TECHNICAL TRANSACTIONS 1/2019
MECHANICS
DOI: 10.4467/2353737XCT.19.016.10056
SUBMISSION OF THE FINAL VERSION: 21/12/2018

Małgorzata Stojek orcid.org/0000-0002-6037-195X
mstojek@pk.edu.pl
Faculty of Civil Engineering, Cracow University of Technology

Engineering safety- and security-related quality requirements

Inżynierskie wymagania jakościowe związane z bezpieczeństwem i ochroną

Abstract

Safety- and security-related problems for software intensive systems are often due to poor or missing relevant engineering requirements. Although there is nothing really new in our presentation, specified problems and methods are well worth revisiting because they are still far from being widely recognized and put into industrial practice.

Keywords: safety, security, software-intensive systems

Streszczenie

W przypadku systemów intensywnie korzystających z oprogramowania problemy związane z bezpieczeństwem i ochroną są często wynikiem złych lub brakujących odpowiednich wymagań technicznych. Chociaż w niniejszym artykule nie ma poruszanych nowych zagadnień, określone problemy i metody są warte ponownego przeanalizowania, ponieważ wciąż są dalekie od powszechnego uznania i przeniesienia ich do praktyki przemysłowej.

Słowa kluczowe: bezpieczeństwo, ochrona, systemy intensywnie wykorzystujące oprogramowanie

1. Introduction

Nuseibeh and Easterbrook [1] advocate for the following definition of the requirements engineering (RE) previously proposed by Zave [2]: "Requirements engineering is the branch of software engineering concerned with the real world goals for, functions of, and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families".

Such broad understanding of RE obviously encompasses safety and security related problems. Despite the fact that Boehm and Papaccio recognized in 1988 that "reworking a software requirements problem once the software is in operation typically costs 50 to 200 times what it would take to rework the problem in the requirements stage" [3], it still remains the fact that requirements engineers often fail to pay sufficient attention to security concerns [4]. In addition, given the fact that software-related accidents are almost all caused by flawed requirements, it is somehow surprising that safety concerns of stakeholders could be mostly restricted to comply with basic standards or legal requirements.

In what follows, we revisit requirements issues related to safety and security engineering [5, 6] and discuss their quality aspects for the development and maintenance of cost effective and secure software dependent systems.

2. Safety and security as quality factors

2.1. Safety and security related definitions

Traditionally, safety, security and requirements engineering have been separate research fields with different backgrounds, journals, and conferences. Safety engineering ensures that a system is sufficiently safe to operate (i.e. free from accidental harm to someone or something); security deals with privacy, authentication, and integrity as immune to their intentional breach. Both have common limitations since they usually address rather prevention of the harm than its detection and proper reaction to its occurrences. Nowadays, safety/security engineering is defined as "the engineering discipline within systems engineering concerned with lowering the risk of unintentional/intentional unauthorized harm to valuable assets to a level that is acceptable to the system's stakeholders by preventing, detecting, and reacting to accidental/malicious harm, mishaps/misuses (i.e., accidents/attacks and incidents), hazards/threats, and safety/security risks" [6].

Let us now see RE as "the discipline consisting of the cohesive collection of all tasks that are primarily performed to produce the requirements and other related requirements work products for an endeavor" [6]. Seeing that safety and security related requirements are included, common collaborative methods are to be established and implemented. Taking into account different cultural engineering backgrounds, such cooperation is with no doubts challenging and requires the right staffing and project management.

2.2. Quality characteristics of requirements

When one thinks about standard requirements engineering tasks, they are usually undertaken after a thorough business analysis and vision statement including capabilities and goals (not requirements yet). A software requirements specification (SRS) is modeled after business (stakeholders) requirements specification (Concept of Operations, CONOPS) by means of the following tasks: requirements identification, analysis, specification, management and validation. The major difficulty is known [7]: "The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is too difficult as establishing the detailed technical requirements, including all the interfaces to people, to machines, and to other software systems. No other part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later".

It is important to make a distinction between goals and operational cases driven by them vs product requirements. The former implies functional requirements while the latter also includes non-functional requirements that are not necessarily driven by operational functionality of the engineering system. The non-functional product requirements cover quality, data, interface requirements as well as system constraints. All requirements are documented and stored in relevant repositories and should have the following characteristics: cohesiveness, completeness, consistency, correctness, currency, customer/user orientation, external observability, feasibility, lack of ambiguity, mandatory, metadata, relevance, usability, validatability, verifiability [8].

By quality requirement one understands a specification of a mandatory amount of a type of product quality. Thus, safety and security requirements fall into minimum acceptable quality consideration. To avoid ambiguity attached to the term 'quality', the quality model defining its meaning must be introduced [9]. A quality model consists of quality characteristics (defining the meaning of a specific type of a quality of a system or its part), quality attributes and quality measurements (scales and methods) [10], where safety and security considerations are parts of external quality characteristics related to the system defensibility (see [10, p. 35]).

2.3. Safety and security as quality characteristic

Safety/security subclass of defensibility seizes control of the following problems [6]:

- accidental/malicious harm to valuable assets;
- safety/security abuses (mishaps/misuses such as accidents/attacks and safety/security incidents);
- safety abusers (people, systems, and the environment);
- security abusers (attackers and malware – systems, software, and hardware);
- safety/security vulnerabilities;
- safety/security dangers (hazards/threats) including the existence (conditions) of non-malicious/malicious abusers who unintentionally exploit/can exploit system vulnerabilities to accidentally/intentionally harm vulnerable valuable assets;
- safety/security risks

and have safety/security solutions: prevented (eliminated, mitigated, keep acceptable low), detected, reacted to, adapted to. The aforementioned defensibility characteristics lead to problem and solution type defensibility attribute, respectively.

We have four types of safety/security defensibility-related requirements: general safety/security defensibility requirements, defensibility significant requirements with four safety/security assurance levels (SAL), defensibility function/subsystem requirements and defensibility constraints. Safety and security requirements are generally negative ones since they specify what the system shall not cause, enable or allow to occur and/or exist.

Quality requirements, as critically important drivers of the architecture and testing, should be based on a quality model defining the specific types of quality and how their measurements scales. By restricting our attention to safety and security requirements, we have the following models [11]:

- safety models: asset models, accident models, hazard models and safety risk models;
- security models: asset models, attacker models, attack models, threat models, security risk models.

It is of utmost importance to ensure that safety and security requirements are adequately complete at any given point in time. Poor and/or missing quality requirements have their negative consequences. For their list and suggestion how to avoid them see e.g. a very comprehensive paper of Firesmith [12].

3. Conclusion

To be more effective, safety and security engineering should be integrated in the architecture design process at a very early stage as necessary for the completeness of project engineering requirements. Although these problems have been already addressed many times, they are well worth revisiting because the associated industry best practices are still far from being widely put into practice. For the recent integration between requirements engineering and safety analysis, see a systematic literature review in the paper of Vilela et al. [12].

References

[1] Nuseibeh B., Easterbrook S., *Requirements engineering: A roadmap*, [in:] Proc. Conf. ICSE '00 The Future of Software Engineering, ACM New York, 2000, 35–46.

[2] Zave P., *Classification of research efforts in requirements engineering*, "ACM Computing Surveys", Vol. 29(4)/1997, 315–321.

[3] Boehm B., Papaccio P., *Understanding and controlling software costs*, "IEEE Transactions on Software Engineering", Vol. 14(10)/1988, 1462–1477.

[4] Mellado D., Blanco C., Sánchez L., Fernández-Medina E., *A systematic review of security requirements engineering*, "Computer Standards & Interfaces", Vol. 32(4)/2010, 153–165.

[5] Firesmith D., *Common concepts underlying safety, security, and survivability engineering*, Technical Note CMU/SEI-2003-TN-033, Carnegie Mellon University 2003.
[6] Firesmith D., *Engineering safety- and security-related requirements for software-intensive systems*, [in:] Proc. ICSE'10 32nd ACM/IEEE Int. Conf. on Software Engineering, Vol. 2, 2010, 489–490.
[7] Brooks F., *No silver bullet – essence and accident in software engineering*, "IEEE Computer", Vol. 20(4)/1987, 10–19.
[8] Firesmith, D., *Specifying good requirements*, "Journal of Object Technology", Vol. 2(4)/2003, 77–87.
[9] Firesmith, D., *Using quality models to engineer quality requirements*, "Journal of Object Technology", Vol. 2(5)/2003, 67–75.
[10] Firesmith, D., *Engineering safety requirements, safety constraints, and safety-critical requirements*, "Journal of Object Technology", Vol. 3(3)/2004, 27–42.
[11] Firesmith D., *Are your requirements complete?*, "Journal of Object Technology", Vol. 4(1)/2005, 27–43.
[12] Firesmith D., *Common requirements problems, their negative consequences, and industry best practices to help solve them*, "Journal of Object Technology", Vol. 6(1)/2007, 17–33.
[13] Vilela J., Castro J., Martins L., Gorschek T., *Integration between requirements engineering and safety analysis: A systematic literature review*, "The Journal of Systems and Software", Vol. 125/2017, 68–92.