system [5, 24, 26].

Transparency is seen as a meta- requirement as well as a quality in use [13, 79]. It works at a meta level compared to functional requirements, enabling the user to know “how requirements can be fulfilled” [13]. As Hosseini [171] discusses, a transparency system must provide information relevant to the contextual user's concern. Additionally, Schwab el al. emphasis that the information needs to be relevant to what the user is doing and about their personal data [5]. When it does it has been shown to elicits a high level of trust if in

combination the software's benefits [5]. Hence, transparency can be used to enhance the stakeholder-system relationship by inducing trust [5, 24, 26].

Meunier have developed a notion called Software Purity which refers to a set of standards that the software has to satisfy in order to establish user trust. In addition, user data would not be used for other purposes different from that which the system is clearly intended [172]. This urge of creating a second concept above transparency which shows the is a need to define the intentions of a system and only then can you judge the implemented transparencies objective. Therefore, this research has defined transparency as enabler for users' trust judgment encompassing this wider concept. Trust judgment is where the user has right and relevant information on which to act upon. We can induce transparency concerns to be considered by a development team, but it is fundamentally down to the team to decide the detailed implementation. A way to resolve this is to have some form of regulatory body which enforces transparency to a standard.

Chapter 5

5. Transparency Engineering Methodology

This chapter describes the Transparency Engineering Methodology (TEM) which has been developed during the course of this Ph.D. project. It first explains the methodology vision and its aim and scope. Afterwards a general section gives a summary of the methodology, the main concepts and terminology used. This is followed by introducing the transparency patterns which formalize the requirements generation process.

5.1 Aim and Vision

The primary aim of the methodology is to identify where a software system is lacking transparency on its data and processes based on its user goals and address these shortcomings using transparency engineering methods and patterns.

Applying TEM during the software development process results in a transparent system in its functions and data concerns, as described above leads to enhance user trust [24, 26]. This methodology encourages the effective implementation of different functional requirements and provides relevant information to the user to increase their trust judgement. To evaluate this a case study is conducted See Chapter 6.

The methodology has been developed to ensure that transparency is leading to the intended effect, as in some case exposing information can be detrimental or misleading. The aim is to enable the user to trust the system by providing the right information about its operations in order to enable them to make and confirm decisions. Users recognizing their available options when completing goals giving them a better experience, which further increases their ongoing trust with the software, providing information about the software in its environment (e.g., the scope of the system goals) advice the user of the software's capabilities and sets their expectations. Meeting the users' expectations is an essential factor that influences their initial trust (see Section 2.1.2), that can be shaped before interacting with the software. Therefore, throughout the design of this methodology, trust was the overarching goal and, transparency is addressed over the system aspects so that it shows the relevant information to the user.

5.2 Scope

Existing research examined transparency as non-functional requirement (NFR), where its interrelations with other systems NFRs and goals [26, 82] are examined. However, there is a research gap on transparency in relation to functional requirements of the system. Thus, this engineering methodology addresses the lack of transparency on system aspects such as personal data and system functionalities [24, 155]. Functional requirements involve technical specifications and data processes. Additionally, functional requirements are being assessed and evaluated differently than non-functional requirements, which ensures that sets of functional requirements are meeting a specific system criterion (see Section 2.3).

The focus in this research is on personal data. Existing literature shows the demands of GDPR in view of systems development, often beyond the actual regulation and including information security into the list of concerns relevant to data subjects. For instance, they recommend that information should be shared with the data subjects regarding where data is stored, how data is protected and who has access to it [173] as well as information about the choices on limiting the processing of their data [6].

When developing this methodology, it is necessary to realize and differentiate transparency requirements from other requirements that are similar, such as usability and privacy. Usability problems, for example, writing too small, or in a different language would mean that the user might miss, not be able to read, or not be able to understand the information provided. TEM focusses on the content of the information provided by the system. How to best present such information falls outside the scope of the patterns, see research on user-friendly representations of privacy-related information and requirements design [22, 174].

The methodology uses end users' goal as the main source of information on which to generate requirements from. This supports user empowerment by keeping the users concerns at the centre of the methodology as well as improves clarity in the methodology's presentation. However, the goals of other stakeholders such as the owners of the system, developers, etc. are not considered when generating the transparency requirements. They are considered during the conflict and consistency check which ensures that no information is revealed that might cause harm to the system or company such as personal data breach, or revealing key secrets of the system.

5.3 Overview of Transparency Requirements Patterns

This Section gives a summary of the Transparency Engineering Methodology (TEM) and describes its requirements patterns. Starting with an overview of the methodology and requirements patterns. Requirements patterns can be used during early stages of software development and support re-usability, consistent vocabulary and enhanced communication [175] while addressing the issue of incomplete requirement [176]. TEM patterns are designed following Withall requirements patterns [177].

The methodology is designed to guide the requirements specialist throughout creating transparency requirements. The transparency patterns provoke the requirements specialist to think about what is missing in their requirements. Selecting and checking the conditions, and data interests (see Section 5.4.2) where the transparency requirements are most needed and helps the requirements specialist to find and bridge the existing gaps (e.g., missing post and preconditions). Additionally, TEM gives the developers presentational choices when implementing these requirements.

Figure 5.1, provides an overview of TEM including the conceptual framework and requirements patterns. This figure reads from left to right highlighting TEM steps. The specific steps are described in Section 5.7. The engineer generates requirements using transparency patterns. These patterns cover the different types of functional requirements in the software see Section 0.

IU	List all labelled uses case Using Patterns Prompt List	IDI	Static	High level requirements	Consistency and conflict check	Yes	Presentation choices	Pull	
			DDP					Dynamic	Push
		UDP	Static					Dynamic /AI	Do not show
			PDP					Static	Dynamic /AI
UIU	Group by relation to system	WNS	No	Discard	Indirect Transparency				
		RWS							
		ESB							
		GAG							

Figure 5.1: Overview of the Transparency Engineering Methodology. [IU= Implemented Use cases, UIU= Unimplemented Use cases, IDI= Identification of Data Interest, DDP= Data Driven Patterns, UDP= Use Case Driven Patterns, PDP= Process Driven Patterns, WNS= Why Not in System, RWS= Relation With System, ESB= Explain System Boundary, GAG= Give Alternative Goal]. The conceptual framework and the requirements patterns are description in details in Sections 5.5, 5.7, 0

5.3.1 Methodology Steps Outline

The methodology has four high level steps:

- Step 0: Identify Data Interests.
- Step 1: Apply transparency patterns to implemented goals.
- Step 2: Add justifications for unimplemented goals.
- Step 3: Consistency check.

The methodology consistent of four steps preceded by the preliminary step of identifying data interests. Based on these, the first step applies the transparency patterns. The second step focuses on the goals that are out of scope for the current implementation of the system and the foreseeable future. The rationale behind this step is to manage the users' expectations, making them aware of limitations in relation to their goals. The third step is a consistency check for the resulting requirements. This methodology uses requirements patterns in Step 1 to solve the transparency issues systematically. Like other patterns in software engineering, transparency requirements patterns (transparency patterns) are reusable solutions to frequent problems. Requirements patterns have an additional benefit of saving project time as they give a starting point to generate the requirements [177]. Specifically, for novice developers the patterns' consistency is useful. More essentially these patterns provide guidance to the developers and requirements specialists for writing their transparency requirements. Further, they provoke the developers and requirements specialist to find and address the gaps in their requirements as outlined in the Aim and Vision (see Section 5.1).

Performing the steps results in high-level requirements which deliver information to the user about the system's functionalities, data and limitations. Research showed that trust can be enhanced "systematically during the system development process" [14]. Performing these steps enhance initial trust by providing static information about the system's intended goals [6, 16]. However TEM is also enhance ongoing trust as it informs users about run-time events [6, 16].

The methodology generates transparency requirements for functional goals represented in the software system with use cases or user stories. It also includes the data the system holds, how and where data is stored and who has access to view and process it etc.

Consequently, use case, data, and process transparency requirements are derived to ensure that this information is communicated to the user.. Use cases consist of two types of actions, Queries, where there are no data changes and Operations where there are data changes. Use cases carry the functionality information that can inform the user on how to operate the system. Therefore, use case patterns are derived to generate the requirements that are responsible for informing the user about system aspects related to these use cases.

User stories are a key component of the agile development process that empowers the user by centralising their concerns in the development cycle process. User stories have similar constituent elements to the use cases, and as a result the use case patterns can be adapted to be used with user stories [178].

The user also needs clarity about the data the system holds, the underlying mechanisms, where their data is stored and who has access to process it etc [179] [95]. Consequently, data transparency patterns are derived to generate the requirements that communicate this information to the user. There are two levels to address in use cases and data patterns which are type and instance levels. The type-level describes the information related to the system schema identified by classes, attributes, and associations. The instance level covers the actual content (data and state) of the system and informs the user how the data is being used.

5.3.2 Transparency Requirements Pattern Classification

The methodology is structured and driven by three distinct domains representing aspects of the system. These domain driven patterns are Data, Use Case, and Process which have their own means of transparency in the system. Literature review show in order for the information to be relevant, transparency cannot be identified for the entire system at once. Thus, transparency is necessary to be identified, only for the particular system aspects (see Sections 4.3.4 4.3.6). [Therefore, the methodology defines and classifies the domain types to extract the related transparency requirements. These domains are driven based on the system aspects (data, operation) that are required to be transparent about. Each domain driven pattern contains sub levels of categorization. For data driven pattern (DDP) as mentioned above: Static (Type) level which describes the information related to the scheme of the system identified by classes (attributes, associations) where transparency is needed.

Whereas the Dynamic (instance) level covers the actual content of the scheme of the system (system state) and informs the user how the data is being used. For example, data change alerts are in the dynamic level to be seen at the run time when data is changing and by whom.

Sub-categorization for the Use case and Process domain driven patterns are also static and dynamic. The static process pattern presents information about data or operation where no input needed at the run time. i.e., when information about how a specific function of the system works. Dynamic process pattern introduces information about data or operation at the run time, where in some cases it takes an input, changing the outcome of the process i.e., how a specific function of the system works in a context defined by user.

5.3.3 Requirements Pattern Template

Requirements patterns generate specific types of requirements. Software patterns can be used during early stages of software developments and provides benefits such as, reusability, consistent vocabulary and enhanced communication [175]. Additionally patterns can be used to solve the issue of incomplete requirement [176]. The structure developed for these patterns in this methodology is based on Stephen With all requirements patterns [177]:

1. Basic details
 - a) Pattern manifestation: the pattern's version number and last update.
 - b) Belongs to domain: the requirements pattern area that this requirement is part of.
 - c) Pattern author: Who wrote the pattern.
2. Applicability specifies in what cases to use or not use the pattern.
3. Discussion specifies how to write a requirement of this pattern.
4. Content specifies the “detailed each list of information that a requirement of this type must conveys”. Each of these items (parameters) are refer to with a name. These parameters are described in (see Section 5.4).
5. How to get the content specifies where and how to get the requirement parameters.
6. Template(s) specifies what to write for a requirement of this pattern by saying what the system must communicate to the end user, and the data interests and conditions

where necessary. Data interest and condition concepts are described in detail in Section Basic concepts below.

7. Example(s) One or more illustrative requirement resulting from using this pattern.

Each transparency pattern demonstrates how to write its resulting requirement. Some of these patterns specify conditions when to be transparent. Conditions are logical statements composed of combinations of data operations with the actors performing them.

Transparency requirements patterns make the developers and designers aware that transparency about information is required and in which condition (see presentation choices Section 5.4.4).

The following Sections present the concepts and terminologies for the development suggestions for requirements specialist and software designer. This provides guidance for types of presentation choices, how to fill patterns parameters and how to communicate information.

5.4 Basic Concepts

There are a number of basic concepts which must be understood in order to successfully apply the transparency engineering methodology.

5.4.1 Conditions

The concept of conditions is provided for the dynamic patterns where to specify when to inform the use by, to addressing cases of critical data change at the run time Each pattern demonstrates how to write the resulting requirement. Some patterns specify conditions when to be transparent. Conditions are logical statements composed of combinations of data operations with actors performing them, of the form:

<< <<actors>> <<data operation>> <<data type>> >>

The actors can be multiple users and or the system. The data operations are the basic functions of the persistent storage as well as the accessibility function on that data. These data operations are CRUD (which is create, read, update, delete) and access control. Data type reflects the data on which the operation is acting.

For GDPR driven patterns data operation is called data processing [6]. Processing under GDPR means “any operation or set of operations which is performed upon personal data or sets of personal data, whether or not by automated means”.

These conditions can be collocated using the logical operators: (AND, OR, NOT). This is indicated in the patterns as <<*List of Conditions*>> where condition can be one condition or combined conditions.

To generate the conditions used by the patterns, use the following steps:

1. Based on the data interest identify the classes that uses this data in the domain model.
2. Select the use cases representing these classes.
3. Extract the data operations and their actors from the use cases.
4. Use these actions and respective actors to generate the conditions.

For instance, a generated condition would look like «Condition» -> “system admin updated address”.

5.4.2 Identification of Data Interests

The requirements specialist uses this process to generate data interests over the whole system for more consistence results. Data interests are a reference to data that are defined under the same concern or information that matters to the user e.g., persona data. This process gives them the flexibility to call out for any data of interest, in some cases the data that normally does not sound like personal data. For example, behavioural data that can potentially identify the individual user.

The process of generating data interests are as follows:

1. The specialist decides identifies discrete sets of data points for the system and gives the grouped data interests names so that they can be easy to use. The data interests come from user concerns based on the goal specification, as well as laws and regulations. Personal data is an example of a data interest that comes from regulation, in this case General Data Protection Regulation [7]. User concern about data is shown in their goals e.g. “I want to see my shipping address before

confirming payments”. A specialist in this case creates a data interest called “personal data- Address” to represent this user data concern.

2. The specialist selects the data types that are mapped to the data interests. This mapping comes from selecting the relevant data classes that store the data from the conceptual data model.

5.4.3 Domain Driven Pattern Prompt List

The methodology suggests a prompt list, which are lists of suggestions to help the requirements specialist analyse and identify the related components (data, use case and process) of their system use cases and associate them with the correct pattern. To generate this list focus group sessions were conducted with the requirement specialists and developers from Spirit Healthcare (See Section 6.1.2) these sessions included brainstorming on the main actions which include exceptions and alternative courses as well as critical data change that would be common user concerns. These actions are identified to help the user TEM to choose the relevant patterns to inform the user. The requirements specialist checks if the use cases contain any of the following, then selects the corresponding patterns accordingly and which then generates the required high-level requirements. The suggested list can be expanded and changed based on the software domain and needs. The prompt list is present in Data Prompt List (DPL) which it identifies cases to use data driven patterns. Common cases identified for this list are:

Critical data change This can be defined by the requirement specialist and create a data interest for it.

Data verification with user input (e.g., When Data status change. An email before been verified is not a personal data, once it is verified it change it is status to personal data.).

Data collection (i.e., When system gathers information about the user or from the user. One concrete example is Personal data, statistical data, behavioural data)

Communicating with the user (i.e., Any communication with user requires their input, or information display to the user, or confirmation messages.)

And Use Case and Process Prompt List (UPPL) which it identifies cases to call for use case or/and process driven patterns. Common cases identified for this list are:

Extended action (e.g., When the use case has extended actions from/to the system)

Potential mismatch (e.g., Mismatch between the post-condition of a use case with pre-condition of another use case.)

Alternative courses (e.g., When use cases have alternative pathways)

Exception courses (e.g., When use cases have exception pathways).

The prompt list also includes Negative label where no pattern is needed for instance, in case of user query for information display See Figure 5.2: Prompt List.

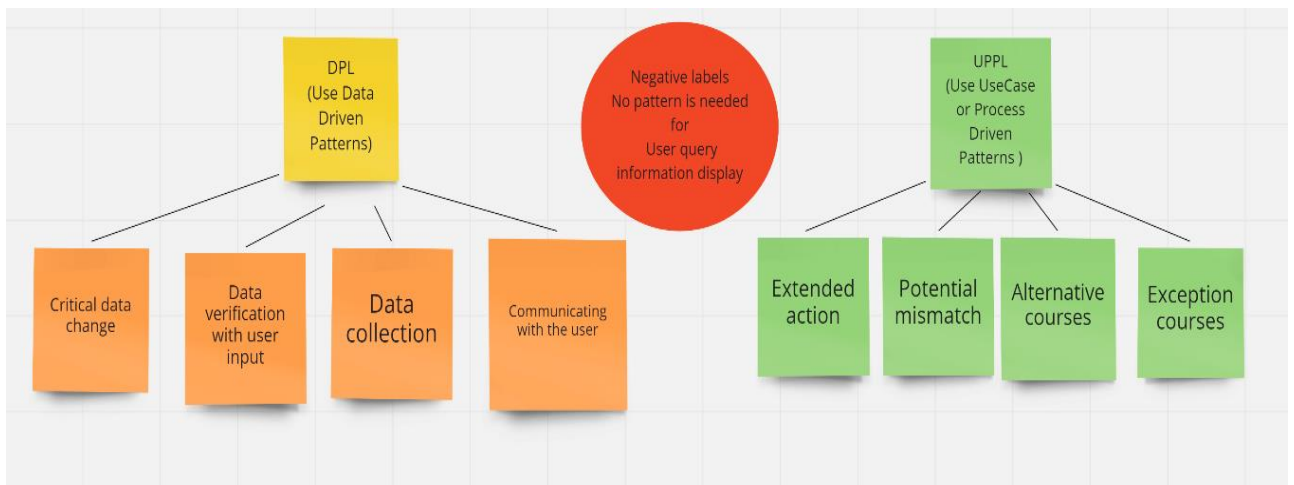


Figure 5.2: Prompt List

5.4.4 Content and Presentation

The presentation choices that may come up every time with a transparency requirement asks to pass some information and in some other cases hide information from the user. The following choices are “Push / Pull / Don’t show / User settings/ Indirect Transparency”. These choices (Pull and Push) are derived from transparency motivation types and transparency guidelines article 29 [180], which explains how this presentation can be decide on (see Section 6.3.1). Use settings option is derived from the Personalization goal of trust [14]. The appropriate of information can be provided by hiding as well as disclosing information, i.e., it can be more or less than currently presented in a system that is not transparent. The motivation for not referring to an over-supply of information as transparent

should be obvious: too much information can hide the essentials needed to make informed decisions. Thus, Don't show and Indirect Transparency are derived.

The pattern (see Section 0) states what needs to be communicated to the end user and the requirements specialist deals with the presentation choices separately. In the patterns templates the expression "must communicate to" is replaced with one of the choices. The following Sections explain these choices and ways to be transparent in more details.

Push- and Pull-Transparency

To determine how the transparency information is provided, two ways can be defined: Push- transparency and Pull- transparency. "Another possible way of providing transparency information is through the use of "push" and "pull" notices. Push notices involve the provision of "just-in-time" transparency information notices while "pull" notices facilitate access to information by methods such as permission management, privacy dashboards and "learn more" tutorials. These allow for a more user centric transparency experience for the data subject" p20 [180].

Pull means that the user has actively to call for the information, push means that the information is presented to the user. Push-transparency is for users and situations that have a high information need (e.g., critical data change) with this need being automatically fulfilled by the system through presenting the right information. An example for illustrating Push transparency would be using a pop-up to display the message "Your address has been changed. The record we hold about you now is University of Leicester University Road Leicester LE17RH". Pull-transparency is when information is not directly related to their current goal with the system. In these cases, the users are fine with retrieving the information themselves, thus pulling the information from the system. Pull transparency information is not actively presented to the end-users, as it is irrelevant for the majority of cases. Instead, the information is located somewhere (e.g., T&Cs, Help, Documentation, ...), where the user must read carefully or utilize a search engine to find it. The same user can have different information needs in different situations. These information needs can be better understood by the transparency motivations (see Section 6.3.1).

There is also an option to improve Push- and Pull- transparency decisions by user rating input. The user rating of information needs can be “primary”, “secondary”, or “irrelevant”. Information rated as primary indicates the need for Push transparency. Information rated as secondary indicates the need for Pull transparency. Information rated as irrelevant indicates “Don’t show”. Eliciting this information from the user allows the requirements specialist to gain insights in their mental model.

When designing transparency, the specialist must consider the mental model of the end user and try to reveal or hide the information needed accordingly. In the perfect settings, this can create a level of abstraction, where both the mental model and the way the system functions are matching.

User Settings

For each requirement that can be generated by any of transparency requirements patterns, the requirements specialist can alternatively create a requirement allowing the user to specify the level and type of transparency (Push, Pull, Don’t show) and the frequency of notifications of the related transparency information. In this case, the system needs to provide the user with the means to provide their input. This type of personalization can help the user to choose the amount and level of information desired, thus helping the system to understand the user needs and resulting with increasing the user trust [14].

Indirect Transparency

Transparency is not just about revealing distinct information to the end user at a given time. It can also be developed by representing the available and unavailable options accessible to the end user with which they interact with the system at that given time. Thus, instead of showing the account balance to the user, the system indicates only if a certain transaction is possible or not (e.g., when transferring money, it does not show the balance before and after the transfer, but “in/sufficient funds”). This can for example be used in a crowded environment to minimize the effects of shoulder surfing or sniffing hacker attacks.

5.5 Prerequisites of the Methodology

There are three information sources for the methodology, use cases and user stories which represent the functional user goals, conceptual data model, and processes. Domain models

are provided to capture the concepts and relations of the problem domain in the form of a high-level data model. That means, we assume representations of:

- The system model (the goals of the end users as implemented by the system) which contains pre- and post-conditions as well as main and alternative sequences, typically with use case diagram.
- The processes and actions in the system, typically with activity diagrams.
- The data in the system by conceptual data models, typically as class diagrams.

The method comes with the following documents.

- Transparency requirements document template.
- Transparency patterns, which also includes TEM steps.

5.6 Overview of the Methodology

The engineering methodology is performed during the software development process. The methodology is part of the requirements engineering process, which can be organised in different ways according to the process model applied.

The steps and patterns are applied by requirements specialist and developers. The requirements specialist generates the transparency requirements using the transparency patterns. The methodology helps the requirements specialist to decide about the presentation choices when passing information to the user through first separating the content from the design options, then guiding the specialist using transparency presentation concepts that explain how to pass the information to the user. The methodology is structured in a way so that the different patterns are decoupled and can be applied in different locations and at different times, by different teams of experts.

The requirements specialist responsibilities are assigned for various roles depending on the development process type. For example, in the agile processes the product owner is the one who is the most heavily involved with the requirement practices, then the SCRUM master and SCRUM team are mainly involved in the requirement analysis processes.

The methodology consistent of four steps. These steps are showing in the conceptual framework see Figure 5.1. The first step applies the transparency patterns. The patterns capture transparency concerns using user stories, use cases and processes, as well as based on data interests. The second step focuses on the goals that are out of the scope for the current implementation of the system (and for the foreseeable future). The rationale behind this step is to not violate the users' expectations about the system by making the user aware of the limitations of the system in relation to their goals. The third step is consistency and conflict check for the high-level resulting requirements. The steps are demonstrated in Figure 5.3. In this methodology the developers can use user stories instead of the use cases. For that to be possible the developers need to make sure that all the components are available in their user stories, for instance the pre- and post-conditions, the main and alternative sequences.

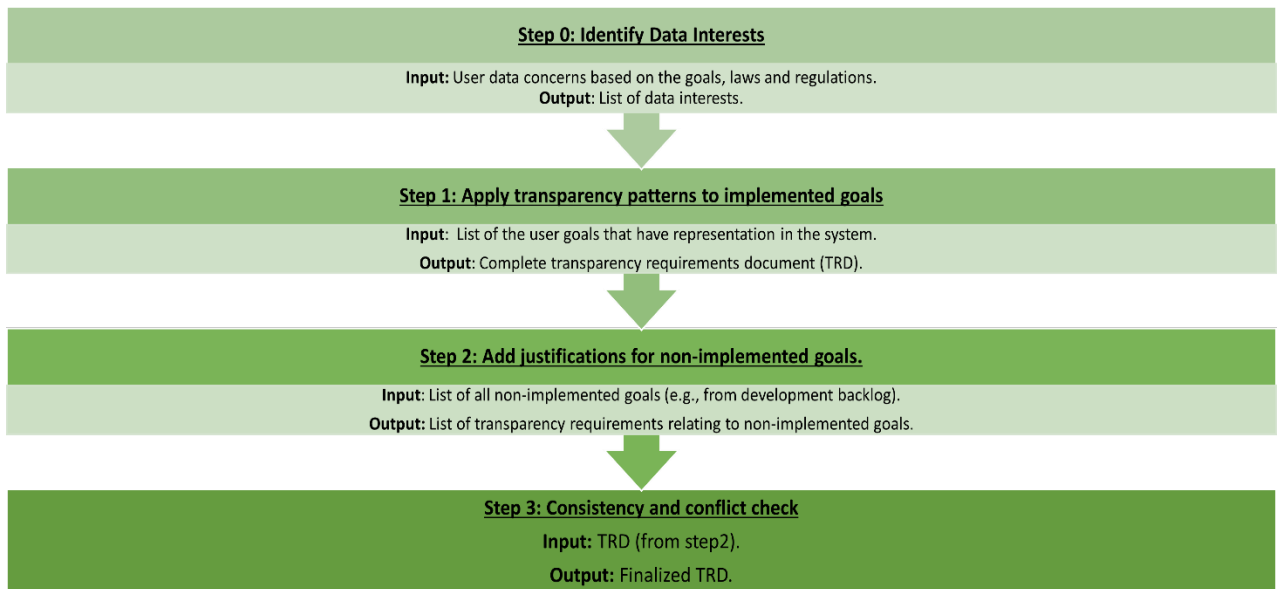


Figure 5.3 TEM steps

5.7 Methodology Steps

Step 0: Identify Data Interests.

Input: User data concerns based on the goals, laws and regulations.

Action: To ensure consistency of requirements, it is important to identify data interests prior to applying the patterns, so that the same data interest used in different areas of the

application can be grouped and used in the same way. Data interests are a reference to data that are defined under the same concern or satisfying the same use goal, e.g., persona data, performance data, account data, etc. This process provides the flexibility to identify any data group related to the user. The process of generating data interests is described in Section 5.4.2

Output: List of data interests.

For example, users' concern about data is often shown in their goals e.g. "I want to see my shipping address before confirming payments". An engineer in this case creates a data interest called "shipping data" to represent this user data concern.

Step 1: Apply transparency patterns to implemented goals (Use Cases, User stories)

Input: List of the user goals that have representation in the system.

Action: Map the goals to one or more of the domain driven patterns (Data, Use case, Process), this can be done directly or with the aid of the prompt list. The goals can refer to functionality already in the system, functionality overlooked by the user or to functionality to be added to the system. In order to achieve this, the implementer must list the relevant use case or user stories representation in the software. Then use the pattern prompt list to identify the related components of system use cases to help you to choose the corresponding patterns. Use the data interest identification process to generate the data interests. For each data interest identify the corresponding data patterns. Write the resulting requirements in the Transparency requirements document (TRD). See a full description on Transparency patterns in 0.

To increase the consistency over the application, it is important to identify all the data interests within the team hierarchy, where the project managers/domain experts can define and then share them with the other team members.

Output: Complete transparency requirements document (TRD).

For example, from a user story such as "As a user, I want to make an online purchase using quick checkout and have it shipped to my current address, so that I can order as fast and hassle free as possible", our patterns will generate the following requirements:

GDPR Data Protection pattern: The system must communicate to the user that it holds data of type Address which is “accessed by delivery service” and “This data is encrypted and will be stored until your account is deleted”.

Static Use case pattern: The system must communicate to the user that “default address must be given before quick checkout”.

Step 2: Add justifications for non-implemented goals.

Input: List of all non-implemented goals (e.g., from development backlog).

Action: Create a list or, use existing list of the goals for that purpose. The low priority goals are to be implemented later or considered out of scope for the foreseeable future. After that, select the related goals which are the ones that could fit into the system scope, but they are currently not implemented.

Then run the following Justifications on the mentioned goals, (WNS = Why Not in System, RWS = Relation with System, ESB = Explain System Boundary, GAG = Give Alternative Goal). Goal and record a reason why that justification has been assigned. The goals are then clustered by reason to create a single transparency requirement to inform the user why a particular goal has not been implemented. This gives the user a better understanding of the limitations of the system and improves their mental model. As these goals are not in the focus of the system, transparency about them is usually achieved through “pull transparency” where the user must actively call for the information.

Output: List of transparency requirements relating to non-implemented goals.

For example, in food delivery systems a user goal can be “I want to order my groceries”. The justification here would be ESB where the system currently only delivers fast food because both fast food and groceries are food types.

Step 3: Consistency and conflict check

Input: TRD (from step2).

Action: Streamline the transparency requirements document by applying a consistency and conflict check that results in removing duplicates and conflicted requirements as well as aggregating requirements that can be combined. The requirements specialist needs to check

if there is any information disclosure or conflict with the other requirements like security or privacy.

Output: Finalized TRD.

For example, sharing information about data servers could cause a security breach.

5.8 Transparency Requirements Patterns

5.8.1 Data Transparency

Static Data Transparency Requirements Pattern

Basic Details

Pattern manifestation:	Standard
Belongs to domain:	Data
Pattern author:	B. Zieni

Applicability

Use the static data type transparency requirements pattern to provide information about the data the system holds about the user and what are the existing system actions on that data type by which actors.

<i>Content</i>	Data interest. Data type System actions and their actors.
----------------	---

How to get the content

The data interest is defined by the requirements specialist based on common and individual concerns.

Data type can be identified based on the data interest where the specialist comes up with the list of information identifiers then select the data types belonging to them. Information identifiers are the label descriptions that explain what kind of data is associated with this data interest.

By checking all the existing use cases that deal with this data type, a list of operations and a list of actors performing them can be identified. The identification process is done by the requirements specialist based on the data interest.

Template(s)

Summary	Definition
<i>Transparency about</i>	<i>//If «Data Type» is a «Data Interest»:</i>

«Data Type»»	The system must communicate to the user that it holds data of the type «Data Type» « list of data operators» can be performed by « list of actors» &/«Reasons»»
--------------	---

Example(s)

Summary	Definition
Transparency about “Address”	The system must communicate to the user that it holds data of the type “Address” and “update” can be performed by “System admin”

Dynamic Data Transparency Requirements Pattern

Basic Details

Pattern manifestation:	Standard
Belongs to domain:	Data
Pattern author:	B. Zieni

Applicability Use the data transparency requirements pattern to generate transparency requirements for the specified data instance that the system holds about the end user. This pattern gives information, during runtime, about what are the system actions being performed and by which actors.

<i>Content</i>	<p>A reference to the data instance.</p> <p>Data Interest.</p> <p>Data type of this data.</p> <p>Data Gathered</p> <p>Condition</p>
----------------	---

How to get the content

The reference to the data is derived from the data type. The data interest is defined by the requirements specialist based on common and individual concerns.

Data type can be identified based on the data interest where the specialist comes up with the list of information identifiers then select the data types belonging to them. Information identifiers are the label descriptions that explain what kind of data is associated with this data interest.

Specify that source of the data and where the data been gathered, inferred, or aggregated. This is become irrelevant when the user is the source.

Conditions are defined by the requirements specialist and get checked during runtime (see 5.4.1)

Template(s)

Summary	Definition
<i>Transparency of Instance of «Data Type» because « condition ».</i>	<i>//If «Data Type» is a «Data Interest»: In case <<condition >>, the system must communicate to the user << the «Data Type» (instance) \&<<data gathered >> by <<actor>> <<condition>> >></i>

Example(s)

Summary	Definition
<i>Transparency of Instance of “Address” because “system admin</i>	<i>In case “system admin updated address”, the system must communicate to the user “University of Leicester, University road, Leicester, UK LE17RH” <<gathered>> by system</i>

updated address”.	<i>admin.</i>
Transparency of Instance of “Address” because “system admin updated address”.	In case “system admin updated address”, the system must communicate to the user “system admin updated address”.

5.8.2 GDPR Transparency

Data Protection Transparency Requirements Pattern

Basic Details

Pattern manifestation: Standard

Belongs to domain: Data

Pattern author: B. Zieni

Applicability

Use the Data Protection Transparency Pattern to generate transparency requirements for the specified data types of the data falling under data protection legislation the system holds about the end user. This pattern illustrates the data accessibility and storage.

Content

Data Interest

Data type

Data storage

Data access

How to get the content

The data interest is defined by the requirements specialist based on common and individual users' concerns.

Data type can be identified based on the data interest where the specialist comes up with the list of information identifiers then select the data types belonging to them. Information identifiers are the label descriptions that explain what kind of data is associated with this data interest. Data type is called category of data under GDPR.

How is data is stored in the software. Is it (encrypted (yes/no), anonymized (yes/no), Pseudonymization (yes/no)? Also, specify for how long it will be stored? Explain why if needed. Where is it stored? Specify if the data storage place, while still complying with security requirements.

Specify with whom it is shared and who has access to it.

Template(s)

Summary	Definition
<i>Transparency about data type «Data Type»</i>	<i>//If «Data Type» is a «Data Interest»: The system must communicate to the user that it holds data of the type «Data Type» which has been <<Data access>> √&<< Data storage >> </i>

Example(s)

Summary	Definition
Transparency about data type "Address" because system admin updated	The system must communicate to the user that it holds data of the type "address" which has been "accessed by system admin". This data is encrypted and will be stored until your account

address.	deletion.
----------	-----------

Data Subject Right Transparency Requirements pattern

Basic Details

Pattern manifestation:	Standard
Belongs to domain:	Data
Pattern author:	B. Zieni

Applicability Use data subject right transparency requirements pattern to provide information about what are data types that system holds about the end user and their rights on that data under GDPR

<i>Content</i>	Data interest.
	Data type
	Data subject's rights

How to get the content The data interest is defined by the requirements specialist based on common and individual concerns.

Data type can be identified based on the data interest where the specialist comes up with the list of information identifiers then select the data types belonging to them. Information identifiers are the label descriptions that explain what kind of data is associated with this data interest. Data type called category of data under GDPR.

Specifying the data subject's rights on the data. They are to (be informed, access, rectify (complete the incomplete data), erasure, restrict processing, data portability, object, rights in

relation to automated decision making and profiling.)

Template(s)

Summary	Definition
Transparency about «Data Type»	//If «Data Type» is a «Data Interest»: The system must communicate to the user that it holds data of the type «Data Type» and << Data subject's rights >> can be performed by data subject

Summary	Definition
Transparency about "Address"	The system must communicate to the user that it holds data of the type "Address" and "erasure, restrict processing, data portability" can be performed by data subject.

5.8.3 Use Case Transparency

Static Use Case Transparency Requirements pattern

Basic Details

Pattern manifestation: Standard

Belongs to domain: Use case

Pattern author: B. Zieni

Applicability

Use the Static Use Case Transparency Requirements pattern to provide information about how the use cases of the system work. The requirements specialist makes a choice on when to use this pattern either when it is a new use case or a use case that needs more information.

Content

Use case

Pre/post conditions for use case actions

Main sequence

How to get the content

The use case from the use case diagrams the specialist chooses to be transparent about.

Pre/post conditions are part of use case description. Both workflows and use case scenarios may define alternative sequences of actions.

Main sequence from the use case diagram.

Template(s)

Summary	Definition
<i>Transparency about</i>	<i>The system must communicate to the user that <<about pre-condition //& about post-condition //& follow main sequence >></i>

«Use Case»	<<before //& after //& to perform>> «Use Case».
------------	---

Summary	Definition
Transparency about “withdraw money”	The system must communicate to the user that “your balance should be above the overdraft” before “withdraw money”.

Example(s)

Alternative Use Case Transparency Requirements Pattern

Basic Details

Pattern manifestation:	Standard
Belongs to domain:	Use case
Pattern author:	B. Zieni

Applicability Use pattern to identify which enabled and available use cases the system needs to reveal to the user in case of a failure to meet pre/ postconditions.

Content Pre/post conditions for use case actions.
 Use case.
 Alternative use cases.
 Reason describes why the system has chosen this sequence.

How to get the content Pre/post conditions are part of use case description. Both workflows and use case scenarios may define alternative sequences of actions.

Use case that has alternative sequence from use case diagram.

The list of alternatives (available and enable) use-cases for the

use case that match the current system state or ways to fulfil the pre and/or postconditions from the use case diagram.

Reason from the use case diagram illustrated from the deviation and mismatch of the pre and postcondition.

Template(s)

Summary	Definition
<i>Transparency about «Use Case» because of unmet <<pre-conditions / post-conditions>>.</i>	<i>If current system state doesn't meet <<pre-conditions "at start" post-conditions "at end" >> of <<use case>> the system must communicate <<list of alternative available and enable use-cases for <<use case>> that match the current system state or ways to fulfil the <<pre &/ post conditions >>>> because of <<reason>></i>

Example(s)

Summary	Definition
Transparency about "transfer money to account at same bank" because of unmet "enough funds in own account".	If current system state doesn't meet "enough funds in own account" at start of "transfer money to account at same bank" the system must communicate "deposit money in bank account does not own by user" because of "not enough funds are less than the minimum amount of 150 GBP".

Dynamic Use Case Transparency Requirements Pattern

Basic Details

Pattern manifestation: Standard

Belongs to domain: Use case

Pattern author: B. Zieni

Applicability Use the Alternative Use Case transparency requirements pattern to identify, which alternative sequence of use cases the system needs to reveal to the user.

Content System use case that has alternative sequences that the system chosen.

Alternative sequence.

How to get the content Use case that has alternative sequences from the use case diagram, where the system is an actor.

The list of alternative sequence of use-case from the use case diagram.

Template(s)

Summary	Definition
<i>Transparency about «Use Case» because of alternative sequence.</i>	<i>If the system uses an alternative sequence of << use case>>. The system must communicate <<alternative sequence >></i>

Example(s)

Summary	Definition
Transparency about “withdraw money” because of alternative sequence.	If the system uses an alternative sequence of “withdraw money” The system must communicate “change the amount or add funds to your account”

5.8.4 Process Transparency

Static Process Transparency Requirements pattern

Basic Details

Pattern manifestation: Standard

Belongs to domain: Process

Pattern author: B. Zieni

Applicability

Use the static process transparency requirements pattern to provide information about how the processes of the system work.

Content

Process

Pre/post conditions for process actions

Main flow

How to get the content Any process from the activity diagram.

Pre/post conditions are part of process description. Both workflows and use case scenarios may define alternative sequences of actions.

Main flow from the activity diagram.

Template(s)

Summary	Definition
<i>Transparency about «Process»</i>	<i>The system must communicate <<about pre-condition //& about post-condition //& follow main flow >> <<before //& after //& to>> «Process» to the user.</i>

Example(s)

Summary	Definition
Transparency about «Withdraw money from another's bank ATM»	The system must communicate “Your bank needs to be in the List off Banks we deal with for free charge” before «Withdraw money from not-my-bank ATM» to the user.
Transparency about <<transaction fee >>	The system must communicate “this product with and without transaction value”

Alternative Process Transparency Requirements Pattern

Basic Details

Pattern manifestation: Standard

Belongs to domain: Process

Pattern author: B. Zieni

Applicability

In case of a failure to meet pre/ postconditions, use pattern to identify which enabled and available alternative processes (sequences of actions) the system must reveal to the user.

Content

Pre/post conditions for process actions

Process.

Alternative flow Processes.

Reason.

How to get the content

Pre/post conditions are part of process description. Both workflows and use case scenarios may define alternative sequences of actions.

Process that has alternative flow of processes from activity diagram.

The list of alternatives (available and enable) flows for the process that match the current system state or ways to fulfill the pre and/or post conditions from the activity diagram.

Reason describes the failure of pre/post condition.

Template(s)

Summary	Definition
<i>Transparency about «Process» because of unmet <<pre-conditions / post-conditions>></i>	<i>If current system state doesn't meet <<pre-conditions / post-conditions>> at <<start end >> of <<process>> the system must communicate <<list of alternative available and enable processes for <<process>> that match the current system state>> to the user or how to fulfill the pre and/or post conditions >> because of <<reason>></i>

Example(s)

Summary	Definition
Transparency about “assurance claim” because of unmet “minimum contract duration”	If current system state does not meet “ <i>minimum contract duration at start</i> ” of “insurance claim” the system must communicate “ <i>call our representative to discuss your options</i> ” because claims can only be made if the contract activation period is one year, and yours has been active for 7 months.

Dynamic Process Transparency Requirements Pattern

Basic Details

Pattern manifestation: Standard

Belongs to domain: Process

Pattern author: B. Zieni

Applicability Use the Alternative process Transparency Requirements pattern to identify, which enabled and available alternative sequence of processes the system needs to reveal to the user.

Content Process with alternative flows.

List of alternative sequences.

How to get the content Processes can be higher-level workflows over use cases or detailed use case scenarios.

The list of alternatives (available and enable) flows for the process that match the current system state or ways to fulfil the pre and/or post conditions.

Template(s)

Summary	Definition
<i>Transparency about « process», because of alternative flow.</i>	<i>If the system uses an alternative flow of << process>>. The system must communicate <<alternative flow >>,</i>

Example(s)

Summary	Definition
Transparency about “ <i>withdraw money</i> ” because of alternative flow	If the system offers an alternative flow of “ <i>withdraw money</i> ” The system must communicate “change the account” to the user,

Chapter 6

6. Evaluation

6.1 Evaluation of the Transparency Engineering Methodology

6.1.1 Research Question and Approach to Evaluation

Based on the conducted literature review (see Chapter 4) it is concluded that transparency properties that can affect trust are various from one application domain to another. Each aspect of the system has a different responsibility towards the user to enhance the user's relationship with the system by enabling user trust judgment on these system aspects. As a consequence, informing the user about these system aspects is essential.

Therefore, TEM is developed in order to encourage the effective implementation of the functional requirements. TEM gives the final decision to the software engineers to decide what are the functionalities and data that needed to be communicated to their targeted users.

To evaluate the methodology following this framework we use a “how” question for the design process of TEM and a “what is” question for knowledge about its impact [181]. Thus formulate the following questions:

- How does TEM help in defining the system requirements?
- How does TEM help in generating transparency requirements?
- How easy is it to apply TEM?
- What is the impact on impact on trust judgment and other system qualities?

As it is common in agile development [182], in our evaluation developers and domain experts act as spokespersons of the end user. This is necessary because trust is a subjective and individual concept based on a range of different factors [16, 53]. Such confounding factors would need to be measured and accounted for to isolate TEM's impact on trust. Also, to be effective the resulting requirements must be refined considering usability (i.e., how the information is best presented to the user) which is beyond the focus of our research but will impact on the effectiveness of transparency and thus on user trust. The use of domain experts as proxies allows to circumvent this problem in a qualitative study.

These questions are not standard measures for effectiveness, but they give an indication of the usefulness of the transparency engineering methodology in refining the functional requirement by generating transparency requirements. Themes are chosen as indicators for measuring the effectiveness of this methodology (see Section 6.1.3). The themes are derived from a combination from reviewing the literature and the research work.

In order to answer these questions, a qualitative analysis of participants questionnaires has been combined with a comparative review from the output of TEM. Points of interest have been grouped together into themes, which are derived from the clusters of the literature review.

Prior to the evaluation, a pilot study was conducted with experts from the requirements engineering, privacy and data protection fields, where they have given their feedback on the transparency engineering methodology. This resulted in changes on the presentational and implementation structure of TEM and the creation of the GDPR patterns tying in with recent legislation affording the user more digital rights. The methodology has been developed to primarily aid the domain experts and requirements specialists; but it also helps the developers in defining and transparency requirements. In the chosen company, the developers and the domain experts were the stakeholders responsible for the requirements engineering activities.

This evaluation is designed to test the validity of the transparency engineering methodology in a real-life scenario. In this experimental evaluation, effectiveness is measured by evaluating the extent to which these themes are covered. And how TEM users feel about using the methodology.

6.1.2 Case Study Setup

To instantiate and test the methodology, a case study with Spirit Healthcare was conducted. Spirit Healthcare is a health organisation with a small technology branch who handle sensitive and individually identifiable medical data across a number of applications including remote patient monitoring and education booking and management. The participants were all team members, and include domain experts, senior developers, and developers. Participation in the case study and the evaluation was voluntary. The evaluation of the case study uses qualitative approaches, including the following instruments: semi-

structured interviews, focus groups, and pre- and post-questionnaires (see Section 3.1.3). The domain experts represent the end user. There is no user involvement in the evaluation process, as HCI aspects are outside of the research focus. The materials used for this evaluation can be found in the Appendix (see Chapter 8).

The main objective behind the case study and expert evaluation was to identify instances of good implementations by experienced developers. This is explored through the themes before and after applying the transparency engineering methodology. These themes are then assigned to the responses using codes (see Section 6.1.3).

The research project has been evaluated through a TEM user-based evaluation. The literature shows that there is great value in a good and trustworthy relationship between the end-users and the software. This value is even more vital in the health care domain, which often has high data sensitivity, there is a high cost, for example, of missing appointments and a reliance on giving accurate information to the user. Trust lowers transaction costs [70]. Therefore, a number of companies which expressed interest in the project have been reviewed, Spirit Healthcare is a very successful company, currently in the top 50 fastest growing company in the UK. The health domain is of particular interest due to the critical nature of its data collection. All appropriate consent was collected from participants and data owners before beginning the project. A variety of voices were heard in this evaluation, because the participants were not only domain experts, but the whole digital and software engineering team.

The team consists of four developers one of which is the project manager, SCRUM master and lead architect. The team also contains a domain expert with a joint role of product manager who sits between the development team and the business. The domain expert has years of clinical experience within the NHS. The case study was used to test the transparency engineering methodology in a real-life situation. Spirit Healthcare is a suitable company due to its manageable size, sensitivity of data concerns, adoption of SCRUM methodology and appropriate applications with critical user information transfer. Indeed, the individual autonomy is a central concept in NHS medical ethics, informing patients of options and empowering them in the ultimate decision, thus aligning closely with TEM objectives.

The *participants' areas of expertise* in software development are:

First developer (D1): Mobile Application Development, Xamarin (Android, iOS, Forms) using C#.

Experience: Four years in Mobile Application Development, eight and a half years in professional software development (One year of that was a placement year). The first four and a half years were mostly spent doing Web Development.

Second developer (D2): Software developer started with C, C++, VB then moved to C#, developed a mix of applications, including windows forms. Mobile device and web projects.

Experience: Software developer for 25+ years degree educated in Computing. Main industries of interest are Retail (Next plc (15 years), in recent years Microsoft dynamics for Ticketing systems.

Third developer (D3): Software developer

Experience: 4 years' experience mostly working C# using Xamarin, Asp.Net and Unity. Also has experience with web development and has worked in Gas industry and the health industry for a number of years.

Fourth developer (D4/ DE4): Head of Development at Spirit. Oversees many aspects of the functionality of the development team. I have been doing this for 1 year at Spirit Healthcare and have been responsible for a team for a year in the past.

Experience: 20 Years' experience in healthcare software development. Spent 2 years as a SQL DB developer.

Domain expert (DE5): Director of product development,

Experience: worked at the coalface of several technology start-ups over the last 10 years and have held Nursing Licence since 1990. Been in their current position for 16 months and prior to that worked within the healthcare arena in business development and pathway design.

Evaluation Stages

The evaluation is divided into three stages: Formalisation and exploration, Training and trial, and Execution. The stages represent important steps in the implementation of TEM and are a guide through the process. Further decisions on how the information is presented,

using the guidelines and concepts that TEM provides, has also been left up to the development team.

These stages include introducing the process to the team, presenting, and gathering information on the team structure and particular problem area, through training the team members and final implementation of the methodology. Each stage has diverse outcomes as presented in the following order:

Formalization and Exploration Stage

This part of the evaluation was mainly to gather data on the participants' experiences in the problem area and to learn about their daily practices as well as orientate and engage them with the topic.

Training and Trial Stage

This stage consisted of a short presentation giving the background of the research project, then explaining the methodology steps. This included a few examples on how to run the transparency patterns. After that the developers were told to first try and run the methodology on two or three of their user stories, then come up with initial questions.

The participants of this stage were developers, including the head software developer and domain expert. The participants were asked to leave comments or raise issues about the methodology including the patterns and the software requirements results. The results are in four documents (see Chapter 8). They also include semi-structured interview with the developers and the domain experts.

After the initial evaluation, a review conducted whether further implementation of the project would align with the company's objectives. Although there were some concerns about efforts, such as overheads from running an unfamiliar methodology, the usefulness in the elicitation of transparency requirement as well as the potential refinement on the requirements was deemed more beneficial.

Execution Stage

The company approved running the methodology over an entire project. They choose Spirit Hub, a diabetes course booking system which contains sensitive data (personal details as

well as medical records, as well as important and costly processes with alternative pathways that can be difficult for the targeted users to follow. Only developers who are specialist for this project ran the methodology. The project was initially scoped by the domain expert who created the user stories and the user pathways. TEM was run across the entire list of the project user stories. The resulting output of TEM, user stories, tasks and transparency requirements were identified and grouped by how it has impacted the user stories' structure. The groups are new transparency feature, better defined functionality, and user ability. They were already well defined with the user stories, and when a pattern has not been directly followed, it has helped to think about the user stories in another way.

6.1.3 Results

The results from the formalization stage are four pre-questionnaires. The results from the training stage are shown in three documents and another three semi structured interview and focused groups recording. The recorded focused groups and semi-structured interviews were mainly discussing the points and topics of the questions that have been asked later on during the day. Therefore, only data collected during these interviews that contains different information or adds information to the participants responses have been noted. This has been followed up with two kinds of post-questionnaires used to gather information from the participants, the first included technical questions and the second including question about the impact of TEM on trust and other system qualities from the domain expert opinion. Further analysis of the output from TEM compared with the original user story definitions.

The pre-questionnaires were used to gauge the current software development context within the Spirit digital team, and then post-questionnaires were used to understand how the methodology would impact that process. The Spirit digital uses agile SCRUM practices in their daily tasks. The team uses Azure DevOps to manage their development life cycle, organising user stories into PBI (product backlog items) which are then broken down into development tasks. The PBIs are arranged into two weeks sprints with free selection of PBI from the sprint by developers. PBIs are written primarily by the domain expert and senior developer and reviewed biweekly in backlog reviews. Daily stand ups manage task progression, communication with the business is managed through the product owner

(domain expert) or through the senior developer (SCRUM master). As a small team SCRUM is implemented with some individual taking on these roles as part of a wider role.

Prequestionnaires were used to gather information on the development cycle of Spirit digital team. The questions were mainly on related agile practice that are used to define their project requirements. The full results from these stages with further details are presented in appendix (see Section 8.1).

To analyse the data from pre/postquestionnaires and semi-structured interviews the responses are coded and clustered to report results and find any similarities. An initial set of themes was extended in the coding processes, some derived from research questions (see Section 6.1.1. The Table 6.1: Themes below groups the results using thematic coding. The responses from the participants are coded to be clustered and report any repetitions.

Thematic coding is used to confirm estimated themes and allow to index the text into groups, then find additional themes in case they exist [183]. Prior to the methodology the output requirements were predicted to fall into a number of themes, as shown in Table 6.1: Themes. These themes were derived from the research questions. Some further themes were discovered during analysing the team responses.

Table 6.1: Themes

Themes	Agile (AG)	Agile + Methodology (AGM)
Time and effort (TE)	(AG/TE)	(AGM/TE)
Transparency requirements (TR)	(AG/TR)	(AGM/TR)
Transparency data -related issues (TD)	(AG/TD)	(AGM/TD)
Missing requirements being covered (MR)	(AG/MR)	(AGM/MR)
Transparent of available software functions (TF)	(AG/TF)	(AGM/TF)
Quality of documentations (Easy to use/reuse and maintain) (QD)	(AG/QD)	(AGM/QD)
Limitations	(AG/L)	(AGM/L)
Newly found Themes		
Defined requirements (DR)	(GM/DR)	(AGM/DR)
Informing the user (IU)	(AG/IU)	(AGM/IU)

Time and Effort (TE)

From the responses it can be seen that all the participants agreed that agile requirements engineering practices reduce the overall workload during a project lifetime.

Moreover, participants D1, D2, D3 agreed and D4 neither agreed nor disagreed that requirements engineering practices in agile process speeds up the work in my project.

After applying TEM in their agile user stories, the developers mentioned that the workload should be the same, if not increased. D3 mentioned that is “due to a more complete description of the tasks”. Moreover, D3 mentioned that applying TEM will also speed up the work, in contrast with Developer (D4) who said that the work would not be speeded up “, but it would improve the quality of the system”.

Transparency Requirements (TR)

The results from applying TEM on the Spirit Hub application, are reported as follows from this software developers and experts, with 94 resulting requirements (see the full detailed results in (see Section 8.3). It can be observed that as familiarity of the process improved as does the resulting output. There is a clear progression of how TEM is implemented, with initial results being incomplete and lacking detail, evolving into substantial outputs with the reported results almost completely following the transparency patterns. This is supported by how the developers described their experience in the postquestionnaire. There are examples comparing the user stories prior and after applying TEM then evaluating the resulting requirements for transparency on data and system functionalities. The developers have grouped the outcome requirements into the following categories where they presented in colour coded manner to show how the results influence on the user stories.

- New Transparency feature (64 Blue)
- Better defined functionality and user ability (18 Green)
- Already well defined with the Use case (2 Brown)

- *When a pattern has not been directly followed but has helped to think about the use case in another way (10 black)*

Transparency Data-related Issues (TD)

The experts stated how important it is to report to the user what data is being collected about them. D3 reported using TEM helps them to consider the data points and interests of the system tasks and followed that up by referring to an issue where the system collects data without telling the user “Made me consider each of the data points and data interests for each task and structure the task description around them. We often collect data without telling the users what it is being used for, so it was good to considered why at each point in the application we are collecting given things. Makes you rethink the validity in collecting certain things, such as user personality ethnic details were removed from the system as we had no real reason to collect it”. This indicates how necessary it is to justify why the user’s data is collected and in case there is no justification, clearly the data_should not be collected. The developers were able to apply data minimisation (one of the pillar principles in the GDPR).

The developers stated that the data interests help in grouping all the data related the information, then added “Data interests are a great extension on the concept of data tables – you can group many tables under a data interest”.

The following Figure 6.1 illustrates a user story from the selected application and the resulting requirements. See full user story and results in the appendix (see Section 8.3), below are excerpt taking from the user stories as points of discussions:

User story 27. Display Structured Education Courses

As a patient I need to be shown the appropriate SE courses for me. This will include courses available online (referred to as Remote) and face to face group courses. I will need to see the detail of what the course is offering, where the course is being held and the date and time...

Static data → requirements result:

- 1- *The system must communicate that Uses Postcode to search for course and “accessed by system admin”. This data is anonymized and will be stored until your account deletion. This is only acted on by the system.*
- 2- *The system must communicate to the user that it holds data of type postcode and locational data which will be sent to Google via google maps and stored on their servers.*

Figure 6.1: User story 27- Static data

Figure 6.1 shows that it is critical to inform the user who has access on their data, as well as justifying where it is not clear which parties the data will be shared with.

Transparent of Available Software Functions (TF)

D4 pointed out that “Regarding other systems that I have worked with, I think there are clear benefits to a process that ensures consideration is given to explaining potentially complex tasks that the system is capable of to the user. This kind of extra detail is often not picked up by developers, so being picked up by the methodology would be helpful.

I have found that once a complex feature is better explained, then it attracts more usage. This is an example of a user trusting the system to do what it says it should. Once that detail was added, users had confidence in customising that part of the system.

Before the Transparency Engineering Methodology (TEM) it was usually the case that features would not be used due to users not knowing how they worked or what data they might affect. It was usually when someone new to the system was asking questions about it or running training sessions on it that questions raised by the TEM would have become clear.”

Another developer stated that “Really great for multi-faceted pathway use case – like the cancel course”.

This aligns with user’s mental model of the software and their expectation of what can be achieved. If at any given context transparency of the available options were clearly stated a user can make an informed decision be reassured that they are using a particular functionality correctly and in line with its intended purpose. Often software has an expectation that the user knows of particular features and how to use them which can lead to frustration and affects user experience.

D3 explained that TEM helps in maintaining user-focused development: “At the task implementation level it is all but lost in a more functional description.” So much of current software development is centred around user focused development especially the elicitation process but this link can be lost when actually under development. D3 is alluding to centring the requirements around communicating with the user as they user stories are broken down into tasks. As a result, TEM can bring user perspective right to the point of writing code.

The results from the technical evaluation illustrated software related processes requirements are being defined see full document (see Section 8.3), a useful example of this results is

User story 27. Display Structured Education Courses

As a patient I need to be shown the appropriate SE courses for me. This will include courses available online (referred to as Remote) and face to face group courses. I will need to see the detail of what the course is offering, where the course is being held and the date and time...

→ Dynamic Use case: Preconditions courses exists → resulting requirement:

In case the patients search parameters do not return results the system must communicate to the user that the search parameters are too narrow and that if they widen them to a particular degree more results would become available.

Another example, User Story 26: Electronically complete a standard template with personal details.

→ Dynamic Use case: Preconditions selected a course email address → resulting requirement: *If the current system state does not meet select the course the system must communicate that in order to create an account the patient must first select a course on which to book onto, this can be done by selecting book a course and searching for available courses.*

Figure 6.2: User story 27- 26 Dynamic Use case

Missing Requirements Being Covered (MR)

The developers have stated in their postquestionnaires that the methodology helped them to cover the missing requirements in their example applications, D3 stated that the methodology “helps fill in many of the gaps particularly around informing the user” D3 added that the user is usually only informed in errors cases, and that creates “lots of gaps as it was often down to the developers discretion” they mentioned that now with TEM” the

structure forces user to be informed about everything so you don't miss the important communications”

Additionally, the project manager mentioned that TEM aids the newer developers, D4 “I think it will make *newer developers* think more laterally and perhaps make them question themselves on how the system should respond in certain areas or situations”. It is thus clear that the methodology helps define and elicitate requirements particularly around user communication and add structure and consistency the way that user communication is managed.

Figure 6.3 presents an original user story and requirements outputs from the data and dynamic use case patterns. It shows that the resulting requirements define the pre and postconditions of the user story as well as the GDPR related data. This example illustrates how TEM can aid in bridging the gaps in the user story in a systematic way. During the focused group session one of the developers pointed on the importance of informing the users about their data and access rights, D3 also mentioned “Makes you think about the data points, if people know their rights, they are more likely the fight for them”.

The pre- and post-conditions were only defined as a prerequisite of TEM methodology, and where not included in the original user stories. In the first case, they are solved by the developers' experience and general understanding of the task. Any wider conflicts of information or significant lack of information can be solved through daily or weekly meetings with the domain experts. This often causes delay on the long run. Some issues can come across accumulatively with other issues and causes a bigger delay.

User Story 29. Receive appropriate reminders and messages about upcoming course.

As a patient Once booked onto an SE course, I need to receive a reminder of my upcoming course with details of time and place. So that This will be in the form of email or text. I will have chosen my preferred means of communication when I created my login.

Test Case 1. The patient will receive relevant reminders of their course and the course details.

→ Dynamic Use case → Preconditions, Postconditions → resulting requirement:

If the current system is within 2 weeks 1 week and 2 days of the start of the course the patient must be sent an email to remind them of the course, stating the alternate options to cancel the course, if any changes to the course are made then the admin will contact them, if they have any questions, they can contact the admin team.

→ GDPR pattern → resulting requirement:

The system must communicate to the user that it holds data of the type Referral information which admins can view and edit, SE Course providers and Educators can view.

Figure 6.3: User story 29 - Dynamic Use case and GDPR

Quality of Documentation (QD)

This is covered by questions three in (Section 8.1) and three in (Section 8.2), and the responses have revealed that the agile documentations can be “at times good”, however, there is a lack of information which can lead to assumptions. They continued to say that “Particular details and functionalities could be more defined. Occasionally a feature is implemented incorrectly to the users desire, and this can lead to lengthy changes due to bad spec in the first place”.

In regard to the quality of documentations of TEM (easy to use/reuse and maintain) (QD), developers noted that it is hard to follow at the start and got easier once more familiar. D3 stated that “Good as a whole, can be difficult to follow at times” however, “found the process got a lot easier as time went on and the process naturally streamlined itself as I become more familiar with the process and documentation”. Also, D4 noted that “I felt that it could do with more complete examples”.

Defined Requirements (DR)

This theme has been covered by the question five of the prequestionnaire when the developers were asked on their past knowledge, D1 stated that “We don’t always have the system aspects required for the task defined and available”. Also, D2 elaborated “In theory yes we do, but in practice there are many times where questions only arise while in mid development.” D3 mentioned in more details what are the missing systems aspects “Sometimes missing text or messages, sometimes missing alternate pathway and error cases, sometimes missing greater detail on the application flow and the exact process”.

This theme has been covered by the question four of the postquestionnaire, D3 was very affirmative on that TEM “Helps to define the functionality more clearly. And has a user focused bent with everything focusing on communication with the user which is often an afterthought for me.”

Developer D4, stated that the methodology helps in defining the requirements “since there is more thought given to the development items.”

Another noted example from the trail stage came from following the requirements pattern to *verbally* inform a particular part of the process to the user outside the software boundary i.e., “the clinician must verbally communicate to the user...”. This shows the scalability of TEM where it can encompass processes outside the software system e.g., prompting the clinical to verbally communicate the information to the user.

Informing the User (IU)

This theme is mainly covered by Question nine in the prequestionnaires (see Section 8.1) and eight in postquestionnaires (see Section 8.2). D1 said “informing the user about the data being collected and the process being performed should be a requirement itself.” D4

mentioned an example from their project experience on where and what they inform the user about “We have a specific text on first login to our system (CliniTouch Vie) that the patient must agree to in order to use the system”. Although that it is clear that the team has showed some transparency it is more on a case-by-case bases, rather than systematic approach across the application. Here transparency is regarded as the need to know as opposed to the right to know and so is not consistent applied throughout the applications.

After applying TEM: The developers have stated that TEM help to sustain the user focus. one of the developers noted that the methodology “Help to maintain user focused development later into the development cycle where at the task implementation level it is all but lost in a more functional description.” They elaborated further “This meant we informed the user about processes we probably wouldn’t have bothered before with but can be informative. On data collection every input is more deeply scrutinized especially peripheral data collection processes which were often overlooked”. D4 also stated that applying the methodology aids in informing the user about their data and other system aspects as it “raises the thought and discussion between stakeholders, which then if carried out will certainly help the user in their use of the system.” They highlighted that TEM aids them to reflect on the data that is stored and, how to communicate to user. During the focus group one of the team members said that TEM brings the user to the implementation and the user stories “by letting us think of what the user wants to know from each systems aspect been addressed in the user story e.g., data, operation”.

One of the developers describes “user centric view” with regards to transparency concerns mostly relates to the user information that they impart. However, a big part of personally identifiable collected data is gathered passively via behavioural, statistical and usage data. This is often not identified as a data concern and the user is rarely directly informed. The developer added “When the system collects information not directly form the user but could be critical – when attempted login – when successful etc...”. For instance, IP address or any information about view time on the website, or any other behavioural data could affect the user or used to identify users. The data interest identification process provides a way to create all different data types needed and inform the end user about them when necessarily.

Postquestionnaire Domain experts

The domain experts are asked to fill out post questionnaire based on their experience, after they learned and executed the methodology in sample examples from their applications. The domain experts filled the questionnaire based on their knowledge about their users. The questions were mainly focused on the influence of implementing the resulting requirements on the user trust with the software. More precisely on the following trust bases and system qualities:

1. *Users' cognitive trust, i.e., perceived understandability, reliability, and technical competence*
2. *Usability and usefulness*
3. *Temporary and permanent reassurance, data traceability*

The responses for the above were all positive, emphasizing that transparency “will lead to traceability throughout the application”. The participants reflected in their answers to these questions, how TEM supports achieving transparency and why this makes users trust in the system. The domain expert noticed, that TEM influences the way they think about the experience of the user (“[TEM] prompts me to think about the user experience from a perspective of trust”) DE5. But besides this general shift of behaviour, it also gives concrete assistance by “identifying places in the system where information about functionality is lacking” (DE4), which allows the designers and developers to address this lack of information by adding it. Trust is equated with making “a user feels very confident about the system” (quote from DE4, but also voiced by DE5). To achieve user confidence, the system needs to give “feedback [...] after carrying out commands” (DE4) and inform the users of their actions (DE5). But you need to be careful, because while “some users would feel reassured” (DE4) if all the TEM outcomes are implemented, others might “feel like the system is trying to tell them too much” (DE4). This concern is addressed in TEM by offering the specialist applying the method, to choose from different options, how to achieve transparency (push, pull, do not show, user settings, see 5.4.4 for details). Especially option for “user settings” leaves the users the freedom to decide, which level of information is right for them. However, selecting this transparency option needs to be

balanced with the number of settings that are presented to the user as well as possible development time and effort restrictions.

4. Users making informed decisions, building their trust, and meeting their expectations.

Domain experts also stated that, when applying TEM, the users' trust can be built, and they will have more informed decisions to make. DE5 noted that TEM will help the user to decide if they want to use the software or not. B4 elaborated on the reason stating, "since if they have a clear understanding of what various system features do then they will know whether it is the appropriate feature that they need to use at that time". During the focus group session B5 mentioned that in our domain area the users tend to assume things and build *false trust* based on them. Transparency in this case is an important requirement. B4 added that taking into consideration that users are very different, TEM will help in meeting user expectation when the system responses "validate what the user did whenever they use any part of the system". DE5 noted on the user expectations that "I think users expect transparency".

5. *Methodology characteristics.*

The domain experts have different insights about TEM characteristics, "*The methodology is concise, consistent and replicable.*" (DE5). Another domain expert noted that TEM work like and "safety net" as it is a reminder for the developers when writing their software specifications followed by the user stories, what system aspects to consider. As well as "serves to generate discussion between developers/stakeholders around points that may not otherwise be considered" (DE4)

6.1.4 Limitations of TEM

The developers and the domain experts noted several limitations of the transparency engineering methodology. They have been asked directly in the postquestionnaires how you think the methodology can be improved. Domain expert found that the time to execute TEM can restrict its use as DE4 stated "The limitations from my perspective really were *time-based*" based on the "the circumstances of the development team as to whether they could fit this detailed analysis into their plans" (DE4). Their suggestion was to bring more

complete and concise examples so that they developers have better expectations and comprehend what is needed to be used for. Another limitation was the structure of the methodology, they noticed that it can be improved so that it become easier to follow. As well as having to group the results themselves can results in inconsistency of the different parts of the application and there is no information presentation unified for the application. During the execution stage some of the results pointed out on consistency issues, where each developer can potentially come up with different style of requirements over the same project. D3 “How it fits together on the application level is the important thing – I can see the same things being repeated and to be able to standardize them across the application would be great”. In the same application D3 noted that not having fresh mind on the results can cause the consistency problem saying, “I tried to use TEM without looking at the user stories details only from the title, that give me fresher mind and the helped to bring more information and consistency to the results”. The consistency issue over the data can be addressed with Data Interest Identification Process which provides a systematic way to generate groups of data types that include a definition of the user encompassing all their data, which the system can communicate to the user when needed. However, this process needs to be used hieratically using top-down approach.

Additionally, D3 noted that the transparency requirement results can overwhelm the end user, with on way to define the type of the user “and the amount of information that they want to see”. The solution to that is by specifying different types of user (see Section 5.4.1), however this means that this option need to be clearer in the condition Section. DE4 also noted on the step two of the methodology where the unimplemented system related goals are clusters by the reasons why they are not implemented for the user to be informed about them, is not necessary as no data would be recorded of these goals or carried out after they are being discarded. However, it is important to note that the developers have not run the presentation choices themselves in this evaluation which potentially help in how to communicate the information to the user over the application (see Section 5.4.4). This issue is partly addressed using the concept of Condition where the communicating a certain information to the user need to be related to their current interaction with the system (see Section 5.4.1). However, the team did not use the concept to its full options and usages, at some points of the training stage, it was important to show the developers and remind them

to use the conditions where they can generate positive or negative expressions depends on their needs. Moreover, the current version of TEM allows the specialist to have the final decision on how the information is communicated to the user with the help of presentation choices. However, that might be not enough, therefore, integrate TEM with an evaluation criterion of the non-functional transparency (where it defines the acceptance criteria) can be done in the future work Section.

In this evaluation the domain experts' feedback and responses are collected on how TEM can enhance user trust. A complete evaluation of the methodology would be ideal, including measuring the impact on user trust, but is out of the scale of this thesis both in terms of time and resources. Therefore, it remains as part of the future work to fully evaluate the methodology on its results on user trust (see Section 7.2.5).

6.1.5 Interpretations of the Analysis

Strengths

This study proved that Transparency Engineering Methodology provided a conceptual framework for transparency requirements that has the capability to capture all transparency requirements for the end users. Once all transparency requirements have been exposed, the requirements engineers and developers can decide what information to show/hide by carrying out a conflict check with other system requirements.

TEM systematically generates transparency requirements and increases the develops ability to structure directly actionable requirements in their applications. The methodology takes in the project high level user stories and depending on the given data interests and processes and generates the required transparency requirements, spanning from who has access to the data to what processes are occurring. TEM, thus guides the developers and requirements specialists to focus on transparency and trust. This is not generally, triggered by traditional requirements engineering methods. Further to this it allows the TEM users to construct the right mindset to think about ways of folding transparency and trust into a system, as users concerns are central to the manner in which the resulting requirements are structured TEM

is useful for bringing the end user's perception to the implementation stages and further drives user focus process to the point of writing code.

Requirement refinement with descriptive user focus, TEM generates transparency requirement, but further to this it generates more complete and more user focus requirements. All outputting requirements are described in terms of communication with the user and the data and operations that are being carried out. This focus on communication seems to give the requirements a holistic nature as all the facets are first gathered and TEM generates a requirement which both includes the operation and the communication. Further to this TEM can directly help fill gaps in requirements by identifying differentials between the pre- and post-conditions and data processes.

Data interests as a way of grouping similar data concerns. The complexity of many applications often means that data models are acted on in multiple location within an application. Data interests span across the whole application grouping similar data concerns and the ways in which the system must be transparent about them together. All the developers recognised the important to create data interests as a beneficial way of grouping data types. Specifically, user data is any data that can identify the user, e.g., personal data, behavioural data...etc and so this can be grouped into a data concern maintain a consistent feedback loop with how this data is being acted upon. The creation of these data interests should be an iterative process and distributed throughout the team via a top-down approach for the consistency reasons.

TEM Exposes transparency and uses of data, particularly secondary processes which are often ignored. Details of what a user needs to know about system aspects and data operations are generated by TEM on all aspect of the system. This expose information about the user which the user has a right to know about and aids in the requirements specialist and developer's decision on what information to show. The hope is that this process is further formalised to a standard, as described in the future work. Too often through other standard elicitation method, this information is not accessible due to not being directly related to the systematic functionality of the application and thus ignored. How a user's data is used in secondary processes was all but completely ignored in the initial elicitation process carried out on the Spirit Hub application. Thus, this indicates that

it can be helpful to place TEM in the development process with other requirements engineering processes, optimizing its use with other models and frameworks. For that purpose, transparency elicitation process has been suggested in the future work (see Section 7.2.3) driving transparency to be considered earlier in the product development cycle.

TEM fits in with agile and SCRUM practices. TEM has been designed to be suitable for agile practices. The Spirit Healthcare development team uses agile practices in their applications and implementation of the methodology fitted well within their team structure. As agile is in the predominant software development framework, it was essential that TEM was developed to support these practices. Current limitations in agile development are that the requirements are not well defined, and the users' perspective is not completely included, because it focuses on rapid iteration results (see Section 2.3). TEM contribute to addressing this issue by extending the agile process meaning teams can continue to work in a similar familiar structure whilst more deeply rooting the users' considerations into the process.

TEM helps the users in achieving their goals with the system. TEM guide the user through a system by describing the available and enabled options and showing reasoning why the system is acting in a certain way when needed at a given context.

Weaknesses

Learning curve. The results from the training and the trial stages show that the developers and the project manager in their first trial seemed to complete the requirements from the related knowledge of that system aspect, without necessarily directly following the pattern. As more of the user stories were elicited there was a substantial improvement of the quality of the output and adherence to the framework. This shows that there is a degree of learning how to implement the methodology. The developers tend to create their own templates to fill out the results the way they see it working. One of the reasons that the developers mentioned is the structure is not clear in some cases. There are several mentioned reasons: to cumbersome, did not understand them fully at that stage, time constrain etc. Although it should be said that as they spent more time with the methodology, they described the process as faster and clearer (see Section Quality of documentation in 6.1.3).

Consistency issue. There was a concern regarding the consistency in the outputs from different patterns and data interests. After one of the focused group sessions, applying the

Data Interest Identification process needed to be clarified that should be used hierarchy. To rationalize the use of patterns as a way of assisting developers in handling the data interest, the head of developers generates the data interests first and share them with the team. This should be clearer in the methodology description.

More knowledge about the problem area might be required in order to get the most out of the engineering methodology. There are decisions and conflict check need to be taken by the developers and requirements specialist during applying TEM, to decide what information to strongly display to the user, what information need to be push or pull (see Section 5.4.4). As well as checking when information discloser might cause any data breach. It is important that the requirements specialist knows well about their targeted audience and application domain to make these decisions.

TEM structure can be difficult to follow at times, the methodology has several steps and patterns and can be difficult to discern which ones to select at times, as the current structure is open and not fixed to prevent that.

6.1.6 Threats to Validity

Although the evaluation is conducted with an entire development team, this team only consists of 5 members. That could potentially affect the scale of answers where some responses can be biased as it relies fully on the members past and current experiences.

The project's application is relatively small consist of only 29 user stories, this was a manageable size but might have not stretched the methodology to it full capability and hitting all its limitation. As a result, the scalability of the methodology was also not tested to make the methodology more broadly applicable using transparency elicitation process and standardization approaches (see Section 7.2.3 7.2.1).

Additionally, the methodology has only been evaluated within the company's current agile practices which makes it difficult to distinguish between poorly implemented agile practices and the mythology's results. The interviews and focus groups helped in defining the line between the two. Further the methodology was only compared to agile which was deemed acceptable as the current industry standard.

Some of the questions could have been designed better, as some of the developer's responses were based on general opinion instead of their particular experiences, for instance the questions about the agile practices.

This evaluation used qualitative analysis methods; however, this has a number of limitations (see Section 3.1.1).

Another weakness with evaluating the results with the company, is that the answers can be biased and effected with the company environment. For instance, some responses from the management team can be diplomatic as oppose of lab environment. To overcome this the methodology could be implemented on a larger scale across multiple companies and projects (see Section 7.2.1).

6.1.7 Discussion (summary)

TEM has been shown to systematically generate transparency requirements leading to the user being informed about how their data is being used and who has access to it. This degree of transparency is extended to backend and passive collection processes where informing the user is often overlooked. Although the methodology takes some time to get used to, as illustrated by the quality of the output and comments from the developers it generates a holistic user story with informing the user at the centre. The responses and the technical results show that the methodology is useful for defining the requirements more systematically with user communication being the key consideration.

Perhaps a serious disadvantage of this method is that the current structure of the steps can be difficult to follow without placing TEM in the development process, therefore, a transparency elicitation process has been introduced in the future work to tackle this issue (see Section 7.2.3).

Integrating transparency across the software system has been unaccomplished task, to make it more possible TEM is designed to fit with agile practices and management tools (see future Sections 7.2.1). To build transparency contextually and in relation to the user goals with the system, that needs a method where it can be embedded in the daily management tools i.e., first time an event happens it might require to present information in order to guild the user in achieving their goal. However, later this information might become less

relevant. Therefore, the concept of conditions is necessary (see Section event-based transparency Section 6.3.2)

6.2 Related Work

Engineering trust and transparency in software systems is an underdeveloped area of research which has been left behind as the main focus has been to drive rapid results, efficiency and quality. Therefore, only a few studies have investigated how to engineer transparency or trust in a software system. Here is a number of key papers in the area have been selected to further investigate their findings. These papers study models, taxonomy, modelling languages and requirements patterns to engineer transparency and/or trust in a software system, where trust and transparency were examined as non-functional requirements.

Since transparency is a complex quality and can hurt as well as improve trust, Meunier et al. [172] have developed a different notion called Software Purity which refers to a set of standards that the software has to satisfy in order to establish user trust. In addition, user data would not be used for other purposes different from that which the system is clearly intended. In this research, transparency is being addressed to enhance trust as TEM is motivated by trust and designed to increase user trust.

Hoffmann et al. [14] stated that users often adopt trust to help reduce complexity and simplify the interaction with a piece of software. Trust is a social contract in that way, where instead of agonising over each interaction a relationship develops and the user entrust an input will be acted on in a particular way, including all subsidiary processes and uses of data. As a result, the researchers stated that trust is not directly tangible and must be analysed through collecting the antecedents of trust. Trust antecedents are “factors or elements that build trust”. The researcher collected antecedents derived from the existing studies and performed a systematic literature review selecting only on the studies on trust in automation. Required patterns were designed around addressing the antecedents and created high-level goals with related forces to drive the domain expert into making decisions with trust more as a focus. The patterns’ results express the goals that the system should satisfy. The patterns present the high-level goals that the system shall achieve without formalising

how to achieve them. These goals are designed to help requirement analysis on deciding whether the pattern is applicable to the system or not. The five patterns' results are Explications of Intentions pattern, to satisfy the systems which should say clearly what "it will act on in a particular way". The second pattern is Understandability, its template solution is saying that "The system shall provide information about current activities". The third pattern is Transparency, its goal (problem) to "satisfy the user need of easily discern the system" and template (solution) says "The system shall provide information for how the output was created." The last two patterns were Information Accuracy with the goal that "The system shall provide possibilities for the user to select data that is applied by the system", and Personalization pattern with the goal that "The system shall provide setting options for system functionalities." The aim of Hoffmann research is to "help requirements analysts to address trust on a basic level". The researcher stated that other research needed to achieve that in a detailed level. That is where is research stands and contribute to addressing the mentioned goals with TEM (see Section 0). TEM helps to specify in more details how to achieve these goals over the systems functional requirements. Thus, helping in reducing the open and abroad interpretations for goals by a team member.

Transparency is examined as a non-functional requirement in Marcio et al. [2] as they believe it "rarely can be satisfied" thus can only be accepted within limits. The researchers used i* to model transparency for that same reason, where they considered intentions behind being transparent on the company actions. Their work focusses on understanding the relationship between transparency and other system qualities like security, trust, and privacy. The researcher used Soft-goal Interdependency Graphs (SIG) to capture these relationships, then used ontology and semantic web later on to collect information about transparency and other non-functional requirements. Likewise, their work also examined the relationship between trust and transparency and found that their relationship can be either positive or negative, however, this research has also found that in some cases they cannot affect each other. Their main view on software transparency is that it can be categories to low or high based on what the system delivers. In this research, transparency is analogous as correctness (see Section 4.3.6); there is no high-transparent software, a system is either transparent or not. Marcio work focuses on information transparency while our research work focuses on both information and process transparency (see Chapter 5).

Hosseini et al. [184] stated that achieving transparency is hard, and therefore, defined transparency usefulness as a measurement to transparency in the system. Transparency here is presented as a non-functional requirement, and it is useful once the user can act on the information provided. To evaluate if the information is transparent, they created a checklist where at the top of this list is the stakeholder actionability where they can act upon the information perceived. The researcher divided transparency into four facets to address the concept and help its integration in business information systems. These facets are stakeholders, meaningfulness, usefulness, and information quality. The facet of meaningfulness refers to the information understandability from the stakeholder and intentions behind providing this information. In the context of their research, the depth of information is defined by data transparency, process transparency (how this data is performed) and policy transparency (why an action is performed). To address all that a modelling language has been created to be used by software engineers. Hussein research studies from a holistic point of view the information provider and receiver where he examined the different level of transparency needed for different stakeholders. Their research on transparency focused on social-technical systems. Similarly, the facets described in Hussein's research are satisfied in this study. However, the research focus was on the end-user. TEM starts with the user goals and generates transparency requirements based on them and in relation to other system requirements. Therefore, it is designed to help users in achieving their goals with the system.

Some authors examined transparency with its interrelations with privacy [6] [162], while others examined transparency with its dependencies with other system goals. Leite [80] have discussed transparency and presented as a non-functional requirement in the context of software engineering as well as organizations. To present what he called transparency knowledge, the researcher defined the concept in a graph consist of 33 soft goals using Softgoal Interdependence SIGs. The graph describes the dependency between the nodes of the graph and transparency. These nodes are all related and influences transparency. Transparency is on the top of the graph then the second level consists of: accessibility, usability, informativeness, understandability and auditability which they have direct influence on transparency. TEM adds to that by addressing transparency with the functional requirements of the system aspects.

Marcio et al. have similarly demonstrated the relationship between trust and transparency in software systems literature. They also found that “trust can either improve or hurt real transparency and vice-versa”. However, they mentioned that there are further and deeper investigations needed. Our research contributes to that with more in-depth analysis on the relationship between the concepts (see chapter 4).

Research has examined the impact of transparency in many ways. Yu [77] have studied the impact of transparency on the effectiveness of requirements documents. According to Yu, more transparency can help in achieving more effective documentation. They developed a working definition of transparency where it consists of three attributes: relevance, accessibility, and understandability and found that more transparent requirement documents aid the stakeholder to “spent less time, answered more questions correctly, and were more confident about their answers”. They measured efficiency “by evaluating how much time participants spend [...] measure satisfaction by how participants feel about using the requirements documents”. This research has studied the impact of transparency on the effectiveness of the functional requirements of the system, and designed TEM to encourage the effective implementation of these requirements. Additionally, TEM has been evaluated similarly with expert evaluation and case study. TEM also addresses the above-mentioned attributes of transparency by generating transparency requirements that help the end-user to access their data and system operations, so they can achieve their goals and understand the system better.

Meis et al. [12]. Their research has mainly examined the flow of the personal data in a system in order to generate the static requirements of privacy that are related to personal information and its corresponding transparency requirements. Those requirements are static in the sense that they help the user understanding what data the system hold on them, but not necessarily about changes that may happen to this data, or how to execute their rights with respect to the data. In our research we develop a set of static and dynamic patterns that generate transparency requirements about user data. These patterns help to inform the user about their data rights under GDPR, who has access to their data, how it has been stored, et cetera. For instance, the user can be informed when data is collected, edited or accessed, as well as they are given the control on who accesses their data. Further to that, we focus on

the application of the patterns in agile practices of development to help the requirements analyst and developers to generate and implement the resulting requirements.

6.3 Extended Discussion on TEM

To elaborated in the discussion further, the following sections explain some of the technical motivations behind designing and developing the engineering methodology.

6.3.1 Transparency Patterns by Motivation Types

The literature review demonstrates that key to implementing trust via transparency are information disclosure, in particular around the users' data, and not violating user expectations. Moreover, transparency must mediate the interaction of the user with the system enabling them to make effective trust judgements. To that end traceability of system states can be used to implement many of these facets and factors. Traceability is the process of revealing to the user, pre- and post-conditions, keeping the user updated about their current and previous data instances and providing available options open to the user at any given point. Leite [82] states that transparency is needed to allow back and forward traceability in the software. TEM is designed to support this by using requirements patterns. Transparency patterns motivation types are distinguished and classified based on their goal and impact on helping the user as follows:

1. **Guidance/ Usefulness/ Utility Transparency:** This type of transparency is mainly used to aid the user in achieving their goals. E.g., operation transparency helps in guiding users through system functions. This transparency has a pre-active role when an action might lead to failure or will have negative consequences. Pre-active is when it applies before the action occurs.
2. **Traceability Transparency:** Traceability can be broken down into two subgroups, temporary and permanent:
 - **Temporary Traceability Transparency/Reassurance:** This type of transparency has a post- passive role for data and activities in the system. Post-passive is when it applies after the action occurs. The two main benefits of this type are helping meeting users' expectations and, therefore, their

satisfaction. The patterns motivated by this type uses push presentation. Moreover, it can be pre-active for the other transparency types. According to Nielsen.s [185] “The system should always keep users informed about current state and actions through appropriate visual cues and feedback within a reasonable time”, e.g. button click.

- Permanent Traceability Transparency/Reassurance: In this type, the user usually queries the information from the system. The patterns motivated by this type uses Pull presentation. This type has post-active and long-lasting effect for monitoring proposes, e.g., balance log.
3. Learning Transparency: Here, the user usually queries the information from the system. The patterns motivated by this type uses Pull presentation. The information is not necessarily needed for a specific process, but it can present general knowledge about the system.

Table 6.2: Transparency patterns by motivation types

Transparency motivation types	Role with system actions	Presentation choice	Transparency patterns
Guidance/Usefulness/ Utility	Pre- active	Pull or Push	Use case patterns
			Process patterns
Temporary Reassurance/Traceability	Post-passive& Pre-active	Push	Data pattern
Permanent Traceability	Post-active	Pull	Data Type patterns
			Data pattern
Learning	Pre-passive or post-passive	Pull	Data Type patterns
			Static Use case patterns
			Static Process patterns

6.3.2 Event-Based Transparency

Information needs to be relevant to the user's current goal with the system. That is an essential factor to not overwhelm the user. Additionally, users tend to forget systems functionalities and need to be always familiar with the system's terminology and presentation. In the designing of software, research does not rely on the user's memory, because users forget details about the system over time [185]. TEM is intended to address this issue with the concept of condition (see Section 5.4.1), which been based on the following analysis.

Information Transparency: The literature review on trust and transparency has shown that information can assure the user on the software qualities and functionalities, e.g., how the system is secure and safe. However, in direct contrast, too much information can do the opposite by overwhelming them. This can be solved by only providing information when the user needs in a contextual manner. Thus, it is vital to realize that in case of information transparency, the needs of a user are not always necessarily constant over time, they can depend on events happening in and around the system. To process this, when defining the transparency requirements, the requirements specialist should also take event-related information during their requirement analysis stage and type of transparency information (data or operation transparency). This approach will help in showing boundaries, only showing the current relevant information, preventing overload. The time-related event-related can be analysed from the pre- and post-conditions of the use case of that relevant event.

The requirements specialist needs to identify possible events in the system where transparency is required. Events are system state changes based on context changes or interactions with the user. For each event (e), the requirements specialist then has to identify the transparency requirements associated with the event. For instance:

- Balance – Needs to always be visible => e. Always
- Fee – Needs only to be displayed if action costs a fee => e. FeeTriggered
- Exchange Rate – Needs to only be displayed, if money is transferred to a foreign country, with different currency => e. TransferForeignCurrency

The outcome of the methodology: Transparency Requirements:

For example:

- If the user tries to Withdraw Money in Germany, the system needs to be transparent about Balance.
- If the user tries to reach Withdraw Money in Germany and Fee is triggered, the system needs to be transparent about Amount of Fee.
- If the user tries to reach Withdraw Money in Germany and Money Exchange is triggered, the system needs to be transparent about Exchange rate.
- If the user tries to reach Withdraw Money in Germany and available balance too low is triggered, the system needs to be transparent about Check available balance, if enough, allow to withdraw, otherwise do not allow to withdraw.

Chapter 7

7. Conclusions and Future Work

7.1 Conclusions

This research aimed to help requirements engineers to manage transparency through a systematic transparency requirements generation methodology in order to enable users' trust judgement by providing them with the relevant information about the related system aspects. It considered the trust relationship between the system and the stakeholders, specifically between end-users and software, and ever more critical relationship as our reliance on technology continues to increase. To achieving this, a definition of this complex quality of trust in the context of software engineering was constructed by looked into its interdisciplinary definitions (see Section 2.1.8).

In addition, this research provides a summary of the relationship between trust and transparency in software systems in different domains and contexts. It includes a clustering of current views from the literature, combined with a concept map developed and used to show the relationship between trust and transparency. A cluster analysis aids in the delineation of trust and transparency and assists in understanding the need for engineering transparency requirements aimed at improving trust in software. Upon reviewing the selected articles, a list of high-impact factors, qualities, and highly reported contexts were identified. Gaps were highlighted pointing out limitations in the existing literature. Our analysis concluded that in order to clarify the relationship between trust and transparency, there are the following steps. Firstly, it is necessary to define the facets of both notions precisely so that these definitions are applicable across a range of sources. Secondly, it is important to consider the properties, factors, qualities, and pitfalls as they provide a clear utilization of the relationship between trust and transparency. This analysis helped us to decide how to integrate transparency properties that enhance trust in their application domains.

Presently research has studied transparency in relation to non-functional requirements. However, there is a gap in research on transparency in relation to functional requirements. In addition, due to the complex nature of the transparency in software, this research shows that it is impossible to achieve transparency for the entire system. However, it should be elicited for individual system aspects.

This research developed an engineering methodology primarily designed for the requirements specialists to help in making decisions and generate transparency requirements in their software. However, it has also been designed to support other stakeholders applying requirements engineering practices, e.g., for developers to refine their requirements more systematically. Crucially, the transparency engineering methodology was developed and designed taking trust facets into consideration. This methodology identifies where a software system is lacking transparency, based on its given user goals, resulting in a more transparent system, which aids users to achieve their goals both through a greater understanding of the system and its functionalities and data (e.g., where their data is stored and how it is processed and by whom). The work has been evaluated by a case study from a health care company. Currently, the transparency engineering methodology works with functional requirements by encouraging the effective implementation of different functional requirements and providing high-quality information to the user to enable their trust judgment.

The results from the evaluation showed that TEM is effective in bringing the end-users' perception into the development process and prompting the developers and requirements specialists to consider the system aspects that the user needs to know about in terms of their data and operations. Furthermore, implementing TEM has advanced the concept of exposing information to the user, which is not directly accessible, thus ignored by standard elicitation methods. The transparency engineering methodology was therefore successful in providing a systematic way for the elicitation of transparency requirements, as well as in informing the users about the software functionalities and data in order to help to achieve their goals with the system.

Furthermore, TEM was effective in integrating the pre- and post-conditions and data processes with a user-focused development procedure, both directly generating transparency requirements and inducing a user-focused development mentality, and aids developers to make user-based decisions.

TEM was developed to support agile requirements engineering practices. A current limitation in agile development is that the requirements are not detailed enough, and that the users' perspective is not included, because the methodologies focus on rapid iterations (see Section 2.3). TEM addresses this issue by extending the agile process, which means that teams can continue to work in a familiar structure whilst deeply rooting the users' considerations into the process.

TEM addressed some of these issues by generating more detailed requirements which were centred around the communication with the user. Most systems are based on user interaction, but often, when speed and efficiency are priorities, communication of information about a system function or process, their data, state, and goal, can be overlooked. TEM enforces this communication and in doing so, yields complete and clear use cases with a closed loop of communication that put the user in the centre of the description.

7.2 Future Work

There are several streams to this research that can be further investigated in the future. This list of suggestions for future work has been developed based on the main contributions of this research, extrapolating ideas and concepts for further logical conclusions.

7.2.1 Standardisation

ISO Standards: Currently, transparency of software is solely down to company discretion and thus mainly driven commercially. Future work could aim to standardize transparency principles for privacy and safety goals where companies will only be certified as transparent if they adhere to the defined practices. Some recent studies suggest a movement in this direction [186].

Intentions are important when it comes to transparency as seen from the evaluation D3 “the data could be collected for one thing and intended to be used for another”. TEM is aimed to

show the intentions of collecting the data; however, the current approach of the transparency engineering methodology is company-focused. With wider implementation across a number of companies and projects, key aspects of the approach can be abstracted out for creating a standard. This standard could then be enforced through customer consensus or even legislation. A plan could be to engage the ISO standards and principles [187] to be applied directly in the transparency engineering methodology, and that should help the requirements specialist and domain expert to follow them. That can also help with the consistency issue that the team mentioned during the evaluation (see 6.1.3).

Project management to Framework: Future work could include TEM being standardized for project management tools such as Microsoft DevOps, which could potentially improve the issues with consistency stated in the limitations. Project management tools can be described as an amalgamation of individuals, processes, and products in order to ensure uninterrupted distribution of value to end-users [188]. They are therefore good platforms on which to integrate requirement engineering. This can help in applying TEM over the daily practices and reduce the consistency issue over the application. The structure of this framework helps in generating the Data Interests as desired using top-down. [171] mentioned “hierarchy of importance and effectiveness of information quality dimensions on the perception of transparency, and their impact on decision-making processes”. TEM, on the other side, can support bridging the limitation of transitioning the changes of the requirements over the system [189] and how it can be carried and maintained throughout the development processes.

7.2.2 Transparency for Non-functional Requirements

Currently, as described, the methodology focused on functional requirements. In the future, this could be further expanded to include non-functional requirements. Some non-functional requirements can be directly judged by the end-user while using the system and will have no need for additional means of transparency. However, other non-functional requirements can be harder to recognized or judge. If this is the case, displaying the indicators will be the way to be transparent. For example, the non-functional requirement “responsiveness” is indirectly “visible” to the user experiencing the response times of the system. The non-functional

requirement “security”, however, is not visible for the user and would take time to determine, even if they have the required knowledge (e.g., by trying to hack the system and failing). Thus, the transparency requirement will have to define other cues demonstrating “security” to the user such as test certificates of trusted organizations, progress messages indicating security steps (e.g., “encrypting message”, “sending an encrypted message through secured channels”), clean and sophisticated look and feel to convey that the application was not put together hastily and insecurely, etc. Some of the latter transparency requirements for non-functional requirements/features might be the same for different non-functional requirements, covering several non-functional requirements at the same time, e.g., “encrypting message” message covers security as well as privacy. Getting the balance right between transparency and exposing secrets which could compromise a system is vital.

7.2.3 Transparency Requirements Elicitation Process

The literature has been reviewed around methods and approaches used to embed transparency in the development phase of the software. The goal is to identify methods and approaches to include TEM in the development phase of software. As a result of the evaluation, it was concluded that even though TEM can be implemented as a standalone methodology once the prerequisites are met (see Section 5.5), it is for maximizing and achieving effective TEM results in the ongoing development of unconscious operations to be embedded from the early stages of the development process of use.

Future work can include a transparency elicitation process, to address the identified concerns from the evaluation with regards to overwhelming the user with information (see Section 6.1.4). This will aid the developers and requirements specialists in gathering transparency requirements by eliciting the related transparency input during the early stages of development. Additionally, to reduce the incidences of overwhelming the user with the information. This proposed process explains how TEM benefits can be maximized. Transparency in the context of this research is analogous to correctness (see Section 4.3.6). To integrate transparency from the early stages of the development process of software, the following approach introduces elicitation processes containing TEM which result in a list of transparency requirements for the developers to engineer into the system, to indicate the

relevant information being disclosed or hidden to the user based on the system aspect. This could, therefore, increase users' ability to perform trust judgement (see Section 2.1).

From the developer's perspective, the approach aims to find out what system aspects to be transparent about to improve the users' trust judgement. The user's needs and expectations are the starting elements of the system to be developed, prototyped, or the system currently running a pre-requisite before being able to apply the approach. An illustrative example can be the expectation of a user that all videos are considered when creating a recommendation list for them. Being transparent about the fact that the YouTube recommender system does not consider "any makeup tutorials" for male viewers as the gender is a precondition of that process, that increases their judgment on the question if YouTube recommender can find relevant videos for them or not. Furthermore, in the future work of this research project aims to integrate transparency into a software system using (user-centric) process. This process would potentially use and improve the users' *mental model*. Along with the process, there are elicitation methods and techniques that have been suggested [140].

7.2.4 Refinement Mechanisms

Refinement mechanisms ensure consistency over the application, and over the other organization applications when required. The future work of this research could suggest requirements templates that contain the broken-down set of requirements of a high-level requirement resulting from TEM. TEM seemed to naturally create transparency blocks of repeat transparency requirements in multiple areas of the application and across an organization. The current TEM solution for that is to gather similar use case and data processes that the application has for a particular pattern. However, the requirements templates would help in creating reusable solutions and maintaining consistency in displaying similar transparency. Any further change to these templates would persist across the whole application, so it does not have to be updated in each instance, helping to keep implementations of these requirements more robust. After doing so, the plan is to apply TEM on bigger projects with Spirit Healthcare, which time constraints did not allow for in earlier stages of the research.

7.2.5 Trust Measurement

Another suggestion could include monitoring the state of trust. Currently, the focus of this research was on how to engineer transparency. To measure the impact of transparency on trust, the level of trust of a user to system needs to be measured for an objective comparison between the same system with and without transparency requirements implemented.

7.2.6 Sustainability

Future work could also consider integrating transparency and trust to improve software *sustainability*. An opportunity started to be looked into during the first year of this PhD project [16] The work of the paper focused on the social dimension of sustainability, specifically on its trust requirements. As noted by Goodland [70], “Social sustainability [...] create[s] the basic framework for society”. Further research on how transparency requirements influence other system qualities is necessary in order to examine the research work results for sustainability.

References

- [1] H. Kniberg and M. Skarin, *Kanban and Scrum-making the most of both*. Lulu. com, 2010.
- [2] L. M. Cysneiros, "Using i* to Elicit and Model Transparency in the Presence of Other Non-Functional Requirements: A Position Paper," in *iStar*, 2013: Citeseer, pp. 19-24.
- [3] Y.-C. Tu, "Transparency in software engineering," ResearchSpace@ Auckland, 2014.
- [4] H. Felzmann, E. F. Villaronga, C. Lutz, and A. Tamò-Larrieux, "Transparency you can trust: Transparency requirements for artificial intelligence between legal norms and contextual concerns," *Big Data & Society*, vol. 6, no. 1, p. 2053951719860542, 2019.
- [5] K. Schwab, A. Marcus, J. Oyola, W. Hoffman, and M. Luzi, "Personal data: The emergence of a new asset class," in *An Initiative of the World Economic Forum*, 2011.
- [6] R. Meis, R. Wirtz, and M. Heisel, "A taxonomy of requirements for the privacy goal transparency," in *International Conference on Trust and Privacy in Digital Business*, 2015: Springer, pp. 195-209.
- [7] D. Spagnuolo, A. Ferreira, and G. Lenzini, "Accomplishing Transparency within the General Data Protection Regulation," in *ICISSP*, 2019, pp. 114-125.
- [8] H.-H. Herrnfeld, "Article 67 Data protection by design and by default," in *European Public Prosecutor's Office*, 2020: Nomos Verlagsgesellschaft mbH & Co. KG, pp. 513-514.
- [9] D. H. McKnight, P. Liu, and B. T. Pentland, "Trust Change in Information Technology Products," *Journal of Management Information Systems*, vol. 37, no. 4, pp. 1015-1046, 2020.
- [10] W. Creed, R. E. Miles, R. Kramer, and T. Tyler, "Trust in organizations," *Trust in organizations: Frontiers of theory and research*, pp. 16-38, 1996.
- [11] K.-U. Schanz, "Maintaining stakeholder trust in difficult times: Some fundamental reflections in light of the credit crisis," *The Geneva Papers on Risk and Insurance-Issues and Practice*, vol. 34, no. 2, pp. 260-270, 2009.
- [12] F. Bannister and R. Connolly, "Trust and transformational government: A proposed framework for research," *Government Information Quarterly*, vol. 28, no. 2, pp. 137-147, 2011.
- [13] M. Hosseini, A. Shahri, K. Phalp, and R. Ali, "Engineering transparency requirements: A modelling and analysis framework," *Information Systems*, vol. 74, pp. 3-22, 2018.
- [14] A. Hoffmann, M. Söllner, H. Hoffmann, and J. M. Leimeister, "Towards trust-based software requirement patterns," in *2012 Second IEEE International Workshop on Requirements Patterns (RePa)*, 2012: IEEE, pp. 7-11.
- [15] D. H. McKnight, "Trust in information technology," *The Blackwell encyclopedia of management*, vol. 7, pp. 329-331, 2005.
- [16] B. Zieni, R. Chitchyan, and R. Heckel, "Trust as a sustainability requirement," in *Proceedings of the 6th International Workshop on Requirements Engineering for Sustainable Systems (RE4SuSy 2017), Lisbon, Portugal, 2017*, vol. 5.

- [17] J.-N. Lee, M. Q. Huynh, and R. Hirschheim, "An integrative model of trust on IT outsourcing: Examining a bilateral perspective," *Information Systems Frontiers*, vol. 10, no. 2, pp. 145-163, 2008.
- [18] M. Söllner, A. Hoffmann, H. Hoffmann, and J. M. Leimeister, "How to use behavioral research insights on trust for HCI system design," in *CHI'12 Extended Abstracts on Human Factors in Computing Systems*, 2012, pp. 1703-1708.
- [19] F. Moyano, C. Fernandez-Gago, and J. Lopez, "Building trust and reputation in: A development framework for trust models implementation," in *International Workshop on Security and Trust Management*, 2012: Springer, pp. 113-128.
- [20] V. E. Solutions, "Verizon 2014 data breach investigations report," *verizon.com verizon.com*, 2016.
- [21] D. Spagnuolo, C. Bartolini, and G. Lenzini, "Qualifying and measuring transparency: A medical data system case study," *Computers & Security*, vol. 91, p. 101717, 2020.
- [22] A. Rossi and G. Lenzini, "Transparency by design in data-informed research: A collection of information design patterns," *Computer Law & Security Review*, vol. 37, p. 105402, 2020.
- [23] M. S. Featherman and P. A. Pavlou, "Predicting e-services adoption: a perceived risk facets perspective," *International journal of human-computer studies*, vol. 59, no. 4, pp. 451-474, 2003.
- [24] R. F. Kizilcec, "How much information? Effects of transparency on trust in an algorithmic interface," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, 2016, pp. 2390-2395.
- [25] H. Cramer *et al.*, "The effects of transparency on trust in and acceptance of a content-based art recommender," *User Modeling and User-adapted interaction*, vol. 18, no. 5, p. 455, 2008.
- [26] M. Hosseini, A. Shahri, K. Phalp, and R. Ali, "Foundations for transparency requirements engineering," in *International Working Conference on Requirements Engineering: Foundation for Software Quality*, 2016: Springer, pp. 225-231.
- [27] R. T. Mercuri, "Trusting in transparency," *Communications of the ACM*, vol. 48, no. 5, pp. 15-19, 2005.
- [28] E. A. Lind, "Transparency, trust and public value," *OPENING GOVERNMENT*, p. 87, 2018.
- [29] H. Sofyani, H. A. Riyadh, and H. Fahlevi, "Improving service quality, accountability and transparency of local government: The intervening role of information technology governance," *Cogent Business & Management*, vol. 7, no. 1, p. 1735690, 2020.
- [30] S. Moghaddam, M. Jamali, M. Ester, and J. Habibi, "FeedbackTrust: using feedback effects in trust-based recommendation systems," in *Proceedings of the third ACM conference on Recommender systems*, 2009, pp. 269-272.
- [31] D. J. Kim, D. L. Ferrin, and H. R. Rao, "A trust-based consumer decision-making model in electronic commerce: The role of trust, perceived risk, and their antecedents," *Decision support systems*, vol. 44, no. 2, pp. 544-564, 2008.
- [32] U.-V. Albrecht, "Transparency of health-apps for trust and decision making," *Journal of medical Internet research*, vol. 15, no. 12, p. e277, 2013.

- [33] M. Alias and Z. Suradi, "Concept mapping: a tool for creating a literature review," in *Proceedings of the 3rd International Conference on Concept Mapping*, 2008, vol. 3, pp. 96-99.
- [34] W. Hayes, "Research synthesis in software engineering: a case for meta-analysis," in *Proceedings Sixth International Software Metrics Symposium (Cat. No. PR00403)*, 1999: IEEE, pp. 143-151.
- [35] T. Das and B.-S. Teng, "The risk-based view of trust: A conceptual framework," *Journal of Business and Psychology*, vol. 19, no. 1, pp. 85-116, 2004.
- [36] C. Becker *et al.*, "Requirements: The key to sustainability," *IEEE Software*, vol. 33, no. 1, pp. 56-65, 2015.
- [37] R. C. Mayer, J. H. Davis, and F. D. Schoorman, "An integrative model of organizational trust," *Academy of management review*, vol. 20, no. 3, pp. 709-734, 1995.
- [38] B. Shneiderman, "Designing trust into online experiences," *Communications of the ACM*, vol. 43, no. 12, pp. 57-59, 2000.
- [39] D. McKnight and N. Chervany, "The meanings of trust. Scientific report, University of Minnesota," ed: Archived 2011-09-30 at the Wayback Machine, 1996.
- [40] B. Brown. "SuperSoul sessions: The Anatomy of Trust." (accessed 01/03/2017, 2017).
- [41] B. Brown, "How the Courage to Be Vulnerable Transforms the Way We Live, Love, Parent, and Lead," ed. New York: Penguin, 2012.
- [42] R. Bachmann and A. Zaheer, *Handbook of advances in trust research*. Edward Elgar Publishing, 2013.
- [43] S. Becker *et al.*, "Trustworthy software systems: a discussion of basic concepts and terminology," *ACM SIGSOFT Software Engineering Notes*, vol. 31, no. 6, pp. 1-18, 2006.
- [44] J. Swait and W. Adamowicz, "Choice environment, market complexity, and consumer behavior: a theoretical and empirical approach for incorporating decision complexity into models of consumer choice," *Organizational behavior and human decision processes*, vol. 86, no. 2, pp. 141-167, 2001.
- [45] "Stakeholders not satisfied with probability assessments." (accessed 09/10/2018).
- [46] C. Julmi, "When rational decision-making becomes irrational: a critical assessment and re-conceptualization of intuition effectiveness," *Business Research*, vol. 12, no. 1, pp. 291-314, 2019.
- [47] L. R. Santos and A. G. Rosati, "The evolutionary roots of human decision making," *Annual review of psychology*, vol. 66, p. 321, 2015.
- [48] H.-H. Lin and Y.-S. Wang, "An examination of the determinants of customer loyalty in mobile commerce contexts," *Information & management*, vol. 43, no. 3, pp. 271-282, 2006.
- [49] J. Huang and M. S. Fox, "Trust judgment in knowledge provenance," in *16th International Workshop on Database and Expert Systems Applications (DEXA'05)*, 2005: IEEE, pp. 524-528.
- [50] P. Madhavan and D. A. Wiegmann, "Similarities and differences between human-human and human-automation trust: an integrative review," *Theoretical Issues in Ergonomics Science*, vol. 8, no. 4, pp. 277-301, 2007.

- [51] M. Zloteanu, N. Harvey, D. Tuckett, and G. Livan, "Digital Identity: The effect of trust and reputation information on user judgement in the Sharing Economy," *PloS one*, vol. 13, no. 12, p. e0209071, 2018.
- [52] D. H. McKnight, V. Choudhury, and C. Kacmar, "The impact of initial consumer trust on intentions to transact with a web site: a trust building model," *The journal of strategic information systems*, vol. 11, no. 3-4, pp. 297-323, 2002.
- [53] X. Luo, H. Li, J. Zhang, and J. P. Shim, "Examining multi-dimensional trust and multi-faceted risk in initial acceptance of emerging technologies: An empirical study of mobile banking services," *Decision support systems*, vol. 49, no. 2, pp. 222-234, 2010.
- [54] J. B. Kim, "An empirical study on consumer first purchase intention in online shopping: integrating initial trust and TAM," *Electronic Commerce Research*, vol. 12, no. 2, pp. 125-150, 2012.
- [55] M. Koufaris and W. Hampton-Sosa, "The development of initial trust in an online company by new customers," *Information & management*, vol. 41, no. 3, pp. 377-397, 2004.
- [56] F. D. Davis, "Perceived usefulness, perceived ease of use, and user acceptance of information technology," *MIS quarterly*, pp. 319-340, 1989.
- [57] D. Gefen and D. Straub, "Managing user trust in B2C e-services," *e-Service*, vol. 2, no. 2, pp. 7-24, 2003.
- [58] D. Artz and Y. Gil, "A survey of trust in computer science and the semantic web," *Journal of Web Semantics*, vol. 5, no. 2, pp. 58-71, 2007.
- [59] M. A. Korsgaard, D. M. Schweiger, and H. J. Sapienza, "Building commitment, attachment, and trust in strategic decision-making teams: The role of procedural justice," *Academy of Management journal*, vol. 38, no. 1, pp. 60-84, 1995.
- [60] A. Vance, C. Elie-Dit-Cosaque, and D. W. Straub, "Examining trust in information technology artifacts: the effects of system quality and culture," *Journal of management information systems*, vol. 24, no. 4, pp. 73-100, 2008.
- [61] D. H. McKnight, V. Choudhury, and C. Kacmar, "Developing and validating trust measures for e-commerce: An integrative typology," *Information systems research*, vol. 13, no. 3, pp. 334-359, 2002.
- [62] D. H. McKnight and N. L. Chervany, "What trust means in e-commerce customer relationships: An interdisciplinary conceptual typology," *International journal of electronic commerce*, vol. 6, no. 2, pp. 35-59, 2001.
- [63] K. C. Lee and N. Chung, "Understanding factors affecting trust in and satisfaction with mobile banking in Korea: A modified DeLone and McLean's model perspective," *Interacting with computers*, vol. 21, no. 5-6, pp. 385-392, 2009.
- [64] R. M. Morgan and S. D. Hunt, "The commitment-trust theory of relationship marketing," *Journal of marketing*, vol. 58, no. 3, pp. 20-38, 1994.
- [65] S. Chandra, S. C. Srivastava, and Y.-L. Theng, "Evaluating the role of trust in consumer adoption of mobile payment systems: An empirical analysis," *Communications of the association for information systems*, vol. 27, no. 1, p. 29, 2010.
- [66] G. H. Brundtland, "World commission on environment and development," *Environmental policy and law*, vol. 14, no. 1, pp. 26-30, 1985.

- [67] X. Li, T. J. Hess, and J. S. Valacich, "Why do we trust new technology? A study of initial trust formation with organizational information systems," *The Journal of Strategic Information Systems*, vol. 17, no. 1, pp. 39-71, 2008.
- [68] A. AlJaafreh, R. Al-Adaileh, A. Gill, A. Al-Ani, and Y. Alzoubi, "A review of literature of initial trust in e-services: the case of internet banking services in Jordanian context," *Journal of Electronic Banking Systems*, 2014.
- [69] D. Cyr, "Modeling web site design across cultures: relationships to trust, satisfaction, and e-loyalty," *Journal of management information systems*, vol. 24, no. 4, pp. 47-72, 2008.
- [70] R. Goodland, "Sustainability: human, social, economic and environmental," *Encyclopedia of global environmental change*, vol. 5, pp. 481-491, 2002.
- [71] M. Drury, K. Conboy, and K. Power, "Obstacles to decision making in Agile software development teams," *Journal of Systems and Software*, vol. 85, no. 6, pp. 1239-1254, 2012.
- [72] G. Elahi and E. Yu, "Trust trade-off analysis for security requirements engineering," in *2009 17th IEEE International Requirements Engineering Conference*, 2009: IEEE, pp. 243-248.
- [73] P. Damián-Reyes, J. Favela, and J. Contreras-Castillo, "Uncertainty management in context-aware applications: Increasing usability and user trust," *Wireless Personal Communications*, vol. 56, no. 1, pp. 37-53, 2011.
- [74] C. Moorman, G. Zaltman, and R. Deshpande, "Relationships between providers and users of market research: The dynamics of trust within and between organizations," *Journal of marketing research*, vol. 29, no. 3, pp. 314-328, 1992.
- [75] S. Ruohomaa and L. Kutvonen, "Trust management survey," in *International Conference on Trust Management*, 2005: Springer, pp. 77-92.
- [76] S. L. Star, G. C. Bowker, and L. J. Neumann, "Transparency at different level of scale: convergence between information artefacts and social worlds," *Library and Information Science, Urbana-Champaign*, 1998.
- [77] Y.-C. Tu, E. Tempero, and C. Thomborson, "An experiment on the impact of transparency on the effectiveness of requirements documents," *Empirical Software Engineering*, vol. 21, no. 3, pp. 1035-1066, 2016.
- [78] M. Turilli and L. Floridi, "The ethics of information transparency," *Ethics and Information Technology*, vol. 11, no. 2, pp. 105-112, 2009.
- [79] N. Bevan and M. Azuma, "Quality in use: Incorporating human factors into the software engineering lifecycle," in *Proceedings of IEEE International Symposium on Software Engineering Standards*, 1997: IEEE, pp. 169-179.
- [80] C. Cappelli and J. Leite, "Software transparency," *Business and Information Systems Engineering*, vol. 2, pp. 127-139, 2010.
- [81] L. Chazette and K. Schneider, "Explainability as a non-functional requirement: challenges and recommendations," *Requirements Engineering*, vol. 25, no. 4, pp. 493-514, 2020.
- [82] J. C. S. do Prado Leite and C. Cappelli, "Exploring i* Characteristics that Support Software Transparency," in *iStar*, 2008, pp. 51-54.
- [83] L. M. Cysneiros and V. M. B. Werneck, "An Initial Analysis on How Software Transparency and Trust Influence each other," in *WER*, 2009.
- [84] E. Yu and L. Liu, "Modelling trust for system design using the i* strategic actors framework," in *Trust in Cyber-societies*: Springer, 2001, pp. 175-194.

- [85] A. K. Schnackenberg and E. C. Tomlinson, "Organizational transparency: A new perspective on managing trust in organization-stakeholder relationships," *Journal of Management*, vol. 42, no. 7, pp. 1784-1810, 2016.
- [86] J. Begole, M. B. Rosson, and C. A. Shaffer, "Flexible collaboration transparency: supporting worker independence in replicated application-sharing systems," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 6, no. 2, pp. 95-132, 1999.
- [87] "Computational Transparency." (accessed 2019).
- [88] D. Ince, *A Dictionary of the Internet*. Oxford University Press, 2017.
- [89] J. Sutherland and K. Schwaber, "The scrum guide," *The definitive guide to scrum: The rules of the game. Scrum.org*, vol. 268, 2013.
- [90] G. F. Coulouris, J. Dollimore, and T. Kindberg, *Distributed systems: concepts and design*. pearson education, 2005.
- [91] S.-Y. Chien, M. Lewis, K. Sycara, A. Kumru, and J.-S. Liu, "Influence of culture, transparency, trust, and degree of automation on automation use," *IEEE Transactions on Human-Machine Systems*, vol. 50, no. 3, pp. 205-214, 2019.
- [92] N. Diakopoulos and M. Koliska, "Algorithmic transparency in the news media," *Digital Journalism*, vol. 5, no. 7, pp. 809-828, 2017.
- [93] K. Hosanagar and V. Jair, "We need transparency in algorithms, but too much can backfire," *Harvard Business Review*, 2018.
- [94] R. W. Buell, "Operational transparency," 2019.
- [95] P. Murmann and S. Fischer-Hübner, "Tools for achieving usable ex post transparency: a survey," *IEEE Access*, vol. 5, pp. 22965-22991, 2017.
- [96] J. C. Bertot, P. T. Jaeger, and J. M. Grimes, "Using ICTs to create a culture of transparency: E-government and social media as openness and anti-corruption tools for societies," *Government information quarterly*, vol. 27, no. 3, pp. 264-271, 2010.
- [97] G. A. Auger, "Trust Me, Trust Me Not: An Experimental Analysis of the Effect of Transparency on Organizations," *Journal of Public Relations Research*, vol. 26, no. 4, pp. 325-343, 2014/08/08 2014, doi: 10.1080/1062726X.2014.908722.
- [98] A. J. M. Beulens, D.-F. Broens, P. Folstar, and G. J. Hofstede, "Food safety and transparency in food chains and networks Relationships and challenges," *Food Control*, vol. 16, no. 6, pp. 481-486, 2005/07/01/ 2005, doi: <https://doi.org/10.1016/j.foodcont.2003.10.010>.
- [99] T. K. Das and B.-S. Teng, "Trust, Control, and Risk in Strategic Alliances: An Integrated Framework," *Organization Studies*, vol. 22, no. 2, pp. 251-283, 2001/03/01 2001, doi: 10.1177/0170840601222004.
- [100] K. R. Fleischmann and W. A. Wallace, "A covenant with transparency: Opening the black box of models," *Communications of the ACM*, vol. 48, no. 5, pp. 93-97, 2005.
- [101] H. Kim and T. H. Lee, "Strategic CSR Communication: A Moderating Role of Transparency in Trust Building," *International Journal of Strategic Communication*, vol. 12, no. 2, pp. 107-124, 2018/03/15 2018, doi: 10.1080/1553118X.2018.1425692.
- [102] S. Kim and J. Lee, "E-participation, transparency, and trust in local government," *Public Administration Review*, vol. 72, no. 6, pp. 819-828, 2012.
- [103] C. Medina and R. Rufín, "Social Media Use and Perception of Transparency in the Generation of Trust in Public Services," in *2015 48th Hawaii International Conference on System Sciences*, 2015: IEEE, pp. 2425-2434.

- [104] K. Menon and K. T. Goh, "Transparency and trust: risk communications and the Singapore experience in managing SARS," *Journal of Communication Management*, vol. 9, no. 4, pp. 375-383, 2005.
- [105] R. R. Moreno and C. M. Molina, "The impact of the transparency policy on university students' trust and intention of continued use," in *2014 47th Hawaii International Conference on System Sciences*, 2014: IEEE, pp. 2083-2092.
- [106] S. M. Norman, B. J. Avolio, and F. Luthans, "The impact of positivity and transparency on trust in leaders and their perceived effectiveness," *The Leadership Quarterly*, vol. 21, no. 3, pp. 350-364, 2010.
- [107] H. Park and J. Blenkinsopp, "The roles of transparency and trust in the relationship between corruption and citizen satisfaction," *International Review of Administrative Sciences*, vol. 77, no. 2, pp. 254-274, 2011.
- [108] T. McManus, Y. Holtzman, H. Lazarus, J. Anderberg, and J. Jahansoozi, "Organization-stakeholder relationships: exploring trust and transparency," *Journal of management development*, 2006.
- [109] G. J. Hofstede, "Trust and transparency in supply netchains: a contradiction?," in *Supply Chain Management: Issues in the New Era of Collaboration and Competition*: IGI Global, 2007, pp. 106-127.
- [110] M. R. Cleary and S. Stokes, *Democracy and the culture of skepticism: the politics of trust in Argentina and Mexico*. Russell Sage Foundation, 2006.
- [111] A. H. Trechsel, R. Kies, F. Mendez, and P. C. Schmitter, *Evaluation of the use of new technologies in order to facilitate democracy in Europe*. C2D-Research and Documentation Centre on Direct Democracy, 2003.
- [112] A. Thurston, "Fostering trust and transparency through information systems," 2005.
- [113] E. Claims. "Tenets of Transparency." (accessed 2018).
- [114] M. Chase and S. Meiklejohn, "Transparency overlays and applications," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 168-179.
- [115] H. S. M. Cramer, "Interaction with user-adaptive information filters.: trust, transparency and acceptance," presented at the CHI '07 Extended Abstracts on Human Factors in Computing Systems, San Jose, CA, USA, 2007. [Online]. Available: <https://doi.org/10.1145/1240866.1240870>.
- [116] K. M. Khan and Q. Malluhi, "Establishing trust in cloud computing," *IT professional*, vol. 12, no. 5, pp. 20-27, 2010.
- [117] M. S. Lund, B. Solhaug, and K. Stølen, "Evolution in relation to risk and trust management," *Computer*, vol. 43, no. 5, pp. 49-55, 2010.
- [118] A. I. Nicolaou and D. H. McKnight, "Perceived information quality in data exchanges: Effects on risk, trust, and intention to use," *Information systems research*, vol. 17, no. 4, pp. 332-351, 2006.
- [119] N. V. Oza, T. Hall, A. Rainer, and S. Grey, "Trust in software outsourcing relationships: An empirical investigation of Indian software companies," *Information and software Technology*, vol. 48, no. 5, pp. 345-354, 2006.
- [120] J. E. Perry, D. Cox, and A. D. Cox, "Trust and transparency: patient perceptions of physicians' financial relationships with pharmaceutical companies," *The Journal of Law, Medicine & Ethics*, vol. 42, no. 4, pp. 475-491, 2014.

- [121] E. Pignotti, S. Beran, and P. Edwards, "What does this device do?," in *URB-IOT'14 Proceedings of the First International Conference on IoT in Urban Space*, 2014: ICST.
- [122] S. Schnorf, M. Ortlieb, and N. Sharma, "Trust, transparency & control in inferred user interest models," in *CHI'14 Extended Abstracts on Human Factors in Computing Systems*, 2014, pp. 2449-2454.
- [123] K. Vaccaro, D. Huang, M. Eslami, C. Sandvig, K. Hamilton, and K. Karahalios, "The illusion of control: Placebo effects of control settings," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018, pp. 1-13.
- [124] T.-W. Chen and S. S. Sundar, "This app would like to use your current location to better serve you: Importance of user assent and system transparency in personalized mobile services," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018, pp. 1-13.
- [125] R. Tomsett, D. Braines, D. Harborne, A. Preece, and S. Chakraborty, "Interpretable to whom? A role-based model for analyzing interpretable machine learning systems," *arXiv preprint arXiv:1806.07552*, 2018.
- [126] B. Y. Lim and A. K. Dey, "Assessing demand for intelligibility in context-aware applications," in *Proceedings of the 11th international conference on Ubiquitous computing*, 2009, pp. 195-204.
- [127] J. Brunk, J. Mattern, and D. M. Riehle, "Effect of transparency and trust on acceptance of automatic online comment moderation systems," in *2019 IEEE 21st Conference on Business Informatics (CBI)*, 2019, vol. 1: IEEE, pp. 429-435.
- [128] A. Springer and S. Whittaker, "'I had a solid theory before but it's falling apart': Polarizing Effects of Algorithmic Transparency," *arXiv preprint arXiv:1811.02163*, 2018.
- [129] B. Y. Lim and A. K. Dey, "Investigating intelligibility for uncertain context-aware applications," in *Proceedings of the 13th international conference on Ubiquitous computing*, 2011, pp. 415-424.
- [130] A. Springer, V. Hollis, and S. Whittaker, "Dice in the black box: User experiences with an inscrutable algorithm," *arXiv preprint arXiv:1812.03219*, 2018.
- [131] R. Parasuraman and V. Riley, "Humans and automation: Use, misuse, disuse, abuse," *Human factors*, vol. 39, no. 2, pp. 230-253, 1997.
- [132] A. Sillitti and G. Succi, "Requirements engineering for agile methods," in *Engineering and Managing Software Requirements*: Springer, 2005, pp. 309-326.
- [133] B. Nuseibeh and S. Easterbrook, "Requirements engineering: a roadmap," in *Proceedings of the Conference on the Future of Software Engineering*, 2000, pp. 35-46.
- [134] Y. Yang, F. Xia, W. Zhang, X. Xiao, Y. Li, and X. Li, "Towards semantic requirement engineering," in *IEEE International Workshop on Semantic Computing and Systems*, 2008: IEEE, pp. 67-71.
- [135] K. Pohl, *Requirements engineering: fundamentals, principles, and techniques*. Springer Publishing Company, Incorporated, 2010.
- [136] K. Voortwis, "Building a giant drawing machine: the build and improvement process off a multi-colour spray-paint wall plotter for performance on the Maker Festival," University of Twente, 2017.
- [137] C. Ebert, "Episode 114: On requirements engineering," ed. Software Engineering Radio, The podcast for Professional Software Developers, 2008.

- [138] K. Wiegers and J. Beatty, *Software requirements*. Pearson Education, 2013.
- [139] A. Van Lamsweerde, *Requirements engineering: From system goals to UML models to software*. Chichester, UK: John Wiley & Sons, 2009.
- [140] D. Zowghi and C. Coulin, "Requirements elicitation: A survey of techniques, approaches, and tools," in *Engineering and managing software requirements*: Springer, 2005, pp. 19-46.
- [141] J. Dick, E. Hull, and K. Jackson, *Requirements engineering*. Springer, 2017.
- [142] Y. Zhu, "Requirements engineering in an agile environment," ed, 2009.
- [143] A. Eberlein and J. Leite, "Agile requirements definition: A view from requirements engineering," in *Proceedings of the International Workshop on Time-Constrained Requirements Engineering (TCRE'02)*, 2002, pp. 4-8.
- [144] J. Erickson, K. Lyytinen, and K. Siau, "Agile modeling, agile software development, and extreme programming: the state of research," *Journal of Database Management (JDM)*, vol. 16, no. 4, pp. 88-100, 2005.
- [145] J. W. Creswell, *A concise introduction to mixed methods research*. SAGE publications, 2014.
- [146] I. Islam, K. M. Munim, S. J. Oishwee, A. N. Islam, and M. N. Islam, "A Critical Review of Concepts, Benefits, and Pitfalls of Blockchain Technology using Concept Map," *IEEE Access*, vol. 8, pp. 68333-68341, 2020.
- [147] P. Goodwin, J. Dargay, and M. Hanly, "Elasticities of road traffic and fuel consumption with respect to price and income: a review," *Transport reviews*, vol. 24, no. 3, pp. 275-292, 2004.
- [148] V. Chandra and A. Hareendran, *Research Methodology by Pearson 1st Edition*. Pearson Education India, 2017.
- [149] R. Kumar, *Research methodology: A step-by-step guide for beginners*. Sage, 2018.
- [150] J. Bell, *Doing Your Research Project: A guide for first-time researchers*. McGraw-Hill Education (UK), 2014.
- [151] A. Galletta, *Mastering the semi-structured interview and beyond: From research design to analysis and publication*. NYU press, 2013.
- [152] P. Stopher, *Collecting, managing, and assessing data using sample surveys*. Cambridge University Press, 2012.
- [153] K. Okamura and S. Yamada, "Adaptive trust calibration for human-AI collaboration," *Plos one*, vol. 15, no. 2, p. e0229132, 2020.
- [154] J. Torous and L. W. Roberts, "Needed innovation in digital health and smartphone applications for mental health: transparency and trust," *JAMA psychiatry*, vol. 74, no. 5, pp. 437-438, 2017.
- [155] U.-V. Albrecht, O. Pramann, and U. von Jan, "Synopsis for health apps: transparency for trust and decision making," in *Social media and mobile technologies for healthcare*: IGI Global, 2014, pp. 94-108.
- [156] S.-Y. Chien, M. Lewis, K. Sycara, J.-S. Liu, and A. Kumru, "The effect of culture on trust in automation: reliability and workload," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 8, no. 4, pp. 1-31, 2018.
- [157] Y.-C. Tu, C. Thomborson, and E. Tempero, "Illusions and perceptions of transparency in software engineering," in *2011 18th Asia-Pacific Software Engineering Conference*, 2011: IEEE, pp. 365-372.

- [158] T. Kulesza *et al.*, "Explanatory debugging: Supporting end-user debugging of machine-learned programs," in *2010 IEEE Symposium on Visual Languages and Human-Centric Computing*, 2010: IEEE, pp. 41-48.
- [159] S. Passera, "Enhancing contract usability and user experience through visualization-an experimental evaluation," in *2012 16th International Conference on Information Visualisation*, 2012: IEEE, pp. 376-382.
- [160] E. Bertino, A. Kundu, and Z. Sura, "Data transparency with blockchain and AI ethics," *Journal of Data and Information Quality (JDIQ)*, vol. 11, no. 4, pp. 1-8, 2019.
- [161] O. Zinovatna and L. M. Cysneiros, "Reusing knowledge on delivering privacy and transparency together," in *2015 IEEE Fifth International Workshop on Requirements Patterns (RePa)*, 2015: IEEE, pp. 17-24.
- [162] H. Hedbom, "A survey on transparency tools for enhancing privacy," in *IFIP Summer School on the Future of Identity in the Information Society*, 2008: Springer, pp. 67-82.
- [163] J. L. Wright, J. Y. Chen, and S. G. Lakhmani, "Agent transparency and reliability in human-robot interaction: the influence on user confidence and perceived reliability," *IEEE Transactions on Human-Machine Systems*, vol. 50, no. 3, pp. 254-263, 2019.
- [164] D. A. Norman, "Some observations on mental models," *Mental models*, vol. 7, no. 112, pp. 7-14, 1983.
- [165] N. Moray, "Intelligent aids, mental models, and the theory of machines," *International journal of man-machine studies*, vol. 27, no. 5-6, pp. 619-629, 1987.
- [166] J. B. Lyons *et al.*, "Shaping trust through transparent design: theoretical and experimental guidelines," in *Advances in human factors in robots and unmanned systems*: Springer, 2017, pp. 127-136.
- [167] M. Chromik, M. Eiband, S. T. Völkel, and D. Buschek, "Dark Patterns of Explainability, Transparency, and User Control for Intelligent Systems," in *IUI Workshops*, 2019, vol. 2327.
- [168] T. P. Vos and S. Craft, "The discursive construction of journalistic transparency," *Journalism Studies*, vol. 18, no. 12, pp. 1505-1522, 2017.
- [169] F. Bannister and R. Connolly, "The trouble with transparency: a critical review of openness in e-government," *Policy & Internet*, vol. 3, no. 1, pp. 1-30, 2011.
- [170] C. Cappelli, J. C. S. do Prado Leite, and A. d. P. A. Oliveira, "Exploring business process transparency concepts," in *15th IEEE International Requirements Engineering Conference (RE 2007)*, 2007: IEEE, pp. 389-390.
- [171] S. M. Hosseini Moghaddam, "Engineering of transparency requirements in business information systems," Bournemouth University, 2016.
- [172] P. Meunier, "Software transparency and purity," *Communications of the ACM*, vol. 51, no. 2, pp. 104-104, 2008.
- [173] D. Spagnuolo, A. Ferreira, and G. Lenzini, "Transparency enhancing tools and the GDPR: Do they match?," in *International Conference on Information Systems Security and Privacy*, 2019: Springer, pp. 162-185.
- [174] P. Murmann and F. Karegar, "From Design Requirements to Effective Privacy Notifications: Empowering Users of Online Services to Make Informed Decisions," *International Journal of Human-Computer Interaction*, pp. 1-26, 2021.

- [175] F. Loizides, M. Winckler, U. Chatterjee, J. Abdelnour-Nocera, and A. Parmaxi, "Human Computer Interaction and Emerging Technologies: Workshop Proceedings from the INTERACT 2019 Workshops," ed: Cardiff University Press, 2020.
- [176] C. Palomares Bonache, "Definition and use of software requirement patterns in requirements engineering activities," in *Proceedings of the REFSQ 2011 Workshops, the REFSQ 2011 Empirical Track, and the REFSQ 2014 Doctoral Symposium*, 2014, pp. 60-66.
- [177] S. Withall, *Software requirement patterns*. Pearson Education, 2007.
- [178] M. Cohn, *User stories applied: For agile software development*. Addison-Wesley Professional, 2004.
- [179] P. Murmann and S. Fischer-Hübner, "Usable transparency enhancing tools: A literature review," ed: Karlstads universitet, 2017.
- [180] Ç. Grubu, "Guidelines on Transparency under Regulation 2016/679," *WP260 rev*, vol. 1, 2018.
- [181] N. H. Thuan, A. Drechsler, and P. Antunes, "Construction of design science research questions," *Communications of the Association for Information Systems*, vol. 44, no. 1, p. 20, 2019.
- [182] A. de Ste Croix and A. Easton, "The product owner team," in *Agile 2008 Conference*, 2008: IEEE, pp. 274-279.
- [183] G. Hickey and C. Kipping, "A multi-stage approach to the coding of data from open-ended questions," *Nurse researcher*, vol. 4, no. 1, pp. 81-91, 1996.
- [184] M. Hosseini, A. Shahri, K. Phalp, and R. Ali, "A modelling language for transparency requirements in business information systems," in *International Conference on Advanced Information Systems Engineering*, 2016: Springer, pp. 239-254.
- [185] J. Nielsen, "10 usability heuristics for user interface design," *Nielsen Norman Group*, vol. 1, no. 1, 1995.
- [186] E. Grünewald and F. Pallas, "TILT: A GDPR-Aligned Transparency Information Language and Toolkit for Practical Privacy Engineering," *arXiv preprint arXiv:2012.10431*, 2020.
- [187] ISO. "STANDARDS BY ISO/PC 317, Consumer protection: privacy by design for consumer goods and services." (accessed 09/09/2020, 2020).
- [188] W. De Kort, *DevOps on the Microsoft Stack*. Springer, 2016.
- [189] A. Hemon, B. Lyonnet, F. Rowe, and B. Fitzgerald, "From agile to DevOps: Smart skills and collaborations," *Information Systems Frontiers*, vol. 22, no. 4, pp. 927-945, 2020.

Chapter 8

8. Appendix

8.1 Prequestionnaires

What is your area of expertise in software development?

Mobile Application Development, Xamarin (Android, iOS, Forms) using C#

How many years of experience do you have in your current area of expertise? What is your position in this project (i.e., developer, senior developer, domain expert., academic, industrial)?

Four years in Mobile Application *Development*, eight and a half years in professional software development (One year of that was a placement year). The first four and a half years were mostly spent doing Web Development.

My title in this project is Developer, but in reality, it is a Senior Developer position. The tiers of development seniority in Spirit Digital at present are not like the traditional Junior → Mid-Level → Senior → Lead Developer structure. We only have Developers and one Senior Developer.

Before applying the methodology

1. *Requirement engineering practices in agile process make the overall workload less during a project lifetime.*

A. *Strongly disagree* B. *Disagree* C. *Neither agree nor disagree* D. *Agree* E. *Strongly Agree*

Please briefly describe why you have selected this answer.

D. I believe that rushing into development without fully understanding the requirements or creating the requirements during development results in a cost later in the project. Time will be spent developing something that will not meet the final requirements, which results in more change and a higher workload.

2. *Requirement engineering practices in agile process speeds up the work in my project.*

A. Strongly disagree, B. Disagree, C. Neither agree nor disagree, D. Agree, E.

Strongly Agree

Agree

Please briefly describe why you have selected this answer.

D. Having clarity with the work that is required of me helps me to focus on delivering that work to a shorter time scale. Having to work out what the requirements are whilst working on a task/story slows me down; in this scenario I may have to re-do some of the work I did as requirements were not fully documented/explained/understood.

3. *What do you think of the quality of the project documentation? How do you think it can be improved?*

We are early in our journey of adopting Agile processes. Currently the quality of our Backlog(s) is poor, which in turn is making refinement more difficult. Stories are often committed into Sprints despite not containing Given When Then acceptance criteria. Some knowledge is assumed and not documented.

4. *What challenges do you face in collecting complete maintaining consistent requirements?*

We are time constrained; I think this is why we often don't have enough well written stories in time for when we commit to starting Sprints.

5. *When implementing requirements do you have all the system aspects required for the task defined and available? If not, can you explain what they are and why?*

We do not always have the system aspects required for the task defined and available. I think the time restraints are having an impact, Agile training for the team might be beneficial as well.

6. *To your knowledge how frequently do you implement a use case that includes transparency requirements?*

I am not sure exactly what transparency requirements are. Assuming they are requirements that are documented to the standards we have agreed (Given When Then), not very often yet, but we are early in our Agile journey.

7. *Which transparency requirements elicitation processes do you have?*

I think we have a loose undefined process at present. It is not something I am that involved with. I think part of the reason we are doing a beta app release to some stakeholders in the business is to refine some final requirements for the app release.

8. *When implementing requirements how do you think about the data you are collecting? And if the data collection meets data protection rules?*

It is something I think about yes, I cannot speak for what the team are doing.

9. *When implementing requirements how do you consider informing the user about the data being collected and the processes being performed? As well as your data protection processes.*

I think informing the user about the data being collected and the process being performed should be a requirement itself. It is not a requirement that has been spoken about between myself and those creating requirements above me in the organization.

10. *To what extent do requirement practices in agile process's help you and other new developers to think about what is **missing** in your and their software **requirements**? can you elaborate?*

I think it is very important. I am a firm believer that a high-quality backlog refined just in time to be committed to Sprints will result in a higher quality product, delivered to more accurate time scales. I believe refinement meetings between the developers are beneficial to creating higher quality requirements that are more explicit and better understood. Having multiple developers, with different ways of thinking really helps to find problems with requirements that can be resolved before the work is committed to.

Prequestionnaire 2

What is your area of expertise in software development?

Software developer started with C, C++, VB then moved to C#, developed a mix of applications, including windows forms. Mobile device and web projects.

How many years of experience do you have in your current area of expertise? What is your position in this project (i.e., developer, senior developer, domain expert., academic, industrial)?

Software developer for 25+ years degree educated in Computing. Main industries of interest are Retail (Next plc (15 years), in recent years Microsoft dynamics for Ticketing systems.

Before applying the methodology

1. *Requirement engineering practices in agile process make the overall workload less during a project lifetime.*

A. Strongly disagree B. Disagree C. Neither agree nor disagree D. Agree E. Strongly Agree

Please briefly describe why you have selected this answer.

d. Agree, but only when the business has full buy in of the process. It is all about being able to deliver a variable product at the end of each sprint.

2. *Requirement engineering practices in agile process speeds up the work in my project.*

A. Strongly disagree B. Disagree C. Neither agree nor disagree D. Agree E. Strongly Agree

Please briefly describe why you have selected this answer.

C. Neither agree nor disagree, depends on largely what is trying to be delivered and the business stresses.

3. *What do you think of the quality of the project documentation? How do you think it can be improved?*

Documentation is very important to ensure the work being carried out is what the business actually require, as we have all been on projects where at the end what they actually want is very different from what they request.

4. *What challenges do you face in collecting complete maintaining consistent requirements?*

From a developer point of view we are ensuring that the change request to be developed is clearly defined and is of a size that can be developed within the sprint without overloading the time available.

5. *When implementing requirements do you have all the system aspects required for the task defined and available? If not, can you explain what they are and why?*

In theory, yes, we do, but in practice there are many times where questions only arise while in mid development.

6. *To your knowledge how frequently do you implement a use case that includes transparency requirements?*

No idea, a new term for me

7. *Which transparency requirements elicitation processes do you have?*

No idea, a new term for me

8. *When implementing requirements how do you think about the data you are collecting? And if the data collection meets data protection rules?*

Yes, as this can have issues for security and of course new GDPR. As we are the data collectors, we must ensure this integrity.

9. *When implementing requirements how do you consider informing the user about the data being collected and the processes being performed? As well as your data protection processes.*

In previous retail would yes, we always ensured that the user knew what data is being collected and its protection

10. *To what extent do requirement practices in agile process's help you and other new developers to think about what is **missing** in your and their software **requirements**? can you elaborate?*

The fact we spend time back log grooming helps to ensure this, discussing a requirement by committee helps to ensure the quality of the Change request before work starts, meaning less rework and change creep.

Prequestionnaires 3

What is your area of expertise in software development?

Software developer

How many years of experience do you have in your current area of expertise? What is your position in this project (i.e., developer, senior developer, domain expert., academic, industrial)?

4 years' experience mostly working C# using Xamarin, Asp.Net and Unity. Also has experience with web development and has worked in Gas industry and the health industry for a number of years.

Before applying the methodology

1. *Requirement engineering practices in agile process make the overall workload less during a project lifetime.*

A. Strongly disagree B. Disagree C. Neither agree nor disagree D. Agree E. Strongly Agree

Please briefly describe why you have selected this answer.

Agree – There are some aspects of the process which are more long-winded than the results which is frustrating but having a good sight of the project aids greatly in the long run. Having to make changes that implement the whole of the application can be very time consuming depending on the decisions you make at the start of the development process so defining this fully beforehand is very useful.

2. *Requirement engineering practices in agile process speeds up the work in my project.*

A. Strongly disagree B. Disagree C. Neither agree nor disagree D. Agree E. Strongly Agree

Please briefly describe why you have selected this answer.

Strongly agree – You know what you need to do before you do it and can group work together.

3. *What do you think of the quality of the project documentation? How do you think it can be improved?*

At times good sometimes lacking information which leads you to sometimes make assumptions.

Work could sometimes be grouped better.

Particular details and functionalities could be more defined. Occasionally a feature is implemented incorrectly to the user's desire and this can lead to lengthy changes due to bad spec in the first place

4. *What challenges do you face in collecting complete maintaining consistent requirements?*

Consistency over the whole application.

Seeing all the possible options

5. *When implementing requirements do you have all the system aspects required for the task defined and available? If not, can you explain what they are and why?*

Sometimes missing text or messages

Sometimes missing alternate pathway and error cases

Sometimes missing greater detail on the application flow and the exact process

6. *To your knowledge how frequently do you implement a use case that includes transparency requirements?*

Rarely – they tend to be messages back to the users and general information messages.

7. *Which transparency requirements elicitation processes do you have?*

None that I know of

8. *When implementing requirements how do you think about the data you are collecting? And if the data collection meets data protection rules?*

Sometimes get nervous at collecting particular data, we were collecting race and religious data for bookinghub. We store the data correctly and restrict access to user that require it but do not have a deeper policy in terms of implementation.

9. *When implementing requirements how do you consider informing the user about the data being collected and the processes being performed? As well as your data protection processes.*

General over the whole application in a privacy policy –

Sometimes if there is a specific use for the data like the address, we might ask for current address and inform them of that.

10. *To what extent do requirement practices in agile process's help you and other new developers to think about what is **missing** in your and their software **requirements**? can you elaborate?*

A fair bit but my experience has been more domain expert focus and understanding what is required through conversation and their expertise.

Prequestionnaire 4

What is your area of expertise in software development?

20 Years' experience in healthcare software development.

Spent 2 years as a SQL DBA.

How many years of experience do you have in your current area of expertise? What is your position in this project (i.e., developer, senior developer, domain expert., academic, industrial)?

I am currently Head of Development. I am currently overseeing many aspects of the functionality of the development team. I have been doing this for 1 year at Spirit Healthcare and have been responsible for a team for a year in the past.

Before applying the methodology

1. *Requirement engineering practices in agile process make the overall workload less during a project lifetime.*

A. Strongly disagree B. Disagree C. Neither agree nor disagree D. Agree E.

Strongly Agree

Please briefly describe why you have selected this answer.

D. Agree. I think agile allows the developer to put more of their thought processes and style into a change request. It is more flexible and allows them to show their flair for something.

2. *Requirement engineering practices in agile process speeds up the work in my project.*

A. Strongly disagree B. Disagree C. Neither agree nor disagree D. Agree E.

Strongly Agree

Please briefly describe why you have selected this answer.

D. Agree – Again, because the developer is working to their strengths, the work will be completed slightly more quickly.

3. *What do you think of the quality of the project documentation? How do you think it can be improved?*

Project documentation was good but was quite abstract. More time is required to write up more detailed change requests.

4. *What challenges do you face in collecting complete maintaining consistent requirements?*

Getting the required parties to focus on accuracy of information.

Having time to write up the item.

5. *When implementing requirements do you have all the system aspects required for the task defined and available? If not, can you explain what they are and why?*

Usually, yes. If not though, it may be that something was overlooked or there may be an issue which arose due to the way something has been coded in the past.

6. *To your knowledge how frequently do you implement a use case that includes transparency requirements?*

Always.

7. *Which transparency requirements elicitation processes do you have?*

We do not have a specific process, although we do publish a development item overview for our clients at the time of each release, which describes the change we implemented.

8. *When implementing requirements how do you think about the data you are collecting? And if the data collection meets data protection rules?*

We make a call during the requirements phase as to whether the information is personal or not.

9. *When implementing requirements how do you consider informing the user about the data being collected and the processes being performed? As well as your data protection processes.*

We have a specific text on first login to our system (CliniTouch Vie) that the patient must agree to in order to use the system.

10. *To what extent do requirement practices in agile processes help you and other new developers to think about what is **missing** in your and their software **requirements**? can you elaborate?*

The backlog grooming meeting helps to identify any missing parts in a requirement, since without that information, we cannot estimate the size of the item, which is critical for that meeting.

8.2 Postquestionnaires

Postquestionnaire1- Developers

1. *Transparency Engineering methodology made the workload less in overall work in my project.*
A. Strongly disagree B. Disagree C. Neither agree nor disagree D. Agree E. Strongly Agree

Please briefly describe why you have selected this answer.

C. The workload is about the same if not increased in some areas due to a more complete description of the tasks.

2. *Transparency Engineering methodology speeds up the work in my project.*
A. Strongly disagree B. Disagree C. Neither agree nor disagree D. Agree E. Strongly Agree

Please briefly describe why you have selected this answer.

D. Better defined tasks make it easier implement. Had to go back over work to add additional features fewer times which can be extremely time consuming.

3. *What do you think of the **quality** of the methodology documentation? How do you think it can be improved? How straight forward is to apply and train for the proposed methodology?*

Good as a whole, can be difficult to follow at times and have to jump between different documents and because of that you miss things. Found the process got a lot easier as time went on and the process naturally streamlined itself as I become more familiar with the process and documentation. Easy once you know what you are doing, can be a bit confusing up to that point.

4. *After applying the proposed methodology when implementing requirements are the system aspects required for the task more defined and available? If not, can you explain what they are and why?*

Yes, very much so. Helps to define the functionality more clearly. And has a user focused bent with everything focusing on communication with the user which is often an afterthought for me.

Really great for multi-faceted pathway use case – like the cancel course.

5. *Knowing about transparency requirements. How **many** transparency **requirements** are discovered with the proposed methodology?*

About 40.

6. *Do you think implementing the proposed methodology's resulting requirements **will improve the software**? If yes, what improvement, why and how?*

Yes, better communication with users.

More transparent about the system

Better defined task and better understanding of similar functionality across the system leading to more robust code.

7. *How did applying the proposed methodology made you think about the data you are collecting? And if the **data collection** meets data protection rules?*

Made me considered each of the data point and data interests for each task and structure the task description around them. We often collect data without telling the users what it is being used for, so it was good to considered why at each point in the application we are collecting given things. Make you rethink the validity in collecting certain things, such as user personality ethnic details were removed from the system as we had no real reason to collect it.

Data interests are a great extension on the concept of table – you can group many tables under a data interest if they are often or always used under the

8. *How did applying the proposed methodology made you consider informing the user about the data being collected and the processes being performed? As well as your data protection processes.*

Help to maintain user focused development later into the development cycle where at the task implementation level it is all but lost in a more functional description. This meant we informed the user about processes we probably would not have bothered before with but can be informative. On data collection every input is more

deeply scrutinized especially peripheral data collection processes which were often overlooked.

9. *Do you think implementing the proposed methodology's requirements help the user to be informed about the system aspects and data?*

Yes

10. *Based on your domain experience and after applying the methodology, do you think implementing the methodology's resulting requirements will help you and other new developers to think about what is **missing** in their software **requirements**? if yes, does following the methodology help in filling in these **gaps**? How and why?*

Yes, it helps fill in many of the gaps particularly around informing the user. Often, we would only inform the user if something were going wrong and this meant that there were lots of gaps as it was often down to the developers' discretion as they were implementing the task, the structure forces user to be informed about everything so you do not miss the important communications.

11. *Are there **limitations** of the methodology you can think of? How do you think it can be improved?*

I think it could overwhelm the user with information. There is no way to consider the type of user and the amount of information that they want to see, some technical users may find it useful to see more whilst a less technical user just wants to see the vital information (for example).

The structure could be improved to make the steps easier to follow.

I found myself linking similar outputs together, formalizing this process could be beneficial and would help link together similar areas of the application and the ways in which the information is presented.

Postquestionnaire 2 - Developers

1. *Transparency Engineering methodology made the workload less in overall work in my project.*

A. *Strongly disagree* B. *Disagree* C. *Neither agree nor disagree* D. *Agree* E. *Strongly Agree*

Please briefly describe why you have selected this answer.

Somewhere between C and D depending on the team.

Depending on the developer doing the work, I think most things are usually picked up and developed when the developer is an experienced hand. If it is a new team, new product, and less experienced developer then the methodology would certainly be beneficial when looking at potential feedback to users that might otherwise be missed.

2. *Transparency Engineering methodology speeds up the work in my project.*

A. *Strongly disagree* B. *Disagree* C. *Neither agree nor disagree* D. *Agree* E. *Strongly Agree*

Please briefly describe why you have selected this answer.

B. It would not speed up the work as it does add more work, but it would improve the quality of the system.

3. *What do you think of the **quality** of the methodology documentation? How do you think it can be improved? How straight forward is to apply and train for the proposed methodology?*

I felt that it could do with more complete examples of the sort of thing each aspect of the methodology requires.

4. *After applying the proposed methodology when implementing requirements are the system aspects required for the task more defined and available? If not, can you explain what they are and why?*

Yes, since there is more thought given to the development items.

5. *Knowing about transparency requirements. How **many** transparency **requirements** are discovered with the proposed methodology?*

Not sure as I did not complete them for all our requirements. I suspect between 10 and 20.

General improvement of the system

6. *Do you think implementing the proposed methodology's resulting requirements **will improve the software**? If yes, what improvement, why and how?*

Yes, as any aspect of the software that provides a greater level of explanation or feedback to it will make it easier or more satisfying to use.

7. *How did applying the proposed methodology make you think about the data you are collecting? And if the data collection meets data protection rules?*

We have had to think about this before when it came up in the initial spec, so it was not a consideration for us.

8. *How did applying the proposed methodology made you consider informing the user about the data being collected and the processes being performed? As well as your data protection processes.*

We have had to think about this before at the implementation stage and decided upon both a clinician user informing a patient user of how the system works and also ensuring that the user agrees to a statement of use before being able to access any system functionality.

9. *Do you think implementing the proposed methodology's requirements help the user to be informed about the system aspects and data?*

Yes. It certainly raises the thought and discussion between stakeholders, which then if carried out will certainly help the user in their use of the system.

10. *Based on your domain experience and after applying the methodology, do you think implementing the methodology's resulting requirements will help you and other new*

*developers to think about what is **missing** in their software **requirements**? if yes, does following the methodology help in filling in these **gaps**? How and why?*

I think it will make newer developers think more laterally and perhaps make them question themselves on how the system should respond in certain areas or situations.

*11. Are there **limitations** of the methodology you can think of? How do you think it can be improved?*

I think the methodology is useful, but it has a distinct time requirement to it. If a development team is experienced and estimates their work well, as well as being given enough time by senior management to complete the work then they should be able to gain some benefit from using the methodology. The limitations from my perspective really were time-based.

The only other aspect was the items that were considered, but not worth implementing. If at a stakeholder meeting, we discussed things that were not going to make it into a system, it is unlikely we would record this in any detail as it would be scoped out of the spec pretty quickly.

Postquestionnaire 2 - Domain Experts

*1. Based on your domain experience, how do you think implementing the methodology's resulting requirements will help you and other developers to inform users about their **data** and the way their data has been used, processed, and protected? Would that influence their knowledge about the system and therefore their trust? How and why? What is your definition of trust here? How has that been done before Transparency Engineering Methodology?*

The methodology offers full transparency to the user and I think this approach can also be useful to reference in the device's Governance policy.

It prompts me to think about the user experience from a perspective of trust – this gives any user confidence in a system when user prompts, behaviours and details are taken into account in relation to transparency of data and in turn the protection of it.

When a user sees that a system is informing them of their actions, I believe this makes for a trusting user experience.

In the current platform users have a permissions page displayed prior to be able to use the system which they can opt out of.

- 2. Are there **limitations** of the methodology you can think of? How do you think it can be improved?*

The methodology seems robust

- 3. How would you describe and characterize the methodology's after applying it?*

The methodology is concise, consistent, and replicable.

- 4. Based on your domain experience, do you think implementing the resulting requirements will result in increasing users' **cognitive trust, i.e., perceived understandability, reliability, and technical competence**? If yes, can you explain How and why and why not?*

Anytime we can be transparent and show how the user experience has been thoughtfully interrogated the better experience for all.

- 5. Based on your experience, how do you think implementing the resulting requirements will result in improved **usability and usefulness** of software functions, in other words, ease-of-use software? Explain why or why not?*

In both the above questions I think the implementation speaks more to the development side of things and prompts questions as to how we can make the platform more robust as we acquire more users

- 6. Based on your domain experience, do you think implementing the resulting requirements will result in increasing **temporary and permanent reassurance, data traceability**? In other words, helping the user to trace their activities and data in the software? Explain why or why not?*

Absolutely yes for the reasons stated prior. Transparency and rigorous methodology will lead to traceability throughout the application

7. *Based on your domain experience, do you think implementing the resulting requirements will help **in building user's trust** in the software? Explain how and why or why not?*

I think this is the same question and as above

8. *Based on your domain experience, do you think implementing the resulting requirements will help users to **make more informed decisions** when interacting with the software? Can you explain how and why or why not?*

It will allow users to ultimately decide if they use the software and what parts of the software, they are happy with/not

9. *Based on your domain experience, do you think implementing the resulting requirements will **help meeting the user's expectations**? Can you elaborate?*

I think users expect transparency, but I am not sure they know how to articulate what that is – it might be even more useful to be explicit about what the methodology actually assures for the user?

Postquestionnaire 2- Domain Experts

1. *Based on your domain experience, how do you think implementing the methodology's resulting requirements will help you and other developers to inform users about their **data** and the way their data has been used, processed, and protected? Would that influence their knowledge about the system and therefore their trust? How and why? What is your definition of trust here? How has that been done before Transparency Engineering Methodology?*

I think the methodology reminds us to consider what **user data** we are storing and whether we need to remind them about how it will be used. For the systems that I am currently working with, it is clear that the user must disclose the information that the clinician requests in order for the service to work for them. This part is usually explained face to face to the user, which many systems do not have the luxury of and is key to them trusting the system.

Regarding other systems that I have worked with, I think there are clear benefits to a **process** that ensures consideration is given to explaining potentially complex tasks that the system is capable of to the user. This kind of extra detail is often not picked up by developers, so being picked up by the methodology would be helpful.

I have found that once a **complex feature** is better explained, it attracts more usage. This is an example of a user trusting the system to do what it says it should. I once worked with a system that allowed users to export a file, modify it to suit their needs and then allow them to upload it again, however there was no information on the page about how to use that facility and what it was for. Once that detail was added, users had confidence in customising that part of the system.

Before the Transparency Engineering Methodology (TEM) it was usually the case that features would not be used due to users not knowing how they worked or what data they might affect. It was usually when someone new to the system was asking questions about it or running training sessions on it that questions raised by the TEM would have become clear.

2. *Are there **limitations** of the methodology you can think of? How do you think it can be improved?*

I think the limitation is in the time required to run it. It is completely down to the circumstances of the development team as to whether they could fit this detailed analysis into their plans. I think creating concise examples of what is expected in a tabular or list format would help the user to understand exactly what is required.

3. *How would you describe and characterize the methodology's after applying it?*

I think it provides a reminder as to some of the aspects of planning that are not always considered when writing a specification. I think it falls somewhere between spec-writing and the writing of the user stories that end up in the project tracking software and serves to generate discussion between developers/stakeholders around points that may not otherwise be considered. It could be characterised somewhat as a safety net.

4. *Based on your domain experience, do you think implementing the resulting requirements will result in increasing users' **cognitive trust**, i.e., **perceived***

understandability, reliability, and technical competence? If yes, can you explain How and why and why not?

It depends on the actions taken as a result of running the TEM. I think it can result in identifying places in the system where information about functionality is lacking, which when applied can make a user feel very confident about the system. I think my story above about the system that allows the export and import of a file explains this quite well. The users I dealt with back then certainly felt more confident in using that facility as a result of adding detailed information on its use.

A lot of the time, when the system is clear in what it does and follows well-thought-out processes that the user understands and they receive feedback from the system after carrying out commands, then users trust it and have confidence in it.

5. *Based on your experience, how do you think implementing the resulting requirements will result in improved **usability and usefulness** of software functions, in other words, ease-of-use software? Explain why or why not?*

I think my answer to the previous question goes into this.

6. *Based on your domain experience, do you think implementing the resulting requirements will result in increasing **temporary and permanent reassurance, data traceability**? In other words, helping the user to trace their activities and data in the software? Explain why or why not?*

If every TEM output was put in place, then I think some users would feel reassured, whereas some may feel like the system is trying to tell them too much. Depending upon what the system is trying to achieve, some users want more information and some just want it to do the job it does without any distractions or potentially unnecessary information.

7. *Based on your domain experience, do you think implementing the resulting requirements will help **in building user's trust** in the software? Explain how and why or why not?*

Yes, see above story around file export/import.

8. *Based on your domain experience, do you think implementing the resulting requirements will help users to **make more informed decisions** when interacting with the software? Can you explain how and why or why not?*

Yes, since if they have a clear understanding of what various system features do then they will know whether it is the appropriate feature that they need to use at that time.

9. *Based on your domain experience, do you think implementing the resulting requirements will **help meeting the user's expectations**? Can you elaborate?*

Users differ hugely, but all users expect some sort of response when they interact with a system. If the system responses validate what the user did whenever they use any part of the system, then the user's needs or expectations should be met. If it is unclear what a part of the system does, then it is important to set that user's expectations before they try to use that functionality. These, sometimes missing items, be it a message box or informational text can make all the difference when it comes to user satisfaction.

8.3 TEM Implementation on SpiritHub

New Transparency feature

Better defined functionality and user ability

Already well defined with the Use case

When a pattern has not been directly followed but has helped to think about the use case in another way

Table 8.1: User stories-TEM

<p>User Story 1. Account creation - (Booking Hub Administrator Portal)</p> <p>As a Hub Service Administrator, I need to be able to create and account using an easily memorable username and self-set password. I should be requested</p>	<p>Data driven pattern</p> <p>Alternative use case pattern</p> <p>GDPR</p>	<p>Data interests</p> <p>-Personal contact details</p>	<p>The system must communicate to the user that it holds email address. This can be edited by the admin team. Used only to set up an account and log in. Keep a record of who is</p>
--	--	--	--

<p>to set the password the first time I log into the system. I should be delivered to the Administrator Booking Hub home page.</p> <p>Test Case 1: When a Hub Service Administrator creates a login to the service for the first time, they should be prompted to setup a memorable password. The username should be their valid email address. They will need to enter a password of X characters with at least X number and a special character. They will need to enter the password twice to ensure they have entered it correctly.</p>			<p>making changes in the system.</p> <p>The system must communicate re-enter details if the email already within the system, email is not a valid email, password not of the correct complexity, passwords do not match</p> <p>The system holds email address, which is accessible by system admin, the details will be stored even after account deletion in order to maintain a record.</p>
<p>User Story 2. Login to Hub Admin Portal</p> <p>As a Hub Administrator I should be able to login to the system using my memorable username and password and have visibility of my administrative tasks.</p> <p>Test Case 1: At subsequent logon attempts the Hub Administrator should be asked to enter their username and password. On a successful logon they should be taken to the relevant homepage for the Hub Service Administrator</p>	<p>Dynamic Use Case Transparency Requirement Pattern Data driven pattern</p>		<p>If current system state doesn't meet login details at end of login, they system must communicate list of alternative available and enabled use-cases – reenter details – reset password Use case 3 in order to set up log in details.</p> <p>The system must communicate to the user that it holds data of the</p>

<p>and this will be a list of tasks that are currently unclaimed for any patients/ HCP/GPs which include SE bookings, referrals, enquiries and self-referrals. The home page of the administration system will provide menus and options to enable the Hub Administrator to carry out the system administration tasks.</p> <p>Test Case 3: On successful login the date and timestamp of the login should be recorded in the system audit tables.</p> <p>Test Case 4: On an unsuccessful login attempt the date and timestamp should be recorded alongside the reason that the login was unsuccessful e.g., Incorrect Password.</p> <p>Ask Nadine if these are all NHS.net passwords</p>			<p>type login record which can be viewed by the admin team</p>
<p>User Story 3. Reset Password</p> <p>As a Hub Service Administrator, I need to be able to perform a password reset if I forget my password, the password reset request link should be clearly visible on the login screen. I should be sent a password reset link to my registered valid email address registered with the system.</p>	<p>Process driven pattern</p> <p>Alternative process pattern</p>		<p>The system must communicate that if valid email address and account with the system and email will be sent to the email address</p> <p>Alternative pattern to contact the admin team</p>

<p>Test Case 1: When I request a password reset, I am asked for my username and if there is a valid email registered with the service an email is sent with a password reset link. Instructions on the screen should state that the password reset link has been sent to the registered email address for the account and to follow the instructions within the email.</p> <p>Test Case 2: When I request a password reset, I am asked for my username and if there is not a valid email registered with the service I should be directed to contact the service support office at Spirit Digital to gain assistance with gaining access to the system and registering a valid email address.</p>			<p>must be shown all the time in order to give the user too much information about the details in the system. The admin details should be displayed all the time.</p>
<p>User Story 4: Edit personal details</p> <p>Test Case 1: When an email address is entered or changed an email requesting confirmation should be sent to the email address. The email should not be deemed valid until the link in the email had been clicked. When the link in the email has been clicked the email should be marked as valid and is used by the system</p>	<p>Alternate Process</p> <p>Dynamic Data</p> <p>GDPR Data protection</p>		<p>In case personal details are updated by admin user the personal details can be viewed and updated by other admins and possible all other user types</p> <p>If email address is updated by admin the system must communicate to the user,</p>

<p>for communication with the user (password resets etc.) and as the users' username when logging into the system.</p> <p>Test Case 2: When a user logs into the system, if their email address has been changed and has not been validated it should be stated that the email address has not been validated in the personal details and the user should be prompted at each login attempt to validate their email address. The username will remain as the previous email address before changes were made. A link should also be presented resend the email validation email. When the email has been validated the link should no longer appear and the user should not receive further reminders when logging into the system. The email should appear as validated in the personal details section.</p> <p>Test Case 3: When any information is entered or edited in the personal details the old and new value should be audited in an audit table and timestamped along with the user who made the change.</p>			<p>the currently in use email address which is being used by the user and other admins – informing them that the email could have been changed.</p> <p>The system must communicate that it holds data of the type personal details which has access by the admin team and selected members of other user types stored even after account has been deleted due to auditing for at least 2 years.</p> <p>If the current system state doesn't meet email confirmed the system must communicate confirm email because it has been changed and not confirmed</p> <p>The system must communicate to the user that any changes made to the data are audited and details of the changer recorded.</p>
--	--	--	---

<p>User Story 5. Visibility of Tasks</p> <p>As a Service Hub Administrator, I must be able to see all referrals and tasks within the Booking Hub which include:</p> <p>Self-referrals</p> <p>Pending patients - flagged</p> <p>Referral by GP or HCP</p> <p>SE Provider Courses - menu of options and details</p> <p>On Hold Referrals- flagged DNAs (reported from the SE Providers)</p> <p>Any other messages e.g., requests/questions from SE Providers</p> <p>Test Case 1: When a Service Hub Administrator is logged into the Hub there must be a drop-down menu of all activities.</p> <p>Test Case 2: When the Service Hub Administrator clicks on an activity a detailed list is accessible showing all activity within the fields including flags (refer to user stories 6 & 7)</p>	<p>Static Use case</p>		<p>The system must communicate that on login in that “You are logged in please select from the tasks”</p>
<p>User Story 6. Patient self-referral process</p> <p>As a Service Hub Administrator, I must be able to see the completed self-referral template, check if patient information is missing and</p>	<p>Dynamic Data</p> <p>GDPR</p> <p>Process</p> <p>Alternate</p>	<p>Referral information including personal medical records.</p>	<p>#Data_Transparency_Referral_info</p> <p>#GDPR_Personal_Referral_info</p>

<p>integrate this template into their GP/HCP systems. There will often be missing data for the patient e.g., HbA1c, Date of diagnosis etc. and I will need to be able to request this missing information from the patient's GP/HCP in a timely manner (less than 2-week turnaround). If medical information is needed, I need to set this patient to 'pending' and flag it.</p> <p>Select from a drop down of General Practices with additional information like address present in the dropdown. Not just name.</p> <p>Test case 1: As a Service Hub Administrator I have access to all self-referring details input by the patient and am able to request information from the patient's GP practice using a standard template. When I request this information, the patient should be flagged to pending to allow for return of the details. This flag will ensure the patient is booked on the course as soon as all information is completed in a timely manner.</p> <p>Test case 2. As a Service Hub</p>		<p>GP information – email address – address etc.</p> <p>SNOMED</p> <p>Courses /Course bookings</p>	<p>The system must communicate to the user that it holds data of the type personal medical records and can be updated by Current user and care must be taken as the medical records must reflect the true records of the patient as they are used in the courses</p> <p>The system must communicate the option to re-request information from the GP, Decline the referral, if the information is incomplete.</p> <p><i>The system must communicate to the user that changes to the patient record that information is recorded to the SNOMED codes and sent back to the GP as part of the patient's medical record</i></p> <p><i>The system must communicate to the SE Provider that a patient</i></p>
---	--	--	---

<p>Administrator, I will be monitoring flagged Pending self-referrals for their completed medical information to be returned from their GP/HCP which will allow me to confirm them on their chosen course. When in receipt of all details, flagged pending will be set to booked and a message to the patient will be generated confirming their course and the details. The confirmed information will also be sent to the SE Provider to confirm this patient's place on the course.</p>			<p><i>has been confirmed onto a course due to the referral being completed.</i></p>
<p>User Story 7. Creation of Unique Referral Code for the patient on receipt of self-referral details</p> <p>The system will generate a Unique Reference Number for each patient (across the entire system) The Unique Reference Number will contain 6 numbers. This is to allow a Service Hub Administrator even greater ability to match patient records and clarify their details. This number will be used along with two other verifiers e.g., DOB, Post code</p> <p>Test case 1: On receipt of a patient's details the system will automatically assign that patient a</p>			<p>This is more of a back end process and doesn't really denote to any of the frameworks as is basic system labelling.</p>

<p>unique 6 letter reference which will be tagged to that patient e.g., Jane Doe -Unique Reference Code 123456</p>			
<p>User Story 8. GP/HCP patient referral process</p> <p>As a GP/ HCP I can refer a patient into The Hub using a standard referral template via e-RS and NHS.net email accounts. I can also direct highly motivated patients to self-refer via The Hub.</p> <p>Test case 1. The GP/HCP completes the patient’s full details which must include Hba1C, Date of diagnosis and mandatory personal fields into the standard template (refer to Emis document in files)</p> <p>Test case 2. The GP/HCP signposts the patient to The Hub following a discussion about courses available and suitable options for their diagnosis</p>	<p>Data transparency</p>	<p>Patient Referral info</p> <p>Course information</p> <p>Actors</p> <p>Patient</p> <p>Admin</p>	<p>#Data_Transparency_Referral_info</p> <p>#GDPR_Personal_Referral_info</p> <p><i>The system must communicate a list of available courses that are suitable for the patient as option for course</i></p>
<p>User story 9. Creation of Unique Referral Code for the patient on receipt of GP/HCP referral details</p> <p>To allow a Service Hub Administrator even greater ability to match patient records and clarify details the system will</p>			<p>This is more of a back end process and doesn’t really denote to any of the frameworks as is basic system labelling.</p>

<p>generate a Unique Reference Number for each patient (across the entire system) The Unique Reference Number will contain 6 numbers. This number will be used along with two other verifiers e.g., DOB, Post code.</p> <p>Test case 1: On receipt of a patient's details the system will automatically assign that patient a unique 6 letter reference which will be tagged to that patient e.g., Jane Doe -Unique Reference Code ABCDEF</p>			
<p>User story 10. The Hub receives and captures the referral then contacts the patient</p> <p>As a Service Hub Administrator, I receive the referral via the standard template and if the information is complete, I contact the patient to discuss the course options with the goal of booking the first course and sign posting the patient to access The Hub for further information.</p> <p>Test case 1. The Service Hub Administrator has all the information necessary to contact the patient and help them make a choice of course suitable for them, book them on the first course and sign post them to the</p>	<p>Process Static</p> <p>Dynamic process</p>	<p>Patient Contact details</p> <p>Course</p> <p>----</p> <p>Admin Patient</p> <p>Precondition- contact details</p>	<p>The system must communicate that because referral process is complete the patient can book onto a course and is encouraged to do so as soon as possible.</p> <p>If the system has not collected all the referral information, it must communicate to the admin that a patient has not all the required referral information and so cannot book onto a course. To fulfill the condition the GP must send a completed referral</p>

<p>website.</p>			<p>form, the admin can re-request a referral form from the users Referral page.</p> <p>If the system doesn't have contact details, then they must be obtained for the referral process on the referral page by contacting the gp.</p>
<p>User Story 11. The Service Hub Administrator books the patient on the first course and communicates the booking</p> <p>As a Service Hub Administrator, I have all the information needed to book the patient on their first SE course. Once the patient has decided on the SE Course, I must ensure to contact the relevant individuals with the relevant information i.e., SE Provider/ Patient (there is an exception - when a patient has no email address The Booking Hub needs to be able to override the mandatory email section when booking a patient on the course - ask Jane about this)</p> <p>Test case 1. The Hub informs the SE Provider of the booking by sending the booking confirmation and referral template information</p>	<p>Data transparency</p> <p>Static process</p>	<p>Actors –</p> <p>Admin</p> <p>Patient</p> <p>Data –</p> <p>Patient contact details</p> <p>Course details</p> <p>Precondition –</p> <p>Patient confirmed desire for course Referral complete (?)</p> <p>Postconditions</p> <p>Course booking has been made</p>	<p>#Data_Transparency_contact_details_patient</p> <p>The system must communicate the course has been booked after the course has been booked to the patient.</p>

<p>to the SE course provider in an electronic format that will easily integrate with their system.</p> <p>Test case 2. The Hub sends the patient confirmation of the booking by email/text (or letter if none of these exists) which will include full details of where and when the course will be with a map attached and other relevant details. This confirmation will also include a link to other resources available on the website. In the instance where a patient has no web access this will need to be sent in paper form.</p> <p>Test case 3. The Hub sends out timely SE Course booking reminder via text/email (letter as last resort if no email or mobile) to the patient</p> <p>Ask Nadine to list the courses and their length and if they have to be attended in order</p>			
<p>User Story 3. Reset Password</p> <p>As a Hub Service Administrator, I need to be able to perform a password reset if I forget my password, the password reset request link should be clearly visible on the login screen. I should be sent a password reset link to my registered valid email</p>	<p>Dynamic process</p> <p>Data transparency</p>	<p>Actors- Admin</p> <p>Data- Personal details</p> <p>Preconditions- Existing account</p>	<p>If the system does not have personal details /Email is not received, then it must communicate – Contact admin team – Or try another email – Check spam folder –</p> <p>#Data_Transparency_con</p>

<p>address registered with the system.</p> <p>Test Case 1: When I request a password reset, I am asked for my username and if there is a valid email registered with the service an email is sent with a password reset link. Instructions on the screen should state that the password reset link has been sent to the registered email address for the account and to follow the instructions within the email.</p> <p>Test Case 2: When I request a password reset, I am asked for my username and if there is not a valid email registered with the service I should be directed to contact the service support office at Spirit Digital to gain assistance with gaining access to the system and registering a valid email address.</p>		<p>Postconditions- Email received</p>	<p>tact_details_patient</p>
<p>User Story 13. Patient not contactable</p> <p>As a Service Hub Administrator if I cannot contact the patient after a set number of attempts (to be determined by Hub) I must send the referral back to the referrer i.e., GP/HCP/Diabetic Clinic to flag this and provide feedback.</p> <p>Test case 1. After the set number of attempts to contact the patient</p>	<p>Dynamic User case</p>	<p>Actor- Admin Patient</p> <p>Data – Patient Details SNOMED</p> <p>Precondition- Patient details</p>	<p><i>If the Patient is uncontactable the system must try to inform the patient that due to a lack of contact returns SNOMED code recording the uncontactable to be returned to the GP and to contact the admin team as soon as possible to prevent this from</i></p>

<p>the Service Hub Supervisor generates outcome information for that individual in an electronic format correctly coded (READ/SNOMED) and returns it to the referrer e.g., GP/HCP so it can be integrated in the patient record. The patient will only be contacted again if the GP/HCP makes another (new) referral.</p>		<p>Postconditions – Contact patient Do not contact patient</p>	<p><i>happening</i></p> <p>#GDPR_Personal_Referral_info</p> <p>*Difficult use case as it is to do with being unable to contact the patient.</p>
<p>User Story 14. A patient is not suitable for SE Course</p> <p>As a Service Hub Administrator, I may determine that the patient is not suitable for the SE course. In which case the patient must be referred back to the referring GP/HCP using an electronic format correctly coded (READ/SNOMED)</p> <p>Test case 1. Service Hub provider uses the electronic format and correct READ/SNOMED codes to communicate unsuitability to the referrer e.g., GP/HCP</p>	<p>GDPR Pattern</p> <p>Data transparency</p> <p>Dynamic use case pattern</p>	<p>Actor- Admin Patient</p> <p>Data- Patient Details Patient Medical details SNOMED Course Course Bookings</p> <p>Preconditions Patient is booked</p> <p>Postcondition Patient is no longer booked Patient is informed</p>	<p>The system must communicate that the patient will be informed by email at the cancelation of the course and their position will be made available to other patients.</p> <p>#GDPR_Personal_Referral_info</p> <p>#Data_Transparency_Referral_info</p> <p>If the user doesn't think that they are unsuitable for the course they must be told the options to contact their gp or contact the admin team.</p> <p>If the system must communicate to the user</p>

			that it holds data of the type SNOMED codes which are returned to the GP as part of your medical record and stored within the EMIS system and maybe kept for some time.
<p>User Story 15. A patient needs to be put on hold</p> <p>As a Service Hub Administrator there may be instances when a patient needs to be put ON HOLD (this could be because they cannot find a course geographically suitable/they need further support prior to receiving SE) I need to ensure I can actively identify and manage the referrals ON HOLD.</p> <p>To do this the information must be held in the database and flagged so I can contact this patient after an agreed length of time to offer them an SE Course if they are ready. An exception to the length of time to contact the patient would be if they are on a waiting list. Whilst these patients are ON HOLD, I will signpost them to other learning services via the website. I will capture the reasons patients are ON HOLD.</p>	Use case	<p>Actor- Patient Admin</p> <p>Data- Patient Referral details Admin Action</p> <p>Preconditions- Patient has been referred</p> <p>Postconditions- Patient cannot book course</p> <p>Patient is informed of their status</p> <p>Admin – Action Course Bookings</p>	<p>If a patient's records cannot be obtained the system must communicate to the user that their referral process has been suspended and will be unable to make a booking at this time. Once the correct information has been obtained, we will contact the patient to book them onto a course, an admin action will be created to remind of that.</p> <p>#GDPR_Personal_Referral_info</p> <p>#Data_Transparency_Referral_info</p> <p>If current system state meets patient has</p>

<p>Test story 1. When a patient is identified as not ready i.e., ON HOLD they must be flagged as such in the data base with a time stamp and a reminder set to re connect with the patient in a timely, set period e.g., 6 weeks.</p> <p>Test story 2. When a patient is ON HOLD because they are waitlisted for a particular course and that course becomes available, they should be contacted and offered the place. These patients should be flagged ON HOLD/WAITLISTED</p> <p>Test story 3. A Service Hub Administrator will capture the reason a patient is ON HOLD and keep a record of where they signposted the patient to in the database. This will be useful for reporting purposes.</p> <p>A Service Hub Provider does not need to inform the GP practice, in this instance, but should a GP practice call to ask the status of a patient, The Hub should easily be able to find that patient and tell the GP practice that they are on hold.</p>			<p>booking, then once the patient has been put on hold all course bookings must be cancelled</p> <ul style="list-style-type: none"> - This could merge with the one above.
<p>User story 16. Patient does not</p>	<p>Alternate</p>	<p>Actor-</p>	<p>Patient is contacted about</p>

<p>attend (DNA) the SE Course</p> <p>As a Service Hub Administrator, I need to be able to identify DNAs from the SE attendance information. I need to attempt to contact that patient and find out the reason for the DNA, reschedule them onto another course or signpost them to other support.</p> <p>Test story 1: The Service Hub Administrator attempts to text/call the patient as a first option to discuss reasons for DNA and tries to rebook (this could also include did not confirm also) If there is no answer then an email is sent (or letter)</p> <p>Test story 2. The Service Hub Administrator reports the DNA (if no contact from patient) to the referrer GP/HCP by means of an electronic format that can easily integrate with the GP/HCP system.</p>	<p>Process</p> <p>GDPR</p>	<p>Patient</p> <p>Se Course</p> <p>Educator</p> <p>Data –</p> <p>Course</p> <p>Attendance</p> <p>Course Details</p> <p>Patient Details</p> <p>SNOMED</p> <p>Preconditions –</p> <p>Patient DNA</p> <p>Postconditions-</p> <p>Patient informed</p> <p>SNOMED</p>	<p>missing the course.</p> <p>The system must communicate the patient is encouraged to book onto another course because they did not attend</p> <p>#Data_Transparency_SNOMED_Code</p>
<p>User Story 17. A Patient attends the SE course</p> <p>As a Service Hub Administrator, I must report the patient’s attendance on SE Course to their referring GP/HCP. I will be informed of this attendance by the SE Provider through their</p>	<p>Alternative</p> <p>Process pattern</p> <p>Data driven</p> <p>pattern</p>	<p>Actor</p> <p>Patient</p> <p>SE Course</p> <p>Educator</p> <p>Data</p> <p>Patient Details</p> <p>SNOMED</p>	<p>#Data_Transparency_contact_details_patient</p> <p>#Data_Transparency_SNOMED_Code</p> <p>The system must communicate book onto</p>

<p>attendance taking data which will be sent to The Hub. I must also contact the patient stating their attendance and signpost them to further resources and support.</p> <p>Test story 1. The Hub sends the attendance/completion letter to the referrer and the GP in an electronic format which will easily integrate into the GP/referrer systems (this could be HCP and the Diabetic Care Provider) with the correct READ/SNOMED codes, stating when and where the attendance was completed, any clinical information and any follow up actions.</p> <p>Test story 2. The Hub sends the patient a message to say they have completed the course. The message will include self-care messages/ follow up information and sign posting to website for other health information. (this process is different for Type 1 as they require much more intensive follow up)</p>		<p>Course Details</p> <p>Course Attendance details</p> <p>Preconditions</p> <p>Patient is booked onto a course</p> <p>Patient has referral</p> <p>Postconditions</p> <p>Updated SNOMED codes</p> <p>Booked onto another course instance if unattended</p> <p>Confirm attendance status with patient</p>	<p>another course if the patient does not book onto the course, alternatively they can be discharged from the system.</p> <p>A member of the admin team will contact you to discuss why you did not attend the course and try and book you onto another course.</p> <p>The system must communicate to the SE Course Educator that it must take the course attendance within 5 days of the course being completed in order to update the SNOMED codes. An email will be sent out to remind the educator of this.</p>
<p>User Story 18. Understanding when a course is completed that has several sessions. This will include the order of these</p>	<p>Use case Static</p>	<p>Actor</p> <p>SE Educator</p> <p>Data-</p>	<p>The system must communicate to the Course creator (SE Provider/Admin) that the</p>

<p>sessions.</p> <p>As a Service Hub Administrator, I need to know the courses duration, in which the order the SE is delivered and how to define 'completed' - which will be decide by the course provider (Talk to Jane for more clarity on this if needed)</p> <p>(Courses have the following indicators - Attended or DNA for each physical session, and then the Course Educator or Provider Administrator (not The Hub admin) should confirm if the course is considered complete. Whether or not a course</p> <p>Test Story 1. The SE course provider will communicate with The Booking Hub when the patient is considered completed and the Booking Hub will in turn complete the appropriate communications to the GP/HCP</p>		<p>Course Data</p> <p>Preconditions- Course exists</p> <p>Postconditions – Course Complete is defined</p>	<p>course complete definition must be defined before the Course creation process can be finished.</p> <p>The system must communicate to the SE Provider that when a patient is considered complete it must communicate to the SE provider that the system will communicate with the User about completion.</p>
<p>User Story 19. Follow up protocol for various courses</p> <p>As a Service Hub Administrator, I need to determine the follow up protocol for Type 1 and Type 2 Diabetics and ensure these patients receive appropriate reminders for all courses</p>	<p>Static data</p> <p>Dynamic Data</p>	<p>Actor Patient Admin</p> <p>Data Other Resources Questionnaire</p>	<p>#Data_Transparency_Referral_info</p> <p>In case course has been completed the system must communicate to the user that Other Resources and alternate course will</p>

<p>including remote.</p> <p>Test story 1. The Hub will send those patients with Type 1 a reminder 6/52 for follow up course. This will require that a patient to be flagged on date of initial course completion to be contacted 6 months (24 weeks for alert) from this date. This will also include collecting follow up quality of life (QOL) and baseline information</p> <p>Test story 2. The Hub will send out a 12-month reminder to those patients with Type 2. This will require that a patient to be flagged on date of initial course completion to be contacted 12 months (50 weeks for alert) months from this date. This will also include collecting follow up QOL and baseline information.</p>		<p>Precondition Completed course</p> <p>Postconditions Informed about other resources</p>	<p>be sent out to the patient and the admin team will contact you about booking onto another course.</p> <p>In case the course has been completed the system must communicate to the user that a quality-of-life questionnaire will be asked to be completed 6 months from the completion of the course as well as a feedback form for the course itself.</p>
<p>User Story 20. Support SE Provider when a course is cancelled.</p> <p>As a Service Hub Administrator, I must assist the SE provider if the provider cancels the course. I must notify all patients on the course of the cancellation by text/email/call (this may have to be by return to ensure the patient</p>	<p>Static data</p> <p>Dynamic data</p> <p>Dynamic process</p> <p>Dynamic use case</p>	<p>Actor - Admin</p> <p>Patient SE Course Educator</p> <p>Data</p> <p>User details</p> <p>Course details</p>	<p>#Data_Transparency_contact_details_patient</p> <p>- Also, for other actors (except admin)</p> <p>If the current system has booked patients it must communicate to the patients that the course has been cancelled and</p>

<p>receives the cancellation notification) and rebook them on other suitable courses.</p> <p>Test case 1. The Hub must be able to efficiently contact all patients booked on the course that is being cancelled. They may require a phone call to ensure all people are notified. All patients should be offered and booked on an alternative course. cancellations should be recorded for reporting purposes.</p>		<p>Preconditions Course exists</p> <p>Postconditions Course cancelled</p> <p>All actors informed Patients rebooked</p>	<p>alternative course are available to be booked onto, the admin team will contact them about booking onto another course.</p> <ul style="list-style-type: none"> - The system must create admin actions for postal patients and the appropriate contact method for all others. <p>If the system has educators on the course the system must communicate to all educators that the course has been cancelled.</p> <p>If the current system does not meet all patients informed of cancellation, then it must display to the admin that all the patients have not yet confirmed the cancellation.</p> <p>The system must communicate to all those using the system that the course has now been cancelled.</p>
--	--	--	--

			The system must communicate as the course has been cancelled the course will no longer be visible to the patients and that bookings can no longer be made. The system must communicate that in order to uncanceled the course the SE Provider must contact the admin team
<p>User Story 21. The Hub must identify any missing attendance information and a reminder sent to the SE Provider</p> <p>As a Service Hub Administrator, I need to identify any missing attendance information and contact the SE Provider to provide this within 5 days of the course completion.</p> <p>Test case 1. Should a course not be fully 'outcomes' within 48 hours, an auto email is sent to the Provider Course Administrator (usually the SE Super Admin, a named email address) saying XYZ course has not been fully outcome. This is repeated again in 48hours if it still remains incomplete</p>	Dynamic process	<p>Actor</p> <p>SE Provider</p> <p>SE Educator</p> <p>Data</p> <p>Course details</p> <p>Course attendance details</p> <p>Preconditions</p> <p>Course has finished</p> <p>Course has incomplete attendance record</p> <p>Postconditions</p> <p>Course has complete</p>	<p>If the current system doesn't meet the complete attendance at end of 48 hours after course the system must communicate to the SE Course provider that the attendance can be completed on the Course View page of the hub where all the attendance must be taken in order for the course to be considered complete</p> <p>The system must communicate that if the course is not completed after another 48 hours then another email will be sent out to the SE</p>

		attendance record	provider asking them to complete the course
<p>User Story 22. Ability to access the website</p> <p>When the patient lands on the home page of the website they have the option to browse the educational material and read about the various courses without needing to log in.</p> <p>As a patient I need to be able to access the website with either a URL link or by typing the website into my browser. I can view SE Courses detail and educational information on the website.</p> <p>(if I want to book a course, I need to create a login using my email and a password on the home page)</p> <p>Home page</p> <p>The home page should incorporate the following aspects:</p> <p>It should be responsive so as to be clear and usable on a laptop, tablet or smartphone.</p> <p>It should initially include a course search list based on a postcode that we can set. See item #518 in relation to the course list.</p> <p>The home page of the site should</p>	Dynamic Use case	Actor All Precondition Website exists and is up	<p>If the system is not up it must communicate to the user that the system is not up and list alternate action to try again later, if the problem persists to contact the administration team.</p> <p>The system must communicate the options available to the user at the home page.</p>

<p>contain:</p> <p>Welcome text, which reads: Booking Hub - Diabetes Courses In XYZ.</p> <p>A phone number of the call Centre as a callable link, which reads: for support in booking courses, please call: 0111 1111111</p> <p>Buttons to allow the visitor to register/sign up to the system. (PBI #514 & #517)</p> <p>A search facility for courses (PBI #518)</p>			
<p>User Story 23. Ability to book a course online by creating an online account Test Case 1.</p> <p>The patient chooses a course to book online (this could be a remote course or a face-to-face group course) To do this they need to be able to create an account using an easily memorable username and self-set password. The username should be their valid email address. They will need to enter a password at least 8 characters long, 1 number and a special character. They will need to enter the password twice to ensure they have entered it correctly.</p>	<p>Dynamic Use case</p> <p>Static Use Case</p> <p>Alternate Use case</p>	<p>Actor Patient</p> <p>Data Patient User details Course details SNOMED</p> <p>Preconditions Course Exists Courses exist 2 weeks for self-referral Patient is registered</p>	<p>If the current system doesn't have courses that exist at point of course search the system must communicate a list of alternatives to adjust the search parameters, contact the administration team to find available courses. It could be that no course is available within the given area.</p> <p>The system must communicate that course has been booked and the patient is able to attend the course with given</p>

<p>When they login they should have the option to save their sign in details for future login.</p> <p>https://diabetesbooking.co.uk/registration1?id=5d8a0b8935c2f for reference of how this is done in DB&L website</p> <p>Test Case 2. When the patient subsequently logs into the system, they should be able to see details of what courses they have previously attended and which courses they have booked onto and allow them to click into the details of those courses to identify things like date, time, location, facilities.</p> <p>Test Case 3. If a patient tries to book a course that they are already on, the system should warn that they are already on that course.</p> <p>Test Case 4. A patient should not be allowed to double book themselves.</p>		<p>Postconditions</p> <p>Course is booked</p> <p>Alternate paths</p> <p>Course is full</p> <p>Course has been cancelled</p> <p>Book a carer</p>	<p>details <<Use case 17>></p> <p>#Data_Transparency_SN OMED_Code</p> <p>The system must communicate that if the course is cancelled you will be informed that the course has been cancelled and to no attend the course.</p> <p>The system must communicate that the course is full and to choose to form other available courses or contact the administration team in order to book onto a course.</p> <p>The system must communicate that the patient can book a carer position to aid in attending the course. The number of carer positions will be made visible on each course. <<Carer use case>></p> <p>If the current system state doesn't meet the</p>
--	--	---	---

			precondition of having an account <<Logged in to an account>> the system must communicate Login, create account, contact admin, <<>> which can allow you to log in with an account.
<p>User story 24. Permission to inform patients of further health information and other courses</p> <p>As a patient I can opt out of receiving more health information, upcoming courses etc. via my preferred method of communication.</p> <p>Test Case 1. When I input my details online, I accept the offer to have further health information served up to me via email or text message.</p>	GDPR	<p>Actor Patient</p> <p>Data Patient personal details</p> <p>Preconditions Patient has account</p> <p>Postconditions Recorded status</p>	The system must communicate to the user that it holds data of the type further health information which is used to continue contacting you about other available courses.
<p>User Story 25. Reset Password</p> <p>As a patient I need to be able to perform a password reset if I forget my password, the password reset request link should be clearly visible on the login screen.</p> <p>I should be sent a password reset link to my registered valid email address registered with the system.</p> <p>Test Case 1: When I request a</p>	<p>Static Use Case</p> <p>Dynamic Data</p> <p>Dynamic Use case</p>	<p>Actor Patient</p> <p>Data Personal details</p> <p>Preconditions Patient has account</p> <p>Postcondition Password</p>	<p>The system must communicate that Patient must have an account and access to the email account associated with that account in order to reset the password.</p> <p>In case the <user><new password><string> is not valid the system must communicate to the</p>

<p>password reset, I am asked for my username and if there is a valid email registered with the service an email is sent with a password reset link. Instructions on the screen should state that the password reset link has been sent to the registered email address for the account and to follow the instructions within the email.</p> <p>Test Case 2: When I request a password reset I am asked for my username and if there is not a valid email registered with the service I should be directed to contact the service support office at The Booking Hub to gain assistance with gaining access to the system and registering a valid email address. The Hub contact number will be displayed on the website and the patient will be able to call for support.</p> <p>I need to be able to perform a password reset if I forget my password.</p>		<p>updated</p> <p>Alternative Password not suitable</p> <p>Not correct email address</p> <p>Passwords don't match</p>	<p>string gathered is not suitable for the given reason. Too short, No Capitals, No numbers, No special characters. Passwords don't match</p> <p>If the current system state doesn't meet correct email address the system must communicate contact admin team to help reset password and allow access to account, also check spam folder.</p> <p>If the current system state updates the password it must communicate to the user that the password has been updated and the new password is now in use.</p>
<p>User Story 26. Electronically complete a standard template with personal details</p> <p>As a patient I need to be able to complete an online template with my details for the purpose of being able to sign up for a</p>	<p>Data</p> <p>Dynamic use case</p>	<p>Actor</p> <p>Patient</p> <p>Data</p> <p>Personal details</p> <p>SNOMED</p>	<p>#Data_Transparency_contact_details_patient</p> <p>#Data_Transparency_Referral_info</p> <p>#Data_Transparency_Ref</p>

<p>Structured Educational (SE) course. I will need to complete: My full name DOB Address GP name and address Email address Mobile phone number Ethnicity (follow NHS standard drop down) Language of preference Preferred means of contact e.g., email, Text message</p> <p>Test Case 1. When a patient log into the system they will be asked to complete a standard template to collect their personal details to enable them to book onto a course. These details will be mandatory fields.</p> <p>(This information will allow The Hub to recognize the correct patient by using their unique details. The patient must complete the GP name and address to allow The Hub to send the patient's details to the Practice to let them know the patient has signed up for SE and for them to complete any information that might be missing e.g. HBA1C blood result - we want to assign a unique referral code to them at this time as per</p>		<p>Preconditions Selected a course Email address</p> <p>Postconditions Account created</p> <p>Alternative path Email already exists</p>	<p>erral_info</p> <p>#Data_Transparency_SN OMED_Code</p> <p>If the current system state doesn't meet select the course the system must communicate that in order to create an account the patient must first select a course on which to book onto, this can be done by selecting book a course and searching for available courses.</p> <p>If the system already has account with that email address the patient must be told to contact the admin team</p> <p>The system must communicate requirement email address before account can be created and to contact the admin team on the given number to create an account and book onto a course.</p>
---	--	---	--

User story 7)			
<p>User story 27. Display Structured Education Courses</p> <p>As a patient I need to be shown the appropriate SE courses for me. This will include courses available online (referred to as Remote) and face to face group courses. I will need to see the detail of what the course is offering, where the course is being held and the date and time.</p> <p>On entry of the home page containing the course location search, the system should display some search controls, followed by a map via Google (details of API key to follow). The system should also have two buttons available to switch between a map view and a list view.</p> <p>The basic search controls should be laid out horizontally across the page and consist of:</p> <ul style="list-style-type: none"> • Postcode text box + dropdown for +1, +2, +5, +10 and +50 miles. (default to +10) • Diabetes Type - Dropdown list (containing Type 1 and Type 2) • Course Type - Dropdown list (containing values from Course Type table) <p>The postcode textbox should</p>	<p>Static data</p> <p>Dynamic data</p> <p>GDPR</p>	<p>Actor</p> <p>Patient</p> <p>Data</p> <p>Course</p> <p>Course booking</p> <p>Patient details</p> <p>Locational data</p> <p>Preconditions</p> <p>Courses exists</p> <p>Postconditions</p> <p>All suitable courses are shown</p>	<p>The system must communicate that Uses Postcode to search for course and is only “accessed by system admin”. This data is anonymized and will be stored until your account deletion.</p> <p>acted on by the system</p> <p>In case the patients search parameters are do not return results the system must communicate to the user that the search parameters are too narrow and that if they widen them to a particular degree more results would become available.</p> <p>The system must communicate to the suer that it holds data of type postcode and locational data which will be sent to Google via google maps and stored on their servers.</p>

<p>attempt to identify the user's location. If that is not possible then a default postcode should be used, recorded in the settings file.</p> <p>When the screen is in map view, which is the default, then the resulting courses should then be marked as pins on the map, which should be visible beneath the search controls.</p> <p>If a user then clicks on a pin, the system should display a list of those courses available at that location.</p> <p>The course list screen should display the following information per course:</p> <p>Course Name</p> <p>Type of course - e.g., Type 2, Face to Face</p> <p>Description</p> <p>Type - e.g., Desmond, Empower, etc.</p> <p>Course Date - e.g., 03 Jun 2020</p> <p>Duration - e.g., Full day.</p> <p>If the user switches to list view, then the system should display a list of courses containing the information above.</p>			
---	--	--	--

<p>If the user wishes to use more detailed filters, then the following should be available via a hidden menu that becomes visible when clicked on:</p> <p>Month (heading) followed by each month name and a checkbox. Limits the course list to courses held on the selected months. This should only select future courses, so if it were April 2020, clicking on a course from February should result in the Feb 2021 courses being displayed.</p> <p>Day (heading) Include every day of the week along with a checkbox for each. Limits the course list to courses held on the selected days.</p> <p>Time of Day (heading) Include two options: daytime, evening. Limits the course list to courses that start before 5pm (daytime) and courses that start after 5pm (evening courses).</p> <p>Course Location Has (heading) Containing the following, plus checkboxes for each: Disabled Access, Parking, Lunch Included,</p>			
---	--	--	--

<p>Wi-Fi, Refreshments. Limits the courses as described. Info held in the Course instance Session table.</p> <p>Session Type (heading) this should include options for: Face to Face, Remote / Digital. DB field yet to be created.</p> <p>Diabetes Type (heading). Either Type 1 or Type 2. DB field yet to be created.</p> <p>The course list screen should allow the user to easily click on more details or simply allow the user to go straight to course booking. If the user is not logged in, then it should take them to the registration/login page.</p>			
<p>User Story 28. Patient receives a message thanking them for signing up to a course and confirming they have a provisional booking</p>	Data		<p>#Data_Transparency_contact_details_patient</p>
<p>User Story 29. Receive appropriate reminders and messages about upcoming course</p> <p>Once booked onto an SE course I need to receive a reminder of my upcoming course with details of time and place. This will be in the form of email or text. I will have chosen my preferred means</p>	<p>GDPR</p> <p>Dynamic Use case</p>	<p>Actor Patient</p> <p>Data Patient Details Course bookings</p> <p>Preconditions Patient is booked onto</p>	<p>#GDPR_Personal_Referral_info</p> <p>#Data_Transparency_contact_details_patient</p> <p>If the current system is within 2 weeks 1 week and 2 days of the start of the course the patient</p>

of communication when I created my login. Test Case 1. The patient will receive relevant reminders of their course and the course details.		course Postconditions Patient is reminded of course	must be sent an email to remind them of the course, stating the alternate options to cancel the course, if any changes to the course are made then the admin will contact them, if they have any questions, they can contact the admin team.
---	--	---	--

The developers created hashtag codes for shared data features during implementing TEM below (see Table 8.2).

Table 8.2: Shared data features

#Data_Transparency_Referral_info	The system must communicate to the user that it holds data of the type Referral and can be view and edited by Admin and Viewed by SE provider and Educator.
#GDPR_Personal_Referral_info	The system must communicate to the user that it holds data of the type Referral information which admins can view and edit, SE Course providers and Educators can view
#Data_Transparency_contact_details_patient	The system must communicate to the user that it holds data of the type Personal Contact information and can be view and edited by Admin and Viewed by SE provider and Educator.
#Data_Transparency_SNOMED_Code	The system must communicate to the user that it collects data which is returned the GP system to update their medical records via SNOMED codes which can be viewed by the GP and admin team, 3 rd parties via the GP