Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 3 / DECEMBER 2010 / 65-77
© 2010 Universidad de Talca - Chile

This paper is available online at www.jtaer.com
DOI:10.4067/S0718-18762010000300007

Practitioner's challenges in designing trust into online systems

Haralambos Mouratidis[1] and Piotr Cofta[2]

[1] School of Computing, IT and Engineering, University of East London, UK, haris@uel.ac.uk
[2] British Telecom Plc, Adastral Park, Martlesham, UK, piotr.cofta@bt.com

Received 14 February 2010; received in revised form 24 September 2010; accepted 14 October 2010

Abstract

It is widely recognised that successful online systems are not those that simply fulfil some functional specifications, but rather systems that are developed to also meet a number of non-functional requirements such as security, reliability and trust. It is also generally accepted that the development of quality online systems is not anymore just the task of single individuals or mono-cultural teams but it often involves diverse and disperse teams with members from different cultures and backgrounds. Some non-functional requirements (such as security or reliability) enjoy a general consensus, independent of the cultural background of the developers. Unfortunately, this is not the case for trust, despite the growing interest in trustworthy information systems. The very nature of trust indicates that it is understood differently by different individuals and relates to personal experiences more than other non-functional requirements. In this paper we identify the field of study to support the inclusion of considerations regarding trust in the design of online systems, to provide the understanding and support that is in par with security or reliability.

Key words: Trustworthy Online Systems, Non-functional Requirements, Multi-cultural software teams, Quality online systems, Security and Trust.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 3 / DECEMBER 2010 / 65-77
© 2010 Universidad de Talca - Chile

This paper is available online at
www.jtaer.com
DOI:10.4067/S0718-18762010000300007

1 Introduction

The wide usage of online software systems, their operation on a worldwide scale, and the global economic dynamics, has introduced new challenges with regards to their development. Such challenges are related not only to the technical issues surrounding the engineering of these systems, such as architectures and interoperability, but also issues related to the human aspects and globalisation such as the nature of the development teams. Quality online systems are not considered anymore only those that meet their functional requirements but rather systems that are developed to also meet a number of non-functional requirements such as security, reliability and trust. From those, security and reliability are the ones that possibly have received the most attention so far, where an understanding exists about the importance of reliable systems and the need to incorporate security considerations from the early stages of the systems development process [31].

On the other hand, the closer analysis of this situation reveals the complex network of issues related to various aspects of trust. Systems are supposed to reflect relationships of trust, but they are also supposed to support assessment of trust and the development of trust. Considerations related to trust are spread along all phases of system lifecycle, from requirements to decommissioning. Further, there are several stakeholders whose trust has to be taken into account, whether operators or users, companies or the society as a whole. Even the process of design is conditioned on trust within the design team and enabled by collaborative software that builds on trust.

It comes therefore as no surprise that the interest in trustworthy ICT is growing, fuelled further by the fact that trust generates significant benefits, whether commercial or sociological. Designing trust into online systems enables security [2], improves quality [8], drives user adoption [6] as well as increases profit margins [37]. Therefore, designers are increasingly required to design trust into the online systems.

The simplistic view on trustworthiness, which equates with the brand or provenance of the system, is not enough. However, the design community is ill-prepared to take the challenge of trust, leading to the situation where trust is mainly ignored during the software engineering processes. While reliable systems have to pass agreed stress tests and secure systems have to acquire certification, trust does not have the similar stringent design or acceptance criteria. The understanding of trust and trustworthiness too, often, rely only on the perception of designers. For as long as they represent the socially coherent group, such an approach was somehow grounded. As the globalisation brought virtual multi-cultural teams, this approach easily fails.

It is now generally agreed that online systems development teams might include members from different cultures and backgrounds, working across time zones and language barriers. The literature provides analysis of cultural dimensions in software engineering teams and experiences have been presented on the literature and discussed against cultural background information based on works from Hoftstede [21]. In an experience report, Borchers [7] indicates that software projects involving multi-cultural teams have been proved quite challenging and a number of “best practices” have been ineffective or difficult to implement.

One of the key reasons for such a possible failure of designing trust into online systems is the fact that trust is perceived differently, and quite often in relation to culture [40]. The cultural dependence of trust is so important that in fact, trust seems to be the connecting element between security and culture, pervasive yet usually overlooked. Such cultural dependence is particularly important when systems are jointly developed by members of different cultures, each one bringing their own salient assumptions about trust. Potential conflicts between those perceptions of trust can easily lead to the failure of the software project.

It is therefore important for designing trust into online systems that software engineering methodologies incorporate objective reasoning and modelling of trust issues in their development stages, so software engineers receive assistance in understanding the various trust related issues introduced not only by the system but also from the environment where the system will be placed. It is only then that appropriate design solutions can be identified and implemented.

We believe that in order to develop trustworthy information systems, practitioners (designers, developers etc.) should be equipped with the standardised body of knowledge and practice that can be directly applied to various stages of the development of software systems. We call such a body 'designing for trust'. This paper discusses the situation, proposes 'designing for trust' as a field of study and argues about future research directions.

The paper is structured as follows. Section 2 sets the practitioner's challenge, highlighting similarities and differences between trustworthiness and security. Section 3 discusses the role of trust in software systems development. highlighting different views on trustworthiness. Section 4 lays the foundation for the 'designing for trust'. Section 6 concludes the paper and presents discussion on future work.

2 The Challenge

As an evocative challenge, let us consider a team of practitioners, coming from different cultures and backgrounds, who have been charged with designing, developing, delivering and operating a complex online software system. In this paper the term online software systems refers to Internet/Web based systems, such as e-commerce systems, e-banking systems etc. It is worth noting that throughout the paper we use the terms “online systems” and “software systems” to refer to online software systems. Buried somewhere in the specification of such a system there is a list of non-functional requirements: the system must be secure, reliable and trusted. The need for trust may have emerged from its value to the business, or from the reality of a marketplace, or possibly from the desire to improve upon customer relationship. Whatever way it happened - it is there.

The practitioners can easily understand how to make the system reliable or secure. They can base their decisions on existing international standards and call certified professionals and discuss with them using the mutually understood technical vocabulary. They have access to design methodologies, tools, products and operational procedures, all developed along the shared meaning of 'reliability' or 'security'. They can finally set the baseline and then monitor progress towards accreditation using generally internationally approved metrics. Culture and different backgrounds do not particularly affect the situation, since technical aspects of reliability and security are well understood and independent of various cultural differences.

Contrasting, the requirement for trust leaves the team in a vacuum. What they face is an endless series of questions. What definition of trust should be applied - behavioural, cognitive, and statistical? - and how this definition fits with their own perception of trust? Is this definition culturally acceptable? Is the requirement really about a system that has to be trusted, that is desired to be trusted or that is supposed to be trustworthy? Does this requirement relate to the design, to operation, to the system or to its social environment? Who should be consulted - psychologist, social scientist, cryptographer or philosopher? - How can requirements be captured, progress measured, designs developed and tested? What design procedures should be applied and what methodologies should be used? What metric can be used to judge the success?

The team can turn to relevant initiatives such as the Trusted Computing [34], only to discover that this is about systems that have to be trusted in order to provide sufficient security, not that they deserve trust. They may consider Trustworthy Computing [33], but such systems promise only security, privacy, reliability and availability, in expectation that this will improve trust in providers of such systems. They may turn to computing with social trust [18], to see that such an approach channels operational trust that is external to the system, not requiring the system itself to be trusted. Whatever way they turn, there are answers that are partial and that require significant academic effort to comprehend and apply to practice. Moreover, such answers might not be acceptable to all the members of the team depending on their cultural understanding of trust.

What they need is a set of generally accepted goals and reference models of trust, methods to assess the state of trust and improve on it, access to tools, products, verification procedures, ability to hire certified professionals etc. Without such an eco-system of pragmatic trust they will not be able to progress towards trust in software systems, mostly because they will not be able to identify the appropriate foundations to allow understanding and progression of such challenge.

3 Trust and Design

Trust is one of the most pervasive yet least understood phenomena, and the concept of trustworthiness follow this suit. Trust has a great social and business potential (albeit not fully explored or understood), ranging from 40% cost savings to 8% increase in profit margin, aiding social stability, improving wellbeing and supporting development. Because of its importance in software systems (online or otherwise), a number of researchers have tried to tackle trust.. Although it is beyond the scope of this paper to present a comprehensive list of works related to trust in software systems, we present below approaches that are related to the idea of “design for trust”.

According to the literature, “…Trust is a term with many meanings” [29], [30] that is difficult to define, convey, measure or specify. The definition of trust presented below is derived from typical constructs found in the literature (e.g. [27].)

> *[Trust is] the willingness of a party (trustor) to be vulnerable to the actions of another party (trustee) based on expectation that the other party will perform a particular action important to the trustor, irrespective of the ability to monitor or control that other party.*

The relationship between trust and ICT technology (such as online systems) is a particularly complex one, and is still a subject of a research debate [12]. Even the simple question whether trust can be extended towards technical systems does not have a clear answer. Jøsang [23] argues that individuals, from different cultures and backgrounds, are willing to trust software systems, as they trust other humans, knowing that there are potential risks. Dennett [13] explains this phenomenon by referring to intentionality, which is simplifying the perception. On the other hand, research by Lacohee et al. [25] emphasises that trust is directed towards the humans behind the systems and not the systems itself, while systems are treated as conveying a message about trustworthiness embedded into their design.

The development of modern software systems went towards increased integration and inter-connectivity, with the Internet or SOA being only one example. Therefore we are painfully aware that the complex modern technology is quite often an obstacle, as it brings yet more questions: why one should trust technology? and how can machines trust each other? In situations where operators of such systems are remote, relationships between systems are volatile and decisions regarding trust have to be made fast. Our everyday experience is usually of a problematic kind where computers refuse to cooperate, network outages stall work and cars tend to fail at the least convenient moments. If technology is to be used to develop trust between people, and support multi-cultural trusted models, then such technology must be first and foremost trusted. As research [10] has shown, if trust is not present, if there is no confidence, expectation, belief and faith in a software system, then there will be no willingness to rely on any such systems.

Further, modern software systems store sensitive and important information while communicate and exchange information with a large number of other systems. In order to guarantee security and privacy of information, software systems must be guaranteed to only exchange information with other systems that are trustworthy. As a result, the importance of assessing and analysing trust (both between software systems and humans; and between software systems that represent its operators) as part of the development process has been brought to light [2]. Recent literature states that “trust is becoming an increasingly important issue in the design of many kinds of software systems” [43] since “uncertainties and concerns of various stakeholders and participants need to be considered and addressed”.

Suttcliffe [38] argues that design and trust intersect in two ways. Firstly, if the design of a software system is not thought-out prior to the development stage, then there is a possibility that the system will not be built as per the user’s requirements. When the user actually utilizes the system, she/he will be made aware that her/his requirements haven’t been fulfilled, hence causing distrust of the system and the rejection of it. The second way that design and trust are intersected is by having ‘technology acting as a mediator of trust between people, organizations or products’ [38].

We can expand on this insight to identify three different types of discussions regarding trust and design of ICT systems. The closer analysis of the social context of a typical on-line system reveals at least three social actors of potentially different requirements regarding trust (Figure 1).

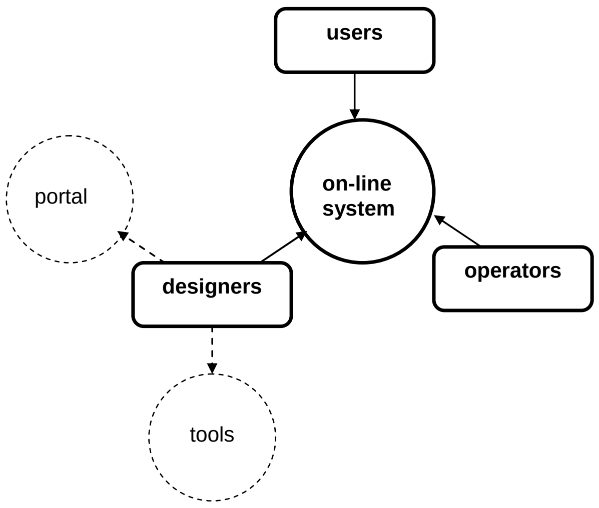

Figure 1: Social Context of the on-line System

1. Designers (and developers), the community that this paper focuses on, are responsible for designing and producing the actual system. They view the system as a product that is reflecting requirements of their customers, and hand-over the system-product to operators.

2. Operators are responsible for the everyday operation of the system, including maintenance, management, supervision, necessary upgrades, and also decommissioning. For on-line systems operators tend to provide the on-line platform, appropriate access etc.

3. Users actually use the system for their own purposes, that may or may not be in accord with the original set of requirements. For modern on-line systems users quite often contribute the major part of the content, either directly (user-generated content) or indirectly (e.g. e-mail).

Note that the honeycomb-like structure can repeat itself, revealing more social actors and systems. For example, the designer community themselves can act as users with regard to design tools that they are using (e.g. compilers, repositories). They can also act as operators e.g. for customer portal that facilitates on-line requirement gathering.

Three types of social actors suggest that the question of trust suggest that there may be at least three different views on the role and position of trust in design of on-line systems. Those views are characterised below, all from the perspective of their impact on the design process.

3.1 Trust-awareness

The trust-aware design takes into account similarities and differences between views on trust and trustworthiness of the variety of social actors that are and will be with the system. The main purpose of the trust-aware design is to capture customer requirements regarding trust, and to reflect them in the architecture of the system. Trust-aware design builds on existing software development methodologies, by enriching them with means to capture and reason about trust.

Trust-awareness closely resembles the view of the design community on the role and position of trust. Here' trust is a property of the operational environment of the system that ahs to be captured, structured and formalised in a form of a requirement. The trust-aware system is the one that satisfies trust-related requirements. While it may look as if this approach exhausts the need of the design community, the complexity of the challenge require an understanding of others approaches.

Typical examples of trust-aware design address the problem by adding trust-related extension to existing design frameworks, thus combining the familiarity of an approach with addressing new challenges. For example, Secure Tropos [17] extends the Tropos methodology with the concepts of trust, delegation, provisioning and ownership in order to allow the developer to capture trust relationships at a social and individual level. Bimrah [3] extends the secure Tropos [31] methodology with the concepts of request, action, trust relationship, trusting intention, repetitive knowledge, recommendation and consequence in order to model trust. The developer is guided through a series of models in order to analyse and reason about trust relationships.

Similarly, in [35] the proposed method makes use of the Goal Requirement Language (GRL) and Use Case Map (UCM) which both of them belong to the User Requirement Notation(URN). In particular, trust is captured as a soft goal because of its uncertainty of whether it is has been satisfied or not and because of its fuzzy nature. Further analysis of trust as a soft goal will eventually lead to well defined tasks. The UMLtrust framework of Uddin and Zulekernine [41] considers trust from the early stages of the development process and it is a scenario based analysis of trust. It uses UML which is well accepted among software developers.

The impact of such considerations on the architecture can be seen e.g. in Yan and Cofta [42] where trust is a set of statements and goals and trust domains are defined as areas of mobile communication where the trust definition is common among the various elements. Because of the subjectivity of trust definitions there are gaps between the trust domains and certain elements bridge the gap and are responsible for ensuring trust at a higher level than the one of the domains.

3.2 Trust-expressiveness

Trust-expressiveness shifts the focus from design to operation, thus reflecting the view of the operator community. It provides tools and methods to express and reason about trust, but it does not reflect any particular relationships of trust or distrust in the design itself. Consequently, the design is 'trust neutral', and the purpose of trust-expressive design is to enable the expressiveness of trust in the operation.

However, while the 'trust neutrality' may be the intention, the potential expressiveness is constrained by design decisions. For example, by including or excluding particular protocols or algorithms, the designer can enable or inhibit particular forms of trust. If, say, the system design does not include a provision for a reputation-based trust, then trust will not be able to be expressed in a form of a reputation.

Requirements regarding trust expressiveness are usually captured as functional ones, as (from the design perspective), it is a function of the target system to deliver a functionality that supports the expression of trust. However, there is a dependency between information captured by trust-aware design and ability to operate a required scheme to express trust. For example, if the assumed operator of the PKI infrastructure is not trusted, then no implementation can overcome it.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 3 / DECEMBER 2010 / 65-77
© 2010 Universidad de Talca - Chile

This paper is available online at www.jtaer.com
DOI:10.4067/S0718-18762010000300007

The trust-expressive approach bears the very close resemblance to current security practices, where there is a clear operational separation between policies and policy execution points. Designers of the actual system can construct effective security enforcement means without knowing what policies will be implemented. The design of actual polices can be separated from the design of an ICT system, without the need for any re-development.

Trust expressiveness is closely related to the well established area of trust management, and the need for trust expressiveness can be often structured as a requirement for the appropriate trust management system. Trust management is defined as “the activity of collecting, encoding, analysing and presenting evidence relating to competence, honesty, security or dependability with the purpose of making assessments and decisions regarding trust relationships for Internet applications” [20]

Earlier approaches focused on trust management mechanisms of internet applications. The PolicyMaker approach [4] is proposed for trust management of internet applications and builds trust relationships between entities [5]. REFEREE [11] is a trust management system for web applications that specifies a language for defining trust polices and provides a general policy evaluation mechanism for web clients and servers. Further, WS-Trust [1], a rich protocol to negotiate and reason about abstracted trust.

Several social sites provide tools for its users to express trust (e.g. in a form of reputation and recommendation). There are several; formal models that support such an approach to expressing trust [9].

3.3 Trust-building

The last approach, trust-building, reflects the user view on trust in the system. Users may be less concerned with trust within the system, but they are greatly concerned with trust that the system can generate, sustain or break within the user community.

The requirement to support trust building is usually not captured by either functional or non-functional requirements. Users are quite often unaware that the system will impact on their social processes, including trust building, and realise the extent of this impact only after the deployment of the system. Only for the relatively small class of systems (e.g. cooperative tools), trust building may be of a primary concern. Others may express certain desires regarding trust in a form of 'usability' requirements.

A number of works have looked at models and frameworks of trust between partners using online systems, for the purpose of supporting decisions to trust. Tan [39] proposes a matrix model which can enable trading partners in e-commerce to analyse trust building services. An e-service is represented in the form of a grid and the grid rows represent a theoretical decomposition of the notion of trust into four reasons namely social signs, personal experience, understanding and communality. The TrustCoM project provided a framework that models trust relationships based on electronic contracts, policies, and reputation systems [22]. The novelty of that framework lies in its ability to support companies to integrate services, processes and resources and form virtual organizations.

Similarly, in Gorski et al. [19] a trust case is represented by UML stereotypes that influence the trust level of the trustor, by providing evidence. The evidence can be in the form of claim, fact and assumption and the Claim Definition Language (CDL) is used to define claims.

Another approach, popular among on-line services, is to concentrate on user experience to elicit trust through appropriate design of user interface [16], or through enhancing the communication channel between users. Yet another approach is to develop systems that are specifically designed to support the development of trust [15].

4 Designing for Trust

While there are three communities involved (designers, operators and users), their responsibilities are not identical, as three approaches to trust in design are hierarchically narrowing. Thus trust-aware design includes into design assumptions that cannot be easily overcome by operators or users. It can be even argued that the structure of trust embedded into the design of the system impacts on future relationships between operators and users. Similarly, operator-drive expressiveness constraints users in their ability to build trust.

Therefore trust-aware design constraints and facilitates what trust-expressiveness can and cannot do. Subsequently, trust-expressiveness constraints what trust-building can do. Because of this dependency, designers should not only be versed with trust-awareness, but they should also anticipate operator and user needs regarding trust-expressiveness and trust building.

As an illustration, let's consider three different statements that may be found among requirements related to trust:

- A is trusted. Such a statement may signals the design-time decision, i.e. it enables the creation of an architecture that is conditioned on the fact that A is trusted. For example, A may become a holder of a top-

This paper is available online at www.jtaer.com
DOI:10.4067/S0718-18762010000300007

level secret that allows him to affect the security of the system. Note that the dynamics of trust in A (e.g. the ability to alter trust or revoke trust) directly impacts on the architecture.

- B can be trusted. Such a statement tends to identify the operational decision: B may be trusted or not, and there must be means to express such a trust (or the lack of it). For example, there should be variable related to B that indicates trust. The specificity of the future trust in B (e.g. who will trust B, on what condition, whether trust can be revoked etc.) must be reflected in the architecture.

- C wants to trust. This is the user-specific decision: C has an intention to build trust with others, trust that currently is unlikely to exist. Such a decision has to be supported in the design, e.g. by providing C with evidence of trust, or by allowing C to review historical information about others - even if those 'others' are not know at the moment.

The challenge that the designer faces is not only to capture the richness of requirements regarding trust, but also to embed them into the design in a way that enable them to be carried forward throughout the lifetime of the system.

The design practitioner, therefore, is faced with not a single design problem, but with three hierarchically dependent design problems. Such an array of problems cannot be handled without a proper preparation. The practitioner's challenge presented earlier clearly illustrate that without equipping designers with the proper body of knowledge and tools, trust is likely to overlooked, ignored or incorrectly incorporated into design. Therefore, the challenge that we are facing is to convert the loosely defined concept of incorporating trust into design into a set of practical design skills, supported by appropriate body of knowledge, methodology and tools, integrated under the banner of 'designing for trust'

It may not be warranted to state that 'designing for trust' should become a discipline, even though it should incorporate several elements of a new design discipline, positioned as a sub-discipline of a design of ICT systems. The word discipline (Latin: disciplina) comes from the Latin word discer, which means to learn. Although the word discipline is widely used (as in school discipline, self-discipline and so on) our aim is to define a field of study that may grow into a discipline, and therefore the rest of this section should be read with this in mind. A research discipline usually refers to a body of knowledge and its associated subject area (or field of study) and it lays the foundations, in terms of focus of study, challenges and so on of that subject area.

We model the proposition of 'designing for trust' after established disciplines of software engineering and information security. In our effort to define “designing for trust” as a field of study, we explore its foundations based on a number of characteristics suggested in the literature [26][32].

4.1 Motivation

The practitioners challenge presented above, has uncovered significant deficiencies in the current design methodology. Those deficiencies motivate the 'designing for trust', and set its requirements.

Variety of Definitions: Trust has been defined differently in different contexts and by different researchers. This forms a real barrier if a culturally independent and widely available definition of trust needs to be defined. It is therefore necessary to develop a definition of trust that is relevant to the design process, while flexible enough to capture the richness of the construct of trust.

Continuum of problems. Three aspects: trust-aware, trust-expressiveness and trust-building demonstrate that there is a continuum of problems related to trust that spans the complete system lifecycle. Requirements that pertain to the different aspects have a different impact on the design phase. It is therefore necessary to develop the methodology that provides a clear link between trust-related requirements and design practices.

Inconsistency of properties of trust: As an extension to the problem of definition, trust is attributed and overloaded with several properties, thus creating different expectations and soliciting various practical solutions. Without an agreed set of properties it is not possible to model trust in a scalable way, or to automate the process. Therefore 'designing for trust' should provide the simple yet concise model of trust, applicable to the design.

Disparity of practice: trust suffers from a significant disparity between everyday practice (such as trust management or trusted interface) and highly theoretical approach delivered by researchers. Design practice should be informed by theory, but there is a need for a supportive framework that will make theory consumable by practitioners.

Isolation of construct: Research [2][40] has argued that trust is not to be considered in isolation but a number of concepts are related to it, such as identity, control, complexity, and reciprocity. However, the richness of the area is not yet met by tools of expression that will enable capturing the variety of constructs related and associated with trust.

Subjectivity of trust: Trust is considered to be subjective and its transitivity is quite often questioned, specifically when different cultures and meanings of trust collide. However, the globalisation of information systems call for the globalisation of trust, and this in turn calls for methods that will construct inter-subjective and culturally- independent trust.

Lack of shared knowledge: It is characteristic to research and practice of trust that different disciplines independently developed their view on trust, which is neither shared nor influenced by other disciplines. However, inter-disciplinary knowledge sharing is a necessary foundation for the progress in better understanding and utilising trust.

Lack of appropriate education: Education in trust, within the context of software systems, stays within natural fault lines of disciplines and seldom venture into practice. Further, there are no recognised qualification and certifications associated with trust, in the context of software systems, so that there is no method to compare, judge and approve practitioners with regard to their expertise in developing trustworthy software systems.

4.2 Positioning

One may argue that the triplicate nature of trust implies that the responsibility should be shared between the design, operator and user communities. While this is fundamentally true, the argument of efficiency and effectiveness strongly favour early integration of trust-related considerations into the system, preferably at the design stage.

Trust in some works is treated as an ad-hoc operational feature, and that treatment tends to be more difficult and costly [36]. When trust requirements are not considered in a consistent way from the early stages of the software development, but in an ad-hoc way, they will result in conflicts with the other functional and non functional requirements of the system. So, in order to resolve these conflicts more valuable resources, such money and time, will be needed, that will still lead to a more complex solution.

It is therefore important that trust should be treated in a holistic manner throughout the whole lifecycle of the system, not through an arbitrary selection and spot application of tools and methods. Only then the software developers will have the knowledge of the problem under investigation and a complete picture of it. In addition, developers of a project might have a different cultural background and have a different understanding of trust. Therefore, not only they need a methodology that can guide them through the software development process, but also a methodology that will establish a common understanding of trust within a technical setting among the developers. Moreover, the existing approaches lead to information systems that prevent individuals, who are using these systems, from their natural ability to assess the trustworthiness of other individuals. Therefore, this leads to requirements of greater control, which are more complex and expensive. So, the existence of a methodology that considers trust issues becomes crucial for the development of trustworthy information systems.

Specifically, there is a growing research theme that recognises that trust, due to its complexity, is not just an issue to be dealt with at application level but rather trust should be looked at from a software engineering point of view and it should be considered from the early stages of a software system development. Yu and Liu [43] attempted to address some of the issues of considering trust at the requirements level of the system development process. They consider trust as a non functional requirement, where trust is a combination of all or some quality attributes of a system under development and they demonstrate their approach by describing the behaviour of a system in the case of attack and examine defences that are needed from trust perspective.

4.3 Definition

We define “Designing for Trust” as the field of study concerned with the socio-technical development of software systems that incorporate trust into its design. In particular, it is concerned with the knowledge (theoretical and practical), principles, practices as well as the establishment of a research agenda regarding the development of trust-related software systems.

The underlying aim of the field of study is to improve the social adoption and quality of software systems by building systems that better implements a variety of trust-related non-functional requirements. Such a field of study will be positioned mid-way between existing everyday practice and the sphere of ideas and concepts as shown in Fig 2.

4.4 Focus of Study

Every field of study aims to address a unique fundamental question or the focus of study. Such a question must have enough substance to evolve into a classical field of study [26] and be independent of technological changes [24].

The fundamental question for the discussed discipline can be formulated as “*how to develop software systems that reflect in their design social considerations regarding trust*”. In answering such a question, many sub-questions need to be formulated and answered. For example, “*how trust can be captured as a requirement?*”, "*how the* *implementation can be verified against trust-related requirements*", "*how to minimise the trust-related abuse of the system?*" or "*does it require a significant re-thinking of current development practices?*".

Usually, different researchers and practitioners will answer differently such questions, especially as some of those questions are inter-disciplinary. However, it is imperative that common answers are established in such fundamental issues, in order to provide a well-developed foundation in which we will be able to base further research questions leading us closer to answer the fundamental question of the discipline. We do not offer any definitive answer to those questions, but rather postulate that answering those questions should contribute to the growth of the discipline.

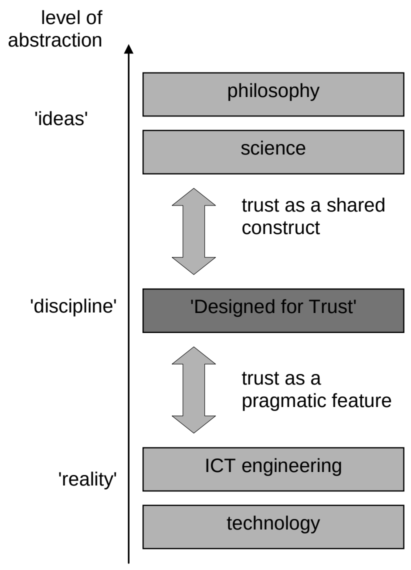

Figure 2: Designing for Trust as a Field of Study

4.5 Paradigm

The way that the 'designing for trust' views the world guides its development through practice and research [14]. We envisage the maturity of the 'designing for trust' in such a degree that software systems developers will be able to model, construct, test, deploy and maintain trusted software systems through well defined and structured processes and with the aid of appropriate modelling languages and methods. In such a vision, development is made even easier with the aid of computer-aided tools that enable to accurately analyse trust related issues and modify system models accordingly.

For that purpose, developers should accept and share certain view on trust, the paradigm, which can be used as a starting point for a discussion that will lead the development of this field of study.

1. **Trust is a pervasive socio-technical phenomenon**. Trust is embedded in and precedes every meaningful communication, making it one of the most pervasive phenomena. Trust must be always studied as the socio-technical phenomenon. Trust is not attributable to technical artefacts, but only to social actors. Technical artefacts (including software) represent trust in social actors. Technology itself does not create trust.

2. **Software systems reflect social trust**. Trust between social actors is reflected in the design of technical systems. Specifically software systems, due to their flexibility, tend to closely reflect social relationship of trust, either in their design or in particulars of their operation.

3. **Design impacts on trust**. It is possible to alter social trust by the design of technical systems, including online systems. Appropriate design can affect the assessment of trust, can impact on the development of trust, and can damage existing trust. Design can also leave certain specific trust-related decisions for the installation or operation. In such a case, design constraints future choices regarding trust.

4. **Trust can be captured, modelled and reasoned about**. Despite its elusive and subjective nature, trust can be processed in a way that provides a sufficient degree of objectivity and automation. While such simplified 'trust' may not capture the compete richness, it is good enough for the vast majority of practical considerations.

5. **Trust is ever-evolving**. The understanding and practice of trust is not fixed, but it continuously evolves. Technology accelerates this process, rendering certain design decisions obsolete. The design process should evolve as well, capturing and disseminating best practices and advances in knowledge.

4.6 Reference Disciplines

'Designing for trust' is an inter-disciplinary effort that is heavily inter-linked with a list of reference disciplines, providing the formal recognition of the contributing body of knowledge and the logical link. [26]. In the case of *'designing for trust*', we can see that it draws from and contributes to at least the following established disciplines:

Software engineering. As software is the key element of modern information and communication technologies, the new discipline directly draws from several of inspirations and methodologies of software engineering, specifically when it comes to requirement analysis, modelling and design of software systems.

Social sciences. The new discipline inherits from social sciences not only the general understanding of trust as a social phenomenon, but also several methodologies to assess, discuss and negotiate trust.

Information security. 'Designing for trust' draws from the rich experience of this relatively new discipline, noting close relationship between some of goals, models and methods set by security (that cannot be satisfied by security alone) and its owns.

Psychology of mind. The new discipline investigates the human understanding of trust, as applied to interaction through and with technology. Theories, methodologies and findings delivered by the psychology of mind are of great importance here.

Human-computer interaction. While the new discipline postulates to look beyond the 'user interface', it acknowledges the body of knowledge that is delivered by studies of human-computer.

Formal logic. As 'designing for trust' seeks ways to reason about trust in complex software systems, it turns to the discipline of formal logic to identify proper formalisms, reasoning methods and systems that will satisfy such a need.

'Designing for trust' not only draws form the existing body of knowledge of other disciplines, but enriches them with its own contribution in a form of a pragmatic (yet founded in theories) methodology of capturing, reasoning about and embedding trust into the design of information systems.

4.7 Principles and Practices

Principles incorporate the world view and define the philosophical approach to solving problems. Practices are the methodologies, models, procedures, and theories used to apply the discipline’s knowledge base. In engineering, the body of abstract knowledge is developed by logical analysis and scientific research. The principles and practices of engineering are embodied in systems of theory, abstraction, design, and implementation. It is the activities which occur inside these systems that differentiates the many engineering disciplines [26].

Main principles of 'designing for trust' have been already discussed above. Following is a non-exhaustive list of practices that the discipline promotes:

Continuous engagement. It is essential to recognise the fundamental role of trust throughout all the stages of lifecycle of a software system. This, in turn, calls for a continuous engagement that should start together with a feasibility study and end after the system has been decommissioned. Frameworks, tools, methods and knowledge should be portable between different stages of a system lifecycle, so that e.g. important insights can be carried throughout the whole lifecycle

Socio-technical fusion. While focusing on the technical system, it is essential to view it intrinsically embedded in its social context. That context then becomes an inseparable part of its design, to the extent that some of the design goals can be achieved only by the fusion of technical and social methods.

Constrained intra-subjectivity. The subjective nature of trust should be seen through the lens of human ability to negotiate meanings in the intra-subjective manner, thus allowing to globally reason about trust, in a manner that approximates objectivity.

Relevance and quality. In recognition that the perfection may not be reachable or even desired, the principle of 'good enough' for the given purpose is proposed, combined with the ability to actually assure the quality.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 3 / DECEMBER 2010 / 65-77
© 2010 Universidad de Talca - Chile

This paper is available online at
www.jtaer.com
DOI:10.4067/S0718-18762010000300007

4.8 Active Research Agenda

An active research agenda implies that hypotheses are being generated which address the fundamental question of the field of study. The agenda stands the test of time, with many researchers and practitioners in the discipline continually expanding the research that builds upon itself [26]. For 'designing for trust' we can see that challenges currently focus around the following areas:

Conceptual foundations of the discipline. Those include basic definitions, reference models, agreed terminology and methods. In case of trust, with its multiplicity of definitions, the lack of shared conceptual foundations is seen as a major obstacle for the discipline, and it has to be addressed in the first place.

Measurement and reasoning. Agreed methods to measure and reason about trust are necessary to achieve large scale adoption. While one cannot expect here the same level of consensus that is necessary for the conceptual foundation, at least the agreement regarding conceptual foundations of measurement and reasoning is essential.

Expressiveness and tools. It is understood that without a shared tools, and shared means of expression, three communities (designers, operators, users) will not be able to communicate effectively. Modelling and meta-modelling is a continuous important activity.

Socio-technical fusion. The necessity of interdisciplinary approach requires the deep harmonisation of methodological frameworks, combined with mutual acceptance and understanding of results coming from different disciplines.

Monitoring and metrics. Finally, results delivered by applying the discipline should be gauged in a manner that allows agreement on progress, determination of weak areas of the discipline, and guidance towards its adoption.

4.9 Education and Professionalism

Education and professionalism are essential to the widespread recognition and adoption of a field of study. A field of study should be identifiable with a research community that sustains its own literature. The written record of knowledge and thought progression is valuable for future researchers and practitioners to reference when developing new theories and methodologies. Conferences and journals provide a forum for researchers and practitioners to exchange ideas, develop new knowledge and identify future lines of research. Separate curricula, professional societies, and journals advance professionalism and are necessary [28].

Focusing on trust, a number of events are organised every year that attract a substantial number of attendees. However, most of these events are domain specific and therefore they do not always achieve an interdisciplinary (or even multi-domain) audience. They also tend to be either research oriented or highly pragmatic, mostly from the area related to information security.

Therefore, we can see the need to create a separate thread of interdisciplinary study in 'designing for trust', with its own curriculum, certifications and professional examinations. Such steps will eventually lead the discipline towards its maturity.

5 Conclusions and Future Work

In this paper, we have introduced the need to develop a field of study to improve software systems quality and eliminate cultural issues related to the inclusion of trust considerations into the development of software systems. We have presented the foundations of such a field of study by defining the challenge and then by looking at different characteristics and introducing principles and practices, illustrated by an example.

Our proposal combines experience from relevant projects and from research both from the academic and industrial world in areas such as telecommunications, health care and education. However, this is not an absolute attempt and the paper aims to motivate a large scale effort towards the development of the field of study, which will hopefully result into a more complete and detailed definition.

References

[1] S. Anderson, Web Services Trust Language, WS-Trust, 2005.
[2] K. K. Bimrah, H. Mouratidis, and D. Preston, Information Systems Development: A Trust Ontology, in 6th International Conference on Ontologies, DataBases, and Applications of Semantics, Algrave, Portugal, Lecture Notes in Computer Science, vol. 4805, Springer-Verlag, 2007.
[3] K. K. Bimrah, A Framework for Modelling Trust During Information Systems Development, PhD Thesis, University of East London, London, March, 1989.

[4] M. Blaze, J. Feigenbaum, and J. Lacy, Decentralized Trust Management, in Proceedings of the IEEE Symposium on Security and Privacy, Oakland, USA, 1996, pp. 164-173.
[5] M. Blaze, K. Feigenbaum, and A. Keromytis, Trust Management for Public-Key Infrastructures, in 6th International Workshop on Security Protocols, UK, Springer, 1998, pp. 625-629.
[6] K. Bohmann, About the Sense of Social Compatibility, AI and Society, vol. 3, no. 4, 1989, pp. 323-331.
[7] G. Borchers, The software engineering impacts of cultural factors on multi-cultural software development teams, In Proceedings of the 25th international Conference on Software Engineering, Portland, Oregon, 2003, pp. 03-10, International Conference on Software Engineering. IEEE Computer Society, Washington, DC, 2003, pp. 540-545.
[8] E. V. Chepaitis, Soft barriers to ICT application in development: trust and information quality in Russia, 2002.
[9] E. Chang, T. Dillion, F. K. Hussain, Trust and Reputation for Service-Oriented Environments: Technologies for Building Business Intelligence and Consumer Confidence. John Wiley & Sons, Ltd, 2006.
[10] K. Chopra, W. A. Wallace, Trust in Electronic Environments, in Proceedings of the 36th Hawaii Conference on System Sciences, Hawaii, 2003, pp. 331.
[11] Y. Chu, REFEREE: Trust Management for Web Applications, Journal of World Wide Web, vol. 2, no. 3, pp. 127-139, 1997.
[12] P. Cofta, Trust, Complexity and Control: Confidence in a Convergent World. John Wiley and Sons, 2007.
[13] D. C. Dennett, The Intentional Stance. Bradford Books, 1989.
[14] M. O. Doheny, C. Cook, M. Stopper, The Discipline of Nursing: an introduction, 2nd edition, Appleton & Lange, Norwalk, Connecticut, 1987.
[15] N. Dwyer, and P. Cofta, Understanding the grounds to trust: game as a cultural probe, presented at Web 2.0 Trust (W2Trust), Trondheim, Norway in conjunction with IFIPTM, 2008.
[16] F. N. Egger, From Interactions to Transactions: Designing the Trust Experience for Business-to-Consumer Electronic Commerce, PhD Thesis, Eindhoven University of Technology, Netherlands, 2000.
[17] P. Giorgini, Requirements Engineering Meets Trust Management, in Proceedings of the Second International iTrust Conference, Oxford, UK, Springer, 2004, pp. 176-190.
[18] J. Golbeck, Computing with Social Trust. Springer, 2008
[19] J. Gorski, Trust Case: Justifying Trust in an IT Solution, Journal of Reliability Engineering and System Safety, Elsevier, vol. 89, no. 1, pp. 33-47, 2001.
[20] T. Grandison, Trust Management for Internet Applications, PhD Thesis, University of London, London, July 2001.
[21] G. Hofstede, Culture's Consequences, Comparing Values, Behaviors, Institutions, and Organizations Across Nations. Thousand Oaks CA: Sage Publications, 2001.
[22] A. Josang, C. Keser, and T. Dimitrakos, Can we Manage Trust?, in Proceedings of the 3rd International Conference of Trust Management (iTrust), Paris, France, May 2005, pp. 93-107.
[23] A. Josang, S. Marsh, and S. Pope, Exploring Different Types of Trust Propagation, in Proceedings 4th International Conference on Trust Management, Springer-Verlag, 2006, pp. 179-192.
[24] P. Keen, MIS Research: Reference Disciplines and Cumulative Tradition, in Proceedings of the First International Conference on Information Systems, Philadelphia, Pennsylvania, December, 1980, pp. 9-18.
[25] H. Lacohée, P. Cofta, A. Phippen, and S. Furnell, Understanding Public Perceptions: Trust and Engagement in ICT Mediated Services. International Engineering Consortium, 2008.
[26] D. H. Liles, M. E. Johnson, L. M. Meade, D. R. Underdown, Enterprise Engineering: A discipline?, in Proceedings of the Society for Enterprise Engineering Conference, Kettering, Ohio, June, 1995.
[27] R. C. Mayer, J. H., Davis, F. D. Schoorman, An integrative model of organizational trust, Academy of Management Review, vol. 20, no. 3, pp. 709-734, 1995.
[28] H. B. Maynard, H. B, Industrial Engineering Handbook, 3rd Edition, McGraw Hill Book Co., New York, 1971.
[29] D. H. McKnight, N. L. Chervany, The Meanings of Trust, Carlson School of Management, University of Minnesota, 1996.
[30] J. B. Michael, D. R. Hestad, C. M. Pedersen, L. T. Gaines, Incorporating the Human Element of Trust into Information Systems. IAnewsletter, vol. 5, pp. 4-8, 2002.
[31] H. Mouratidis and P. Giorgini, Secure Tropos: A Security-Oriented Extension of the Tropos methodology, International Journal of Software Engineering and Knowledge Engineering (IJSEKE), World Scientific, vol. 17, no. 2, pp. 285-309, 2007.
[32] H. Mouratidis, Secure Information Systems Engineering: A Manifesto, International Journal of Electronic Security and Digital Forensics, Inderscience Publishers, vol.1, no. 1, pp. 27-41, 2007.
[33] C. Mundie, P. de Vries, P. Haynes, and M. Corwine, Trustworthy Computing. Microsoft White Paper, 2002
[34] S. Pearson, Trusted Computing Platforms: TCPA Technology In Context. Prentice-Hall, 2002.
[35] A. Pourshahid, and T. Tran, Modelling Trust in E-Commerce: An Approach Based on User Requirement, Proceedings of the 9th International Conference on Electronic Commerce, Minneapolis, Minnesota, USA, August, 2007, pp. 413-422.
[36] S. L. Presti, Holistic Trust Design of E-Services. In*:* Trust in E-services: Technologies, Practices and Challenges, London: Idea Group Publishing, 2006, pp. 113-139.
[37] P. Resnick, The value of reputation on eBay: a controlled experiment, Experimental Economics, vol. 9, no. 2, Jun 2006, pp. 79-101.
[38] A. Sutcliffe, Trust: From Cognition to Conceptual Models and Design, presented at the 18th International Conference, CAiSE 2006, Luxembourg, Springer-Verlag Berlin Heidelberg, June 5-9, 2006.
[39] Y. H. Tan, A Trust Matrix Model for Electronic Commerce, Proceedings of the First International iTrust Conference, Crete, Greece, Springer, 2003, pp. 33-45.

This paper is available online at www.jtaer.com
DOI:10.4067/S0718-18762010000300007

[40] A. Tsohou, M. Theoharidou, S. Kokolakis, and D. Gritzalis, Addressing Cultural Dissimilarity in the Information Security Management Outsourcing Relationship, Lecture Notes in Computer Science, vol. 4657, pp. 24-33, 2007.
[41] M. G. Uddin and M. Zulkernine, UMLtrust: Towards Developing Trust-Aware Software, Proceedings of the 2008 ACM Symposium on Applied Computing, Fortaleza, Ceara, Brazil, March, 2008, pp. 831-836.
[42] Z. Yan and P. Cofta, Methodology to Bridge Different Domains of Trust in Mobile Communications, The First International Conference on Trust Management (iTrust'03), LNCS, vol. 2692/2003, Greece, 2003, pp. 211-224.
[43] E. Yu, L. Liu, Modelling Trust for System Design Using the i* Strategic Actors Framework, in Proceedings of the workshop on Deception, Fraud, and Trust in Agent Societies held during the Autonomous Agents Conference: Trust in Cyber-societies, Integrating the Human and Artificial Perspectives, 2001, pp. 175-194.