

Engineering Resilient Systems: Models, Methods and Tools

Edited by

Maritta Heisel¹, Mohamed Kaaniche², Alexander Romanovsky³,
and Elena Troubitsyna⁴

1 Universität Duisburg-Essen, DE, maritta.heisel@uni-duisburg-essen.de

2 LAAS – Toulouse, FR, mohamed.kaaniche@laas.fr

3 Newcastle University, GB, alexander.romanovsky@newcastle.ac.uk

4 Abo Akademi University – Turku, FI, Elena.Troubitsyna@abo.fi

Abstract

Software-intensive systems are becoming widely used in such critical infrastructures as railway, air- and road traffic, power management, health care and banking. In spite of drastically increased complexity and need to operate in unpredictable volatile environment, high dependability remains a must for such systems. Resilience – the ability to deliver services that can be justifiably trusted despite changes – is an evolution of the dependability concept. It adds several new dimensions to dependability concepts including adaptability to evolving requirements and proactive error prevention. To address these challenges we need novel models, methods and tools that enable explicit modeling of resilience aspects and reasoning about them. The Dagstuhl Seminar 13022 “Engineering Resilient Systems: Models, Methods and Tools” discussed the most promising techniques for achieving resilience both at the system design stage and at runtime. It brought together researchers from dependability, formal methods, fault tolerance and software engineering communities that promoted vivid cross-disciplinary discussions.

Seminar 7.–11. January, 2013 – www.dagstuhl.de/13022

1998 ACM Subject Classification B.8.1 Reliability, Testing, and Fault-Tolerance, D.2.1 Requirements/Specifications, D.2.2 Design Tools and Techniques, D.2.11 Software Architectures, D.4.5 Reliability, F.3.1 Specifying and Verifying and Reasoning about Programs

Keywords and phrases Resilience, modelling, verification, evaluation, fault tolerance, evolution

Digital Object Identifier 10.4230/DagRep.3.1.30

1 Executive Summary

Maritta Heisel

Mohamed Kaaniche

Alexander Romanovsky

Elena Troubitsyna

License © Creative Commons BY 3.0 Unported license

© Maritta Heisel, Mohamed Kaaniche, Alexander Romanovsky, and Elena Troubitsyna

The Dagstuhl Seminar 13022 – Engineering Resilient Systems: Models, Methods and Tools has brought together prominent researchers from different fields to discuss the problems of engineering resilient systems. The seminar was run in a highly interactive manner. The discussions were centered around the following topics:



Except where otherwise noted, content of this report is licensed under a Creative Commons BY 3.0 Unported license

Engineering Resilient Systems: Models, Methods and Tools, *Dagstuhl Reports*, Vol. 3, Issue 1, pp. 30–46
Editors: Maritta Heisel, Mohamed Kaaniche, Alexander Romanovsky, and Elena Troubitsyna



DAGSTUHL
REPORTS

Dagstuhl Reports
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

- defining resilience
- resilience in modelling languages for requirement analysis and system design
- resilience in implementation languages and frameworks
- verifying resilience using testing, model checking and static analysis
- assessing resilience using probabilist models
- resilience mechanisms at architectural and implementation level

The concept of resilience has been introduced to capture the move towards a greater adaptability and flexibility. However, the notion of resilience is still a subject of debates. The seminar has discussed various proposed definitions and converged to defining resilience as *dependability in presence of changes*.

Over the last decades a remarkable progress has been achieved in engineering of highly dependable systems, i.e., the systems that can be justifiably trusted to provide critical services to a society. However, novel computing paradigms pose new scientific and technological challenges to the dependability field. To deliver critical services in a dependable way, the systems should smoothly adapt to changes. At the seminar, we had a dedicated session discussing the nature of changes. Among the proposed categories were

- evolving user requirements
- changing operating environment
- unforeseen failure modes
- scalability challenge

Resilience is strongly linked with the entire life-cycle of a system. Engineering of resilient systems should empower the systems with capabilities to cope with changes in a predictable way, cater for evolution and ensure robust behavior in spite of faults. These require novel techniques that explicitly address resilience through the entire system development cycle. Our seminar has explored challenges in formal modelling and verification of resilient systems. At the seminar we discussed suitable models for resilience, resilience-explicit development methods and verification techniques enabling both quantitative and qualitative resilience evaluation.

Modelling is the primary vehicle driving development of resilient systems. However, system modelling area is still highly fragmented. The most acute problems are caused by

- the gap between the requirements and models and
- heterogeneity of models used to represent different aspects of system behaviour

Indeed, over the last few years the problem of poor flow-down of system requirements to software requirements has started to receive a proper attention. The vast majority of development relate the severe design problems with the flawed requirements and misunderstandings about what the software should do. Requirements tend to focus on describing nominal behaviour while omitting or poorly describing off-nominal conditions, safety constraints and fault tolerance mechanisms.

During the seminar we have brainstormed the examples of requirements that would be specific to resilient systems and tried to link them with the modelling techniques.

While developing resilient systems the designers use dedicated models to reason about different (often antagonistic) aspects of system behaviour. Hence, the design space is inherently heterogeneous. On the one hand, specialised models provide the designers with expressive and powerful techniques to analyse various aspects of system behaviour. On the other hand, it becomes hard to obtain a holistic view on the system characteristics and analyse trade-offs between several potentially conflicting goals, define the mechanisms for

adapting to volatile operating conditions and devise appropriate mechanisms for proactive fault tolerance.

We have discussed the advances in formal modelling of resilient systems and in particular proactive fault tolerance and adaptive fault tolerance mechanisms at various frameworks. We have reviewed the advances achieved in the area of formal modelling of resilient systems and brain-stormed the techniques leveraging an integration of various models to facilitate emergence of integrated modelling approaches.

Essentially, any design flow can be seen as a set of well-defined abstraction levels. The design flow should allow the designer to optimize design decision at each level and move freely between abstraction layers. At our seminar we discussed the principles of mapping abstract models onto architectural models and design implementation. We addressed the problem of achieving architectural plasticity and brain-stormed architectural patterns supporting adaptation as well as mechanisms guaranteeing adequate predictable system reaction on changes. A significant attention has also been paid to the methods and tools for resilience assessment.

Open Problems

Engineering resilient systems is a young research area. The participants of the seminar have agreed that often it is hard to distinguish a traditional dependability research from the resilience research. We have converged to the view that the system ability to scale, cope with changes and evolve emphasizes the resilience aspect.

It was also noted that the area of resilience engineering lacks a comprehensive reference guide that would allow the designers of resilient systems understand how various proposed methods and tools can facilitate design of resilient systems. The participants of the seminar has decided to work on such a book.

2 Table of Contents

Executive Summary

Maritta Heisel, Mohamed Kaaniche, Alexander Romanovsky, and Elena Troubitsyna 30

Overview of Talks

Testing and monitoring of dynamic systems: a governance-based framework <i>Antonia Bertolino</i>	35
Model-based dependability and performance assessment in evolving contexts: the CONNECT experience <i>Felicita De Giandomenico</i>	36
A Model-based Assessment Framework to Analyse the Impact of Interdependencies in Power Systems <i>Felicita De Giandomenico</i>	36
Assessing Self-Organising Systems Resilience using DREF <i>Giovanna Di Marzo Serugendo</i>	37
Analytical Architecture Fault Models <i>Peter H. Feiler</i>	37
A model checking approach in the engineering of resilient systems <i>Stefania Gnesi</i>	38
Adaptability Metrics for QoS-driven Adaptable Systems <i>Vincenzo Grassi</i>	39
Pattern and Component-based Development of Dependable Systems <i>Denis Hatebur</i>	40
Resilience – The ReSIST perspective <i>Mohamed Kaaniche</i>	40
Developing Mode-Rich Satellite Software by Refinement in Event-B <i>Linus Laibnis</i>	41
Assessing the resilience of medical device user interfaces with verification tools <i>Paolo Masci</i>	41
A “SERENE” overview of the SOTA on Engineering Resilient Systems <i>Henry Muccini</i>	42
From observations to models of resilient systems <i>Andras Pataricza</i>	42
Engineering an open-source platform for mission planning of autonomous quadrotors <i>Patrizio Pellicione</i>	43
The shape of resilience <i>Matteo Risoldi</i>	43
Concurrency & Resilience – challenges in modern IT systems <i>Thomas Santen</i>	43
Reengineering of systems supposed to be resilient <i>Rolf Schumacher</i>	44


34 **13022 – Engineering Resilient Systems: Models, Methods and Tools**

Resilience in Cyber-Physical Systems	
<i>Janos Sztipanovits</i>	44
Participants	46

3 Overview of Talks

3.1 Testing and monitoring of dynamic systems: a governance-based framework

Antonia Bertolino (ISTI-CNR, IT)

License  Creative Commons BY 3.0 Unported license
© Antonia Bertolino

For nowadays dynamic systems, resilience to failures needs to be planned in advance by collaborative agreements among involved stakeholders, ruling how components and services must be designed, documented, deployed, assessed. In our view resilience of dynamic systems should be supported by a governance framework establishing policies to be followed for off-line and on-line validation. By monitoring we can timely detect deviations of behaviour from functional and non-functional requirements. However, if the monitored system is obtained by the dynamic composition of independently developed services, who is the actor to whom monitor should report? and what action should (could) be taken? Moreover, whereas monitoring can only passively detect problems after they have occurred, a proactive approach of triggering selected behaviours might help anticipate possible future problems. We called such approach on-line testing. The idea of continuing to test a system after deployment and during real execution is appealing, but its implementation poses complex challenges: how to prevent or mitigate side effects? how can a tester simulate real or realistic service requests? In this presentation I will overview ongoing work along such directions by my group within the ongoing Choreos European Project. In particular, we are currently implementing a governance-based framework for on-line testing and monitoring of service choreographies. The presentation will outline some preliminary approaches with proposed tools and policies, but will mostly focus on open challenges for steering discussion and potential collaborations.


An annotated list of some related papers:

References

- 1 Antonia Bertolino, Guglielmo De Angelis, Sampo Kellomaki, Andrea Polini: Enhancing Service Federation Trustworthiness through Online Testing. *IEEE Computer* 45(1):66–72 (2012): *gives an overview of the idea beyond collaborative on-line testing and how this can be implemented within a service federation.*
- 2 Guglielmo De Angelis, Antonia Bertolino, Andrea Polini: Validation and Verification Policies for Governance of Service Choreographies. *WEBIST 2012*:58–70: *introduces some policies that could be considered for governing the testing of services.*
- 3 Antonia Bertolino and Andrea Polini. 2009. SOA Test Governance: Enabling Service Integration Testing across Organization and Technology Borders. In *Proc. IEEE International Conference on Software Testing, Verification, and Validation Workshops (ICSTW '09)*. IEEE Computer Society, Washington, DC, USA, 277–286 : *A keynote talk introducing the motivation and vision for test governance framework.*
- 4 Francesco De Angelis, Andrea Polini, Guglielmo De Angelis: A Counter-Example Testing Approach for Orchestrated Services. *ICST 2010*:373–382: *the solution implementing within Choreos on-line testing approach*
- 5 Antonia Bertolino, Antonello Calabro, Francesca Lonetti, Antinisca Di Marco, Antonino Sabetta: Towards a Model-Driven Infrastructure for Runtime Monitoring. *SERENE 2011*:130–144: *outlines the GLIMPSE monitoring framework*

3.2 Model-based dependability and performance assessment in evolving contexts: the CONNECT experience

Felicita De Giandomenico (ISTI-CNR, IT)

License  Creative Commons BY 3.0 Unported license
© Felicita De Giandomenico

The European FP7 Future and Emerging Technology Project CONNECT aimed at enabling seamless and dependable interoperability among networked systems in spite of technology diversity and evolution. The ambitious goal of the project was to have eternally functioning distributed systems within a dynamically evolving open-world context. This is pursued through the on-the-fly synthesis of the CONNECTors through which heterogeneous networked systems can communicate in dependable and secure way. Indeed, effective interoperability requires ensuring that such on-the-fly CONNECTed systems provide the required nonfunctional properties and continue to do so even in presence of evolution, thus calling for enhanced and adaptive assessment frameworks.

In the context of the CONNECT project, approaches to both off-line and runtime analysis have been investigated to analyze and ensure the synthesis of CONNECTors with required dependability and performance levels. In particular, an assessment framework has been proposed which combines continuous on-line assessment of non-functional properties through a lightweight flexible monitoring infrastructure with stochastic model-based analysis. The goal is to assess complex dependability and performance metrics through accurate analysis that adapts to the evolving context. Although not novel in its basic principles, this off-line and run-time integrated framework is proposed as a general, automated approach to fulfill the dependability and performance assessment needs in dynamic and evolving contexts.

3.3 A Model-based Assessment Framework to Analyse the Impact of Interdependencies in Power Systems

Felicita De Giandomenico (ISTI-CNR, IT)

License  Creative Commons BY 3.0 Unported license
© Felicita De Giandomenico

Critical Infrastructures (CI) are complex and highly interdependent systems, networks and assets that provide essential services in our daily life. Given the increasing dependence upon such critical infrastructures, research and investments in identifying their vulnerabilities and devising survivability enhancements are recognized paramount by many countries. Understanding and analyzing interdependencies and interoperabilities between different critical infrastructures and between the several heterogeneous subsystems each infrastructure is composed of, are among the most challenging aspects faced today by designers, developers and operators in these critical sectors. Assessing the impact of interdependencies on the ability of the system to provide resilient and secure services is of primary importance; following this analysis, steps can be taken to mitigate vulnerabilities revealed in critical assets. This presentation focuses on Electric Power Systems (EPS), one of the prominent representatives of CI systems, and overviews a model-based assessment framework for EPS, which explicitly accounts for interdependencies between the two infrastructures composing EPS: the Electric Infrastructure (EI) and the information infrastructure (II). The major achievements in this research line by the dependability group in Pisa along several years are shown.

3.4 Assessing Self-Organising Systems Resilience using DREF

Giovanna Di Marzo Serugendo (University of Geneva, CH)

License © Creative Commons BY 3.0 Unported license
© Giovanna Di Marzo Serugendo

One of the central features of self-organizing (SO) systems is their natural resilience to changes and faults, due to their ability to adapt their behavior. Assessing this resilience is generally done with experiments and simulations. Robustness and adaptation to some changes is obtained through specific self-organizing mechanisms, which have their limits and do not help overcoming any type of faults or change. For instance, digital pheromone in ant-based systems help overcome the appearance of obstacles in their environment or the disappearance of food, but is of limited help in case of faults (malicious or not) in the agent behavior (e.g. not properly following the pheromone). Therefore, in the process of development of a SO system, a developer will often want to achieve better resilience by adding, removing or modifying the system's behavior, then testing the system to see whether and how it has improved. Due to the complex behaviors of SO systems, however, it is not easy to quantify how a new version of a SO system compares to the one it replaces. Informal methods of comparison are generally effective only for relatively simple and small-scale systems. As SO systems are often used to model large complex behaviors, a structured, systematic and repeatable way to compare the properties of different versions of a system is necessary.

In this article we show how the evolution process taking place during the development of a SO system can benefit from a quantification of the satisfaction of properties by different versions of the system. To this end, we will enrich the classical “trial and error” development process (varying parameters and performing simulations) with a formal framework for the quantification of resilience called DREF (Dependability and Resilience Engineering Framework). The main goal of this article is to show how a precise, quantitative definition of resilience measures helps the developer in the choice of a particular version of a system.

3.5 Analytical Architecture Fault Models

Peter H. Feiler (CMU – Pittsburgh, US)

License © Creative Commons BY 3.0 Unported license
© Peter H. Feiler

This presentation discusses how challenges in safety-critical software-intensive systems are addressed by the concept of an analyzable architecture fault model – expressed in SAE AADL and its revised Error Model Annex standard. The revised Error Model Annex includes a multi-set based type system and an error propagation type ontology. The presentation illustrates its use in avionics systems to address safety and resilience concerns through analysis early in the development life cycle. Examples show fault impact analysis, compositional specification and analysis of tolerance to failures, discuss the interaction between operational and failure modes, and conclude with an illustration of resilience to the intricacies of timing behavior in safety-critical systems.

References

- 1 P. Feiler, “Analytical Architecture Faults Models”, Keynote presentation at the 3rd International Analytic Virtual Integration of Cyber-Physical Systems Workshop held in conjunction with RTSS 2012.
- 2 P. Feiler, SAE AS5506/3 Revised Error Model Annex Standard, Draft Oct 2012.
- 3 P. Feiler, D. Gluch, “Model-Based Engineering with AADL: An Introduction to the SAE Architecture Analysis & Design Language”, Addison Wesley, Sept 2012.
- 4 P. Feiler, L. Wrage, J. Hansson, System Architecture Virtual Integration: A Case Study Embedded Real-time Software and Systems Conference (ERTS2010), May 2010.
- 5 P. Feiler, A. Rugina, Dependability Modeling with the Architecture Analysis & Design Language (AADL), Software Engineering Institute (SEI) Technical Report CMU/SEI-2007-TN-043, July 2007. <http://www.sei.cmu.edu/library/abstracts/reports/07tn043.cfm>

3.6 A model checking approach in the engineering of resilient systems

Stefania Gnesi (CNR – Pisa, IT)

License  Creative Commons BY 3.0 Unported license
© Stefania Gnesi

Software Product Line Engineering (SPLE) is a paradigm for developing a diversity of software products and software-intensive systems based on the underlying architecture of an organisation’s product platform. In the context of Software Product Lines (SPLs) the introduction of variability in the software development cycle has been extensively studied [5, 7]. At all abstraction levels, a product line description is composed of a constant part and a variable part. Variability among products is made explicit by variation points, i.e., places in design artifacts where a specific decision is reduced to several features but the feature to be chosen for a particular product is left open (like optional, mandatory, or alternative features). Variety from a single product platform is achieved by identifying such variability points. Variability management is the key aspect differentiating SPLE from conventional software engineering. Modelling variability in product families has been studied extensively in the literature on SPLs, especially that concerning feature modeling [6]. Formal methods have been developed to show that a product belongs to a family or to derive instead a product from a family. Deontic-style logics [1, 8] have become popular to formalize descriptive and behavioural aspects of computer systems, mainly because they provide a natural way to formalise concepts like violation, obligation, permission and prohibition. Intuitively, these concepts permit one to distinguish correct (normative) states and actions from non-compliant ones. Hence, deontic logic is a natural candidate for expressing the conformance of members of a family of products with respect to variabilities. A number of models, logics and associated tools for the qualitative analysis of variability aspects and their use to deal with adaptability and evolvability of systems have recently been proposed. In these lectures, we will focus on the approach presented in [2, 3, 4], where the introduction of the action-based branching-time temporal logic MHML allows expressing constraints over the products of a family as well as constraints over their behaviour in a single logical framework. Based on model-checking techniques for MHML, a modelling and verification framework will be presented that can automatically generate all the family’s valid products, visualise the family/products behaviour and efficiently model check properties expressed in MHML over products and families alike. The use of the above methods, techniques and tools will be applied to a scenario derived from a family of resilient systems.

References

- 1 Åqvist, L.: Deontic logic. In: Gabbay, D., Guenther, F. (eds.) Handbook of Philosophical Logic, 2nd edition, vol. 8, pp. 147–264. Kluwer (2002)
- 2 Asirelli, P., ter Beek, M.H., Fantechi, A., Gnesi, S.: A logical framework to deal with variability. In: Mery, D., Merz, S. (eds.) IFM 2010. Lecture Notes in Computer Science, vol. 6396, pp. 43–58. Springer (2010)
- 3 Asirelli, P., ter Beek, M.H., Fantechi, A., Gnesi, S.: A verification environment for families of services. In: Bruni, R., Dingel, J. (eds.) FMOODS/FORTE 2011. Lecture Notes in Computer Science, vol. 6722, pp. 44–58. Springer (2011)
- 4 Asirelli, P., ter Beek, M.H., Gnesi, S., Fantechi, A.: Formal description of variability in product families. In: SPLC 2011, pp. 130–139. IEEE (2011)
- 5 Clements, P.C., Northrop, L.: Software Product Lines-Practices and Patterns. Addison-Wesley (2002)
- 6 Kang, K., Choen, S., Hess, J., Novak, W., Peterson, S.: Feature Oriented Domain Analysis (FODA) Feasibility Study. Technical Report SEI-90-TR-21, Carnegie Mellon University (1990)
- 7 Pohl, K., Bockle, G., van der Linden, F.: Software Product Line Engineering-Foundations, Principles, and Techniques. Springer (2005)
- 8 Meyer, J.-J.Ch., Wieringa, R.J. (eds.): Deontic Logic in Computer Science – Normative System Specification. John Wiley & Sons (1994)

3.7 Adaptability Metrics for QoS-driven Adaptable Systems

Vincenzo Grassi (Università di Roma “Tor Vergata”, IT)

License © Creative Commons BY 3.0 Unported license
© Vincenzo Grassi

Joint work of Grassi, Vincenzo; Mirandola Raffaella

One of the major current research trends in Software Engineering is the focus on the development of new methodologies and techniques to deal efficiently with the design of more resilient systems that are able to evolve and adapt to rapid changes of their requirements or their execution environment. We present some ideas about the definition of metrics able to quantify and evaluate the adaptability of a software system at the architectural level. In particular, we focus on metrics measuring the system adaptability with respect to its ability to fulfill Quality of Service requirements. These metrics could be used by the software architect to drive design decisions concerning QoS-oriented adaptability features. They could also be helpful during runtime – when human intervention is not possible – to drive self-adaptation actions based on the impact they can have on the architecture adaptability.

References

- 1 A. DeLoach, V. A. Kolesnikov “Using Design Metrics for Predicting System Flexibility” FASE 2006, LNCS 3922, pp. 184–198, 2006
- 2 E. Kaddoum, C. Raibulet, J.-P. Georg, G. Picard, M.-P. Gleizes “Criteria for the evaluation of self-* systems” ACM SEAMS (2010), pp. 29–38
- 3 D. Perez-Palacin, R. Mirandola, J. Merseguer, “Software Architecture Adaptability Metrics for QoS-based Self-Adaptation” ACM QoSA (2011)
- 4 C. Raibulet, Masciadri, “Evaluation of dynamic adaptivity through metrics: an achievable target?” IEEE WICSA/ECSA (2009), pp. 341–344

- 5 P. Reinecke, K. Wolter, A. van Moorsel., “Evaluating the adaptivity of computing systems” *Performance Evaluation* 67 (2010), pp. 676–693
- 6 H. Schmeck, C. Muller-Schloer, E. Cakar, M. Mnif, U. Richter “Adaptivity and self-organization in organic computing systems” *ACM Trans. on Autonomous and Adaptive Systems*, 5(3)10, Sept. 2010
- 7 N. Subramanian, L. Chung “Metrics for software adaptability” *Software Quality Management* (2001) pp. 95–108

3.8 Pattern and Component-based Development of Dependable Systems

Denis Hatebur (Universität Duisburg-Essen, DE)

License  Creative Commons BY 3.0 Unported license
© Denis Hatebur

The presentation discusses a process for pattern and component-based development of dependable systems. Dependability is an important aspect of resilient systems. The process is based on an description of the environment and covers analysis, design, implementation and testing. It is supported by an extended UML tool that checks the consistency of different artifacts. The process is extended by patterns for dependability requirements (confidentiality, integrity, availability, and reliability) that are part of a pattern system used to identify missing and conflicting requirements. It also covers a structured development of the architecture for dependable systems that fulfils the requirements.

3.9 Resilience – The ReSIST perspective


Mohamed Kaaniche (LAAS – Toulouse, FR)

License  Creative Commons BY 3.0 Unported license
© Mohamed Kaaniche

This talk is aimed at presenting the definition of Resilience that has been proposed in the context of The ReSIST Network of Excellence funded by the European Commission. This definition is used as a starting point for initiating interactions and discussions about: 1) the difference between resilience and other similar concepts such as dependability, trustworthiness, survivability, etc., 2) whether this definition needs to be extended, and 3) the main new challenges that need to be addressed for the development of resilient systems compared to traditional existing approaches used the development of fault tolerant and dependable computing systems.

3.10 Developing Mode-Rich Satellite Software by Refinement in Event-B

Linas Laibinis (Abo Akademi University – Turku, FI)

License  Creative Commons BY 3.0 Unported license
© Linas Laibinis

One of the guarantees that the designers of on-board satellite systems need to provide, so as to ensure their dependability, is that the mode transition scheme is implemented correctly, i.e. that the states of system components are consistent with the global system mode. There is still, however, a lack of scalable approaches to developing and verifying systems with complex mode transitions. This paper presents an approach to formal development of mode-rich systems by refinement in Event-B. We formalise the concepts of modes and mode transitions as well as deriving specification and refinement patterns which support correct-by-construction system development. The proposed approach is validated by a formal development of the Attitude and Orbit Control System (AOCS) undertaken within the ICT DEPLOY project. The experience gained in the course of developing such a complex industrial system as AOCS shows that Event-B refinement provides the engineers with a scalable formal technique. Moreover, the case study has demonstrated that Event-B can facilitate formal development of mode-rich systems and, in particular, proof-based verification of their mode consistency.

3.11 Assessing the resilience of medical device user interfaces with verification tools


Paolo Masci (Queen Mary University of London, GB)

License  Creative Commons BY 3.0 Unported license
© Paolo Masci

Medical device regulators such as the US Food and Drug Administration (FDA) aim to make sure that medical devices are reasonably safe before entering the market. To expedite the approval process and make it more uniform and rigorous, regulators are considering the development of reference models that encapsulate safety requirements against which software incorporated in to medical devices must be verified. Safety, insofar as it relates to interactive systems and its regulation, is generally a neglected topic, particularly in the context of medical systems. An example is presented that illustrates how the interactive behaviour of a commercial Patient Controlled Analgesia (PCA) infusion pump can be verified against a reference model. Infusion pumps are medical devices used in healthcare to deliver drugs to patients, and PCA pumps are particular infusion pump devices that are often used to provide pain relief to patients on demand. The reference model encapsulates the Generic PCA safety requirements provided by the FDA, and the verification is performed using a refinement approach.

3.12 A “SERENE” overview of the SOTA on Engineering Resilient Systems

Henry Muccini (Univ. degli Studi di L’Aquila, IT)

License  Creative Commons BY 3.0 Unported license
© Henry Muccini

This short talk will introduce a preliminary analysis and classification of papers on software resilience presented in the past SERENE workshops. It wants to serve the purpose to create a starting point for realizing a survey on methods, approaches, and tools for engineering resilient systems.

3.13 From observations to models of resilient systems

Andras Pataricza (Budapest Univ. of Technology & Economics, HU)

License  Creative Commons BY 3.0 Unported license
© Andras Pataricza

Modeling, analysis, design of resilience related phenomena all need empirical substantiation due to the complexity of the typical target system and the underlying mechanisms and phenomena, as well. While model based design paradigms are well-proven and highly efficient in addressing complex problems, the faithfulness of the model is a crucial factor deciding in the very first moment all the quality of the results generated by the subsequent analysis/-synthesis steps. Accordingly, the extraction of a proper model from initial observations is the indispensable precondition for a well-substantiated approach in any resilience related engineering problem. Analyzing empirical resilience related data (like operation logs, or data sequences gained in fault injection campaigns or benchmarking experiments) is extremely difficult in the terms of statistics despite of decades long research efforts.


Analysis of resilience related data is clearly a big data problem over long sequences of many dimensional data reaching frequently the order of magnitude of several gigasamples of tens of thousands of signals. Moreover, as systems are dependable enough, fault manifestations are typically only rare outliers in a huge amount of samples corresponding to the correct behavior. As a consequence most popular algorithms widely used in other fields of statistics are of little use, as they simply suppress outliers and the hard class of rare event processing has to be used instead.

The problem of model creation is in itself a multi-phase process. The first, exploratory phase serves on the identification and rough characterization of the phenomena observed including the analysis of the individual signals, detection of the outliers and the relation between different signals. The derived characteristics help in the later phases of resilience analysis the estimation of principal factors leading to the individual failure modes, control clustering identifying typical operation domains and failure modes, etc.

Typically, in this initial phase of analysis, there is no or little statistical knowledge on the observed data to be analyzed. Exploratory data analysis (EDA) is an effective visual analysis approach to extract the main characteristics without the necessity of using not well founded and unnecessarily restrictive mathematical assumptions. The outcome of EDA is on the one hand a qualitative phenomenological model to be used in the formal analysis, and guiding heuristics for the detailed, already algorithmic analysis.

3.14 Engineering an open-source platform for mission planning of autonomous quadrotors

Patrizio Pelliccione (Univ. degli Studi di L'Aquila, IT)

License  Creative Commons BY 3.0 Unported license
© Patrizio Pelliccione

Several projects exist to specify environmental monitoring missions. However, existent projects specify missions by means of programming languages which are too distant from the knowledge and terminology of the kind of users typically involved in such tasks. In this paper we propose an open-source platform, which enables the specification of monitoring missions that will be performed by fleets of autonomous quadrotors. The specification is performed at a high-level of abstraction and permits to graphically specify missions in the ground station. The mission will be then automatically decomposed by the platform in instructions to be performed by each quadrotor in order to fulfill the common goal. The platform enables users with limited IT knowledge, but domain experts in environmental missions to plan missions easily. A reconfiguration engine is specifically designed to autonomously react to faults and external events in order to accomplish the designed mission. Moreover, under some limitations explained in the paper, the reconfiguration engine permits to change the mission at run-time.

3.15 The shape of resilience

Matteo Risoldi (University of Luxembourg, LU)

License  Creative Commons BY 3.0 Unported license
© Matteo Risoldi

There exist several different definitions of resilience, based on different scientific domains, intended purposes of the definition, requirements, and other scientific and cultural biases. We think however that, in most cases, these different definitions simply identify different ways that resilience manifests itself, rather than actually different “types” of resilience. In this talk, we present DREF, a formal framework for dependability and resilience, that allows quantification and visualization of many concepts related to resilience. In addition to being used for the quantitative assessment of resilience, DREF can be usefully employed for visualizing and identifying the fundamental ways that resilient behaviour manifests itself, through the variation of resilience-related parameters. We give a few examples of how different definitions of resilience may be represented in DREF, and give a (non-exhaustive) list of common traits of resilient behaviour.

3.16 Concurrency & Resilience – challenges in modern IT systems

Thomas Santen (European Microsoft Innovation Center – Aachen, DE)

License  Creative Commons BY 3.0 Unported license
© Thomas Santen

Performance is one of the key requirements on modern IT systems from a user’s perspective. This has major implications on the way that those systems are designed and implemented.

Many safe technologies that avoid certain types of faults by design such as type safe languages often cannot be used because they would impact the performance of the resulting system in an unacceptable way. True concurrency as induced by many-core systems is another source of complexity. To handle this complexity, techniques like formal code verification and model-based approaches can be helpful. Looking at the future of service-based systems resilience in providing services despite failures in data centers, communication, or other effects on the service infrastructure, is an increasingly relevant concern, and more technology to cope with those effects is needed that should take into account the other constraints of industrial software production like the key requirement of performance.

3.17 Reengineering of systems supposed to be resilient

Rolf Schumacher (Ingenieur-Büro Rolf Schumacher – Buchholz, DE)

License  Creative Commons BY 3.0 Unported license
© Rolf Schumacher

With the event of clever virtual (as opposed to physical) attacks to public cyber physical systems or the ever increasing frequency of technological changes there will be a demanding need to improve the architecture of existing systems regarding yet unconsidered resilient properties. However, many of today's average software systems, including cyber physical systems, lack a reliable and complete set of maintainable requirements and architecture from where to start improvements in a controlled manner. As there are many thinkable approaches for re-engineering existing software systems in place and many of them failed or turned out to be overly expensive this contribution presents a process that worked in practice. The presented process has been applied to improve software for dependability. The process proceeds to be applied to similar average software systems, proving it to be general enough to be applied to a variety of existing software systems to be improved.

After a description of some challenging dependability aspects, a use-case driven top-down approach is presented for a behavioral model. It limits the significance of static models in favor of the more robust service oriented view. It leads to a set of requirements and a manageable architecture fulfilling its desired goals for the re-engineered system, e.g. dependability assessment. It is observed that one key success factor has been the renouncement of completeness of system states in favor of completeness of component functions usage. Having this experience in place we can concentrate on resilience requirements improving existing systems.

3.18 Resilience in Cyber-Physical Systems

Janos Sztipanovits (Vanderbilt University, US)

License  Creative Commons BY 3.0 Unported license
© Janos Sztipanovits

DARPA Adaptive Vehicle Make (AVM) Program is a major DARPA program a decade after MoBIES:

- End-to-end model- and component-based design and integrated manufacturing of a next generation amphibious infantry vehicle – a complex, real-life cyber-physical sys-

tem. From infrastructure to manufactured vehicle prototype in five years (2010-2014).
Engineering/economic goals:

- Decrease development time by 80% in defense systems (brings productivity consistent with other industries)
- Enable the adoption of fables design and foundry concept in CPS
- “Democratize” design by open source tool chain, crowd-sourced model library and prize-based design challenges

Scientific challenge: achieve AVM goals by pushing the limits of “correct-by-construction” design

- “Separation of concerns” principle need to be re-examined META pursues multi-physics, multi-abstraction and integrated cyber-physical design flows: modeling cross-domain interactions is in focus
- Multi-modeling makes model integration a fundamental challenge in the META design automation tool chain. META extensively uses model integration and includes CYPhy – a model integration language.

Participants

- Antonia Bertolino
CNR – Pisa, IT
- Felicità Di Giandomenico
CNR – Pisa, IT
- Giovanna Di Marzo Serugendo
University of Geneva, CH
- Peter H. Feiler
CMU – Pittsburgh, US
- Stefania Gnesi
CNR – Pisa, IT
- Vincenzo Grassi
Università di Roma “Tor Vergata”, IT
- Denis Hatebur
Universität Duisburg-Essen, DE
- Maritta Heisel
Universität Duisburg-Essen, DE
- Mohamed Kaaniche
LAAS – Toulouse, FR
- Linas Laibinis
Abo Akademi University –
Turku, FI
- Paolo Masci
Queen Mary University of
London, GB
- Henry Muccini
Univ. degli Studi di L’Aquila, IT
- Andras Pataricza
Budapest Univ. of Technology &
Economics, HU
- Patrizio Pelliccione
Univ. degli Studi di L’Aquila, IT
- Matteo Risoldi
University of Luxembourg, LU
- Alexander Romanovsky
Newcastle University, GB
- Thomas Santen
European Microsoft Innovation
Center – Aachen, DE
- Rolf Schumacher
Ingenieur-Büro Rolf Schumacher –
Buchholz, DE
- Janos Sztipanovits
Vanderbilt University, US
- Anton Tarasyuk
Abo Akademi University, FI
- Elena Troubitsyna
Abo Akademi University, FI
- Marco Vieira
University of Coimbra, PT

