Analysis of enterprise IT service availability

Enterprise architecture modeling for assessment, prediction, and decision-making

ULRIK FRANKE

Doctoral Thesis
Stockholm, Sweden 2012

TRITA EE 2012:032
ISBN 978-91-7501-443-2
ISSN 1653-5146
ISRN KTH/ICS/R--12/02--SE

Industrial Information and Control Systems
KTH, Royal Institute of Technology
Stockholm, Sweden

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Set in LaTeX by the author
Printed by Universitetsservice US AB

Abstract

Information technology has become increasingly important to individuals and organizations alike. Not only does IT allow us to do what we always did faster and more effectively, but it also allows us to do new things, organize ourselves differently, and work in ways previously unimaginable. However, these advantages come at a cost: as we become increasingly dependent upon IT services, we also demand that they are continuously and uninterruptedly available for use. Despite advances in reliability engineering, the complexity of today's increasingly integrated systems offers a non-trivial challenge in this respect. How can high availability of enterprise IT services be maintained in the face of constant additions and upgrades, decade-long life-cycles, dependencies upon third-parties and the ever-present business-imposed requirement of flexible and agile IT services?

The contribution of this thesis includes (i) an enterprise architecture framework that offers a unique and action-guiding way to analyze service availability, (ii) identification of causal factors that affect the availability of enterprise IT services, (iii) a study of the use of fault trees for enterprise architecture availability analysis, and (iv) principles for how to think about availability management.

This thesis is a composite thesis of five papers. Paper 1 offers a framework for thinking about enterprise IT service availability management, highlighting the importance of variance of outage costs. Paper 2 shows how enterprise architecture (EA) frameworks for dependency analysis can be extended with Fault Tree Analysis (FTA) and Bayesian networks (BN) techniques. FTA and BN are proven formal methods for reliability and availability modeling. Paper 3 describes a Bayesian prediction model for systems availability, based on expert elicitation from 50 experts. Paper 4 combines FTA and constructs from the ArchiMate EA language into a method for availability analysis on the enterprise level. The method is validated by five case studies, where annual downtime estimates were always within eight hours from the actual values. Paper 5 extends the Bayesian prediction model from paper 3 and the modeling method from paper 4 into a full-blown enterprise architecture framework, expressed in a probabilistic version of the Object Constraint Language. The resulting modeling framework is tested in nine case studies of enterprise information systems.

Keywords: Service Level Agreement, outage costs, Enterprise Architecture, enterprise IT service availability, decision-making, metamodeling, Enterprise Architecture analysis, Bayesian networks, fault trees, Predictive Probabilistic Architecture Modeling Framework

Sammanfattning

Informationsteknik blir allt viktigare för både enskilda individer och för organisationer. IT låter oss inte bara arbeta snabbare och effektivare med det vi redan gör, utan låter oss också göra helt nya saker, organisera oss annorlunda och arbeta på nya sätt. Tyvärr har dessa fördelar ett pris: i takt med att vi blir alltmer beroende av IT-tjänster ökar också våra krav på att de är ständigt tillgängliga för oss, utan avbrott. Trots att tillförlitlighetstekniken går framåt utgör dagens alltmer sammankopplade system en svår utmaning i detta avseende. Hur kan man säkerställa hög tillgänglighet hos IT-tjänster som ständigt byggs ut och uppgraderas, som har livscykler på tiotals år, som är beroende av tredjepartsleverantörer och som dessutom måste leva upp till verksamhetskrav på att vara flexibla och agila?

Den här avhandlingen innehåller (i) ett arkitekturramverk som på ett unikt sätt kan analysera IT-tjänsters tillgänglighet och ta fram rekommenderade åtgärder, (ii) ett antal identifierade kausalfaktorer som påverkar IT-tjänsters tillgänglighet, (iii) en studie av hur felträd kan användas för arkitekturanalys av tillgänglighet samt (iv) en uppsättning principer för beslutsfattande kring tillgänglighet.

Avhandlingen är en sammanläggningsavhandling med fem artiklar. Artikel 1 innehåller ett konceptuellt ramverk för beslutsfattande kring IT-tjänsters tillgänglighet som understryker vikten av variansen hos nertidskostnaderna. Artikel 2 visar hur ramverk för organisationsövergripande arkitektur (s.k. *enterprise architecture* – EA) kan utvidgas med felträdsanalys (FTA) och bayesianska nätverk (BN) för analys av beroenden mellan komponenter. FTA och BN är bägge etablerade metoder för tillförlitlighets- och tillgänglighetsmodellering. Artikel 3 beskriver en bayesiansk prediktionsmodell för systemtillgänglighet, baserad på utlåtanden från 50 experter. Artikel 4 kombinerar FTA med modelleringselement från EA-ramverket ArchiMate till en metod för tillgänglighetsanalys på verksamhetsnivå. Metoden har validerats i fem fallstudier, där de estimerade årliga nertiderna alltid låg inom åtta timmar från de faktiska värdena. Artikel 5 utvidgar den bayesianska prediktionsmodellen från artikel 3 och modelleringsmetoden från artikel 4 till ett fullständigt EA-ramverk som uttrycks i en probabilistisk version av Object Constraint Language (OCL). Det resulterande modelleringsramverket har testats i nio fallstudier på verksamhetsstödjande IT-system.

Nyckelord: Service Level Agreement, nertidskostnader, Enterprise Architecture, tillgänglighet hos IT-tjänster, beslutsfattande, metamodellering, arkitekturanalys, bayesianska nätverk, felträd, Predictive Probabilistic Architecture Modeling Framework

Acknowledgments

My first debt of gratitude is to my supervisors Pontus Johnson, Göran Ericsson, and Lars Nordström, each of whom has made a distinct impact on the final result. As many of the good things in life, your guidance will probably be the most missed only when it is no longer part of the daily routine. And of course, it was all done in the spirit of the grand old man, Torsten Cegrell, who initially hired me and made me part of a great team.

Being a PhD student is sometimes said to be a lonely endeavor. However, it feels considerably less so in the company of good colleagues. Waldo Rocha Flores, Johan König, Liv Gingnell, Per Närman, Markus Buschle, and Mathias Ekstedt, my co-authors in the papers making up this thesis, deserve a special nod, but Joakim Lilliesköld, Robert Lagerström, Pia Närman, Moustafa Chenine, Johan Ullberg, Teodor Sommestad, David Höök, Kun Zhu, Evelina Ericsson, Hannes Holm, Claes Sandels, Magnus Österlind, Judith Westerlund, Annica Johannesson, Erik Johansson, and Mårten Simonsson have all played their parts, as have all the good friends at the Technische Universität Berlin, Technische Universität München and Universität St. Gallen.

This is a thesis on an applied subject, and it would not have been feasible without close cooperation with real-world companies. I am particularly grateful to Niclas Almlöv, Jalal Matini, Ulf Karlsson, Christoffer Eile, Peter Eriksson, Mattias Nyman, Erik Lundin, Michael Mirbaha and Jakob Raderius for invaluable help with the empirics and with looking through the practitioner lens.

Getting a PhD is a lot more than just doing research and writing a thesis. At its best, it is a time of intellectual, personal and professional growth and exploration. All in their own ways, Jonas Andersson, Asmus Pandikow, Erik Herzog, Michael Stolz, Niclas Holmquist, Stefan Varga, Imre Juhasz, Tobias Lindenkäll, Jenny Remén, Mattias Wallén, Lars Gezelius and Emil Gelebo have contributed to making me a better person.

During the final sprint of the thesis work, I started working as a scientist at the Swedish Defence Research Agency (FOI). I am grateful to Lars Lindberg, Pontus Svenson, and Vahid Mojtahed for welcoming me into this new research environment, and for generously allowing me to share my time between FOI and KTH as needed to complete my PhD.

On a more personal note, I am grateful to my family – Cecilia, Peter, Oskar, and Axel – and to my friends and loved ones, particularly to Evelina Lorentzon, Petrus Boström, Katarina O'Nils and Jens Bäckbom.

En route to the PhD, I have received financial support from numerous institutions. The Swedish national grid (Svenska Kraftnät) has been a main source of funding for the latter half of the project, for which I am grateful. Additionally, I have received generous travel grants from Ångpanneföreningens Forskningsstiftelse, Frans Georg och Gull Liljenroths stiftelse, E C Ericssons fond and Sten och Lisa Velanders fond. Of course, such debts should not be repaid, but I do aspire to pay them *forward*, by some day joining the ranks of donors

rather than recipients of private research funding. (David Schmidtz wonderfully coined the term *transitive reciprocity* for this phenomenon.)

Thank you all!

Stockholm, August 2012
Ulrik Franke

Papers

Papers included in the thesis

[1] U. Franke, "Optimal IT Service Availability: Shorter Outages, or Fewer?" *Network and Service Management, IEEE Transactions on*, vol. 9, no. 1, pp. 22–33, Mar. 2012, DOI: 10.1109/TNSM.2011.110811.110122.

[2] U. Franke, W. Rocha Flores, and P. Johnson, "Enterprise architecture dependency analysis using fault trees and Bayesian networks," in *Proc. 42nd Annual Simulation Symposium (ANSS)*, San Diego, California, March 23-25 2009, pp. 209–216.

[3] U. Franke, P. Johnson, J. König, and L. Marcks von Würtemberg, "Availability of enterprise IT systems: an expert-based Bayesian framework," *Software Quality Journal*, vol. 20, pp. 369–394, 2012, DOI: 10.1007/s11219-011-9141-z.

[4] P. Närman, U. Franke, J. König, M. Buschle, and M. Ekstedt, "Enterprise architecture availability analysis using fault trees and stakeholder interviews," *Enterprise Information Systems*, 2012, DOI: 10.1080/17517575.2011.647092.

[5] U. Franke, P. Johnson, and J. König, "An architecture framework for enterprise IT service availability analysis," 2012, submitted manuscript.

Author contributions

[1] was fully authored by Franke.

In [2], the general research concept is due to Franke and Johnson, whereas the article was mostly authored by Franke and Rocha Flores, and presented by Franke at the conference.

In [3], the general research concept is due to Franke and Johnson, the construction of the survey was mostly done by Franke and König, and the respondents were found by Franke, König and Marcks von Würtemberg, who also shared most of the authoring.

In [4], the general research concept is due to Närman, Ekstedt, and Franke, the metamodel was constructed by Närman, Franke, Ekstedt, König, and Buschle, the empirical data collection was done by Närman and Franke, and the authoring was mostly done by Närman, Franke, König, and Buschle.

In [5], the general research concept is due to Johnson and Franke, the metamodel construction and P^2AMF implementation was done by Franke and Johnson, and the ITIL operationalization and empirical data collection was done by Franke and König, who also shared most of the authoring.

Related papers not included in the thesis

[6] A. Fazlollahi, U. Franke, and J. Ullberg, "Benefits of Enterprise Integration: Review, Classification, and Suggestions for Future Research," in *International IFIP Working Conference on Enterprise Interoperability (IWEI 2012)*, Sep. 2012.

[7] H. Holm, T. Sommestad, U. Franke, and M. Ekstedt, "Success rate of remote code execution attacks – expert assessments and observations," *Journal of Universal Computer Science*, vol. 18, no. 6, pp. 732–749, Mar. 2012.

[8] C. Sandels, U. Franke, and L. Nordström, "Vehicle to grid system reference architectures and Monte Carlo simulations," *International Journal of Vehicle Autonomous Systems*, 2011, to appear.

[9] ——, "Vehicle to grid communication Monte Carlo simulations based on Automated Meter Reading reliability," in *Power Systems Computation Conference 2011*, Aug. 2011.

[10] L. Marcks von Würtemberg, U. Franke, R. Lagerström, E. Ericsson, and J. Lilliesköld, "IT project success factors – an experience report," in *Portland International Conference on Management of Engineering and Technology (PICMET)*, Jul. 2011.

[11] H. Holm, T. Sommestad, U. Franke, and M. Ekstedt, "Expert assessment on the probability of successful remote code execution attacks," in *8th International Workshop on Security in Information Systems – WOSIS 2011*, Jun. 2011.

[12] J. König, P. Närman, U. Franke, and L. Nordström, "An extended framework for reliability analysis of ICT for power systems," in *Proceedings of IEEE Power Tech 2011*, Jun. 2011.

[13] J. Saat, R. Winter, U. Franke, R. Lagerström, and M. Ekstedt, "Analysis of IT/business alignment situations as a precondition for the design and engineering of situated IT/business alignment solutions," in *Proceedings of the Hawaii International Conference on System Sciences (HICSS-44)*, Jan. 2011.

[14] U. Franke, R. Lagerström, M. Ekstedt, J. Saat, and R. Winter, "Trends in enterprise architecture practice – a survey," in *Proc. 5th Trends in Enterprise Architecture Research (TEAR2010) workshop*, Nov. 2010.

[15] M. Buschle, J. Ullberg, U. Franke, R. Lagerström, and T. Sommestad, "A tool for enterprise architecture analysis using the PRM formalism," in *CAiSE2010 Forum Post-Proceedings*, Oct. 2010.

[16] U. Franke, O. Holschke, M. Buschle, P. Närman, and J. Rake-Revelant, "IT consolidation – an optimization approach," in *Enterprise Distributed Object Computing Conference Workshops, 2010. EDOCW 2010. 14th*, Oct. 2010.

[17] J. Saat, U. Franke, R. Lagerström, and M. Ekstedt, "Enterprise architecture meta models for IT/business alignment situations," in *Proc. 14th IEEE International EDOC Conference (EDOC 2010)*, Oct. 2010.

[18] C. Sandels, U. Franke, N. Ingvar, L. Nordström, and R. Hamrén, "Vehicle to grid – Monte Carlo simulations for optimal aggregator strategies," in *Proc. 2010 International Conference on Power System Technology (PowerCon 2010)*, Oct. 2010.

[19] M. Jensen, C. Sel, U. Franke, H. Holm, and L. Nordström, "Availability of a SCADA/OMS/DMS system – a case study," in *Proc. IEEE PES Conference on Innovative Smart Grid Technologies Europe*, Oct. 2010.

[20] C. Sandels, U. Franke, N. Ingvar, L. Nordström, and R. Hamrén, "Vehicle to grid – reference architectures for the control markets in Sweden and Germany," in *Proc. IEEE PES Conference on Innovative Smart Grid Technologies Europe*, Oct. 2010.

[21] U. Franke, P. Närman, D. Höök, and J. Lilliesköld, "Factors affecting successful project management of technology-intense projects," in *Proc. Portland International Conference on Management of Engineering & Technology (PICMET) 2010, Bangkok*, Jul. 2010.

[22] J. König, U. Franke, and L. Nordström, "Probabilistic availability analysis of control and automation systems for active distribution networks," in *In proceedings of IEEE PES Transmission and Distribution Conference and Exposition*, Jun. 2010.

[23] M. Buschle, J. Ullberg, U. Franke, R. Lagerström, and T. Sommestad, "A tool for enterprise architecture analysis using the PRM formalism," in *Proc. CAiSE Forum 2010*, vol. 592, Jun. 2010, pp. 68–75, ISSN 1613-0073.

[24] P. Närman, U. Franke, and M. Stolz, "The service-orientation of MODAF – conceptual framework and analysis," in *Proc. 7th bi-annual European Systems Engineering Conference (EuSEC 2010)*, May 2010.

[25] J. Ullberg, U. Franke, M. Buschle, and P. Johnson, "A tool for interoperability analysis of enterprise architecture models using pi-OCL," in *Proceedings of The international conference on Interoperability for Enterprise Software and Applications (I-ESA)*, Apr. 2010.

[26] U. Franke, P. Johnson, J. König, and L. M. von Würtemberg, "Availability of enterprise IT systems – an expert-based Bayesian model," in *Proc. Fourth International Workshop on Software Quality and Maintainability (SQM 2010), Madrid*, Mar. 2010.

[27] R. Lagerström, U. Franke, P. Johnson, and J. Ullberg, "A method for creating enterprise architecture metamodels – applied to systems modifiability analysis," *International Journal of Computer Science & Applications*, vol. VI, pp. 89–120, Dec. 2009.

[28] U. Franke, J. Ullberg, T. Sommestad, R. Lagerström, and P. Johnson, "Decision support oriented enterprise architecture metamodel management using classification trees," in *Enterprise Distributed Object Computing Conference Workshops, 2009. EDOCW 2009. 13th*, Sep. 2009.

[29] U. Franke and P. Johnson, "An enterprise architecture framework for application consolidation in the Swedish armed forces," in *Enterprise Distributed Object Computing Conference Workshops, 2009. EDOCW 2009. 13th*, Sep. 2009.

[30] S. Aier, S. Buckl, U. Franke, B. Gleichauf, P. Johnson, P. Närman, C. M. Schweda, and J. Ullberg, "A survival analysis of application life spans based on enterprise architecture models," in *Lecture Notes in Informatics*, vol. P-152, Sep. 2009, pp. 141–154, proc. 3rd International Workshop on Enterprise Modelling and Information Systems Architectures (EMISA 2009).

[31] P. Gustafsson, D. Höök, U. Franke, and P. Johnson, "Modeling the IT impact on organizational structure," in *Proc. 13th IEEE International EDOC Conference (EDOC 2009)*, Sep. 2009.

[32] S. Buckl, U. Franke, O. Holschke, F. Matthes, C. M. Schweda, T. Sommestad, and J. Ullberg, "A pattern-based approach to quantitative enterprise architecture analysis," in *Proc. 15th Americas Conference on Information Systems (AMCIS), San Francisco, USA*, no. Paper 318, Aug. 2009.

[33] R. Lagerström, J. Saat, U. Franke, S. Aier, and M. Ekstedt, "Enterprise meta modeling methods – combining a stakeholder-oriented and a causality-based approach," in *Enterprise, Business-Process and Information Systems Modeling, Lecture Notes in Business Information Processing*, vol. 29, Springer Berlin Heidelberg, Jun. 2009, pp. 381–393, ISSN 1865-1348.

[34] P. Närman, U. Franke, and L. Nordström, "Assessing the quality of service of Powel," in *Proc. 20th International Conference on Electricity Distribution*, Jun. 2009.

[35] U. Franke, P. Johnson, E. Ericsson, W. R. Flores, and K. Zhu, "Enterprise architecture analysis using fault trees and MODAF," in *Proc. CAiSE Forum 2009*, vol. 453, Jun. 2009, pp. 61–66, ISSN 1613-0073.

[36] U. Franke, D. Höök, J. König, R. Lagerström, P. Närman, J. Ullberg, P. Gustafsson, and M. Ekstedt, "EAF2 – a framework for categorizing enterprise architecture frameworks," in *Proc. 10th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, May 2009, pp. 327–332.

[37] U. Franke, P. Johnson, R. Lagerström, J. Ullberg, D. Höök, M. Ekstedt, and J. König, "A formal method for cost and accuracy trade-off analysis in software assessment measures," in *Proc. 3rd International Conference on Research Challenges in Information Science (RCIS), Fès, Morocco*, Apr. 2009.

[38] P. Närman, P. Johnson, R. Lagerström, U. Franke, and M. Ekstedt, "Data collection prioritization for system quality analysis," in *Electronic Notes in Theoretical Computer Science*, vol. 233, Mar. 2009, pp. 29–42.

[39] U. Franke, P. Johnson, R. Lagerström, J. Ullberg, D. Höök, M. Ekstedt, and J. König, "A method for choosing software assessment measures using Bayesian networks and diagnosis," in *Proc. 13th European Conference on Software Maintenance and Reengineering*, Mar. 2009.

[40] M. Ekstedt, U. Franke, P. Johnson, R. Lagerström, T. Sommestad, J. Ullberg, and M. Buschle, "A tool for enterprise architecture analysis of maintainability," in *Proc. 13th European Conference on Software Maintenance and Reengineering*, Mar. 2009.

[41] U. Franke, T. Sommestad, M. Ekstedt, and P. Johnson, "Defense graphs and enterprise architecture for information assurance analysis," in *Proceedings of the 26th Army Science Conference*, Dec. 2008.

[42] P. Gustafsson, U. Franke, D. Höök, and P. Johnson, "Quantifying IT impacts on organizational structure and business value with extended influence diagrams," in *Springer Lecture Notes in Business Information Processing*, vol. Volume 15, Nov. 2008, pp. 138–152.

[43] P. Närman, U. Franke, and L. Nordström, "Assessing the Quality of Service of Powel's Netbas at a Nordic Utility," in *Nordic Distribution and Asset Management Conference (NORDAC 2008)*, Sep. 2008.

[44] R. Lagerström, M. Chenine, P. Johnson, and U. Franke, "Probabilistic metamodel merging," in *Proceedings of the Forum at the 20th International Conference on Advanced Information Systems*, vol. 344, Jun. 2008, pp. 25–28.

[45] P. Gustafsson, U. Franke, P. Johnson, and J. Lilliesköld, "Identifying IT impacts on organizational structure and business value," in *Proceedings of the Third International Workshop on Business/IT Alignment and Interoperability*, vol. 344, Jun. 2008, pp. 44–57.

Table of contents

Part I

Introduction

Chapter 1

Introduction

1.1 Outline of the thesis

This thesis consists of two parts. The first part is an introduction to the subject, research question, methods and results presented in greater detail in the second part, which is the locus of the main contribution. The second part consists of five papers that have either been published in (papers 1, 3, and 4), or submitted to (paper 5) peer-reviewed academic journals, or presented at and published in the proceedings of a peer-reviewed academic conference (paper 2).

1.2 Background

In the modern world, it is becoming increasingly difficult to imagine life without IT systems. Not only are computers becoming ever smaller, faster and more pervasive in a physical sense, but they also increasingly affect how organizations are set up and how activities are organized. While this has brought about many advantages, such as the automation of manual labor, economies of scale in information management, and decreasing transaction costs for businesses and consumers, it has has also added new challenges, such as the complexity of increasingly integrated systems, the life-cycle management of long-living solutions, and the contingency planning for outages. The last challenge is importantly connected to this thesis: with technology dependence comes the need for continuous and uninterrupted IT service availability.

As businesses increasingly come to rely on IT services, the requirements on availability levels continue to increase [136]. Nevertheless, the understanding of the cost-benefit relation between a desired availability level and its associated cost is often wanting [34]. This is partly due to a lack of maturity on the part of businesses, but also to a lack of proper investigations into the costs of downtime. One often cited source is an IBM study from the nineties reporting that American companies lost $4.54 billion in 1996 due to lost system outages [65]. While there is a consensus that the costs of downtime are generally on the rise, it is also important to understand that this does not justify any amount of spending on mitigation [135]. What is required is an enlightened trade-off between the costs and benefits of availability improvements. This is a key topic of this thesis.

1.3 Enterprise IT service availability

A suitable point of departure for a study of enterprise IT service availability is to survey the meaning of the term.

IT services

Starting off with the notion of an *IT service* (or just service, for short), Taylor et al., in the ITIL series of publications, give the following definition of an IT service [145]:

> "A Service provided to one or more Customers by an IT Service Provider. An IT Service is based on the use of Information Technology and supports the Customer's Business Processes. An IT Service is made up from a combination of people, Processes, and technology and should be defined in a Service Level Agreement."

Though the ITIL framework and its definitions are widely adopted among practitioners, this definition is slightly enigmatic. First of all, it immediately calls for further definitions of customers and service providers, respectively. The corresponding ITIL definition of *customer* reads as follows [145]:

> "Someone who buys goods or Services. The Customer of an IT Service Provider is the person or group that defines and agrees the Service Level Targets. The term Customer is also sometimes informally used to mean Users, for example 'this is a Customer-focused Organization'."

For *service provider*, we have [145]:

> "An Organization supplying Services to one or more Internal Customers or External Customers. Service Provider is often used as an abbreviation for IT Service Provider."

Taken together, it is clear that the ITIL definitions of customers and service providers are quite empty – they rely heavily on a previous conception of the service concept to be properly (or at all) understood. (ITIL alone should not be faulted for its circular definitions, though. The International Standardization Organization offers a similarly unenlightening definition in the ISO/IEC 20000 (Information technology – Service management) standard where "service provider" is defined as "the organization aiming to achieve ISO/IEC 20000".) The kind of informal background understanding of IT services required by the ITIL definition can be articulated as follows (from Johnson and Ekstedt [78]):

> "Although system users might sometimes feel that the systems fail to deliver, the information systems in a company are there to provide value to the business. Even when successful, however, the information systems themselves need support to continue delivering services to its users. As briefly mentioned in the previous chapter there is thus a causal flow from the IT organization through the information systems to the business [...]"

This kind of description makes it easier to understand what an IT service is. Importantly, it is not only about technology, but about technology in an organizational setting, where it delivers some kind of value to whatever that organization is doing. Indeed, it could be argued that the main point of the ITIL definition is its trichotomy of people, processes and technology – highlighting that a service is more than the technology upon which it is built.

The causal flow of Johnson and Ekstedt is useful to understand how an information system (a piece of technology) relates to the customer's business process: even if it is a necessary precondition for a successful service, it is not a sufficient one. This causal flow is important for the purpose of this thesis, as we consider IT – and its availability – in its business operations context. Unavailability that does not impact a customer's business process is of no consequence or concern in this setting. This is also importantly connected to the next term to be scrutinized, viz. *enterprise* IT. The IT service concept will also be revisited in section 1.6 (related work).

Enterprise IT

The focus of this thesis is *enterprise* IT services, not IT services in general. So how do enterprise services, enterprise software, and enterprise computing differ from the non-enterprise counterparts? Turning to industry practice first, the renowned consultancy firm Gartner defines enterprise applications as follows [46]:

> "Software products designed to integrate computer systems that run all phases of an enterprise's operations to facilitate cooperation and coordination of work across the enterprise. The intent is to integrate core business processes (e.g., sales, accounting, finance, human resources, inventory and manufacturing). The ideal enterprise system could control all major business processes in real time via a single software architecture on a client/server platform. Enterprise software is expanding its scope to link the enterprise with suppliers, business partners and customers."

Fowler offers a good ostensive definition (i.e. definition "by pointing") of enterprise applications [41]:

> "Enterprise applications include payroll, patient records, shipping tracking, cost analysis, credit scoring, insurance, supply chain, accounting, customer service, and foreign exchange trading. Enterprise applications don't include automobile fuel injection, word processors, elevator controllers, chemical plant controllers, telephone switches, operating systems, compilers, and games."

However, the simplicity of Fowler's definition is a mixed blessing. His exclusion list indeed includes systems that arguable can be thought of as enterprise systems. A chemical plant controller in the sense of a Supervisory Control And Data Acquisition (SCADA) system is, arguably, an enterprise application, whereas a chemical plant controller that merely regulates the flow in a pump certainly is not. Similarly, a telephone switch that is interconnected to an office network, integrated with an enterprise telephone directory and the calendars of the office clerks is, arguably, an enterprise application, whereas software that merely routes calls on a switchboard is not. To accommodate such distinctions, an ostensive definition is not enough. Johnson defines enterprise software systems in the following manner [77]:

> "An enterprise software system is the interconnected set of systems that is owned and managed by organizations whose primary interest is to use rather than develop the systems. Typical *components* in enterprise software systems are thus considered as proper *systems* in most other cases. They bear names such as process control systems, billing systems, customer information systems, and geographical information systems." (Emphasis in original.)

This wording accurately describes the definition adhered to in this thesis. In particular, the use-not-develop concept is useful – in the enterprise context, IT is a tool to be used, and the less it has to be developed the better. Similarly to Gartner's definition, Johnson goes on to highlight the increasing importance of *integration*, and its consequences for enterprise systems (or services) management:

> "In the early days, these components were separated from each other physically, logically, and managerially. During the last decades, however, an ever-increasing integration frenzy has gripped the enterprises of the computerized world. Today's enterprise software system is thus multi-vendor based and enterprise-wide, characterized by heterogeneous and large-grained components."

This integration aspect resolves the chemical plant controller and telephone switch ambiguities in an elegant manner.

Furthermore, this heterogeneity and growing complexity are key driving forces behind the advent of *enterprise architecture* (cf. next section, where the enterprise concept is revisited), a discipline aiming to holistically address the management problems of today's complex enterprise computing environments. Looking at the scope of one of the leading academic conferences in the area – the IEEE Enterprise Distributed Object Computing Conference (EDOC), the same lines of thought can be seen [21]:

> "The IEEE EDOC Conference emphasizes a holistic view on enterprise applications engineering and management, fostering integrated approaches that can address and relate processes, people and technology. The themes of openness and distributed computing, based on services, components and objects, provide a useful and unifying conceptual framework."

Note that the processes, people and technology trichotomy familiar from the ITIL service definition reappears here as well. *Enterprise Information Systems* is an academic journal published by Taylor & Francis. In its aims and scope, a similar concern with integration can be seen [35]:

> "[...] the Journal focuses on both the technical and application aspects of enterprise information systems technology, and the complex and cross-disciplinary problems of enterprise integration that arise in integrating extended enterprises in a contemporary global supply chain environment. Techniques developed in mathematical science, computer science, manufacturing engineering, operations management used in the design or operation of enterprise information systems will also be considered."

This mission statement also hints to the multitude of methodologies that can be deemed relevant in the field of enterprise IT.

Availability

The literature offers several relevant definitions of *availability*. Schmidt makes the following definition [134]:

> "Availability is the measure of how often or how long a service or a system is available for use."

This gives an informal feeling for the term, but as a definition it is vague in that it leaves the "how often or how long" question open. The International Telecommunications Union is a bit more precise, defining availability as [72]:

> "the ability of an item to be in a state to perform a required function at a given instant of time, or at any instant of time within a given time interval, assuming that the external resources, if required, are provided."

Here, availability is specified to a given time interval, and some preconditions are stated. Indeed, the proviso "assuming that the external resources, if required, are provided" offers an important insight into why availability work can be a source of conflict in an organization: under this definition, determining whether a service is available requires common pictures both of what is required and what has been provided in any given case. (See Barki and Hartwick [6] for an investigation of conflicts in information system development.) Rausand and Høyland, following British Standard BS4778, make the following definition [127]:

> "The ability of an item (under combined aspects of its reliability, maintainability and maintenance support) to perform its required function at a stated instant of time or over a stated period of time."

This definition offers a choice: whether to measure availability at an instant or over a period of time. This distinction will be revisited in a moment. The International Organization for Standardization makes the following definition [68] in the ISO/IEC 9126 standard (Software engineering – Product quality), part 1:

> "Availability is the capability of the software product to be in a state to perform a required function at a given point in time, under stated conditions of use."

In 2011, ISO/IEC 9126 was superseded by ISO/IEC 25010 (Systems and software engineering – Systems and software Quality Requirements and Evaluation (SquaRE) – System and software quality models), which defines availability as the [71]:

> "degree to which a system, product or component is operational and accessible when required for use"

Yet another useful ISO/IEC definition is found in ISO/IEC 20000 (Information technology – Service management), where availability is defined as the [70]:

> "ability of a component or service to perform its required function at a stated instant or over a stated period of time"

The ISO/IEC definitions are all very similar. Perhaps their most important difference resides in the definition of the relevant time window: Again, the given point (of ISO/IEC 9126) is contrasted with the instant/period options (of ISO/IEC 20000). The less formal but quite demanding "when required for use" is added by ISO/IEC 25010. To resolve the issue of measurement window, we follow Milanovic and distinguish the *steady state* availability from *instantaneous availability*, defined as "the probability that the system is operational (delivers the satisfactory service) at a given time instant" [108]. The steady state availability is defined mathematically in the following fashion:

$$A = \frac{\text{MTTF}}{\text{MTTF} + \text{MTTR}} \tag{1.1}$$

MTTF denotes "Mean Time To Failure" and MTTR "Mean Time To Repair" or "Mean Time To Restore", respectively. The term MTBF, "Mean Time Between Failures", is sometimes used for repairable systems (i.e. when there can be several failures). In this thesis MTTF and MTBF are used interchangeably, as only repairable systems (enterprise IT services) are considered. It should be emphasized that since *mean* times are used, Eq. 1.1

measures the long-term performance of a system, i.e. the *steady state* system availability. In this thesis, availability refers to steady state availability, unless explicitly stated otherwise.

Indeed, this is precisely the operationalization prescribed by ISO/IEC 25010, revealing the meaning of "when required for use" [71]:

> "Externally, availability can be assessed by the proportion of total time during which the system, product or component is in an up state. Availability is therefore a combination of maturity (which governs the frequency of failure), fault tolerance and recoverability (which governs the length of down time following each failure)."

The second sentence is a more verbose explanation of the simple mathematical observation that to improve availability, as defined by Eq. 1.1, there are two strategies: Increase MTTF or decrease MTTR. While increasing MTTF is the typical traditional strategy of reliability engineering, some have argued that decreasing MTTR is actually a more cost-effective way of providing the desired availability [4].

However, one more conceptual aspect needs to be mentioned. The point can be illustrated by the ITIL definition of availability [149]:

> "Ability of a Configuration Item or IT Service to perform its agreed Function when required."

It comes along with an operationalization [145]:

> "It is Best Practice to calculate Availability using measurements of the Business output of the IT Service."

In the enterprise IT service context, this operationalization is important. Remembering the causal flow of Johnson and Ekstedt, it is not enough that particular databases, network connections or servers can be shown to be available – what matters is the availability of the service as whole, the so-called *end-to-end* availability. As pointed out by Gartner, achieving any given end-to-end service availability level requires a substantially higher average availability level of the constituent components [136]. Of course, this is not to say that measuring component availability levels is useless. However, such measurements need to be aggregated to be appropriate for the enterprise IT service context, e.g. as done in paper 4.

To summarize, the availability addressed in this thesis is the steady state availability of enterprise IT services, as measured by their business output.

However, before moving on, it is useful to contrast availability with the closely related, yet different, notion of *reliability*. Rausand and Høyland (following ISO/IEC 8402) define it as follows [127]:

> "The ability of an item to perform a required function, under given environmental and operational conditions and for a stated period of time."

ISO/IEC 25010 defines reliability as the [71]:

> "degree to which a system, product or component performs specified functions under specified conditions for a specified period of time"

These definitions are confusingly similar to some of the definitions of availability given above. Milanovic explains the difference pedagogically: Reliability is about failure-free operations; about the probability that no failure occurs during a time-interval. Availability, on the other hand, allows for systems to fail and then be repaired again. Only for non-repairable systems

are the two equivalent [108]. Gray makes the point very succinctly [57]: "Reliability and availability are different: Availability is doing the right thing within the specified response time. Reliability is not doing the wrong thing." In the excellent wording of Goyal et al., written 25 years ago [55]:

> "System availability is becoming an increasingly important factor in evaluating the behavior of commercial computer systems. This is due to the increased dependence of enterprises on continuously operating computer systems and to the emphasis on fault-tolerant designs. Thus, we expect availability modeling to be of increasing interest to computer system analysts and for performance models and availability models to be used to evaluate combined performance/availability (performability) measures. Since commercial computer systems are repairable, availability measures are of greater interest than reliability measures."

It should be noted that some of these confusing reliability definitions have been criticized. Immonen and Niemelä explain the apparent lack of studies on availability [67]:

> "One reason is the confusing definitions of the ISO/IEC 9126-1 quality model [66] that defines reliability as the capability of a software system to maintain a specified level of performance when used under the specified conditions. According to the quality model, reliability is mixed with performance, and availability is a sub-characteristic of reliability."

Regrettably, the unfortunate characterization of availability as a sub-characteristic of reliability is retained in ISO/IEC 25010, that superseded ISO/IEC 9126 in 2011 [71].

1.4 Enterprise architecture

At the intersection of information technology and business operations, new planning and evaluation tools are needed to keep track of the bigger picture. In this area, the long established notions of *software architecture* and *systems architecture* are now being accompanied by the new notion of *enterprise architecture* (EA). EA does not replace, but rather complements, previous descriptions of information systems. It assumes a new level of abstraction, much like a city plan does not replace, but rather complements, the technical drawing of the individual house. At this level of abstraction, abstract properties of information systems such as availability, modifiability, security and interoperability become increasingly important to understand, in order to be able to effectively manage the landscape of IT and business activities. Recall the distinction made by Johnson about the components in enterprise software systems being considered full-scale systems in other contexts [77]. This is helpful for our understanding of the proper abstraction level of enterprise architecture: its components certainly can be opened up and considered in greater detail – but that would entail losing the bigger picture.

The science and practice of enterprise architecture have evolved in concert over the past two decades, trying to find an appropriate granularity for describing the technology, organization and business processes of enterprises in a single coherent fashion. A typical feature of enterprise architecture is the use of *models*, made up of *entities*, related by *relationships*, and equipped with *attributes* describing their properties [78, 91, 17]. Looking at a few definitions of enterprise architecture, some similarities and differences can be found:

> "The formal description of the structure and function of the components of an enterprise, their inter-relationships, and the principles and guidelines governing

> their design and evolution over time. (Note: 'Components of the enterprise' can be any elements that go to make up the enterprise and can include people, processes and physical structures as well as engineering and information systems.)" (Ministry of Defence, [109])

(Note how this definition echoes the service definition from above, i.e. the trichotomy of people, processes, and technology – broadly construed.)

> "A strategic information asset base, which defines the business, the information necessary to operate the business, the technologies necessary to support the business operations, and the transitional processes necessary for implementing new technologies in response to the changing business needs. It is a representation or blueprint" (CIO Council, [24])

> "A rigorous description of the structure of an enterprise, its decomposition into subsystems, the relationships between the subsystems, the relationships with the external environment, the terminology to use, and the guiding principles for the design and evolution of an enterprise" (Giachetti, [49])

> "fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution" (Hilliard, [62])

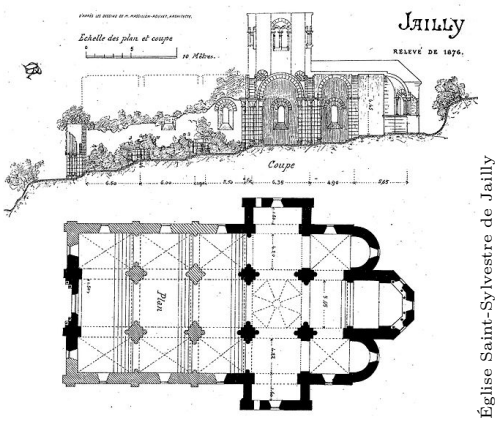
The TOGAF framework embraces the IEEE (Hilliard) definition, but argues that "architecture" has two different, context-dependent, meanings:

> "1. A formal description of a system, or a detailed plan of the system at component level to guide its implementation
> 2. The structure of components, their inter-relationships, and the principles and guidelines governing their design and evolution over time"
>
> (The Open Group, [150])

A more action-oriented definition is offered by the Gartner consultancy [46]:

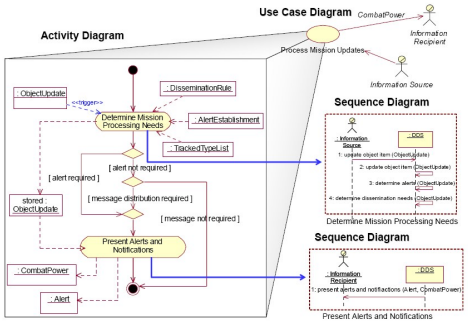
> "Enterprise architecture (EA) is the process of translating business vision and strategy into effective enterprise change by creating, communicating and improving the key requirements, principles and models that describe the enterprise's future state and enable its evolution.
>
> The scope of the enterprise architecture includes the people, processes, information and technology of the enterprise, and their relationships to one another and to the external environment. Enterprise architects compose holistic solutions that address the business challenges of the enterprise and support the governance needed to implement them."

The thesis is part of an ongoing enterprise architecture research program at the department of Industrial Information and Control Systems, where non-functional qualities of information systems are analyzed using enterprise architecture methods. A key feature of this program is the role of quantitative *analysis*. This quantitative strain of enterprise architecture aims for its models to allow predictions about the behavior of future architectures. Thus, rather than using trial and error to govern and modify enterprise information systems, decision-makers can get model-based decision-support beforehand [80].



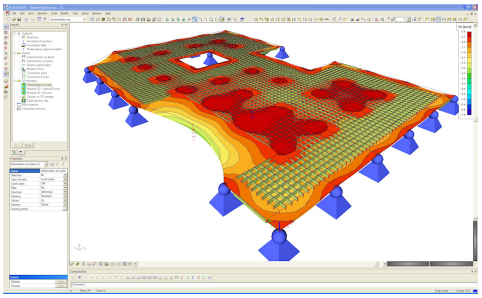
Église Saint-Sylvestre de Jailly
From Wikimedia Commons

(a) Traditional architecture



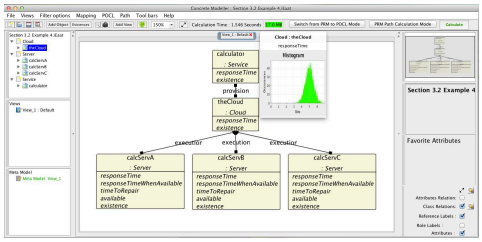
DoDAF OV-5 example
From Wikimedia Commons

(b) Traditional EA model



Finite element model
From Wikimedia Commons

(c) Computer aided engineering



The KTH EA^{2}T tool

(d) EA analysis model

Figure 1.1: From documentation and communication to analysis and explanation – an analogy: 1.1c is to 1.1a as 1.1d is to 1.1b.

Enterprise architecture models serve several purposes, e.g. (i) documentation and communication, (ii) analysis and explanation, and (iii) design [88]. While the quantitative EA analysis program that this thesis is a part of emphasizes the second purpose, mainstream EA is still more focused on the first. Documentation and communication is certainly important and should not be denigrated. However, the aim of EA analysis is to take the additional leap of processing and visualizing the information inherent in an architectural description in such a way that non-trivial properties, that cannot be assessed at first sight, emerge. Conceptually, this is equivalent to the leap taken by traditional architecture from drawings indicating how many columns support a roof and where they are placed to computer aided design (CAD) and computer aided engineering (CAE) models that offer automatic calculations not only of column loads and buckling, but also of acoustics and lighting.

The analogy extends to the underlying theoretical foundations: creating a sophisticated CAE tool is not only a matter of software engineering, but requires scientific understanding of solid mechanics, acoustics and optics. Similarly, EA analysis requires not only application of modeling principles, but also domain-specific theory. Thus forecasting the performance of services and databases might require models from queuing theory and forecasting their availability might require a fault tree-like dependency structure. An EA analysis model of

enterprise IT service availability thus requires both finding adequate theoretical underpinnings and integrating them into a suitable modeling formalism.

The EA analysis program was first outlined by Johnson and Ekstedt in 2007 [78]. Two PhD theses have been published so far by Lagerström on modifiability [90] and Närman on on multiple non-functional system qualities [114]. Forthcoming theses from the program include Sommestad on cyber security [142, 143], Ullberg on interoperability [152], and König on reliability [85].

1.5 Purpose of the thesis

The purpose of this thesis is to offer support for decision-making related to the availability of enterprise IT services. As described above, such services span not only traditional software architecture, but rather a combination of technology with processes and people. Therefore, it is suitable to make enterprise architecture models – that are designed to span at least the technology and processes parts – a key part of the approach. Such enterprise architecture models should support analysis of future scenarios and offer estimates of how different courses of actions would impact the availability of the services within the decision-maker's domain. More specifically, therefore, the main purpose of the thesis is:

- To develop a method for enterprise IT service availability analysis using enterprise architecture models.

To achieve this, the method needs to appropriately reflect modern availability management practices (including outsourcing of services that are controlled only through Service Level Agreements), it needs to be based on an adequate understanding of the factors affecting availability, it needs to be properly formalized and it needs to be empirically investigated and validated. Thus, three important subgoals can be discerned:

1. To investigate the causal factors contributing to availability levels.
2. To create an enterprise architecture metamodel for availability analysis and prediction.
3. To empirically test, investigate and validate the approach.

Delimitations

According to the ITIL definition cited above, an (enterprise) IT service is made up from a combination of people, processes, and technology. While most previous work on enterprise IT service availability tends to focus on technology, this thesis also includes the process perspective. However, the people component has been excluded from the constituent papers, in order not to over-extend the research scope.

Furthermore, this thesis is delimited to steady state availability, rather than instantaneous availability. It is reasonable to assume that when studying enterprise IT service availability from an EA perspective (looking at the "bigger picture" of structure and processes), long-term averages changes can be tracked, explained and predicted in terms of process maturity, whereas instantaneous outages need to be predicted with other means (if they can be predicted at all).

1.6 Related work

Surveying the scientific literature on availability analysis methods in 2008, Immonen and Niemelä find it surprisingly scarce [67]:

> "The literature that addresses the availability analysis methods is very scarce; except two methods [57,58]. The availability analysis has not been studied, or at least, we could not find any evidence."

As noted above, they attribute this lack of literature partly to the confusing definition of the ISO/IEC 9126 standard. Still, they note the importance of such analysis and call for more research [67]:

> "However, future software systems are service oriented and the availability of services is a critical quality factor for systems end-users. Therefore, availability is an issue that requires specific consideration, extensive research, and development of the appropriate analysis methods, techniques, and tools."

As explained in Section 1.5, availability analysis is at the heart of this thesis. However, even if there is precious little literature on *availability analysis*, there is still a lot of related work on reliability and availability. In the following review, a few of the most important lines of thought are described and contrasted with the approach of this thesis.

From hardware to software reliability

Functioning hardware is the most fundamental precondition for working enterprise IT services. In the early days of computing, hardware failure was a major source of outages. However, the impact of hardware failure on the availability of enterprise IT systems and services has been steadily decreasing since then [47]. Already in 1985, Gray showed that IT administration and software are the major contributors to failure [57], and in a later work Gray shows that the sources of failures have changed from hardware to software over the 1985-1990 time period [58]. Similarly, Pertet and Narasimhan investigate the causes and prevalence of failure in web applications, concluding that software failures and human error account for about 80% of failures [124]. A similar (but not identical) claim is made by Malik and Scott, who assert that people and process failures cause approximately 80% of "mission-critical" downtime [99].

The increasing relative importance of software over hardware spawned the well-researched field of *software reliability*. Software reliability differs importantly from hardware reliability. As it is succinctly put in ISO/IEC 25010 [71]:

> "Wear does not occur in software. Limitations in reliability are due to faults in requirements, design and implementation, or due to contextual changes."

Software reliability models play a key role in this field. *Probabilistic* models include *failure rate models* that describe how failure rates change as a function of the remaining faults in a program. Well-known failure rate models include the Jelinski-Moranda model [74] from 1972, the Schick-Wolverton model [133] from 1978, the Jelinski-Moranda geometric model [111] from 1981, the Moranda geometric Poisson model [110] from 1975, the negative-binomial model, the modified Schick-Wolverton model [144] from 1977, and the Goel-Okumoto imperfect debugging model [51] from 1979. *Reliability growth models* such as Coutinho's [28] from 1973 or that of Wall and Ferguson [157] from 1977 explain the reliability improvement throughout testing and debugging. *None-homogeneous Poisson process*

(NHPP) models describe reliability by letting an NHPP fit the number of failures experienced up until a certain time t. Models include the Musa exponential model [113] from 1987, S-shaped growth [163] from 1983 and generalized NHPP [126] from 1997. There are also *deterministic* models, the most famous of which are probably the Halstead [59] (1975) and McCabe [102] (1976) measures of complexity. (The taxonomy of software reliability models partly re-used in this section is due to Pham, who also offers brief model descriptions [125].)

New technologies also push for new analysis methods. The advent of peer-to-peer file sharing systems prompted Bhagwan et al. to model users periodically leaving and joining the system as intermittently available components [13], and others to similarly analyze availability in terms of end-user behavior [132, 23, 31].

Another strand of software reliability deals with causes of failure. This is a diverse strand, ranging from general considerations on what causes systems to go down [156] over well-delimited particular subjects such as the reliability of Windows 2000 [112] to high-impact events such as large-scale Internet failures [120]. There is also a methodological diversity. For example, Zhang and Pham present an effort to identify factors affecting software reliability by surveys directed to stakeholders in software development or research [170].

Over the decades, numerous review articles have been written on software reliability methods and models, e.g. Shanthikumar [139], Goel [50] and Cai et al. [19]. However, despite prolific theory development, practical applicability remains limited. Thus Koziolek et al. (2010) summarize their research [87]:

> "Although researchers have proposed more than 20 methods in this area, empirical case studies applying these methods on large-scale industrial systems are rare. The costs and benefits of these methods remain unknown. [...] We found that architecture-based software reliability analysis is still difficult to apply and that more effective data collection techniques are required."

Gokhale (2007) identifies related limitations in architecture-based software reliability analysis, since existing techniques [53]:

> "1. cannot consider many characteristics and aspects that are commonly present in modern software applications (for example, existing techniques cannot consider concurrent execution of components),
> 2. provide limited analysis capabilities,
> 3. cannot be used to analyze the reliability of real software applications due to the lack of parameter estimation techniques, and
> 4. are not validated experimentally."

On a similarly skeptical note, Milanovic (2010) concludes [108]:

> "Software reliability measures can at present be achieved with pretty good accuracy if programming team has a substantial track data and lots of reliability data to support it, which is rarely the case. As no standard or widely accepted reliability model exists, curve fitting seems to be most popular in practice."

Of course, the difficulty to obtain accurate data for reliability models has been addressed [54]. However, there is a more profound reason why software reliability is not at the forefront of this thesis, related to its scope of investigation. To see why this is so, consider the work by Gokhale and Mullen, a thorough investigation of software repair rates based on

more than 10 000 software defect repair times collected from Cisco systems [52]. Recalling the definitions given in Section 1.2, it would seem that joining the work of Gokhale and Mullen on repair times with the large existing body of knowledge on software failure times, good availability models would be achieved. However, the usefulness of such an availability measure is highly limited. Gokhale and Mullen investigate finding and fixing bugs in software. Finding and fixing such bugs can take days, weeks, or months, but businesses restoring their IT services on those time scales go out of business. The discrepancy is easy to understand: to get a service back on-line, it is not necessary to repair the bug in a software engineering sense. Indeed, probably the most common service restore is a rollback – simply going back to the last working configuration. Any bugs in the flawed version can then be found and fixed off-line. This is the reason why applying Eq. 1.1 to software failure and repair rates as they are mostly investigated in the software reliability literature is not a very relevant availability measure for any business with high availability requirements.

Service availability

The limitations of software reliability research, together with the advent of the service-oriented paradigm in software engineering [37] brought about a new way to look at availability. In fora such as the International Service Availability Symposium mentioned above, availability started to be analyzed not from the perspective of software failure, but rather from that of the *consequences* of service failure. The program chair's message at the first symposium (2005) sets the tone [97]:

> "No matter whether we call the computing services of the future "autonomic," "trustworthy" or simply "reliable/available" the fact of the matter is that they will have to be there seven days a week, 24 hours a day, independent of the environment, location and the mode of use or the education level of the user. This is an ambitious challenge which will have to be met. Service availability cannot be compromised; it will have to be delivered. The economic impact of unreliable, incorrect services is simply unpredictable."

The research challenge thus outlined is quite close to the scope of this thesis.

Within this paradigm, some important strands can be identified. As technologies important to enterprise IT services, the availability of databases [32, 38, 161, 165] and middleware [128, 117, 122, 105] have received a lot of attention. While these subjects were also studied within the software reliability paradigm, external services that fail [115] and service composition [140, 33, 107] – particularly quality of service-aware service composition [168, 167, 104] – are topics more native to the perspective of service availability. So are the interest in user perceived availability [162, 82, 158, 151, 121] and the SLA management perspective on availability [137, 11, 5, 160].

On the method side, the service-oriented paradigm makes the *architecture* a useful level of analysis [131, 86, 89, 67, 39, 103], reflecting the fact that architectures composed from constituent services are subject to change. This is closely related to the area of component-based reliability assessment [30, 60, 56] and more recently component-based availability modeling [20, 106].

Milanovic offers what is probably the most comprehensive survey to date of tools for availability assessment in the service-oriented paradigm [108], comparing more than 60 commercial, public domain and academic tools. It is worth quoting the result, as it is closely linked to the difference between traditional software systems and the modern service-oriented enterprise systems:

> "Based on this part of the study, the following conclusion was drawn: existing methods and tools are powerful, but difficult to apply directly for modeling service availability. Historical development of availability models and tools has associated them with mission-critical systems that rarely change during their lifetime, such as real-time systems, avionics, or telecommunication circuits. The availability assessment procedures they offer are unable to adapt to fast-paced changes in modern SOA [service-oriented architecture] systems. Each time the IT infrastructure or business process changes, it is required to manually intervene and update, verify and evaluate availability model."

This is the raison d'être for architecture-based availability analysis in the service-oriented paradigm. Architectures need to be quickly and accurately (i.e. automatically) created and then used to compute the availability of the design alternative described. Indeed, this is closely related to the model-driven paradigm in software engineering, and the literature is abundant with model-oriented assessment formalisms for availability and reliability. A model for predicting the reliability of future systems based on component reliability, represented in UML, is proposed in a series of papers by Singh et al. [141, 27, 26]. Along related lines, Bocciarelli and D'Ambrogio use web services described in the business process execution language (BPEL) to create UML models annotated with reliability data so that the reliability characteristics of composite services can be computed [16]. Other work in the same area include Leangsuksun et al., Rodrigues, Bernardi and Merseguer and Majzik et al., who all offer methods that integrate system design aspects as expressed in UML models with computational formalisms that enable the prediction of the reliability of software systems that are still in the design stage [92, 130, 12, 96]. Unlike this thesis, these approaches do not account for the governance aspects of service availability, e.g. the impact of IT service management process maturities. The same holds for Immonen, who uses a design-stage simulation to detect relations between system components that impact the reliability of the execution paths of the final architecture [66].

Milanovic's work is also similar. Following his survey of available tools and formalisms, he offers an architecture-based availability assessment tool, where an availability model (fault tree or Markov model) is generated from an architecture description of the underlying ICT infrastructure, populated with data from an infrastructure data repository and solved with respect to a business process modeled in BPMN or a similar language [108]. This is state of the art architecture-based availability analysis, and quite similar to this thesis. The main difference is that whereas Milanovic limits his model to assessing availability in terms of the underlying ICT infrastructure, our aim is to also take some account of how IT service management processes affect service availability. These factors cannot be localized to any particular place in the service architecture, but nevertheless they impact its availability. In this sense, this thesis is also somewhat similar to design guides for high availability such as those offered by Vargas [154, 153], Holub [64] and Kalinsky [83] or general availability risk management models such as that offered by Zambon et al. [166].

Availability in ever-changing enterprise environments

The service availability research paradigm properly accounts for the dynamics introduced by the ad hoc service composition envisioned by SOA advocates: if a service is dynamically re-configured to include other atomic services, a new architecture for availability evaluation can be automatically generated within a framework such as that proposed by Milanovic. However, there is another kind of dynamics that is unaccounted for. Malek, in a tutorial

delivered at the 2008 International Service Availability Symposium, describes this challenge [98]:

> "Also, due to dynamicity (frequent configurations, reconfigurations, updates, upgrades and patches) an already tested system can transform into a new untested system and additional faults may be inserted."

Indeed, this is not news. Already in 1985, Gray noted that system administration was the main source of failures, accounting for 42% of reported system failures in his study [57]. This is an important reason why methods that consider only a static environment where humans do not intervene will be unable to capture a lot of important causes of unavailability. The service availability paradigm accounts for dynamic reconfiguration of services – but not for the propensity of human error. This thesis attempts to include such outage causes as well, and to reconcile that with the traditional, architectural approach. The Bayesian model used (as presented in Article 3) has some similarities to the work of Zhang et al. [169], but whereras he uses system logs , we base our model on expert judgements.

The impact of system administration puts additional emphasis on the human part of the technology-processes-humans trichotomy. Naturally, the literature in this field is less technology-centered, instead focusing on for instance the importance of communication with and support from senior executives [61, 118, 73, 2], and the use of proper key performance indicators (KPIs) [8, 94]. Paper 1 addresses this issue to some extent, pointing out that the single figure notion of availability is seriously inadequate. However, the human part of the technology-processes-humans trichotomy is mostly excluded from the thesis work.

1.7 Results

This section summarizes the contribution of each of the papers constituting this composite thesis and relates them to the goals identified in Section 1.5 above.

Papers 2, 3, 4, and 5 form the main pillar of the thesis. Paper 2 (2009) introduces an architectural approach to availability modeling, addressing the second subgoal identified in Section 1.5 above. Paper 3 (2011) introduces a causal factor approach to availability modeling, addressing the first subgoal identified in Section 1.5 above. Paper 4 (2012) extends the architectural approach of paper 2, and its five case studies verify that the combination of fault trees and architectural descriptions yields useful and accurate results, addressing the second and third subgoals. Paper 5 (2012) integrates the fault tree and causal factors approaches into a single framework and tests it empirically, addressing the second and third subgoals. This line of work constitutes the long-term direction of the thesis project, aligned with the overall department research program on enterprise architecture.

Paper 1 (2012) was spawned by the realization – during the work on the main pillar – that the single figure notion of availability (e.g. 99.8%) is seriously inadequate.

The following short summaries are based on the abstracts of each of the constituent papers of the thesis.

Paper 1: Optimal IT Service Availability: Shorter Outages, or Fewer?

Even though many companies are beginning to grasp the economic importance of availability management, the implications for management of Service Level Agreements (SLAs) and availability risk management still need more thought. This paper offers a framework for thinking about availability management, highlighting the importance of variance of outage costs. Using simulations on previously existing data sets of revenue data, variance is shown

to be important. From this follows the observation that when outage costs are proportional to outage duration, more but shorter outages should be preferred to fewer but longer, in order to minimize variance. However, sometimes the cost of an outage depends non-linearly on its duration. Using two archetypal cases of such non-linearity, an optimal outage length is derived, along with some guidance for its application in the context of hourly downtime cost variance. The feasibility of the method is evaluated, as is its practitioner relevance, in particular with regard to SLA management.

Paper 2: Enterprise Architecture Dependency Analysis using Fault Trees and Bayesian Networks

Analysis of the dependencies between technical systems and business processes is at the heart of Enterprise Architecture. Nevertheless, most EA models focus on visualizations and qualitative decision-support. This paper builds on two existing quantitative methods for reliability and availability modeling – Fault Tree Analysis (FTA) and Bayesian networks (BN) – and incorporates them into the DoDAF (the Department of Defense Architecture Framework) EA framework, thus enabling automatic dependency analysis. Furthermore, a method for how to adapt DoDAF models for use with FTA and BN is offered. Finally, an example of how to perform dependency analysis on a DoDAF model under different scenarios is given.

Paper 3: Availability of enterprise IT systems – an expert-based Bayesian framework

There are many factors that determine the availability of enterprise IT systems, and their impact on various system architectures is difficult to predict. This paper presents a model of the most such important factors. It is based on expert judgments from 50 experts on IT systems availability that were collected through an electronic questionnaire. The expert answers were used to construct a 16 factor Bayesian leaky Noisy-OR causal model, suitable for evaluating the consequences of various enterprise IT system decision-making scenarios. The great advantage of such a model is that the probable outcomes of decisions can be predicted beforehand, and various scenarios can be experimented with in order to foster prudent decision-making with regard systems availability. Furthermore, a way to integrate the Bayesian model with a reliability block diagram formalism is proposed, thus obtaining a model for assessing availability on the architecture level. Some examples of the use of this unified model for availability modeling are presented.

Paper 4: Enterprise architecture availability analysis using fault trees and stakeholder interviews

The applicability of theories needs to bested in practice. This paper offers a metamodel with an accompanying method for enterprise IT service availability modeling, based on Fault Tree Analysis and constructs from the ArchiMate EA language. Five case studies within the banking and electrical utility industries were performed in order to assess the method in practice. Data was collected using stakeholder interviews, from which the appropriate architecture models were built. These models were then used to calculate annual average availability figures for the services represented. These results were compared with availability figures from SLA reports and system logs. In the five cases thus compared, the annual downtime estimates from the model were all within eight hours from the actual

downtimes. Furthermore, no modeling effort required more than 20 man-hours of work, thus making the method highly applicable in practice.

Paper 5: An architecture framework for enterprise IT service availability analysis

This paper presents an integrated enterprise architecture framework that unites the strands from the previous contributions. The aim of the framework is to facilitate qualitative and quantitative modeling and assessment of enterprise IT service availability. In the first part of the paper, the framework is described, further developing the fault tree concept familiar from papers 2 and 4. However, to also account for the causal model of paper 3, the framework also features a formal computational model written in a probabilistic version of the Object Constraint Language. This model is based on a refined version of the causal model from paper 3, that also takes the structural features of the service architecture into account. The structural modeling is achieved by a metamodel similar to that presented in paper 4. Integrating the complementary fault tree and causal factor approaches to availability modeling is a major contribution of the paper.

The second part of the paper offers an empirical evaluation of the framework. Results from 9 enterprise information system case studies are reported. Using an initial availability baseline, data on how the 14 factors of the model developed over the years is used to create annual availability predictions for each of the modeled services. These figures are compared to the actual outcomes as reported in SLA reports and system logs. The paper ends with a discussion of the practical usefulness of the method based on the outcome of a workshop conducted with the participating enterprises.

1.8 Thesis contribution

This section describes the contribution of the thesis. More details are found in each of the included papers.

Shorter outages, or fewer? Availability is often presented as a single figure (e.g. 99.8%), as is evident from Section 1.2. This obscures the importance of considering both time to failure and time to restore. A given availability level can be achived by many combinations of time to failure and time to restore, non-equivalent to each other. Paper 1 offers a structured way to think about these matters, including a way to find optimal outage durations when the cost of an outage depends non-linearly on its duration.

The causal factors. The causal factors presented in Paper 3 are a unique contribution. Even though the factors themselves are based on the work of Marcus and Stern [101], their relative strengths as determined by the expert questionnaire offer important guidance for how to achieve better availability. This addresses subgoal 1 from section 1.5.

The use of fault trees in enterprise architecture. While fault trees as such are an old and proven method, their use in enterprise architecture modeling has not been studied in-depth before. Papers 2 and 4 illustrate how to integrate fault trees into enterprise architecture models, and empirically validate that such models indeed yield accurate results at a low modeling cost. This addresses subgoals 2 and 3 from section 1.5.

The enterprise architecture framework. While there are many frameworks for availability calculations, few or none share the service abstraction level of the framework presented in Paper 5. Being able to account both for the architectural structure and availability properties of its constituent components, *and* for process maturities that cannot be localized to any particular place in the architecture, this framework offers a unique and action-guiding way to analyze service availability. The framework, integrating the other contributions, addresses all of subgoals 1 through 3 from section 1.5.

1.9 Research design

Research strategies

There is no single way to do research on information systems [7, 155]. Indeed, this diversity has been embraced and itself proposed as a useful source of good research strategies [84, 44], but has also received some criticism [9]. Availability is no exception, and the section on related work (1.6) offers a wealth of methods.

To understand the methodological relation between the papers of this thesis, it is useful to consider the description of the nature of operations research given by Hillier and Lieberman, explaining how the scientific method is applied to operational and management problems [63]:

1. Carefully observe and formulate the problem
2. Construct a scientific (typically mathematical) model that captures the essence of the real problem
3. Hypothesize that the model is precise enough that its solutions are also valid for the real problem
4. Test this hypothesis and modify it as needed
5. Verify some form of the hypothesis

This method rather well approximates the method adopted by the EA analysis research program [78] of which this thesis is a part.

Paper 1 does not make an empirical contribution. Instead, its contribution resides in steps 1-3 of the Hillier and Lieberman process. The problem is formulated based on the definition of availability (Eq. 1.1), and a mathematical model is constructed from which inferences about the proper trade-off between shorter outages and fewer can be made. The use of empirical data (taken from available third party sources) merely illustrates the deductive reasoning, justifying that the model is precise enough that its solutions are also valid for the real problem. Testing and verification, however, remains for future work.

Paper 2 has a similar, deductive, approach. A mathematical model is constructed for axiomatic reasoning in the form of fault trees applied to architectural descriptions of large systems. A hypothesis is made that the model is precise enough that its solutions are also valid for the real problem. Even though paper 2 itself stops there, the hypothesis that the combination of fault trees with architectural descriptions yields useful and accurate results is a key result of paper 4.

Papers 3-5 have more empirical content. To understand them methodologically, it is appropriate to consider the five different empirical research strategies delineated by Yin, following COSMOS Corporation [164]. As illustrated in Table 1.1, the choice of empirical

method for investigation depends not only on the research question, but also on the researcher's ability to exert control over behavioral events, and whether contemporary events are in focus or not.

Table 1.1: Research strategies, following [164].

Strategy	Form of research question	Requires control of behavioral events?	Focuses on contemporary events?
Experiment	how, why?	Yes	Yes
Survey	who, what, where, how many, how much?	No	Yes
Archival analysis	who, what, where, how many, how much?	No	Yes/No
History	how, why?	No	No
Case study	how, why?	No	Yes

In paper 3, an expert *survey* was used to assign numeric values to the qualitative causal relations previously inferred from the literature. In the Hillier and Lieberman process, this contribution resides in step 2. The results were used to build a Bayesian model (cf. the next section, on analysis formalisms). Such *expert elicitation* is often used when creating Bayesian models, when available datasets are sparse compared to the number of nodes that need to be parameterized [76].

Paper 4, building on paper 2, presents a metamodel and a method for availability modeling, the use and precision of which is illustrated in five *case studies*. In the Hillier and Lieberman process, this contribution resides in steps 2-5, since a model is both constructed and tested in the case studies.

Paper 5, building on papers 2, 3, and 4, presents a framework for availability modeling, assessment and prediction, based upon architectural structure and causal factors. The framework's abilities are illustrated with nine *case studies*, conducted at five different enterprises. In the Hillier and Lieberman process, this contribution resides in steps 2-5, similarly to paper 4.

Analysis formalisms

This thesis makes use of several analysis methods. More detailed descriptions can be found in each of the papers.

Paper 1 uses only standard mathematical methods from calculus and probability.

Paper 2 is largely based upon Fault Tree Analysis (FTA) and Bayesian networks (BN), two methods that have previously been integrated [15, 95, 159]. Fault Tree Analysis (FTA) is based on reliability theory, Boolean algebra and probability theory [36]. A fault tree represents how combinations of basic events lead to the occurrence of a particular undesired event (viz. system failure) called the top event. A graph is drawn, where the nodes are either events or gates. Events represent the failure of components, subsystems or of the whole system. Gates represent the logic relations (e.g. AND and OR) between events. For more comprehensive treatments on FTA, cf. Ericson [36], Codetta-Raiteri [25], Leveson and Harvey [93], and Bobbio et al. [15].

A Bayesian network is described by Friedman et al. as a representation of a joint probability distribution, where a vertex set denotes a domain of random variables, an edge set denotes the causal dependencies between these random variables, and a probability distribution describes distributions for each of the random variables, given the values of its causal parents [43]. The advantage of the graph is its compact representation of which variables are conditionally independent given other variables. The conditional probabilities are represented in matrices, called Conditional Probability Matrices (CPMs). More comprehensive treatments on Bayesian networks can be found in e.g. [116], [75], [138] and [123].

Paper 3 also uses Bayesian networks, namely a canonical form of CPM called the leaky Noisy-OR gate. Canonical CPMs are a useful formalism, as they make the number of parameters required from expert estimation smaller. The Noisy-OR gate [123, 119] is typically used to describe the interaction of several causes to an effect. It is assumed that each cause has a probability of bringing about the effect, and that the abilities to bring about the effect are independent. The *leaky* Noisy-OR gate allows additional un-modeled causes to be represented as a leak probability.

Paper 4, building on paper 2, also uses FTA, but extends the Bayesian networks into a probabilistic formalism known as Probabilistic Relational Models (PRMs) [42]. While Bayesian networks are suitable for fixed domains, where the random variables and their probabilistic dependencies are known in advance, PRMs can also cope with changing domains. This is accomplished by defining a probability distribution over a metamodel, rather than a model. Whenever such a metamodel is instantiated, the resulting models will be equipped with a corresponding probability distribution. As it is succinctly put by Getoor et al., PRMs "are to Bayesian networks as relational logic is to propositional logic" [48].

Paper 5 retains the probabilistic formalisms of papers 2 and 3, but also requires the ability to traverse architectural models. This is accomplished by the Predictive, Probabilistic Architecture Modeling Framework, P^{2}AMF, a framework for generic software system analysis [81]. P^{2}AMF is based on the Object Constraint Language (OCL), a formal language used to describe expressions on models expressed in the Unified Modeling Language (UML) [1]. P^{2}AMF is fully implemented in the Enterprise Architecture Analysis Tool (EA^{2}T) [79, 18].[1] P^{2}AMF enriches standard entity-relationship models with probabilistic features, viz. stochastic attributes and uncertain existence of entities and relationships. The probabilistic aspects are implemented in a Monte Carlo fashion in the EA^{2}T tool.

Validity and reliability

Papers 1 and 2 do not make empirical contributions in their own right. Therefore, the standard concepts of validity and reliability have little applicability in these cases. However, it should be noted that the methods described require empirical data to be applicable. To *fully* apply the decision-making procedure outlined in paper 1, a functional relationship between actions taken to improve availability and their resultant effects is required. Obtaining such production functions is a challenging (though not impossible) task. Paper 3 indeed takes a first step in this direction. To *partially* apply the results from paper 1, however, it is sufficient that the reader (decision-maker) understands qualitative concepts such as the archetypal cases *fixed restart cost* and *snowball effect* used to characterize the enterprise IT services that is within her domain of decision-making. Relating to the Hillier and Lieberman process described above, the problems of validity and reliability then re-surface in steps 3 and 4, when the models are applied to a particular enterprise. The issue of validity, in this corporate context, is mostly about meticulously including *all* the costs related to outages in

[1] `http://www.ics.kth.se/eaat`

the data collected. If some costs are systematically neglected, the model will not be valid. Similarly, in the corporate context, reliability is mostly about having a documented process for data collection that is being strictly adhered to. If data is collected differently every time the method is employed, the resulting model will yield different results each time.

Paper 2 is similar to paper 1, in that issues of validity and reliability appear only when the proposed logic gate model is applied to a particular enterprise. While this is out of scope of paper 2, it is one of the main contributions of paper 4.

Paper 3 collects expert knowledge using a questionnaire. In order to ensure validity and reliability, the elicitation process adhered to recommendations by Renooij [129], and the construction of the questionnaire further benefited from guidelines taken from Mangione [100], Fowler [40], Blaxter et al. [14], Czaja and Blair [29], Baecher [3], and Garthwaite et al. [45]. One remaining threat to validity relates to respondent interpretation of the questions. The concepts used employed some subjective wordings, such as "best practice". This has both advantages and disadvantages. The obvious drawback is that every respondent forms his own conception of the concept. The upside, however, is that the respondents are not limited by the assumptions and misconceptions of the questionnaire authors. In this case, the respondents were experts selected based on academic publications, and the authors of paper 3 do not claim to know the technical details of "best practice" any better than the respondents did.

Validity was further ensured (i) by the respondent selection process, (ii) by accounting for the uncertainty of their answers, and (iii) by the self-selection that made sure that the 50 final respondents were among the most qualified. Furthermore, the respondents were asked whether there were causes of downtime missing that should be added. The majority of respondents did not have any causal factors to add, and no single factor was pointed out by more than 4% of the respondents.

Paper 4 tests a modeling method, based on fault trees and stakeholder interviews, in practice. The method was tested by comparing the results from the models with the data available from system logs and SLA reports.

The use of the very metrics that are reported to the business for SLA follow-up is a strong argument for the validity of the study. The downtime logged and reported in SLA reports should be precisely the kind of downtime that is the concern of this thesis.

The fact that the modeling data is based on stakeholder interviews naturally introduces a reliability issue. However, determining the extent of this problem is one of the article's core aims, and the results indicate that interview-based models are indeed quite reliable as data sources for architecture models.

The fact that the case studies are from five different companies bring about reliability and validity problems of their own. Different enterprises tend to have different definitions of availability – and different procedures for measuring and documenting it. However, this is an unavoidable consequence of empirical studies of less than perfectly mature areas. Unfortunately, it is not feasible to approach an enterprise and ask not only for a carefully measured series of availability data (that far from every company has), but for a carefully measured series of availability data using a different definition than the one regularly used within that particular enterprise. Sometimes, it is necessary to make do with the study material available.

Paper 5 re-uses the 16 factors of paper 3. However, they have undergone some changes, affecting validity and reliability. Whereas in paper 3, the factors were used in a survey addressed to academics, paper 5 includes empirical data collection from practitioners. To increase construct validity (i.e. "correct operational measures for the concepts being studied" [164]) the factors were cast in the language of the popular ITIL standard [146, 149, 145, 147, 148] (these wordings are found in an appendix to paper 5). In this process, two pairs of

factors were merged together, so that a total of 14 separate factors were retained. Paper 5 includes the application of the framework for availability assessment and prediction to 9 different enterprise IT services. It is certain that reliability and comparability between these cases was increased by the ITIL operationalization. On the other hand, it is also certain that the validity of the original 16 factor model was affected when it was refurbished into the new 14 factor model.

The use of nine different enterprise IT services from 5 different companies bring about reliability and validity problems of their own. Different enterprises tend to have different definitions of availability – and different procedures for measuring and documenting it. Despite best efforts (i.a. the ITIL wordings referred to above), it is highly unlikely that all differences have been resolved. Nevertheless, the high reliability of the interview-based methodology in paper 4 is an argument in favor of the corresponding results in paper 5.

Another concern is the external validity (i.e. "the domain to which a study's findings can be generalized" [164]) of the 14 factor model. The expert questionnaire explicitly made no delimitation: experts were invited to answer based on their experience with any kind of IT system. The goal was to form a general model. Of course, tailoring a model for each and every system would be highly desirable from a precision point of view, but also prohibitively costly. Indeed, the very idea of statistics is to build general models from particular samples. The framework of paper 5 makes a trade-off in this respect, offering a framework that is built upon general principles, thought to be valid across a large domain.

The case studies carried out in paper 5 are all retrospective: the framework based on the 14 factor model is applied to historical enterprise IT services to build models of their behavior. The benefit is clear: there are logs and SLA reports that tell the truth of the state of availability over time. Furthermore, these availability metrics are of business relevance to the enterprises – otherwise they would not be measured and reported. However, the data required for the 14 factor model was collected at a point in time when the respondents were also retrospectively looking back – knowing the outcome in terms of availability figures. Clearly, a much preferred research design would have been to continuously collect architectural descriptions and assessment of the 14 factors over the years, and compare them to the outcome afterwards. However, that would also require a five-year data collection process to start only *after* the framework was devised – which is not possible within the framework of a PhD thesis.

Practitioner relevance

Some have argued that information systems research all too often lacks relevance to practice [10, 22]. For a practically oriented research area, it is important to avoid this tendency. In its later stage, this thesis project has made use of practitioner workshops in order to ensure relevance and applicability.

In a workshop conducted in February 2011, the practitioners who participated in the paper 4 study commented on the accuracy of the method. They found it fully sufficient for the kind of high level decision-support required when planning architectural changes at an early stage. More details can be found in the paper itself.

A similar workshop was conducted in May 2011, where the practitioners who participated in the paper 5 study made a number of important remarks. The following description is based on the analysis of paper 5, and more details can be found in the paper itself.

First of all, the participants identified a gap between the business requirements and practitioner demands for availability modeling and prediction on the one hand, and the actual state of the practice on the other. This constitutes an important confirmation that the overall thesis goal is relevant. Indeed, most of the companies lack sophisticated means

to predict future availability levels. Instead, when planning for future systems, or changes to existing ones, it is common practice to re-use last year's availability figure, sometimes with a tiny inflation factor. Nevertheless, the participants identified several important areas where the ability to model and predict availability of enterprise IT services would be highly useful:

- Oftentimes, practitioners fear that availability levels will suffer when large changes are made to an architecture, yet they cannot quantify this gut feeling. More mature modeling and prediction techniques could improve risk management in projects.
- When communicating with senior management, it is sometimes difficult to convey messages regarding availability levels. This is also corroborated in the literature [118]. However, prognoses of future service availability based on credible and tested methods could be an efficient way to communicate relevant information up the chain of command. One of the workshop participants gave an example of when such a prognosis convinced the business that the predicted availability was too poor, leading to re-allocation of budget in order to improve availability.
- Many enterprises have check lists for improving availability. However, they are rarely prioritized. Prediction methods for each measure could enable more efficient allocation of resources.
- With the advent of outsourcing and cloud computing, service provision becomes less transparent. Indeed, sometimes services are delivered by sub-contractors to sub-contractors, making it very difficult to be on top of expected availability levels. The practitioners held that many companies are immature in the area of writing service level agreements (SLAs), but hoped that architectural modeling and availability prediction could enable better decision-support in this respect.
- IT service architecture is a younger and less mature engineering discipline than for instance mechanical engineering. A workshop participant with degrees in both construction and computer engineering, noted that while bridges are often built with four- or eightfold safety factors, IT systems are often just built to cope precisely with the foreseen load. This is an area where techniques for modeling and prediction of availability can help advancing the field of IT service provision.

To summarize, the kind of decision-support offered by the framework of this thesis has a number of useful applications identified by the practitioner community. The participants in the second workshop were not aware of any existing method with similar characteristics.

Bibliography

[1] Object constraint language, version 2.2. Technical report, Object Management Group, OMG, Feb. 2010. URL `http://www.omg.org/spec/OCL/2.2`. OMG Document Number: formal/2010-02-01.

[2] D. Aron. Escape IT Leadership Deadlocks Using Hybrid Thinking. Technical report, Gartner, Inc., Oct. 2010.

[3] G. Baecher. Judgemental probability in geotechnical risk assessment. Technical report, The Office of the Chief, U.S. Army Corps of Engineers, 1988.

[4] M. Baker and J. Ousterhout. Availability in the sprite distributed file system. *ACM SIGOPS Operating Systems Review*, 25(2):95–98, 1991.

[5] R. Baldoni, S. Fuligni, M. Mecella, and F. Tortorelli. The italian e-government enterprise architecture: A comprehensive introduction with focus on the sla issue. In T. Nanya, F. Maruyama, A. Pataricza, and M. Malek, editors, *Service Availability*, volume 5017 of *Lecture Notes in Computer Science*, pages 1–12. Springer Berlin / Heidelberg, 2008. ISBN 978-3-540-68128-1. 10.1007/978-3-540-68129-8_1.

[6] H. Barki and J. Hartwick. Interpersonal conflict and its management in information system development. *MIS Quarterly*, 25(2):pp. 195–228, 2001. ISSN 02767783. URL `http://www.jstor.org/stable/3250929`.

[7] R. Baskerville and A. Wood-Harper. Diversity in information systems action research methods. *European Journal of Information Systems*, 7(2):90–107, 1998.

[8] S. Beatham, C. Anumba, T. Thorpe, and I. Hedges. KPIs: a critical appraisal of their use in construction. *Benchmarking: An International Journal*, 11(1):93–117, 2004. ISSN 1463-5771.

[9] I. Benbasat and R. Weber. Research commentary: rethinking diversity in information systems research. *Information Systems Research*, 7(4):389–399, 1996.

[10] I. Benbasat and R. Zmud. Empirical research in information systems: the practice of relevance. *MIS Quarterly*, pages 3–16, 1999.

[11] S. Benlarbi. Estimating slas availability/reliability in multi-services ip networks. In D. Penkler, M. Reitenspiess, and F. Tam, editors, *Service Availability*, volume 4328 of *Lecture Notes in Computer Science*, pages 30–42. Springer Berlin / Heidelberg, 2006. ISBN 978-3-540-68724-5. 10.1007/11955498_3.

[12] S. Bernardi and J. Merseguer. A UML profile for dependability analysis of real-time embedded systems. In *Proceedings of the 6th international workshop on Software and performance*, pages 115–124. ACM, 2007.

[13] R. Bhagwan, S. Savage, and G. Voelker. Understanding availability. *Peer-to-Peer Systems II*, pages 256–267, 2003.

[14] L. Blaxter, C. Hughes, and M. Tight. *How to research*. Open University Press, 2006.

[15] A. Bobbio, L. Portinale, M. Minichino, and E. Ciancamerla. Comparing fault trees and bayesian networks for dependability analysis. In *SAFECOMP '99: Proceedings of the 18th International Conference on Computer Computer Safety, Reliability and Security*, pages 310–322, London, UK, 1999. Springer-Verlag. ISBN 3-540-66488-2.

[16] P. Bocciarelli and A. D'Ambrogio. A model-driven method for describing and predicting the reliability of composite services. *Software and Systems Modeling*, 10:265–280, 2011. ISSN 1619-1366. URL `http://dx.doi.org/10.1007/s10270-010-0150-3`. 10.1007/s10270-010-0150-3.

[17] S. Buckl, A. Ernst, J. Lankes, K. Schneider, and C. Schweda. A pattern based approach for constructing enterprise architecture management information models. *Wirtschaftsinformatik*, 2:145–162, 2007.

[18] M. Buschle, J. Ullberg, U. Franke, R. Lagerström, and T. Sommestad. A tool for enterprise architecture analysis using the PRM formalism. In *CAiSE2010 Forum PostProceedings*, Oct. 2010.

[19] K. Cai, C. Wen, and M. Zhang. A critical review on software reliability modeling. *Reliability Engineering & System Safety*, 32(3):357–371, 1991.

[20] H. Cervantes and R. S. Hall. Autonomous adaptation to dynamic availability using a service-oriented component model. In *Proceedings of the 26th International Conference on Software Engineering*, ICSE '04, pages 614–623, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 0-7695-2163-0. URL `http://dl.acm.org/citation.cfm?id=998675.999465`.

[21] C.-H. Chi and P. Johnson. Message from the program co-chairs. In *Enterprise Distributed Object Computing Conference (EDOC), 2011 15th IEEE International*, Aug. 29 - Sept. 2 2011.

[22] M. W. Chiasson and E. Davidson. Taking industry seriously in information systems research. *MIS Quarterly*, 29(4):pp. 591–605, 2005. ISSN 02767783. URL `http://www.jstor.org/stable/25148701`.

[23] J. Chu, K. Labonte, and B. Levine. Availability and locality measurements of peer-to-peer file systems. In *Proceedings of SPIE*, volume 4868, page 310, 2002.

[24] CIO Council. Federal Enterprise Architecture Framework (FEAF). Technical report, Office of Management and Budget, USA, Sept. 1999.

[25] D. Codetta-Raiteri. *Extended Fault Trees Analysis supported by Stochastic Petri Nets*. PhD thesis, University of Torino, Torino, Italy, 2005.

[26] V. Cortellessa and A. Pompei. Towards a UML profile for qos: a contribution in the reliability domain. In *ACM SIGSOFT Software Engineering Notes*, volume 29, pages 197–206. ACM, 2004.

[27] V. Cortellessa, H. Singh, and B. Cukic. Early reliability assessment of UML based software models. In *Proceedings of the 3rd international workshop on Software and performance*, WOSP '02, pages 302–309, New York, NY, USA, 2002. ACM. ISBN 1-58113-563-7. doi: http://doi.acm.org/10.1145/584369.584415. URL `http://doi.acm.org/10.1145/584369.584415`.

[28] J. Coutinho. Software reliability growth. In *Proceedings of the IEEE Symposium on Computer Software Reliability, New York, New York, USA*, pages 58–64, 1973.

[29] R. Czaja and J. Blair. *Designing surveys*. Sage Publications Inc., 2005.

[30] J. Dolbec and T. Shepard. A component based software reliability model. In *Proceedings of the 1995 conference of the Centre for Advanced Studies on Collaborative research*, page 19. IBM Press, 1995.

[31] J. Douceur. Is remote host availability governed by a universal law? *ACM SIGMETRICS Performance Evaluation Review*, 31(3):25–29, 2003.

[32] S. Drake, W. Hu, D. McInnis, M. Sköld, A. Srivastava, L. Thalmann, M. Tikkanen, y. Torbjørnsen, and A. Wolski. Architecture of highly available databases. In M. Malek, M. Reitenspieß, and J. Kaiser, editors, *Service Availability*, volume 3335 of *Lecture Notes in Computer Science*, pages 1–16. Springer Berlin / Heidelberg, 2005. ISBN 978-3-540-24420-2. 10.1007/978-3-540-30225-4_1.

[33] O. Drugan, I. Dionysiou, D. Bakken, T. Plagemann, C. Hauser, and D. Frincke. On the importance of composability of ad hoc mobile middleware and trust management. In M. Malek, E. Nett, and N. Suri, editors, *Service Availability*, volume 3694 of *Lecture Notes in Computer Science*, pages 149–163. Springer Berlin / Heidelberg, 2005. ISBN 978-3-540-29103-9. 10.1007/11560333_13.

[34] D. Durkee. Why cloud computing will never be free. *Queue*, 8(4):20, 2010.

[35] *Enterprise Information Systems*. Aims & scope. http://www.tandf.co.uk/journals/TEIS, 2012. Accessed February 27, 2012.

[36] C. Ericson. Fault tree analysis – a history. In *17th International System Safety Conference*, 1999.

[37] T. Erl. *Service-oriented architecture: concepts, technology, and design*. Prentice Hall PTR, 2005.

[38] C. Fetzer and T. Jim. Dependable distributed computing using free databases. In M. Malek, E. Nett, and N. Suri, editors, *Service Availability*, volume 3694 of *Lecture Notes in Computer Science*, pages 123–136. Springer Berlin / Heidelberg, 2005. ISBN 978-3-540-29103-9. 10.1007/11560333_11.

[39] L. Fiondella and S. Gokhale. Software reliability with architectural uncertainties. In *Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on*, pages 1 –5, april 2008. doi: 10.1109/IPDPS.2008.4536436.

[40] F. J. J. Fowler. *Survey research methods*. Sage Publications Inc., 2002.

[41] M. Fowler. *Patterns of enterprise application architecture*. Addison-Wesley Professional, 2003.

[42] N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *Proc. of the 16th International Joint Conference on Artificial Intelligence*, pages 1300–1309. Morgan Kaufman, 1999.

[43] N. Friedman, M. Linial, and I. Nachman. Using bayesian networks to analyze expression data. *Journal of Computational Biology*, 7:601–620, 2000.

[44] G. Gable. Integrating case study and survey research methods: an example in information systems. *European Journal of Information Systems*, 3(2):112–126, 1994.

[45] P. H. Garthwaite, J. B. Kadane, and A. O'Hagan. Statistical methods for eliciting probability distributions. *Journal of the American Statistical Association*, 100:680–701, June 2005. URL `http://ideas.repec.org/a/bes/jnlasa/v100y2005p680-701.html`.

[46] Gartner, Inc. It definitions and glossary. http://www.gartner.com/technology/it-glossary/, 2012. Accessed February 27, 2012.

[47] T. Genadis. A cost optimization model for determining optimal burn-in times at the module/system level of an electronic product. *International Journal of Quality & Reliability Management*, 13(9):61–74, 1996.

[48] L. Getoor, N. Friedman, D. Koller, A. Pfeffer, and B. Taskar. Probabilistic relational models. In L. Getoor and B. Taskar, editors, *An Introduction to Statistical Relational Learning*. MIT Press, 2007.

[49] R. Giachetti. *Design of enterprise systems: Theory, architecture, and methods*. CRC Press, 2010.

[50] A. Goel. Software reliability models: Assumptions, limitations, and applicability. *Software Engineering, IEEE Transactions on*, (12):1411–1423, 1985.

[51] A. L. Goel and K. Okumoto. A markovian model for reliability and other performance measures of software systems. *Managing Requirements Knowledge, International Workshop on*, 0:769, 1979. doi: http://doi.ieeecomputersociety.org/10.1109/AFIPS.1979.9.

[52] S. Gokhale and R. Mullen. A multiplicative model of software defect repair times. *Empirical Software Engineering*, 15:296–319, 2010. ISSN 1382-3256. URL `http://dx.doi.org/10.1007/s10664-009-9115-y`. 10.1007/s10664-009-9115-y.

[53] S. S. Gokhale. Architecture-based software reliability analysis: Overview and limitations. *Dependable and Secure Computing, IEEE Transactions on*, 4(1):32 –40, 2007. ISSN 1545-5971. doi: 10.1109/TDSC.2007.4.

[54] S. S. Gokhale, W. E. Wong, J. Horgan, and K. S. Trivedi. An analytical approach to architecture-based software performance and reliability prediction. *Performance Evaluation*, 58(4):391 – 412, 2004. ISSN 0166-5316. doi: DOI:10.1016/j.peva.2004.04.003. URL `http://www.sciencedirect.com/science/article/B6V13-4D984KF-1/2/b2fa8ede6a31fef2e898b82c1b5b0a62`.

[55] A. Goyal, S. Lavenberg, and K. Trivedi. Probabilistic modeling of computer system availability. *Annals of Operations Research*, 8:285–306, 1987. ISSN 0254-5330. URL `http://dx.doi.org/10.1007/BF02187098`. 10.1007/BF02187098.

[56] V. Grassi, R. Mirandola, E. Randazzo, and A. Sabetta. Klaper: An intermediate language for model-driven predictive analysis of performance and reliability. *The Common Component Modeling Example*, pages 327–356, 2008.

[57] J. Gray. Why Do Computers Stop and What Can Be Done About It? Technical report, Tandem Computers Inc., June 1985.

[58] J. Gray. A census of tandem system availability between 1985 and 1990. *Reliability, IEEE Transactions on*, 39(4):409–418, 1990.

[59] M. H. Halstead. Toward a theoretical basis for estimating programming effort. In *ACM 75: Proceedings of the 1975 annual conference*, pages 222–224, New York, NY, USA, 1975. ACM. doi: 10.1145/800181.810326.

[60] D. Hamlet, D. Mason, and D. Woit. Theory of software reliability based on components. In *Proceedings of the 23rd International Conference on Software Engineering*, pages 361–370. IEEE Computer Society, 2001.

[61] M. Harper, C. Lawler, and M. Thornton. IT Application Downtime, Executive Visibility and Disaster Tolerant Computing. In *CITSA, 2nd International Conference on Cybernetics and Information Technologies, Systems and Applications*. Citeseer, 2005.

[62] R. Hilliard. IEEE-Std-1471-2000 Recommended Practice for Architectural Description of Software-Intensive Systems. Technical report, IEEE, 2000.

[63] F. S. Hillier and G. J. Lieberman. *Introduction to operations research*. McGraw-Hill New York, NY, 2005. Eighth edition.

[64] E. Holub. There's Gold at the End of the ITIL Rainbow. Technical report, Gartner, Inc., Oct. 2009.

[65] IBM Global Services. Improving systems availability. Technical report, IBM Global Services, 1998.

[66] A. Immonen. A method for predicting reliability and availability at the architecture level. In T. Käköla and J. C. Duenas, editors, *Software Product Lines*, pages 373–422. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-33253-4. doi: 10.1007/978-3-540-33253-4_10.

[67] A. Immonen and E. Niemelä. Survey of reliability and availability prediction methods from the viewpoint of software architecture. *Software and Systems Modeling*, 7:49–65, 2008. ISSN 1619-1366. doi: 10.1007/s10270-006-0040-x.

[68] International Organization for Standardization. Software engineering – product quality – part 1: Quality model. International standard ISO/IEC TR 9126-1:2001(E), International Organization for Standardization, June 2001.

[69] International Organization for Standardization. Software engineering – product quality – part 2: External metrics. International standard ISO/IEC TR 9126-2:2003(E), International Organization for Standardization, July 2003.

[70] International Organization for Standardization. Information technology – service management – part 1: Specification. International standard ISO/IEC 20000-1:2005(E), International Organization for Standardization, December 2005.

[71] International Organization for Standardization. Systems and software engineering – systems and software quality requirements and evaluation (square) – system and software quality models. International standard ISO/IEC 25010:2011(E), International Organization for Standardization, March 2011.

[72] International Telecommunications Union. Final draft of revised recommendation e.800. Technical report, International Telecommunications Union, May 2008.

[73] S. L. Jarvenpaa and B. Ives. Executive involvement and participation in the management of information technology. *MIS Quarterly*, 15(2):pp. 205–227, 1991. ISSN 02767783. URL `http://www.jstor.org/stable/249382`.

[74] Z. Jelinski and P. Moranda. Software reliability research. *Statistical computer performance evaluation*, pages 465–484, 1972.

[75] F. V. Jensen. *Bayesian Networks and Decision Graphs*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2001. ISBN 0387952594.

[76] F. Johansson and G. Falkman. Implementation and integration of a bayesian network for prediction of tactical intention into a ground target simulator. In *Information Fusion, 2006 9th International Conference on*, pages 1–7, July 2006. doi: 10.1109/ICIF.2006.301605.

[77] P. Johnson. *Enterprise Software System Integration: An Architectural Perspective*. PhD thesis, Royal Institute of Technology (KTH), Apr. 2002.

[78] P. Johnson and M. Ekstedt. *Enterprise Architecture: Models and Analyses for Information Systems Decision Making*. Studentlitteratur, 2007. ISBN 9144027524.

[79] P. Johnson, E. Johansson, T. Sommestad, and J. Ullberg. A tool for enterprise architecture analysis. In *Proceedings of the 11th IEEE International Enterprise Computing Conference (EDOC 2007)*, Oct. 2007.

[80] P. Johnson, R. Lagerström, P. Närman, and M. Simonsson. Enterprise architecture analysis with extended influence diagrams. *Information Systems Frontiers*, 9(2), May 2007.

[81] P. Johnson, J. Ullberg, M. Buschle, K. Shahzad, and U. Franke. P^2AMF: Predictive, Probabilistic Architecture Modeling Framework. 2011. Submitted manuscript.

[82] M. Kaâniche, K. Kanoun, and M. Martinello. A user-perceived availability evaluation of a web based travel agency. In *Proceedings of the 2003 International Conference on Dependable Systems and Networks (DSN'03)*, pages 709–718, 2003.

[83] D. Kalinsky. Design patterns for high availability. *Embedded Systems Programming*, 15(8):46–61, 2002.

[84] B. Kaplan and D. Duchon. Combining qualitative and quantitative methods in information systems research: a case study. *MIS quarterly*, pages 571–586, 1988.

[85] J. König and L. Nordström. Reliability analysis of substation automation system functions. In *Reliability and Maintainability Symposium (RAMS), 2012 Proceedings - Annual*, Jan. 2012. To Appear.

[86] H. Kopetz. Tta supported service availability. In M. Malek, E. Nett, and N. Suri, editors, *Service Availability*, volume 3694 of *Lecture Notes in Computer Science*, pages 1–14. Springer Berlin / Heidelberg, 2005. ISBN 978-3-540-29103-9. 10.1007/11560333_1.

[87] H. Koziolek, B. Schlich, and C. Bilich. A Large-Scale Industrial Case Study on Architecture-based Software Reliability Analysis. In *Software Reliability Engineering (ISSRE), 2010 IEEE 21st International Symposium on*, pages 279–288. IEEE, 2010.

[88] S. Kurpjuweit and R. Winter. Viewpoint-based meta model engineering. In *Enterprise Modelling and Information Systems Architectures – Concepts and Applications, Proceedings of the 2nd Int'l Workshop EMISA*, volume 119, pages 143–161, 2007.

[89] A. Kövi, D. Varró, and Z. Németh. Making legacy services highly available with openais: An experience report. In D. Penkler, M. Reitenspiess, and F. Tam, editors, *Service Availability*, volume 4328 of *Lecture Notes in Computer Science*, pages 206–216. Springer Berlin / Heidelberg, 2006. ISBN 978-3-540-68724-5. 10.1007/11955498_15.

[90] R. Lagerström. *Enterprise Systems Modifiability Analysis - An Enterprise Architecture Modeling Approach for Decision Making.* PhD thesis, Royal Institute of Technology (KTH), Apr. 2010.

[91] M. Lankhorst. *Enterprise Architecture at Work: Modelling, Communication and Analysis.* Springer, 2005.

[92] C. Leangsuksun, H. Song, and L. Shen. Reliability modeling using UML. In *Proceeding of The 2003 International Conference on Software Engineering Research and Practice*, 2003.

[93] N. Leveson and P. Harvey. Software fault tree analysis. *The Journal of Systems and Software*, vol. 3, no 2:173–181, 1983.

[94] A. Likierman. Performance indicators: 20 early lessons from managerial use. *Public Money & Management*, 13(4):15–22, 1993. ISSN 0954-0962.

[95] X. Liu, H. Li, and L. Li. Building method of diagnostic model of Bayesian networks based on fault tree. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 7127 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, Nov. 2008. doi: 10.1117/12.806736. URL `http://adsabs.harvard.edu/abs/2008SPIE.7127E..83L`.

[96] I. Majzik, A. Pataricza, and A. Bondavalli. Stochastic dependability analysis of system architecture based on UML models. In R. de Lemos, C. Gacek, and A. Romanovsky, editors, *Architecting Dependable Systems*, volume 2677 of *Lecture Notes in Computer Science*, pages 219–244. Springer Berlin / Heidelberg, 2003. URL `http://dx.doi.org/10.1007/3-540-45177-3_10`. 10.1007/3-540-45177-3_10.

[97] M. Malek. Program chair's message. In M. Malek, M. Reitenspieß, and J. Kaiser, editors, *Service Availability*, volume 3335 of *Lecture Notes in Computer Science*, pages VII–VIII. Springer Berlin / Heidelberg, 2005. ISBN 978-3-540-24420-2. 10.1007/978-3-540-30225-4_1.

[98] M. Malek. Predictive algorithms and technologies for availability enhancement. In *Service availability: 5th International Service Availability Symposium, ISAS 2008, Tokyo, Japan, May 19-21, 2008: proceedings*, volume 5017, page 17. Springer-Verlag New York Inc, 2008.

[99] B. Malik and D. Scott. How to Calculate the Cost of Continuously Available IT Services. Technical report, Gartner, Inc., Nov. 2010.

[100] T. W. Mangione. *Mail surveys. Improving the quality.* Sage Publications Inc., 1995.

[101] E. Marcus and H. Stern. *Blueprints for high availability, second edition.* John Wiley & Sons, Inc., Indianapolis, IN, USA, 2003.

[102] T. J. McCabe. A complexity measure. In *ICSE '76: Proceedings of the 2nd international conference on Software engineering*, page 407, Los Alamitos, CA, USA, 1976. IEEE Computer Society Press.

[103] I. Meedeniya, I. Moser, A. Aleti, and L. Grunske. Architecturebased reliability evaluation under uncertainty. In *Proc. 7th Int. Conf. on the Quality of Software Architectures (QoSA'11)*, pages 85–94, 2011.

[104] D. Menasce. Composing web services: A qos view. *Internet Computing, IEEE*, 8(6): 88 – 90, nov.-dec. 2004. ISSN 1089-7801. doi: 10.1109/MIC.2004.57.

[105] Z. Micskei, I. Majzik, and F. Tam. Comparing robustness of ais-based middleware implementations. In *Proceedings of the 4th international symposium on Service Availability*, pages 20–30. Springer-Verlag, 2007.

[106] M. Mikic-Rakic, S. Malek, and N. Medvidovic. Improving availability in large, distributed component-based systems via redeployment. In A. Dearle and S. Eisenbach, editors, *Component Deployment*, volume 3798 of *Lecture Notes in Computer Science*, pages 83–98. Springer Berlin / Heidelberg, 2005. ISBN 978-3-540-30517-0. URL `http://dx.doi.org/10.1007/11590712_7`. 10.1007/11590712_7.

[107] N. Milanovic. Contract-based web service composition framework with correctness guarantees. In M. Malek, E. Nett, and N. Suri, editors, *Service Availability*, volume 3694 of *Lecture Notes in Computer Science*, pages 52–67. Springer Berlin / Heidelberg, 2005. ISBN 978-3-540-29103-9. 10.1007/11560333_5.

[108] N. Milanovic. *Models, Methods and Tools for Availability Assessment of IT-Services and Business Processes.* Universitätsbibliothek, 2010. Habilitationsschrift.

[109] Ministry of Defence. MODAF Glossary version 1.2. Technical report, Ministry of Defence, UK, July 2008.

[110] P. Moranda. An comparison of software error-rate models. In *Proc. Texas Conference on Computing Systems.* IEEE CS, 1975.

[111] P. B. Moranda. An error detection model for application during software development. *Reliability, IEEE Transactions on*, R-30(4):309 –312, oct. 1981. ISSN 0018-9529. doi: 10.1109/TR.1981.5221096.

[112] B. Murphy and B. Levidow. Windows 2000 dependability. Technical report, Technical Report MSR-TR-2000-56, Microsoft Research, 2000.

[113] J. Musa. A. lannino, and k. okumoto. *Software reliability: measurement, prediction, application*, 1:5–15, 1987.

[114] P. Närman. *Enterprise architecture for information system analysis – Modeling and assessing performance, availability, data accuracy and application usage*. PhD thesis, Royal Institute of Technology (KTH), Sept. 2012.

[115] B. Nassu and T. Nanya. Interaction faults caused by third-party external systems – a case study and challenges. In T. Nanya, F. Maruyama, A. Pataricza, and M. Malek, editors, *Service Availability*, volume 5017 of *Lecture Notes in Computer Science*, pages 59–74. Springer Berlin / Heidelberg, 2008. ISBN 978-3-540-68128-1. 10.1007/978-3-540-68129-8_7.

[116] R. E. Neapolitan. *Learning Bayesian Networks*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2003. ISBN 0130125342.

[117] J. Neises. Benefit evaluation of high-availability middleware. In *Service availability: first International Service Availability Symposium, ISAS 2004, Munich, Germany, May 13-14, 2004: revised selected papers*, volume 3335, page 73. Springer, 2005.

[118] D. Olwell. Reliability leadership. In *Reliability and Maintainability Symposium, 2001. Proceedings. Annual*, pages 139 –141, 2001. doi: 10.1109/RAMS.2001.902456.

[119] A. Onisko, M. J. Druzdzel, and H. Wasyluk. Learning bayesian network parameters from small data sets: application of noisy-or gates. *International Journal of Approximate Reasoning*, 27(2):165–182, 2001. ISSN 0888-613X. doi: DOI: 10.1016/S0888-613X(01)00039-1.

[120] D. Oppenheimer. Why do internet services fail, and what can be done about it? In *Proceedings of USITS'03: 4th USENIX Symposium on Internet Technologies and Systems*, Mar. 2003.

[121] H. Ozaki, A. Kara, and Z. Cheng. User-perceived availability of priority shared protection systems. *International Journal of Quality & Reliability Management*, 28(1): 95–108, 2011.

[122] S. Parkin, D. Ingham, and G. Morgan. A message oriented middleware solution enabling non-repudiation evidence generation for reliable web services. In *Service availability: 4th International Service Availability Symposium, ISAS 2007, Durham, NH, USA, May 21-22, 2007; proceedings*, volume 4526, page 9. Springer-Verlag New York Inc, 2007.

[123] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988. ISBN 1558604790.

[124] S. Pertet and P. Narasimhan. Causes of failure in web applications. Technical report, Parallel Data Laboratory, Carnegie Mellon University, CMU-PDL-05-109, 2005.

[125] H. Pham. *Software Reliability*. Springer-Verlag, Singapore, 2000.

[126] H. Pham and L. Nordmann. A generalized nhpp software reliability model. In *Third International Conference on Reliability and Quality in Design, Anaheim, ISSAT Press*, 1997.

[127] M. Rausand and A. Høyland. *System Reliability Theory: Models, Statistical Methods, and Applications*. John Wiley & Sons, Inc., Hoboken, NJ, USA, 2nd edition, 2004. URL `http://www.ntnu.no/ross/srt`.

[128] H. Reisinger. Quality of service control by middleware. In *Service availability: first International Service Availability Symposium, ISAS 2004, Munich, Germany, May 13-14, 2004: revised selected papers*, volume 3335, page 61. Springer, 2005.

[129] S. Renooij. *Qualitative approaches to quantifying probabilistic networks*. PhD thesis, Utrecht University, Utrecht, the Netherlands, 2001.

[130] G. Rodrigues. *A Model Driven Approach for Software Reliability Prediction*. PhD thesis, University College London, 2008.

[131] R. Roshandel and N. Medvidovic. Toward architecture-based reliability estimation. In *Proc. Twin Workshops on Architecting Dependable Systems*, 2004.

[132] S. Saroiu, P. Gummadi, S. Gribble, et al. A measurement study of peer-to-peer file sharing systems. In *proceedings of Multimedia Computing and Networking*, volume 2002, page 152, 2002.

[133] G. Schick and R. Wolverton. An analysis of competing software reliability models. *Software Engineering, IEEE Transactions on*, (2):104–120, 1978.

[134] K. Schmidt. *High availability and disaster recovery*. Springer, 2006.

[135] D. Scott. How to Assess Your IT Service Availability Levels. Technical report, Gartner, Inc., Apr. 2009.

[136] D. Scott. Benchmarking Your IT Service Availability Levels. Technical report, Gartner, Inc., Dec. 2011.

[137] T. Setzer, K. Bhattacharya, and H. Ludwig. Change scheduling based on business impact analysis of change-related risk. *Network and Service Management, IEEE Transactions on*, 7(1):58 –71, 2010. ISSN 1932-4537. doi: 10.1109/TNSM.2010.I9P0305.

[138] R. D. Shachter. Probabilistic inference and influence diagrams. *Oper. Res.*, 36(4): 589–604, 1988. ISSN 0030-364X.

[139] J. Shanthikumar. Software reliability models: a review. *Microelectronics Reliability*, 23(5):903–943, 1983.

[140] I. Shin and I. Lee. A compositional framework for real-time embedded systems. In M. Malek, E. Nett, and N. Suri, editors, *Service Availability*, volume 3694 of *Lecture Notes in Computer Science*, pages 137–148. Springer Berlin / Heidelberg, 2005. ISBN 978-3-540-29103-9. 10.1007/11560333_12.

[141] H. Singh, V. Cortellessa, B. Cukic, E. Gunel, and V. Bharadwaj. A bayesian approach to reliability prediction and assessment of component based systems. In *Software Reliability Engineering, 2001. ISSRE 2001. Proceedings. 12th International Symposium on*, pages 12 – 21, nov. 2001. doi: 10.1109/ISSRE.2001.989454.

[142] T. Sommestad, M. Ekstedt, and P. Johnson. A probabilistic relational model for security risk analysis. *Computers & Security*, 29(6):659–679, Sept. 2010.

[143] T. Sommestad, H. Holm, and M. Ekstedt. Estimates of success rates of remote arbitrary code execution attacks. *Information Management & Computer Security*, Jan. 2012. To appear.

[144] A. Sukert. An investigation of software reliability models. In *Annual Reliability and Maintainability Symposium, Philadelphia, Pa*, pages 478–484, 1977.

[145] S. Taylor, D. Cannon, and D. Wheeldon. *Service Operation (ITIL)*. The Stationery Office, TSO, 2007. ISBN 9780113310463.

[146] S. Taylor, G. Case, and G. Spalding. *Continual Service Improvement (ITIL)*. The Stationery Office, TSO, 2007. ISBN 9780113310494.

[147] S. Taylor, M. Iqbal, and M. Nieves. *Service Strategy (ITIL)*. The Stationery Office, TSO, 2007. ISBN 9780113310456.

[148] S. Taylor, S. Lacy, and I. Macfarlane. *Service Transition (ITIL)*. The Stationery Office, TSO, 2007. ISBN 9780113310487.

[149] S. Taylor, V. Lloyd, and C. Rudd. *Service Design (ITIL)*. The Stationery Office, TSO, 2007. ISBN 978011310470.

[150] The Open Group. TOGAF 9.1, TOGAF Online. http://pubs.opengroup.org/architecture/togaf9-doc/arch/ index.html, 2009.

[151] K. Tokuno and S. Yamada. User-perceived software service availability modeling with reliability growth. In T. Nanya, F. Maruyama, A. Pataricza, and M. Malek, editors, *Service Availability*, volume 5017 of *Lecture Notes in Computer Science*, pages 75–89. Springer Berlin / Heidelberg, 2008. ISBN 978-3-540-68128-1. 10.1007/978-3-540-68129-8_8.

[152] J. Ullberg, P. Johnson, and M. Buschle. A modeling language for interoperability assessments. In *Third International IFIP working Conference on Enterprise Interoperability*, Mar. 2011.

[153] E. Vargas. High availability Best Practices. Technical report, Sun Microsystems, Inc., 2000.

[154] E. Vargas. High availability fundamentals. Technical report, Sun Microsystems, Inc., 2000.

[155] I. Vessey, V. Ramesh, and R. L. Glass. Research in information systems: An empirical study of diversity in the discipline and its journals. *J. Manage. Inf. Syst.*, 19:129–174, October 2002. ISSN 0742-1222. URL `http://dl.acm.org/citation.cfm?id=1289776.1289781`.

[156] M. Vujosevic. Uncertainty in reliability evaluation processes and a simulation approach to treating it. *European Journal of Operational Research*, 32(2):245 – 250, 1987. ISSN 0377-2217. doi: DOI:10.1016/S0377-2217(87)80146-7. URL `http://www.sciencedirect.com/science/article/B6VCT-4GJVBRY-8/2/95f829187bb7b9886848474161a72c61`. Third EURO Summer Institute Special Issue Decision Making in an Uncertain World.

[157] J. Wall and P. Ferguson. Pragmatic software reliability prediction. In *Annual Reliability and Maintainability Symposium, Philadelphia, Pa*, pages 485–488, 1977.

[158] D. Wang and K. Trivedi. Modeling user-perceived service availability. In M. Malek, E. Nett, and N. Suri, editors, *Service Availability*, volume 3694 of *Lecture Notes in Computer Science*, pages 107–122. Springer Berlin / Heidelberg, 2005. ISBN 978-3-540-29103-9. 10.1007/11560333_10.

[159] P. Weber and L. Jouffe. Complex system reliability modelling with dynamic object oriented bayesian networks (DOOBN). *Reliability Engineering & System Safety*, 91(2):149 – 162, 2006. ISSN 0951-8320. doi: DOI:10.1016/j.ress.2005.03.006. URL `http://www.sciencedirect.com/science/article/B6V4T-4G7NSTN-1/2/ecb3643f9e79004fbed06692b8eb9d2d`. Selected Papers Presented at QUALITA 2003.

[160] D. Williams. The Challenges and Approaches of Establishing IT Infrastructure Monitoring SLAs in IT Operations. Technical report, Gartner, Inc., Sept. 2010.

[161] A. Wolski and V. Raatikka. Performance measurement and tuning of hot-standby databases. In D. Penkler, M. Reitenspiess, and F. Tam, editors, *Service Availability*, volume 4328 of *Lecture Notes in Computer Science*, pages 149–161. Springer Berlin / Heidelberg, 2006. ISBN 978-3-540-68724-5. 10.1007/11955498_11.

[162] W. Xie, H. Sun, Y. Cao, and K. Trivedi. Modeling of user perceived webserver availability. In *Communications, 2003. ICC '03. IEEE International Conference on*, volume 3, pages 1796 – 1800 vol.3, may 2003. doi: 10.1109/ICC.2003.1203909.

[163] S. Yamada, M. Ohba, and S. Osaki. S-shaped reliability growth modeling for software error detection. *Reliability, IEEE Transactions on*, 32(5):475–484, 1983.

[164] R. K. Yin. *Case study research: Design and methods*, volume 5. Sage publications, INC, 1994.

[165] T. Yoshida, M. Hisada, and S. Tomita. Implementation of highly available memory database as saf component. In M. Malek, M. Reitenspieß, and A. van Moorsel, editors, *Service Availability*, volume 4526 of *Lecture Notes in Computer Science*, pages 43–51. Springer Berlin / Heidelberg, 2007. ISBN 978-3-540-72735-4. 10.1007/978-3-540-72736-1_5.

[166] E. Zambon, S. Etalle, R. Wieringa, and P. Hartel. Model-based qualitative risk assessment for availability of it infrastructures. *Software and Systems Modeling*, pages 1–28, 2010. ISSN 1619-1366. URL `http://dx.doi.org/10.1007/s10270-010-0166-8`. 10.1007/s10270-010-0166-8.

[167] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng. Quality driven web services composition. In *Proceedings of the 12th international conference on World Wide Web*, WWW '03, pages 411–421, New York, NY, USA, 2003. ACM. ISBN 1-58113-680-3. doi: 10.1145/775152.775211. URL `http://doi.acm.org/10.1145/775152.775211`.

[168] L. Zeng, B. Benatallah, A. Ngu, M. Dumas, J. Kalagnanam, and H. Chang. Qos-aware middleware for web services composition. *Software Engineering, IEEE Transactions on*, 30(5):311 – 327, may 2004. ISSN 0098-5589. doi: 10.1109/TSE.2004.11.

[169] R. Zhang, E. Cope, L. Heusler, and F. Cheng. A Bayesian Network Approach to Modeling IT Service Availability using System Logs. In *Workshop on the Analysis of System Logs*, Oct. 2009.

[170] X. Zhang and H. Pham. An analysis of factors affecting software reliability. *J. Syst. Softw.*, 50(1):43–56, 2000. ISSN 0164-1212. doi: 10.1016/S0164-1212(99)00075-8.

Part II

Papers 1 to 5

Paper 1

Optimal IT service availability: Shorter outages, or fewer?

Paper 2

Enterprise architecture dependency analysis using fault trees and Bayesian networks

Paper 3

Availability of enterprise IT systems: an expert-based Bayesian framework

Paper 4

Enterprise architecture availability analysis using fault trees and stakeholder interviews

Paper 5

An architecture framework for enterprise IT service availability analysis