Foreword

Enforcing the dependability of software systems has been a very active and productive area of research for over 30 years, addressing support for fault prevention, fault tolerance, fault removal and fault forecasting. Such an effort has in particular led to introducing a number of dependability concepts, and related principled methods and tools guiding the structuring of dependable systems, and further allowing reasoning about the systems' dependability. As such, research results in the dependability area impact upon the overall software development process. However, dependability has for long been considered as an aside property in the development of software systems, except for specific classes of systems such as safety-critical systems that cannot tolerate unexpected behavior following the occurrence of failures.

The increasing reliance on software systems that are now surrounding our everyday's life, being embedded in most devices, and providing us with access to a huge amount of content and services via the Internet, makes dependability a prime requirement for today's systems. In particular, dependability requirements should be accounted for in the early phase of the development process, since dependability means significantly impact design choices. In this context, architectural modelling of software systems offers much benefit towards assisting the design of dependable systems and assessing their dependability. By abstracting the low-level details of the system, architectural modelling allows effective exploitation of formal methods for reasoning about the systems' behavior, which constitutes a key dependability means. Architectural modelling further allows developers to comprehensively deal with dependability requirements in the structuring of their systems.

Bringing together researchers from the dependability and software architecture communities, via dedicated workshops and books on improvement to the state of the art on architecting dependable systems, can only be acknowledged as a valuable effort towards eliciting architecture modelling languages, methods and tools that support the thorough development of dependable software systems. This book, the second of an undoubtedly promising series, introduces research results that show how dependability and software architecture research conveniently complement and benefit from each other, addressing specific system architecting for dependability, integration of fault tolerance means with software architecture, and architecture modelling for dependability analysis. Last but not least, reports on industrial experience highlight how such an effort meets industrial practices in dependable software system development. As a result, the breadth and depth of the coverage that is provided by this book on recent research in architecting dependable systems is particularly impressive, and the editors and authors are to be congratulated.

June 2004

Valérie Issarny
INRIA
Research Unit of Rocquencourt

Preface

System dependability is defined as reliance that can be justifiably placed on the service delivered by the system. It has become an essential aspect of computer systems as everyday life increasingly depends on software. It is therefore a matter for concern that dependability issues are usually left until too late in the process of system development. This is why, even though there is a large body of research on dependability, reasoning about dependability at the architectural level is only just emerging as an important theme. It is a theme that needs to be actively pursued since architectural representations have been shown to be effective in helping to understand broader system characteristics by abstracting away from details. Apart from this, there are other factors that make it urgent to consider dependability at the architectural level, such as the complexity of emerging applications and the need for building trustworthy systems from the existing untrustworthy components.

This book comes as a result of an effort to bring together the research communities of software architectures and dependability. It was inspired by the ICSE 2003 Workshop on Software Architectures for Dependable Systems (WADS 2003), where many interesting papers were presented and lively discussions took place. The book addresses issues that are currently relevant to improving the state of the art in architecting dependable systems. It presents a selection of peer-reviewed papers stemming from some original WADS 2003 contributions and several invited ones. The book consists of four parts: architectures for dependability, fault tolerance in software architectures, dependability analysis in software architecture, and industrial experience.

The first part of this book focuses on software architectures for dependability. Its first paper, by Koutsoukos, Loureno, Avillez, Gouveia, Andrade, Fiadeiro, and Wermelinger, is entitled "Enhancing Dependability Through Flexible Adaptation to Changing Requirements". This paper describes an architectural approach that relies on coordination contracts to facilitate the dynamic adaptation of systems to changing domain rules. The approach is illustrated through a case study in the financial systems area, where agreed policies and conditions are negotiated on a case-by-case basis. The paper concludes by reporting on an information system that ATX Software developed for a company specialized in recovering bad credit. The second paper in this part is "A Self-optimizing Run-Time Architecture for Configurable Dependability of Services" by Tichy and Giese. In this paper, Tichy and Giese identify a set of architectural principles that can be used to improve the dependability of service-based architectures. These architectural principles have been instantiated by extending Jini, and have been evaluated qualitatively and quantitatively for a configuration of multiple identical services, showing how the different parameters affect the resulting dependability. The paper by Knight and Strunk on "Achieving Critical System Survivability Through Software Architectures" addresses the idea of making a system survivable rather than highly reliable or highly available by exploring the motivation for survivability, how it might be used, what the concept means in a precise and testable sense, and how it is being implemented in two very different application areas. The subsequent

paper is authored by Rodrigues, Roberts and Emmerich. It is on "Reliability Support for the Model Driven Architecture" and elaborates on how the provision of reliability can be suitably realized through Model Driven Architectures (MDA). It is based on a platform-independent reference model that can be mapped to specific platforms. The UML metamodeling language is extended to show how design profile elements reflect on the deployment of the components when transformation rules are applied to the model. The last paper in this part, "Supporting Dependable Distributed Applications Through a Component-Oriented Middleware-Based Group Service" by Saikoski and Coulson, presents a group-based middleware platform that aims at supporting flexibility for controlled redundancy, replication, and recovery of components and services. This flexibility is provided at design time, deployment time and run-time. Their approach is based on concepts of software component technology and computational reflection.

The second part of this book is related to fault tolerance in software architectures. In the first paper, "Architecting Distributed Control Applications Based on (Re-) Configurable Middleware", Deconinck, De Florio and Belmans introduce the DepAuDE architecture developed for industrial distributed automation applications. This architecture provides a fault tolerance middleware, a library for error detection and recovery and fault treatments, and a specialized language called ARIEL for specifying fault tolerance and configuration actions. The paper concludes with a thorough discussion of a case study: a demonstrator of a Primary Substation Automation System controlling a substation for electricity distribution. The second paper of this part is entitled "A Dependable Architecture for COTS-Based Software Systems Using Protective Wrappers". The authors, Guerra, Rubira, Romanovsky and de Lemos, combine the concepts of an idealized architectural component and protective wrappers to develop an architectural solution that provides an effective and systematic way for building dependable software systems from COTS software components. The approach is evaluated using a PID controller case study. The next paper entitled "A Framework for Reconfiguration-Based Fault-Tolerance in Distributed Systems" is co-authored by Porcarelli, Castaldi, Di Giandomenico, Bondavalli and Inverardi. In this framework fault tolerance of components-based applications is provided by detecting failures using system monitoring, and by recovery employing system reconfiguration. The framework is based on Lira, an agent distributed infrastructure employed for component and application level monitoring and reconfiguration, and a decision maker used for selecting new configurations using the feedbacks provided by the evaluation of stochastic Petri net models. In the next paper, "On Designing Dependable Services with Diverse Off-The- Shelf SQL Servers", Gashi, Popov, Stankovic and Strigini argue, based on empirical results from their ongoing research with diverse SQL servers, in favor of diverse redundancy as a way of improving dependability and performance of a SQL server. The paper provides evidence that current data replication solutions are insufficient to protect against the range of faults documented for database servers, outlines possible fault-tolerant architectures using diverse servers, discusses the design problems involved, and offers evidence of performance improvement through diverse redundancy. The last paper of part two, "A New Model and a Design Approach to Building Quality of Service (QoS) Adaptive Systems", is co-authored by Ezhilchelvan and Shrivastava. The focus is on developing Internet-based services provisioning systems. The authors propose a system architecture and identify

a model appropriate for developing distributed programs that would implement such systems. The probabilistic asynchronous model proposed abstracts the network performance and dependability guarantees typically offered by the Internet service providers. The system architecture prescribes the role of QoS management algorithms to be: evaluating the feasibility of QoS requests from the end users and adapting system protocols in response to changes in the environment.

Part three of this book deals with dependability analysis in software architectures. In the first paper, which is entitled "Multi-view Software Component Modeling for Dependability", the authors Roshandel and Medvidovic focus on a more comprehensive approach for modelling components. Instead of relying just on the description of the components interfaces, and their respective pre- and post-conditions, the authors propose an approach to modelling components using four primary functional aspects of a software component (known as the Quartet): interface, static behavior, dynamic behavior, and interaction protocol. In addition to describing individually the four aspects, the paper also discusses the relationships between them for ensuring their compatibility. The goal of the work is to obtain support for the architectural-level modelling and analysis of system dependability, in particular, reliability. The second paper, "Quantifiable Software Architecture for Dependable Systems of Systems" by Liang, Puett and Luqi presents an approach for the development and evolution of dependable systems-of-systems. Based on the architectural description of these systems, the approach involves establishing consensus between the different dependability attributes associated with component systems, and translating them into quantifiable constraints. The approach illustrates that with reusable architectural facilities and associated tools support, the quantifiable architecture with multiple perspectives can be effective in supporting the engineering of dependable systems-of-systems. In the last paper of this part, which is entitled "Dependability Modeling of Self-healing Client-Server Applications", the authors Das and Woodside present an analytical model for evaluating the combined performance and dependability attributes of fault-tolerant distributed applications. The authors consider a layered software architecture in which the application and management components can fail and be repaired. It also considers the management of connections, and the application's layered failure dependencies, together with the application performance. In order to show the capability of the approach in evaluating large-scale systems, the authors apply their analytical model to an air traffic control system.

The final part of the book contains two papers that report on existing industrial experiences involving dependability in the context of software architectures. In the first paper, entitled "A Dependable Platform for Industrial Robots", the authors Mustapic, Andersson, Norström and Wall discuss the design of an open software platform for an ABB Robotic System. For them a software platform is the basis for a product-line architecture that aims to increase the number of variations between the different software systems, while maintaining the integrity of the whole robotic system. An initial step in their approach is to model at the architectural level the quality constraints of the platform, which include several dependability attributes. The second paper of this final part, "Model Driven Architecture an Industry Perspective" by Raistrick and Bloomfield, discusses some of the research work that is currently being undertaken within the avionics industry on the usage of Model Driven Architectures (MDA), an initiative of the Object

Management Group (OMG). It has been recognized that the MDA approach might become fundamental in reducing costs in the development and maintenance of software. The authors of this paper identify several fronts in which the usage of the MDA might be effective. These are: the automation of software development with the support of tools, the management of legacy systems, the mapping of avionic applications into standard modular computer systems, and the incremental certification of avionics systems.

We believe that the introduction of the topic of architecting dependable systems is very timely and that work should continue in this area. The first book of the same title, published in the summer of 2003, included expanded papers based on selected contributions to the WADS ICSE 2002 workshop and a number of invited papers. The forthcoming ICSE/DSN 2004 Twin Workshops on Architecting Dependable Systems is another ambitious project, which aims to promote cross-fertilization between the communities of software architectures and dependability.

As editors of this book, we are certain that its contents will prove valuable for researchers in the area and are genuinely grateful to the many people who made it possible. Our thanks go to the authors of the contributions for their excellent work, the WADS 2003 participants for their active support and lively discussions, and Alfred Hofmann from Springer-Verlag for believing in the idea of this book and helping us to get it published. Last but not least, we appreciate the time and effort our reviewers devoted to guaranteeing the high quality of the contributions. They are D. Akehurst, T. Bloomfield, A. Bondavalli, F.V. Brasileiro, M. Castaldi, G. Deconinck, F. Di Giandomenico, M. Correia, G. Coulson, I. Crnkovic, S. Crook-Dawkins, W. Emmerich, J.L. Fiadeiro, G. Fohler, P. Inverardi, V. Issarny, J. Knight, N. Levy, N. Medvidovic, C. Norstrom, A. Pataricza, P. Popov, S. Riddle, G. Roberts, C.M.F. Rubira, S. Shrivastava, F. van der Linden, P. Veríssimo, M. Wermelinger, C.M. Woodside, and several anonymous reviewers.

June 2004

Rogério de Lemos
Cristina Gacek
Alexander Romanovsky