1 Dependability

Terms like reliability, availability, maintainability, safety, and dependability are often used loosely in colloquial conversation without relying on their precise definition and representation. However, in scientific literature it is important to define these terms precisely so that they can be described, represented, and evaluated without any loss of precision. This chapter is an attempt to present this latter view in a comprehensive manner and set the stage for the detailed exposition that follows in the subsequent chapters.

1.1 Definition

The scientific use of the terms dependability, availability, reliability, maintainability, and safety were first defined rigorously by Laprie [1]. The terminology was further refined and comprehensive definitions relating to dependability were given in a landmark paper by Avizienis *et al.* [2].

Dependability is treated as an umbrella concept that encompasses attributes such as reliability, availability, safety, integrity, and maintainability. Security-related concepts such as confidentiality, availability, and integrity were also brought under the overall framework in this paper. Further elaboration on these ideas has been suggested elsewhere, including [3]. Let us now examine the term *dependability* in some detail.

The IEC international vocabulary [4] defines the dependability (of an item) as the "ability to meet success criteria, under given conditions of use and maintenance." In the information technology area [5], the meaning of the word "dependability" has been thoroughly investigated by the International Federation for Information Processing (IFIP) working group WG10.4 (on dependable and fault-tolerant computing), and a recent definition issued by this group appears in [2]:

> The dependability of a computer system is the ability to deliver a service that can justifiably be trusted. The service delivered by a system is its behavior as it is perceived by its user(s); a user is another system (physical, human) that interacts with the former at the service interface.

A 1988 survey of several definitions of computer-based system dependability resulted in the following summary [6]:

> Dependability of a computer system may be defined as the justifiable confidence the manufacturer has that it will perform specified actions or deliver specified results in a trustworthy and timely manner.

https://doi.org/10.1017/9781316163047.003 Published online by Cambridge University Press

In this context a system can be a single component, a module, a subsystem of a complex system, such as a power or chemical or nuclear plant, a computer system, a software system such as a web server, a data center, a telephone network, smart grid, cloud, an avionic traffic control system or an aircraft flight control system.

Dependability is the result of a large number of physical, technological, and structural characteristics of the system, in addition to the impact of the environmental and operating (application) conditions: the materials with which the elementary components are built, the assembly techniques, the operating conditions including the workload being processed and the thermal conditions, the interaction with the human operator, the technical assistance and maintenance policies. Understanding, modeling, and analyzing the dependability of a system with the aim of pinpointing potential weaknesses and improving the capability to operate correctly requires the harmonious combination of different disciplines, from materials science to probability and statistics, from manufacturing engineering to man–machine interaction and production organization.

The degree to which a system is able to provide the expected operation or service for which it is designed needs to be quantitatively assessed by defining proper measurable quantities. The quantitative assessment of system dependability thus becomes essential in system design, planning, implementation, validation, manufacturing, and field operation. A number of requirements, methods, and techniques have been established and standardized to quantitatively evaluate the capability of a system for providing the desired operation.

Along the various phases of the design and manufacturing process of a system there is a need to predict with the lowest possible degree of uncertainty: how the system will behave in operation during its entire useful life; how malfunctions or failures will appear during operation and with what frequency; how long the duration of the outages will be and what resources are needed to maintain the system in a correct operating state. The quantitative evaluation of these attributes plays a critical role in assessing the effectiveness of design alternatives, the choice of appropriate materials or parts, and in determining the success of a product or system.

In order to quantitatively assess and assure the dependability attributes of a system, different measures are introduced and a set of modeling and analysis techniques have been developed to derive and evaluate the measures. The following sections examine these in further detail.

1.2 Dependability Measures and Metrics

Dependability, as defined earlier, is a qualitative property, an aptitude of the system to conform to design specifications and user expectations. However, in order to establish a quantitative theory of dependability, specific quantities that characterize different aspects or attributes of dependability have to be formally defined so that they can be evaluated through unambiguous mathematical techniques.

IFIP WG10.4 views dependability issues from three aspects: threats, attributes, and means. The taxonomy of the precise terms and their relationship is delineated in the

https://doi.org/10.1017/9781316163047.003 Published online by Cambridge University Press

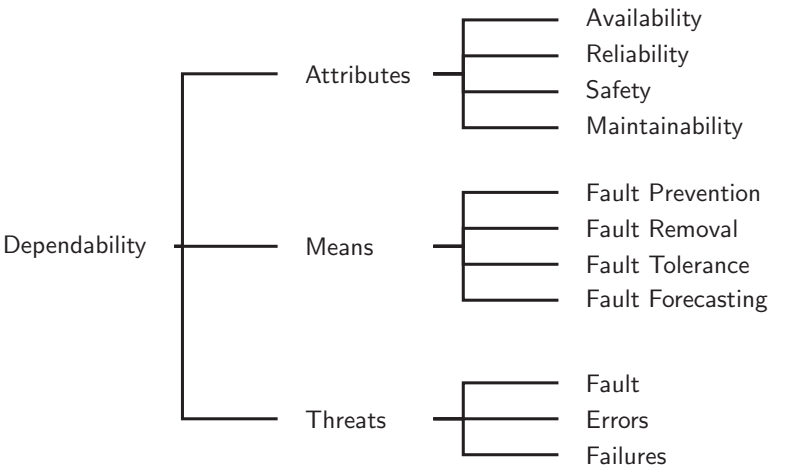

Figure 1.1 IFIP WG10.4 dependability tree.

dependability tree as shown in Figure 1.1 [2, 5]. The dependability tree has subsequently been extended to include security as well [2].

The threats to dependability can be viewed in terms of faults, errors, and failures. Faults are adjudged causes of errors and failures. A fault, when exercised, may produce an internal error. Errors, either singly or on accumulation, may give rise to a failure. Failure of a subsystem in turn becomes a fault at the system level. Faults for electronic hardware have been further classified into permanent, transient, and intermittent [7]. For software, faults were recently classified into "Bohrbugs," (non-aging-related) "Mandelbugs," and aging-related bugs [8], and for networks errors have been classified into single-bit, multiple-bit, and correlated errors in [9]. Failures have been classified into omission failures, value (or content) failures, and timing failures [10]. Timing failures are also known as performance failures or dynamic failures. Failures are also classified by their severity [11] or criticality [12]. For instance, failures in safety-critical and life-critical systems can be classified into safe and unsafe failures. For capturing security metrics, failures have been classified into those compromising confidentiality, those compromising integrity and those leading to a lack of access (that is, unavailability) [13, 14]. For further discussion on this topic, see [2].

The means to achieving and assuring dependability include fault prevention (or fault avoidance), which consists of carefully designing a system with a minimal number of faults, and fault removal, which is the process of finding and fixing bugs during testing or during operation. Fault tolerance constitutes a set of techniques that allow the system as a whole to continue to function in spite of component or subsystem failures. The use of redundancy is an essential part of fault tolerance. Using more components than required (massive redundancy), repeating an operation (time redundancy) or using more bits than required (information redundancy) are commonly used techniques. Furthermore, the management of redundancy including detection, location, reconfiguration and recovery is an essential aspect of fault tolerance. Finally,

https://doi.org/10.1017/9781316163047.003 Published online by Cambridge University Press

fault forecasting consists of methods to predict the occurrence of faults/errors/failures or associated dependability attributes.

The attributes of dependability that will be considered in the subsequent chapters of this book are informally defined next, by combining the definitions taken from both the IEC international vocabulary [4] and the IFIP WG10.4 working group [2].

Reliability

- ability to perform a required function under given conditions for a given time interval [4];
- continuity of correct service [2].

Availability

- ability to be in a state to perform a required function, under given conditions, at a given instant of time, or after enough time has elapsed, assuming that the required external resources are provided [4];
- readiness for correct service [2].

Maintainability

- ability to be retained in, or restored to, a state in which it can perform a required function, under given conditions of use and maintenance [4];
- ability to undergo modifications and repairs [2].

A fundamental difference between reliability and availability is that reliability refers to failure-free operation during an interval, while availability refers to failure-free operation at a given instant of time, usually the time when a device or system is accessed to provide a required function or service. Reliability is a measure that characterizes the failure process of an item, while availability combines the failure process with the restoration or repair process and looks at the probability that at a given time instant the item is operational independently of the number of failure/repair cycles already undergone by the item.

Each one of the above attributes can be characterized by different metrics that will be formally defined in Chapter 3. In the probabilistic approach to the quantitative dependability assessment, the above attributes (reliability, availability, maintainability) will be computed as the probability of occurrence of specific events.

- For computing the attribute *reliability*, we refer to the event that the system is operating continuously without failures in a given interval under specified operating and environmental conditions. The *reliability* is the probability that this event occurs.
- For computing the attribute *availability*, we refer to the event that the system is operating at a given point in time independently of the number of failures (and repairs) already incurred by the system. The *availability* is the probability that this event occurs.

https://doi.org/10.1017/9781316163047.003 Published online by Cambridge University Press

- For computing the attribute *maintainability*, we refer to the event that when the system is not operational due to failures, it will be recovered to an operating condition. The *maintainability* is the probability that this event occurs.

The traditional dependability metrics defined above take a *system-oriented* perspective [2]. *Service-oriented* dependability metrics as defined by Tortorella [15], on the other hand, take a user-oriented perspective. Tortorella [15] classifies the figures of merit for service dependability into three attributes:

Service accessibility: the ability to initiate the service request when desired.
Service continuity: the successful continuation of a successfully initiated service request.
Service release: the successful completion of an initiated service request.

Bauer and Adams [16] state that since most services are reliable, it is more convenient to focus on the much smaller number of unreliable service events or service defects. These service defects are conveniently normalized as the number of calls or customer demands not served per million attempts, referred to as defects per million (DPM) [16, 17].

1.3 Examples of System Dependability Evaluation

So far, we have examined several conceptual and theoretical aspects of dependability. We present a brief overview of several examples of dependability evaluation in this section. The motivation behind presenting these examples is threefold:

- provide concrete evidence of the applicability of the techniques that will be encountered in the subsequent chapters of this book;
- illustrate how the dependability techniques are employed in the evaluation of specific systems;
- showcase different real-life case studies that span the whole gamut of dependability evaluation, ranging from pure reliability evaluation to systems showing attributes of availability, maintainability, and beyond.

1.3.1 Pure Reliability Evaluation

Mission-critical systems like aircraft and spacecraft flight control, nuclear reactor control systems and telecommunication systems are characterized by stringent requirements imposed by government regulatory bodies on the probability of catastrophic failure. As an example, the US Federal Aviation Administration (FAA) mandates that the catastrophic failure probability of aircraft should be below 10^{-9}/flight-hour [18, 19]. These are classic examples of systems requiring pure reliability evaluation. Ramesh *et al.* [20] consider the reliability evaluation of an aircraft in a flight operation time management scenario. They develop analytical and numerical methods for probabilistic risk analysis using fault trees, Markov chains, and stochastic Petri

https://doi.org/10.1017/9781316163047.003 Published online by Cambridge University Press

nets. Hjelmgren *et al.* [21] carry out the reliability analysis of a fault-tolerant Full Authority Digital Electronic Control system (FADEC) used for control of an aircraft gas turbine engine on an aircraft equipped with a single engine. Two redundancy options, including a two-channel hot standby and a three-channel triple modular redundancy (TMR) system that is reconfigured into a two-channel hot standby after an error has been detected and located, are evaluated. Markovian models are used to assess the probability of failure for the system. The operational framework for fault detection, identification and recovery in autonomous spacecraft has been studied in [22] using dynamic fault trees and Bayesian networks.

1.3.2 Safety Analysis of Critical Systems

In safety-critical systems, like chemical, nuclear or power plants, or intensive care units for emergency rooms or operating theatres in a hospital, the occurrence of a system failure may entail catastrophic consequences for human life or the environment. The interest in the analysis is to evaluate the probability that such a catastrophic condition may occur, and the time of the first occurrence of such a catastrophic condition.

1.3.3 Availability and Maintainability Evaluation

High-availability systems, such as the IBM BladeCenter®, are designed to provide commercial services such as e-commerce, financial, stock trading, and telephone communication services [23]. Blade servers are widely adopted because of their modular design, with industry-standard racks accommodating multiple servers together with shared power, cooling, and other services within the rack chassis. Availability requirements for such systems are in the region of 0.999 99 ("five nines"), with annual mean system downtime requirements below six minutes. Smith *et al.* [23] present a detailed availability analysis of an IBM BladeCenter comprising up to 14 separate blade servers contained within a single chassis. They construct a comprehensive two-level hierarchical model with a higher-level fault tree model of the system with the underlying subsystems and components being modeled as lower-level Markov chains. Specific dependability metrics for this model include the system availability and the downtime in minutes/year. In addition, they conduct sensitivity analysis of the system, including the contribution of each component to a single blade downtime.

1.3.4 Software Dependability

Software *aging* [24] is now a well-recognized phenomenon, resulting from degradation in the software state or the execution environment. This degradation is due to such causes as memory bloating and leaking, unreleased file locks, data corruption, storage space fragmentation and accumulation of roundoff errors. The net result is performance degradation of the software, possibly resulting in a crash/hang failure. Software aging-related bugs are included in the software fault classification (Bohrbugs or Mandelbugs) by [24]. To counteract the effects of aging-related problems, software

https://doi.org/10.1017/9781316163047.003 Published online by Cambridge University Press

rejuvenation is often employed, which entails occasionally stopping the software, cleaning up its state and the environment and restarting the software. The primary motivation behind this example [24] is to derive optimal rejuvenation schedules to maximize availability or minimize downtime cost. Toward this goal, the authors first construct a semi-Markov reward model based on workload and resource usage data collected from a Unix environment. The model is then solved to obtain estimated times to exhaustion for each resource. The results from the semi-Markov reward model are then fed into a higher-level semi-Markov availability model that accounts for failure followed by reactive recovery, as well as proactive recovery, to determine the optimal rejuvenation schedule.

1.3.5 Service-Oriented Dependability

Service-oriented environments like telecommunication systems are better evaluated using service-oriented dependability metrics. In this example, we focus on an environment that supports Voice over IP (VoIP) functionality using the session initiation protocol (SIP), an application-layer control protocol for creating, modifying, and terminating sessions, including Internet telephone calls, multimedia distribution, and multimedia teleconferences. A typical SIP-based communication involves interaction between two parties: the user agent client (UAC) and the user agent server (UAS). To establish a SIP call session, signaling messages are exchanged with the mediation of an application server. The application server, such as the IBM's WebSphere Application Server (WAS) [25] or BEA's WebLogic Server [26], is implemented as a software process and provides rich SIP functionality for the users. This example [17, 27] considers that the SIP application installed on the application server is a back-to-back user agent (B2BUA), which acts as a proxy for SIP messages in VoIP call sessions [28]. Any failure of the application server will result in lost calls, either newly arriving calls due to blocking, or in-progress calls due to cutoff. A blocked call event is defined as a call that was prevented from being successfully established due to failures. A cutoff call event occurs when a stable call is terminated prior to either party going on-hook. In this example, we are primarily interested in computing DPM, a commonly used service (un)reliability measure for telecommunication systems. DPM accounts for all types of call losses, including blocked and cutoff calls.

1.3.6 Task-Oriented Dependability

A task, which requires a specified amount of work to be executed, is processed by a system that may change its computational power in time due to failures and repairs or to variations in the performance level. A task-oriented view of such a system recognizes the fact that the task completion time is affected by changes in the system computational power. For example, the occurrence of a fault during the execution of a task may cause the task to be dropped, or to be preempted and then resumed at a later system recovery, or to be preempted and then restarted from the beginning when the system is up again. Analysis of the distribution of the task completion time under different interruption and

https://doi.org/10.1017/9781316163047.003 Published online by Cambridge University Press

recovery policies when the computing system changes its mode of operation randomly can be found in [29–31].

1.4 Predictive Dependability Assessment

Quantitative dependability analysis is conveniently applied in the earliest phases of a design process with the goal of evaluating the conformity to specifications and of comparing different design alternatives. Such methods have also been used in later phases such as verification or operational phases. Some areas where predictive dependability techniques are invaluable, not only from a technical but also from a social and economic point of view, and to determine the success of a project, are:

Risk assessment and safety analysis: The risk level associated with a technological activity is related to the probability that some malfunction appears in a part of a system and the consequences that this malfunction may have on the system as a whole, on the workers or persons working in the proximity of the malfunctioning system or on the surrounding environment [32]. Dependability theory offers a framework to evaluate the probability that a malfunction may appear in the system, thus providing a necessary input in the construction of a quantitative risk assessment [33]. Quantitative evaluation of safety-critical systems is also the objective of international electronic standards [34].

Design and contract specifications: In the design specifications of complex or safety-critical systems, dependability clauses are often included. We have already seen examples in Sections 1.3.1 and 1.3.3. These clauses may be in the form of a guaranteed availability, or in the form of an upper bound on the total expected downtime in a given period (e.g., one year), or in the form of probability of correct operation in a given mission time. The fulfillment of the dependability clauses requires the capability of quantitatively predicting the required measures from the earliest phases of system design.

Technical assistance and maintenance: The cost of the technical assistance and maintenance services depends on the expected number of requests that will be issued in a given period of time to keep or to restore the system in proper operating conditions. The number of requests depends on the expected number of failures, on their type, and severity, and on the complexity of the actions that are needed to recover to operating condition. Planning the technical assistance service requires an *a priori* estimation of the number of times the assistance will be needed. Furthermore, the knowledge of the behavior of the system in time may allow one to predict the expected cost of system outages in a given time interval of operation and to infer the logistics of the maintenance services – number of spares for each component or subsystem, scheduling of repair crews, location of repair facilities, and so forth. Warranty periods can also be seen as an aspect of guaranteed dependability, since, in this case, the manufacturers assume the cost of the potential malfunctions and resultant downtime.

https://doi.org/10.1017/9781316163047.003 Published online by Cambridge University Press

Optimal preventive maintenance: Scheduling of preventive maintenance can be based on dependability quantification. This has been applied both for hardware preventive maintenance [35–37] and software preventive maintenance [24, 38].

Lifecycle cost: The cost of a system, considered over its complete lifecycle, may be considered as composed of several parts: an initial acquisition cost known as *CapX*, and the operational, labor, and a deferred maintenance cost distributed over the life of the system, collectively known as *OpX*. These cost components are often in conflict; a highly dependable system tends to have a higher initial acquisition cost but a reduced deferred cost. Dependability prediction techniques may help in finding an optimal tradeoff between these cost factors, by balancing the amount to invest in the initial system dependability to reduce maintenance, operational, and labor costs distributed over the useful life. Formal cost minimization techniques can be applied to solve this problem [39].

Market competitiveness: Dependability is often a key ingredient in product differentiation and valorization leading to the commercial success of a product.

Summary

In conclusion, we have presented in this chapter formal definitions of various dependability-related terms: availability, reliability, maintainability, and safety. We have briefly referred to various measures of dependability. Several case studies to illustrate the practical applicability of the techniques were briefly presented. This chapter sets the stage for the reader to start with a clear understanding of the terminology and expectations for the rest of the chapters.

1.5 Further Reading

Any journey into understanding dependability begins with [1, 2], where a comprehensive overview of the dependability terminology was presented. Another source of information is the IEC international vocabulary [4]. Two edited volumes by K. B. Misra provide excellent source material on all the topics discussed here [40, 41]. We caution the reader, though, that the terminology we use is somewhat different from that used in [41]. For instance, compare, and contrast our Figure 1.1 with Figure 1.2 in [41]. For software reliability engineering, a good source is [42]; for safe software in critical applications, [43]. More recent discussions on software failures and their mitigation can be found in [8, 24, 44]. Tortorella [15] presents detailed definitions of service-oriented dependability metrics. Computing of service-oriented metrics is discussed in [45]. The related field of risk assessment is covered in [33, 46].

References

[1] J.-C. Laprie, "Dependable computing and fault-tolerance," in *Proc. 15th Int. Symp. on Fault-Tolerant Computing (FTCS-15)*, 1985, pp. 2–11.

https://doi.org/10.1017/9781316163047.003 Published online by Cambridge University Press

[2] A. Avizienis, J. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 11–33, 2004.
[3] K. S. Trivedi, D. S. Kim, A. Roy, and D. Medhi, "Dependability and security models," in *Proc. 7th Int. Workshop on Design of Reliable Communication Networks (DRCN 2009)*, 2009, pp. 11–20.
[4] IEC 60050, *International Electrotechnical Vocabulary: Chapter 191: Dependability and Quality of Service*. IEC Standard No. 60050-191, 2nd edn., 2001.
[5] A. Avizienis and J.-C. Laprie, "Dependable computing: From concepts to design diversity," *Proceedings of the IEEE*, vol. 74, no. 5, pp. 629–638, 1986.
[6] B. Parhami, "From defects to failures: A view of dependable computing," *SIGARCH Comput. Archit. News*, vol. 16, no. 4, pp. 157–168, Sep. 1988.
[7] D. P. Siewiorek and R. S. Swarz, *Reliable Computer Systems: Design and Evaluation*, 3rd edn. A K Peters/CRC Press, 1998.
[8] M. Grottke and K. Trivedi, "Fighting bugs: Remove, retry, replicate, and rejuvenate," *Computer*, vol. 40, no. 2, pp. 107–109, 2007.
[9] D. Chen, S. Garg, and K. S. Trivedi, "Network survivability performance evaluation: A quantitative approach with applications in wireless ad-hoc networks," in *Proc. 5th ACM Int. Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems*, 2002, pp. 61–68.
[10] F. Cristian, "Understanding fault-tolerant distributed systems," *Commun. ACM*, vol. 34, no. 2, pp. 56–78, Feb. 1991.
[11] M. Grottke, A. Nikora, and K. Trivedi, "An empirical investigation of fault types in space mission system software," in *IEEE/IFIP Int. Conference on Dependable Systems and Networks (DSN)*, 2010, pp. 447–456.
[12] J. Musa, "Software reliability-engineered testing," *Computer*, vol. 29, no. 11, pp. 61–68, 1996.
[13] B. B. Madan, K. Goševa-Popstojanova, K. Vaidyanathan, and K. S. Trivedi, "A method for modeling and quantifying the security attributes of intrusion tolerant systems," *Performance Evaluation*, vol. 56, pp. 167–186, 2004.
[14] F. B. Schneider, "Blueprint for a science of cybersecurity," *The Next Wave*, vol. 19, no. 2, pp. 47–57, 2012.
[15] M. Tortorella, "Service reliability theory and engineering I: Foundations," *Quality Technology & Quantitative Management*, vol. 2, no. 1, pp. 1–16, 2005.
[16] E. Bauer and R. Adams, *Reliability and Availability of Cloud Computing*. Wiley-IEEE Press, 2012.
[17] K. S. Trivedi, D. Wang, and J. Hunt, "Computing the number of calls dropped due to failures," in *Proc. IEEE 21st Int. Symp. Software Reliability Engineering (ISSRE)*, 2010, pp. 11–20.
[18] FAA, *Certification Maintenance Requirements*, Advisory Circular num 25-19, Nov. 28, 1994.
[19] FAA, *System Design and Analysis*, Advisory Circular/Advisory Material Joint 25.1309, May 24, 1996.
[20] A. Ramesh, D. Twigg, U. Sandadi, and T. Sharma, "Reliability analysis of systems with operation-time management," *IEEE Transactions on Reliability*, vol. 51, no. 1, pp. 39–48, 2002.

[21] K. Hjelmgren, S. Svensson, and O. Hannius, "Reliability analysis of a single-engine aircraft FADEC," in *Proc. Annual Reliability and Maintainability Symposium*, 1998, pp. 401–407.

[22] A. Bobbio, D. Codetta-Raiteri, L. Portinale, A. Guiotto, and Y. Yushtein, "A unified modelling and operational framework for fault detection, identification, and recovery in autonomous spacecrafts," in *Theory and Application of Multi-Formalism Modeling*, eds. M. Gribaudo and M. Iacono. IGI-Global, 2014, ch. 11, pp. 239–258.

[23] W. E. Smith, K. S. Trivedi, L. Tomek, and J. Ackaret, "Availability analysis of blade server systems," *IBM Systems Journal*, vol. 47, no. 4, pp. 621–640, 2008.

[24] K. Vaidyanathan and K. S. Trivedi, "A comprehensive model for software rejuvenation," *IEEE Transactions Dependable and Secure Computing*, vol. 2, no. 2, pp. 124–137, Apr. 2005.

[25] IBM's WebSphere Application Server (WAS). [Online]. Available: www.ibm.com/software/webservers/appserv/was/

[26] BEA's Web Logic Server. [Online]. Available: www.bea.com/content/products/weblogic/server/

[27] S. Mondal, X. Yin, J. Muppala, J. Alonso Lopez, and K. S. Trivedi, "Defects per million computation in service-oriented environments," *IEEE Transactions on Services Computing*, vol. 8, no. 1, pp. 32–46, Jan. 2015.

[28] Back-To-Back User Agent (B2BUA) SIP Servers Powering Next Generation Networks – A Functional and Architectural Look At Back-To-Back User Agent (B2BUA) SIP Servers. RADVISION – an Avaya Company. [Online]. Available: www.radvision.com

[29] V. Kulkarni, V. Nicola, and K. Trivedi, "The completion time of a job on a multi-mode system," *Advances in Applied Probability*, vol. 19, pp. 932–954, 1987.

[30] P. Chimento and K. Trivedi, "The completion time of programs on processors subject to failure and repair," *IEEE Transactions on Computers*, vol. 42, pp. 1184–1194, 1993.

[31] A. Bobbio and M. Telek, "Task completion time in degradable systems," in *Performability Modelling: Techniques and Tools*, eds. B. R. Haverkort, R. Marie, G. Rubino, and K. S. Trivedi. Wiley, 2001, ch. 7, pp. 139–161.

[32] E. Zio, "Challenges in the vulnerability and risk analysis of critical infrastructures," *Reliability Engineering & System Safety*, vol. 152, pp. 137–150, 2016.

[33] E. Henley and H. Kumamoto, *Reliability Engineering and Risk Assessment*. Prentice Hall, 1981.

[34] IEC 61508, *Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems*. IEC Standard No. 61508, 2011.

[35] D. G. Nguyen and D. N. P. Murthy, "Optimal preventive maintenance policies for repairable systems," *Operations Research*, vol. 29, no. 6, pp. 1181–1194, 1981.

[36] D. Chen and K. Trivedi, "Analysis of periodic preventive maintenance with general system failure distribution," in *Proc. Pacific Rim Int. Symp. on Dependable Computing (PRDC)*, 2001, pp. 103–107.

[37] D. Chen and K. S. Trivedi, "Closed-form analytical results for condition-based maintenance," *Reliability Engineering & System Safety*, vol. 76, no. 1, pp. 43–51, 2002.

[38] Y. Huang, C. M. R. Kintala, N. Kolettis, and N. D. Fultoni, "Software rejuvenation: Analysis, module and applications," in *Proc. Int. Symp. on Fault-Tolerant Computing (FTCS)*, 1995, pp. 381–390.

[39] R. Pietrantuono, S. Russo, and K. Trivedi, "Software reliability and testing time allocation: An architecture-based approach," *IEEE Transactions on Software Engineering*, vol. 36, no. 3, pp. 323–337, 2010.
[40] K. B. Misra, *New Trends in System Reliability Evaluation*. Elsevier, 1993.
[41] K. B. Misra, *Handbook of Performability Engineering*. Springer-Verlag, 2008.
[42] M. R. Lyu, *Software Fault Tolerance*. John Wiley & Sons, 1995.
[43] N. Leveson, *Safeware: System Safety and Computers*. Addison-Wesley, 1995.
[44] M. Grottke, D. S. Kim, R. K. Mansharamani, M. K. Nambiar, R. Natella, and K. S. Trivedi, "Recovery from software failures caused by Mandelbugs," *IEEE Trans. Reliability*, vol. 65, no. 1, pp. 70–87, 2016.
[45] D. Wang and K. S. Trivedi, "Modeling user-perceived reliability based on user behavior graphs," *International Journal of Reliability, Quality and Safety Engineering*, vol. 16, no. 04, pp. 303–329, 2009.
[46] H. Kumamoto and E. Henley, *Probabilistic Risk Assessment and Management for Engineers and Scientists*. IEEE Press, 1996.

https://doi.org/10.1017/9781316163047.003 Published online by Cambridge University Press