Extending Software Development Methodologies to Support Trustworthiness-by-Design

Nazila Gol Mohammadi[1], Torsten Bandyszak[1], Sachar Paulus[2], Per Håkon Meland[3], Thorsten Weyer[1] and Klaus Pohl[1]

[1]paluno – The Ruhr Institute for Software Technology, University of Duisburg-Essen, 45127 Essen, Germany
{nazila.golmohammadi, torsten.bandyszak, thorsten.weyer, klaus.pohl}@paluno.uni-due.de
[2]Mannheim University of Applied Sciences, Paul-Wittsack-Straße 10, 68163 Mannheim, Germany
s.paulus@hs-mannheim.de
[3]SINTEF ICT, Strindveien 4, N-7465 Trondheim, Norway
per.h.meland@sintef.no

Abstract. People are increasingly concerned about the trustworthiness of software that they use when acting within socio-technical systems. Ideally, software development projects have to address trustworthiness requirements from the very early stages of development using constructive methods to enable trustworthiness-by-design. We analyze the development methodologies with respect to their capabilities for supporting the development of trustworthy software. Our analysis reveals that well-established development methodologies do not specifically support the realization of trustworthy software. Based on findings, we propose a generic mechanism for extending development methodologies by incorporating process chunks that represent best practices and explicitly address the systematical design of trustworthy software. We demonstrate the application of our approach by extending a design methodology to foster the development of trustworthy software for socio-technical systems.

Keywords: Trustworthiness, Trustworthiness-by-design, Software Development Methodology

1 Introduction

Trustworthiness is a major issue for the development of software-intensive socio-technical systems [1, 2]. For instance, for the users of today's web applications and services it becomes increasingly difficult to track or control who stores personal and business-critical data. Thus, software-intensive systems need to be trustworthy to address concerns of their users and thereby foster the trust in these systems. Understanding how to address trustworthiness in early design phases is crucial for the successful development of software systems. Software development methodologies and processes should address the different challenges of engineering trustworthy software.

Trustworthiness is an important quality that needs to be engineered. There is a strong dependency between the degree of trustworthiness an information system exhibits and the suitability of the applied development methodology [3].

There are limited contributions that approach the trustworthiness issues other than those related to security. Most existing approaches assume that one-dimensional properties of services lead to trustworthiness of such services, and even to trust in it by users, such as a certification (e.g., Common Criteria [4]), the presence of certain technologies (e.g., encryption), or the use of certain methodologies (e.g., SSE-CMM [5]). In contrast, the Trusted Software Methodology [3] as a comprehensive and holistic methodology that explicitly focuses on trustworthiness is not flexible, since it is based on a certain development process. Though in principle the application of any development process model may result in trustworthy products, commonly used and well-established methodologies, such as user-centered [6] or test-driven development [7], do not specifically foster the systematic establishment of trustworthiness properties within the system. In order to address this gap, we believe that specific techniques and developer guidance should be defined as generic and reusable process building blocks. Defining reusable process chunks that can be integrated into well-established development methodologies instead of defining yet another development methodology brings flexibility, enables a powerful process modeling tool support, and reduces the complexity of tailoring already established development processes.

First, we review and analyze well-established software development methodologies by studying their characteristics that are promising to build trustworthy information systems, and the ones indicating improvement potential. In this paper, we build upon an outline of our approach sketched in [8], and provide a generic mechanism for enhancing software development by incorporating process chunks that explicitly address and enable trustworthiness-by-design. In particular, we propose an extension of the Software Process Engineering Meta-model (SPEM) [9], which allows for integrating and tailoring certain “trustworthy” process chunks into different development methodologies. These capability patterns can represent a broad range of trustworthiness-related practices, such as the preparation for run-time maintenance [10]. As an example, we analyzed the User-Centered Design (UCD) methodology [6] with respect to trustworthiness potentials and drawbacks as an example. Based on these findings, we demonstrate our approach by exemplarily extending the UCD process model.

The remainder of this paper is structured as follows: Section 2 provides a brief overview on the fundamental notions of trust and trustworthiness of STS. Section 3 presents our approach for extending development methodologies by trustworthiness-by-design capabilities and illustrates its application by showing how a popular engineering methodology can be extended to support trustworthiness-by-design. Section 4 summarizes the paper and gives an outlook on future work.

2 Fundamentals and Related work

Trust is defined as “a bet about the future contingent actions of others” [11]. Regarding software-intensive socio-technical systems (STS), which include humans, organizations, and the information systems [7], the scope of this definition can be broadened in order to include these systems as potential trustees. Because of delegation of tasks to STS, it can be said that the trustworthiness of such systems is a key concern that needs to be fostered and even engineered into these systems to maintain high levels of trust within society. Trustworthiness requirements are project-specific, and depend on domain and application. Software trustworthiness is highly dependent on the prescribed, yet evolving, set of requirements, technical decisions, and management decisions throughout the development process life cycle. A comprehensive list of trustworthiness attributes (e.g., correctness, reliability, safety, usability, security) should be taken into consideration when developing trustworthy software [2]. Hence, we focus on a multitude of software quality attributes that contribute to trustworthiness as analyzed in [2]. For example, trustworthiness may be evaluated with respect to the availability, confidentiality and integrity of stored information, the response time, or accuracy of outputs [12, 13, 14].

To the best of our knowledge, the Trusted Software Methodology (TSM) [3] is the only comprehensive approach that describes processes and guidance for engineering and assessing trustworthy software. It covers multiple quality attributes, and focuses on processes instead of evaluating development artifacts. TSM provides a set of Trust Principles, which describe established development practices or process characteristics that enhance software trustworthiness. A development process can be assessed by means of five different levels of trustworthiness, according to the conformance to the trust principles. This also constitutes the basis for process improvement with respect to trustworthiness. Though the principles constitute general best practices, the methodology, however, is assumed to be applied following a military standard for software development [15]. In contrast, our focus is on enhancing a broad spectrum of general software development methodologies in order to incorporate the consideration of trustworthiness and use them to create trustworthy software.

Yang et al. [3] review a set of software development methodologies in order to derive a meta-model for trustworthy development processes. They define process trustworthiness as “the degree of confidence that the software process produces expected trustworthy work products that satisfy their requirements” [3]. The meta-model includes, for example, trustworthy products that depend on a trustworthy process. It also depicts the connection to trustworthiness requirements. For modeling process trustworthiness, they adopt the Process Area concept from CMMI [16] and extend it by the Trust Principles, then constituting Trustworthy Process Areas (TPAs) [3]. The TPAs, in turn, can be refined by three categories, i.e. regarding trustworthiness assurance, trustworthiness monitoring, and trustworthiness engineering process areas. Thus, the approach covers the whole system life-cycle. Yang et al. also present their efforts towards designing a comprehensive Trustworthy Process Management Framework, which e.g., additionally involves a measurement model based on metrics [3].

In contrast, our approach relies on the SPEM [7], which provides a meta-model for describing software development processes. In our approach, we will use the Delivery Process and Capability Pattern concepts from SPEM. Capability patterns are process building blocks that are independent of specific process phases, and represent best development practices to be incorporated into a process [7]. The Delivery Process and Capability Pattern concepts originate from the SPEM. SPEM provides adequate concepts that allow for describing capability patterns on a fine-granular level, i.e. assigning concrete tasks, responsible roles, guidance, or involved artifacts.

The concepts introduced can be compared to the work of Yang et al. [3]. However, we propose a different structure and different concepts, e.g., using SPEM capability patterns instead of CMMI process areas (cf. [16]), or combining design and assessment in one meta-model.

3 Integrating Trustworthiness-by-Design in Development Methodologies

Characteristics of Trustworthiness-by-Design Processes. In order to incorporate the notion of trustworthiness-by-design into development methodologies, we consider and extend the SPEM meta-model [7] by specializing the *Delivery Process* concept so that it subsumes trustworthiness-by-design processes. We also utilize the concept of *Capability Patterns*. We define a *Trustworthy Product* (i.e. work product, development artifact) as a product that holds a range of its trustworthiness attributes for satisfying its trustworthiness requirements. Fig. 1 shows a corresponding ontology for Trustworthiness-by-Design Processes. The meta-model presented here shows the concepts that we have introduced in addition to SPEM (highlighted in grey in Fig. 1), specifically: *Trustworthiness-by-design Process* is a specialization of a delivery process and contains a set of capability patterns. A properly applied trustworthiness-by-design process will create a *Trustworthy Product* that exhibits certain trustworthiness attributes to meet its *Trustworthiness Requirements*. Trustworthiness requirements specify requirements that a Trustworthy Product should fulfill. *Assessment Model* verifies if the trustworthiness requirements have been met. Metrics could be used to evaluate the products. *Trustworthiness Evidence* is some kind of evidence to show that a trustworthiness-by-design process has been followed. Though this will not guarantee trustworthiness, it is at least an indication that planned measures have been taken into account to ensure it.
We define capability patterns that particularly address trustworthiness to improve existing design process models. For describing capability patterns, we provide the necessary content, e.g., concrete tasks, responsible roles, guidance, and involved artifacts [7].

An exemplary capability pattern for trustworthiness-by-design is the identification of threats and mitigating controls. This capability pattern involves analyzing system models or specifications in order to anticipate risks that might corrupt the system's trustworthiness across the whole life-cycle (e.g., also considering system operation).
To provide tool support for designing, tailoring and sharing trustworthy development processes, we use the Eclipse Process Framework (EPF[1]), which has an underlying

[1] Eclipse Process Framework Project (EPF), `http://www.eclipse.org/epf/`

meta-model based on SPEM. The EPF is a customizable software process-engineering framework for authoring, tailoring, and deploying development processes. All our capability patterns are organized in a plug-in that can be imported into any EPF project, which again can be exported to online process handbooks.

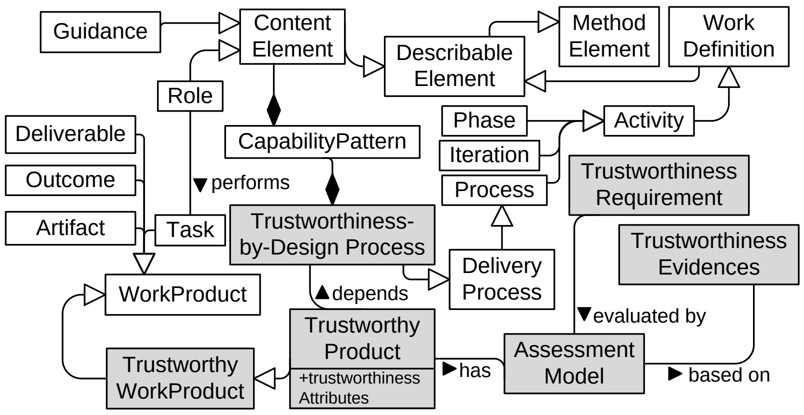


Fig. 1. Ontology for Trustworthiness-by-Design Processes

Extending the User-centered Design Methodology. The nature of engineering of trustworthy systems is different from simply engineering usable software. The key here is that trustworthiness is a subjective value judgment of stakeholders in a STS. There is a need to understand what trustworthiness attributes of the system will enhance the trust of a stakeholder in that system and how system design can thus help to circumvent any distrust-related concerns that the stakeholders have about the service. This makes it necessary to not only elicit requirements with respect to the way in which people will use the system, as would be done in a standard UCD [6] process, but also to draw up a set of requirements about which trustworthiness attributes will address the potential trust issues that the end users of the system highlight.

In order to assess the product with respect to the satisfaction of trustworthiness requirements the overall structure of the UCD approach can remain the same, with the only difference that in the process, besides usability and usefulness, trust and trustworthiness needs are specifically addressed.

We suggest the following extensions of the four major phases of the UCD methodology:

- In the initial *specify context of use* phase (Phase 1 in Fig. 2), a usability expert elicits from the future end users what the potential trust concerns are that they have with respect to using the system.
- In the specify user requirements phase (Phase 2 in Fig. 2), these concerns can then be turned into use case descriptions of situations in which the trust issues become apparent to the user. To this end, the Trustworthiness Capability Pattern “Identification of threats and mitigation controls” should be incorporated into UCD. By means of the involved analysis tools, threats to trustworthiness can be derived. It

should also be determined which controls can be applied in the design to mitigate the identified trustworthiness and trust issues.

- The produce design solutions phase (Phase 3 in Fig. 2) should then implement (e.g., in a prototype) the identified trustworthiness requirements.
- The "Measurement of end-to-end trustworthiness" capability pattern can enhance the *evaluation against requirements* phase by providing appropriate metrics and measurement approaches to validate that the system satisfied the required trustworthiness level (this can enhance Phase 4 in Fig. 2).

Name: User-Centered Design Process
Description: User-centered design processes [6] consist of the following general phases:

1) Knowledge elicitation and attempt at understanding the context of use;
2) Defining user requirements;
3) Prototyping the system and
4) Evaluation, which provides input for the refinement of the design.

This process model is generally used iteratively and by going through the process multiple times, developers converge on a user-friendly system.

Elements interesting for trustworthiness:

- User-centered design is a specialization of incremental development and therefore shares the same trustworthiness characteristics.
- By using an incremental user-centered process, it is possible that throughout the design process the design is validated to establish whether the trustworthiness attributes designed into the system appropriately address any concerns with respect to trust that the system users might have.

Improvement potential:

- Documenting trustworthiness requirements and thereafter generation of trustworthiness evaluation results for explicit documentation of trustworthiness evidences in order to support designers when making design decisions. Additionally, these documents bring awareness about the designed system to the end-users.
- Involvement of end-user to derive their trustworthiness expectations and to evaluate the system design towards the satisfaction of those expectations.

Usability for modeling trustworthiness: The user centered design processes are unrelated to trustworthiness modeling. Only the use of modeling techniques in general for a user centered design will enable to also model trustworthiness requirements.

Fig. 2. Indicative trustworthiness analysis for User-Centered Design

Fig. 3 illustrates the extension of UCD by plugging one of the proposed capability patterns namely "Identify Threats and Controls" in early stage of specifying user requirements.

As we have already shown above, the extension of the UCD process can be supported by using the EPF Composer. Fig. 4 shows an excerpt from the description of the extended UCD methodology as part of a software development process model (i.e. delivery process as defined by SPEM).

As the excerpt sketches, the corresponding "*Trustworthy User Centered Design*" methodology integrates the two additional capability patterns "Identification of threats and mitigation controls" (Fig. 4, Index no. 6) and "Measurement of end-to-end trustworthiness" (Fig. 4, Index no. 13) into specific phases of the original user-centered design process model.

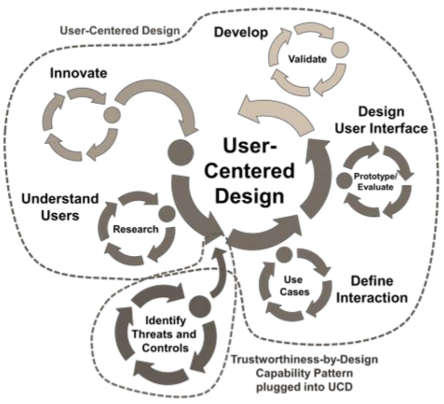
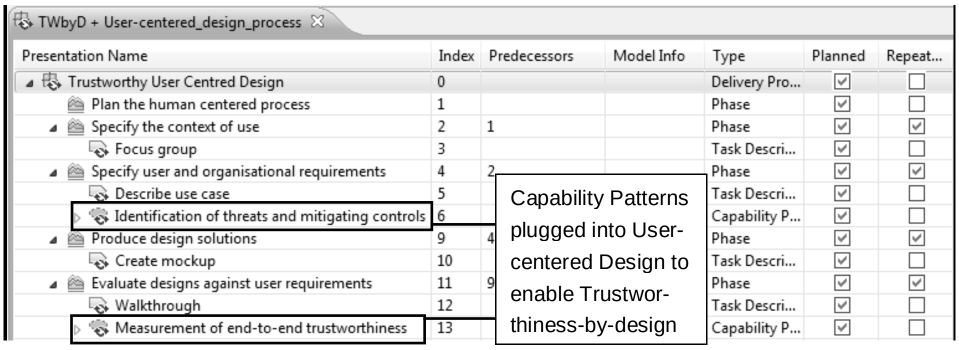


Fig. 3. Extending the User-Centered Design[2] for enabling Trustworthiness-by-Design [8]


TWbyD + User-centered_design_process

Presentation Name	Index	Predecessors	Model Info	Type	Planned	Repeat...
Trustworthy User Centred Design	0			Delivery Pro...	✓	☐
Plan the human centered process	1			Phase	✓	☐
Specify the context of use	2	1		Phase	✓	✓
Focus group	3			Task Descri...	✓	☐
Specify user and organisational requirements	4	2		Phase	✓	✓
Describe use case	5			Task Descri...	✓	☐
Identification of threats and mitigating controls	6			Capability P...	✓	☐
Produce design solutions	9	4		Phase	✓	✓
Create mockup	10			Task Descri...	✓	☐
Evaluate designs against user requirements	11	9		Phase	✓	✓
Walkthrough	12			Task Descri...	✓	☐
Measurement of end-to-end trustworthiness	13			Capability P...	✓	☐

Capability Patterns plugged into User-centered Design to enable Trustworthiness-by-design

Fig. 4. Enhancing the UCD methodology with trustworthiness capability patterns

4 Conclusion and Future Work

Existing software design methodologies have some capacities in ensuring security and a few other trustworthiness attributes. However, the treatment of a complete set trustworthiness attributes and requirements in software development is not yet well studied. We analyzed development methodologies for trustworthy development.

As a result, we concluded that none of them fully assures or addresses the development of trustworthy software. Consequently, individual activities, so-called "trustworthy development practices", must be identified and tailored into these processes in order to proceed towards systematically developing trustworthy software. The concept and an initial set of reusable, trustworthiness-enhancing process chunks in the form of Capability Patterns have been introduced. We have observed that the usage of appropriate trustworthiness capability patterns increases the confidence that the software development processes will result in trustworthy software.

Our work is still in progress, and the main ideas and findings will be further investigated. Further work is needed to evaluate the recommended extensions to these

[2] Based on http://www.sapdesignguild.org/editions/edition10/ucd_overview.asp

methodologies, how to combine capability patterns and investigate how trustworthiness attributes can be treated in a measurable and comparable way.

References

1. Whitworth, B.: A Brief Introduction to Socio-technical Systems. In: Encyclopedia of Information Science and Technology, pp. 394–400. IGI Global (2009)
2. Gol Mohammadi, N.; Paulus, S.; Bishr, M.; Metzger, A.; Könnecke, H.; Hartenstein, S.; Weyer, T.; Pohl, K.: Trustworthiness Attributes and Metrics for Engineering Trusted Internet-based Software Systems. In: Cloud Computing and Service Science 2013 (Selected Papers from CLOSER), CCIS, Springer (2013)
3. Yang, Y., Wang, Q., Li, M.: Process Trustworthiness as a Capability Indicator for Measuring and Improving Software Trustworthiness. In: Trustworthy Software Development Processes, Int'l. Conf. on Software Process. LNCS, vol. 5543, pp. 389-401. Springer (2009)
4. International Organization for Standardization: ISO 15408-1, Common Criteria, Information technology -- Security techniques -- Evaluation criteria for IT security. International Standard (2009)
5. International Organization for Standardization: ISO/IEC 21827, Information technology, Security techniques, Systems Security Engineering -- Capability Maturity Model® (SSE-CMM®). International Standard (2008)
6. Sutcliffe, A. G.: Convergence or Competition between Software Engineering and Human Computer Interaction. In: Human-Centered Software Eng. - Integrating Usability in the Software Development Lifecycle, Human-Computer Inter. Series, vol. 8, pp. 71-84 (2005)
7. Sommerville, I., Software Engineering. 9th Edition, Pearson, Boston (2011)
8. Gol Mohammadi, N.; Bandyszak, T.; Paulus, S.; Håkon Meland, P.; Weyer, T.; Pohl, K.: Extending Development Methodologies with Trustworthiness-By-Design for Socio-Technical Systems, In: Proceedings 7th Int'l. Conf. TRUST (2014)
9. Object Management Group: Software & Systems Process Engineering Meta-Model Specification, Version 2.0. Technical Report, Object Management Group (2008)
10. Bandyszak, T.; Gol Mohammadi, N.; Bishr, B.; Goldsteen, A.; Moffie, M.; Nasser, B. I.; Hartenstein, S.; Meichanetzoglou, S.: Cyber-Physical Systems Design for Runtime Trustworthiness Maintenance Supported by Tools. In: 1st Int'l. Workshop on Requirements Engineering for Self-Adaptive systems and Cyber Physical Systems (2015)
11. Sztompka, P.: Trust: A Sociological Theory. Cambridge University Press (1999)
12. Avizienis, A., Laprie, J. C., Randell, B., Landwehr, C.: Basic Concepts and Taxonomy of Dependable and Secure Computing. In: IEEE Transactions on Dependable and Secure Computing, vol. 1 issue 1, 11–33 (2004)
13. Gómez, M., Carbó, J., Benac-Earle, C.: An Anticipatory Trust Model for Open Distributed Systems. In: Anticipatory Behavior in Adaptive Learning Systems. LNCS, vol. 4520, pp. 307–324. Springer (2007)
14. Yolum, P., and Singh, M. P.: Engineering Self-Organizing Referral Networks for Trustworthy Service Selection. In: IEEE Trans. on Systems, Man and Cybernetics, Part A: Systems and Humans vol. 35 no. 3, 396–407 (2005)
15. U.S. Department of Defense: Trusted Software Methodology, SDI-SD-91-000007, Volumes 1 and 2. Technical Report, U.S. Department of Defense, Strategic Defense Initiative Organization (1992)
16. Software Engineering Institute: Capability Maturity Model® Integration for Software Engineering, Version 1.1. Technical Report, Software Engineering Institute, Carnegie Mellon University (2002)