Available online at www.sciencedirect.com

SCIENCE DIRECT®

Reliability Engineering and System Safety 89 (2005) 1–5

RELIABILITY ENGINEERING & SYSTEM SAFETY

www.elsevier.com/locate/ress

Guest Editorial

Safety, reliability and security of industrial computer systems

This special issue consists of the best papers presented at SAFECOMP 2003 [1] held in Edinburgh, Scotland, UK, 23–26 September 2003. Since its establishment in 1979 by the European Workshop on Industrial Computer Systems[1] (EWICS), SAFECOMP, the series of conferences on Computer Safety, Reliability and Security[2], has contributed to progress in the state of the art in dependable applications of computer systems. SAFECOMP provides ample opportunity to exchange insights and experiences in emerging methods and practices across the borders of different disciplines. Previous SAFECOMP proceedings[3] and journal special issues [2–5] have already registered the increasing interest for multi-disciplinary approaches dealing with system dependability. SAFECOMP 2003 further addresses multi-disciplinarity by supporting the collaborations of different scientific communities towards the identification of communication means that may enhance our ability in designing, building, deploying and assessing computer systems.

The recent developments in system dependability emphasize a shift from process-based development towards model-based development from early stage to certification, deployment, evaluation, evolution, and decommissioning in the system life cycle. Moreover, the increasing ubiquity and pervasiveness of computer systems stress how the modern Information Society (IS) relies on the services these systems support. Therefore, systems have further to comply with stringent dependability requirements in order to guarantee and support specific safety-critical as well as normal daily activities. In particular, it is possible to stress a shift from safety-critical systems to services, hence to dependability of services. Services are more difficult to conceptualize than software and hardware components. Moreover, they often rely on emergent non-functional features. Therefore, it is necessary to engage broad multi-disciplinary approaches in order to understand and capture a comprehensive viewpoint of system dependability. Hence, the properties of technical systems have strong connections with quality of service. In respect of these concerns, Martinello et al. in Ref. [6] address analytical models for evaluating the service availability of web cluster architectures. These models support the design of Internet services (e.g. online banking, e-commerce, etc.). They enable us to analyze how hardware, software and performance related failures affect the quality of service delivered to remote system users. Martinello et al. in Ref. [6] propose a composite performance and availability modeling approach that takes into account various causes of service unavailability. In particular, they are concerned with web cluster system modeling that involves two error recovery strategies (i.e. client-transparent and non-client-transparent) and two traffic models (i.e. Poisson and modulated Poisson). The sensitivity analyses using the proposed models point out how the traffic model affects the predicted web service availability. These analyses provide useful guidelines for the design of web-based systems, since they support the dimensioning of web architectures in order to identify the most suitable tradeoffs for achieving high availability with acceptable performance levels.

Gilmore and Kloul in Ref. [7] further develop performance modeling. They describe a performability modeling approach, which supports the analysis of performance models extracted from high-level system descriptions in Unified Modeling Language (UML). The proposed approach relies on a structured performance-engineering platform that allows the connection of a specification environment with a verification environment, so that they communicate each other. It is, therefore, possible to move forward and backward from one environment to the other one by two main communication means: *extractors* and *reflectors*. Extractors enable the translation of UML designs into performance models used in the verification environment. This translation omits any design aspect that is irrelevant for the verification task. Reflectors convert performance analysis results, obtained in the verification environment, back into design models. It is possible sequentially to compose extractors in order to provide a path from one specification formalism to another one. Similarly, it is possible sequentially to compose reflectors in order to represent analysis results into design models. The combination of extractors and reflectors allows

[1] http://www.ewics.org/.

[2] http://www.safecomp.org/.

[3] Recent SAFECOMP proceedings have been published by Springer-Verlag in the series LNCS. They are accessible online via the SAFECOMP web site or directly via the Springer LNCS web site.

0951-8320/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ress.2004.08.002

the connection of the specification environment with various verification environments. Gilmore and Kloul in Ref. [7] use the MTBDD-based PRISM probabilistic model checker in order to analyze the underlying performance models. A case study drawn from the mobile phone domain illustrates the proposed approach. The first step complies UML models into an intermediate language, i.e. the stochastic process algebra PEPA. Sequentially, PEPA models are translated into MTBDDs for the verification phase.

Another identifiable shift is that safety-critical systems often rely on evolvable information infrastructures. These infrastructures involve diverse systems and applications that act as a whole. Infrastructures are unbounded and open, scalable to new systems and new applications. Therefore, information infrastructures use diverse resources that can change in order to fulfill evolving demands. Systems rely on information infrastructures for different purposes in different situations. Unfortunately, the scenarios of use are often unknown or of no concern to system designers. Moreover, information infrastructures evolve incessantly. Therefore, it is necessary further to integrate system designs with evolvable information infrastructures. Increasingly, systems depend on information infrastructures in order to provide specific services. Understanding this dependency is important in order to assess systems safety. Górski et al. in Ref. [8] present an approach to develop trust cases for information infrastructures. They apply the proposed approach to develop a trust case for a complex IT infrastructure, i.e. the DRIVE solution. The DRIVE objectives include safe and cost-effective distribution and application of drugs. The notion of trust covers both safety and security (e.g. privacy and accountability) aspects. A trust case highlights all assumptions that support the argumentation for the trustworthiness of the system under analysis. A trust case consists of claims that support the trustworthiness of a computer system. Claims refine high-level aspects of trust at different abstraction levels and scopes. Górski et al. in Ref. [8] use UML to represent claims and the related contexts of trust cases. UML stereotypes capture the contexts of high-level claims. These claims usually have a broad scope that includes people and physical objects as well as pharmaceutical rules and knowledge (i.e. work practice). Unfortunately, system design provides limited support to the construction of these high-level models, which can be part of the trust case development. Górski et al. in Ref. [8] define a Claim Definition Language (CDL), as a conceptual model, which allows the specification of claims. The claim context model, in CDL, enables to control the scope and the language associated with a claim. The combination of the formal specification with the natural language description of claims and assumptions supports the communication of trust cases within the development environment. In practice, this allows the parallel development of different parts of trust cases, which will form the trust case as a whole.

Rae et al. in Ref. [9] present another case study drawn from the medical domain. They analyze the software implementation of the emergency shutdown feature in a major radiotherapy system. The analysis uses a directed form of code review based on module dependences. Dependences between modules are labeled by particular assumptions. This allows one to trace through the code, and identify those fragments responsible for critical features. The analysis involves a traversal through the dependency graph of the code. This traversal, in parallel, generates a tree of assumptions. An assumption tree highlights the assumptions that each module makes about others. Examination of these assumptions may reveal flaws in the system. For instance, critical features depend on unwarranted assumptions. The root of the assumption tree is the critical feature under examination. The tree leaves consist of the assumptions that, if invalid, might cause the critical feature to fail. Rae et al. in Ref. [9] report a case study that shows how the proposed analysis reveals some unexpected assumptions that require code improvements (i.e. software changes). The analysis highlights those parts of the code that are relevant to the safety-critical feature under examination. Although the proposed analysis is simple and effective, a tool support would make it less burdensome, and would make mistakes less likely to occur.

The benefit of tools for any methodology is 2-fold. On the one hand, tools allow the automation of tasks that require specific expertise. On the other hand, tools capture organizational knowledge. That is, tools capture work practice in use contexts. Weber et al. in Ref. [10] present a general procedure in combination with a tool that adapts the Software Fault Tree Analysis (SFTA) to assembler code and demonstrate its applicability to complex flight critical software. Weber et al. in Ref. [10] are concerned with any incorrect output of the software as the undesired top-level event. They generate a software fault tree from the instructions that implement software outputs. Then, they track back all instructions that contribute to these outputs in order to identify a hierarchical system of references. The resulting fault tree captures these references. The SFTA allows the detection of data flow deficiencies (e.g. incorrect usage of registers and memory areas), especially at the interfaces between code units. This highlights certain kinds of software weaknesses, which can raise safety issues with future software releases. However, other verification techniques should always complement SFTA at code level. SFTA at code level provides additional evidence for the safety and quality of safety-critical software. It provides a different viewpoint that emphasizes the semantics of the code and introduces diversity to the verification process. Weber et al. in Ref. [10] present a tool suite that automatically performs most steps from analyzing the code to drawing the fault trees. The tool suite manages the large number of fault trees generated by the process. The total number of events over all fault trees clearly inhibits manual analysis. Weber et al. in Ref. [10] apply the tool to the operational software of an inertial measurement unit, which provides safety-critical signals for artificial stabilization of an aircraft. Their experience highlights the benefits of the tool and the methodology. The tool points out coding errors that were undetected either by testing or by code inspection. Therefore, the tool is useful to complement human activities (e.g. test case selection or code inspection) that may result error-prone.

Safety-critical systems often expose the limitations of design methodologies and practices. On the one hand, it is necessary to improve the state of the art in designing safety-critical systems. On the other hand, it is evident that safety in industry contexts results from the combination of design methodologies and practices together with safety culture. Rooks et al. in Ref. [11] describe the design of a drive-by-wire computer system. The integration of drive-by-wire systems into the future generations of vehicles requires reliable and safe processing of the driver input requests. Currently, automotive computer systems use redundant hardware. Most available solutions use specialized control units as well as communication systems, which inevitably increase the production cost of automotive control systems in large scale. Unfortunately, this results in expensive systems and long developments. Rooks et al. in Ref. [11] describe a safety relevant controller composed from Commercial-Off-The-Shelf (COTS) components designed for automotive applications. The hardware architecture is a duo duplex system, which consists of four Electronic Control Units (ECU) connected via the Controller Area Network (CAN). Moreover, a hardware component, i.e. BUSPWR block, guarantees the independence of failures between ECUs. That is, the BUSPWR block stops the communication of faulty ECUs. Thus, the BUSPWR block is a non-redundant element with stringent dependability requirements. Rooks et al. in Ref. [11] describe the test procedures as well as the signaling protocols that enhance the reliability of this component. Software controls the redundancy management. Each computer node runs a version of the control software. Hence, there exist stringent safety requirements for the software components. Rooks et al. in Ref. [11], therefore, discuss the software development process and the tools used. In particular, they discuss the application of automated code generation for safety relevant drive-by-wire systems. Automated code generation increases software testability, hence, improves the correctness of software components. Moreover, automated code generation supports software portability (to other ECUs) without major software changes.

Embedded control applications such as drive-by-wire in cars require dependable interaction between various sensors, processors and actuators. Kandasamy et al. in Ref. [12] address the design of low-cost communication networks complying with both performance and fault-tolerance requirements for such distributed applications. They propose a fault-tolerant allocation and scheduling method that maps messages on to a low-cost multiple-bus system to ensure predictable inter-processor communication. The proposed method targets Time-Division Multiple Access (TDMA) communication protocols. Therefore, the method is applicable to popular protocols (e.g. FlexRay and TTP) for embedded systems such as automobile controllers. The proposed fault-tolerant clustering method allocates and schedules messages on the minimum number of buses to provide dependable transmission. Kandasamy et al. in Ref. [12] illustrate the method on a case study drawn from the automotive domain. The case study shows that sharing transmission slots among multiple messages reduces bandwidth consumption while preserving predictable communication. Hence, they conclude that the method can potentially reduce topology cost when applied to large embedded systems.

Hazard and safety analyses for safety-critical systems are major activities in order to assess whether the risk associated with any specific design is acceptable. However, they require sufficient coverage and rigor to provide enough evidence that the proposed solutions mitigate the identified risk. This often implies exhaustive hazard and safety analyses, although these are time consuming and error-prone. Reuse is a strategy to reduce the cost of developing hazard and safety analyses. Unfortunately, reuse strategies may affect the effectiveness, correctness and validity of hazard and safety analyses. Smith and Harrison in Ref. [13] propose a method for identifying the amount of reuse in hazard analysis. The proposed method relies on an edit distance algorithm that highlights argument clusters. Smith and Harrison in Ref. [13] investigate hazard analysis reuse in two case studies. Both case studies uncover a considerable amount of reuse. The first case study highlights reuse in the construction of safety arguments. The analysis identifies the argument structures in the hazard analysis. Therefore, it is possible to assess the amount of reuse. If reuse is substantial, it can give a misleading impression of rigorous coverage. This is obviously undesirable for the dependability analysis of safety-critical systems. However, the amount of reuse provides limited indication about the validity of safety cases. In order to support the construction of safety arguments, the analysis needs to determine whether there exist suitable reusable arguments in the current context. The second study is concerned with reuse changes resulting from tool support. Although tool support enables customize of reused arguments. Most reused arguments are customized trivially. The proposed edit distance algorithm allows the identification and the enumeration of reused safety arguments.

Conventionally, safety analysis consists of three main activities: definition and identification of system(s), risk analysis and definition of mitigation actions. Diverse domains (e.g. nuclear, chemical or transportation) adopt this general approach. Recently, the Air Traffic Management (ATM) domain has adopted similar safety analyses for the introduction of new systems and their related procedures. However, ATM systems and procedures present specific features that expose limitations of conventional safety analyses. In particular, the complete identification of the system under analysis is always very critical, because it affects the cost and the effectiveness of the safety analysis. If the definition of the system is too narrow and inadequately considers its relation with other heterogeneous parts of the system (e.g. human operators, procedures, interacting computer systems, etc.), then the safety analysis is inadequate. On the other hand, if the definition of the system is too broad, then the safety analysis becomes complex, extensive and unmanageable. This conventional approach is acceptable in domains such as the nuclear or the chemical sector. Nuclear or chemical plants are well-confined entities with limited predictable interactions with the surroundings. In nuclear and chemical plant design, the separation of safety related components from other plant systems is stressed. This ensures the independence of failures. Therefore, in these application domains, it is possible to identify acceptable tradeoffs between completeness and manageability during the definition and identification of the system under analysis. In contrast, ATM systems operate in open and dynamic environments. Hence, it is difficult to identify the full picture of system interactions in ATM contexts. Unfortunately, unforeseen interactions can cause catastrophic failures. Pasquini and Pozzi in Ref. [14] discuss the application of safety assessment methodologies to a pre-operational project drawn from the ATM domain. They present the major open issues faced in conducting the safety assessment of an experimental project. An interesting aspect of the case study in Ref. [14] is the necessity effectively to assess new operational procedures and tools. In particular, they exploit and integrate methodologies to evaluate computer-based applications and their interactions with the operational environment. Pasquini and Pozzi in Ref. [14] review current safety practices, methodologies, guidelines and standards in ATM domain in order to understand how to apply them to their project. Thus, they highlight specific problematic areas for the safety assessment in a pre-operational experimental project. On the basis of theoretical principles, they take into account some possible solutions. They discuss possible solutions highlighting the rationale of most relevant decisions in order to provide guidance for generalization or reuse.

This special issue collects the best SAFECOMP 2003 papers. These papers provide new insights and contribute to the progress of the state of the art in dependable applications of computer systems. As a whole they represent a step forward towards system dependability. The papers in this special issue draw directions than blend practical experience and research studies in engineering dependable systems. The papers selected for this special issue highlight challenges in the development of dependable systems. Future research should further investigate our ability in integrating different approaches. The selected papers point out that the integration of different methodologies is still a process that requires considerable effort and skilled expertise. Future research should also assess our ability to transfer specific technology to industrial contexts by integrating specific methodologies into work practice. This requires further integration of diverse methodologies throughout the system life cycle. The role of certification activities and standards still remain central to the trustworthiness of dependable systems. The systems arising in the modern IS may require different model of trust. Unfortunately, multi-disciplinary work on human factors in dependable systems is still scarce and patchy. Additional effort must be spent in cross-fertilizations between different research communities. Conventional engineering methodologies fail to capture human factors. Socio-technical systems give rise to new failure modes that have origins in lack of understanding of human factors as well as in the emergent and evolving nature of these systems. On the other hand, methodologies grounded in human related theories (e.g. Cognitive Science, Social Science, Activity Theory, Distributed Cognition, Situation Awareness, etc.) need to perform a step towards engineering methodologies.

A three-step process has selected the papers forming this special issue. The SAFECOMP 2003 International Program Committee selected a subset of potential best papers. The authors were then invited to submit an extended and revised version of their SAFECOMP 2003 papers. A second review process was then arranged in order to review the extended and revised versions of the papers. This allowed the authors to provide further technical details without the length limit that constrains the proceedings papers. The final selection for this special issue was then made according to the initial reviews for the SAFECOMP 2003 papers and the additional reviews on the extended and revised versions of the papers. The authors of the accepted papers were then required to take into account the additional comments in order to prepare the final version for this special issue. This selection process would have been impossible without the effort and support by the SAFECOMP 2003 International Program Committee and the external reviewers. We have been furthermore able to speed-up the entire review process thanks to the prompt collaboration of the authors. We would like to thank the SAFECOMP 2003 International Program Committee and the external reviewers for their tremendous work, the authors for their excellent papers. Special thanks go to the editor-in-chief of this journal, Prof. G.E. Apostolakis, for his support throughout the entire process.

References

[1] Anderson S, Littlewood B, Felici M, editors. Computer safety, reliability and security. Proceedings of the 22nd international conference, SAFECOMP 2003, Edinburgh, UK, September 23–26, LNCS 2788. Berlin: Springer; 2003.

[2] Kanoun K, Pasquini A. Guest editorial: safety, reliability and security of industrial computer systems. Reliab Eng Syst Safety 2001;71(3): 227–8.

[3] van der Meulen M, Koornneef F. Guest editorial: safety, reliability and security of industrial computer systems. Safety Sci 2002;40(9): 715–7.

[4] Anderson S, Felici M. Guest editorial: safety reliability and security of industrial computer systems. Reliab Eng Syst Safety 2003;81(3): 235–8.

[5] Voges U. Guest editorial: safety, reliability and security of industrial computer systems. Safety Sci 2004;42(5):351–4.

[6] Martinello M, Kaâniche M, Kanoun K. Web service availability—impact of error recovery and traffic model. Reliab Eng Syst Safety; this issue.

[7] Gilmore S, Kloul L. A unified tool for performance modelling and prediction. Reliab Eng Syst Safety; this issue.

[8] Górski J, Jarzębowicz A, Leszczyna R, Miler J, Olszewski. Trust case: justifying trust in an IT solution. Reliab Eng Syst Safety; this issue.

[9] Rae A, Jackson D, Ramanan P, Flanz J, Leyman D. Critical feature analysis of a radiotherapy machine. Reliab Eng Syst Safety; this issue.

[10] Weber W, Tondok H, Bachmayer M. Enhancing software safety by fault trees: experiences from an application to flight critical software. Reliab Eng Syst Safety; this issue.

[11] Rooks O, Armbruster M, Sulzmann A, Spiegelberg G, Kiencke U. Duo duplex drive-by-wire computer system. Reliab Eng Syst Safety; this issue.

[12] Kandasamy N, Hayes JP, Murray BT. Dependable communication synthesis for distributed embedded systems. Reliab Eng Syst Safety; this issue.

[13] Smith SP, Harrison MD. Measuring reuse in hazard analysis. Reliab Eng Syst Safety; this issue.

[14] Pasquini A, Pozzi S. Evaluation of air traffic management procedures—safety assessment in an experimental environment. Reliab Eng Syst Safety; this issue.

Stuart Anderson, Massimo Felici*
LFCS, School of Informatics,
The University of Edinburgh,
JCMB-KB, Mayfield Road, Edinburgh,
Scotland EH9 3JZ, UK
E-mail address. massimo.felici@ed.ac.uk

Available online 6 October 2004

*Corresponding author. Tel.: +44 131 6505899; fax: +44 131 6677209.