Practices in Model Component Reuse for Efficient Dependability Analysis

Fumio Machida
Department of Computer Science
University of Tsukuba
Tsukuba, Ibaraki, Japan
machida@cs.tsukuba.ac.jp

ABSTRACT

Model-based dependability analysis provides an effective manner to evaluate and design the dependability of critical IT systems by abstracting the system architecture and operations. As the size and the complexity of systems increase, however, the process to compose the dependability model becomes complicated and time-consuming. Improving the efficiency of modeling process is practically an important challenge of dependability engineering. In this paper, we review the techniques for model component reuse that makes dependability model composition and analysis more efficient. In particular, component-based modeling approaches for reliability, availability, maintainability and safety analysis presented in the literature are summarized. In order to effectively apply model component reuse, we advocate the importance of asset-based dependability analysis approach that associates the reusable model components with underlying system development process. Finally, we discuss the necessary extensions of these techniques toward efficient dependability analysis for IoT systems which are significantly affecting real world.

CCS CONCEPTS

• Computer systems organization → Dependable and fault-tolerant systems and networks → Availability

KEYWORDS

Availability; Dependability; Internet of Things; Safety; Reuse.

ACM Reference format:

Fumio Machida. 2019. Practices in model component reuse for efficient dependability analysis. In *Proceedings of ACM Workshop on Education and Practice of Performance Engineering (WEPPE'19). April 7–11, 2019, Mumbai, India,* ACM, NY, NY, USA. *6 pages.* https://doi.org/10.1145/3302541.3311525

ICPE '19 Companion, April 7–11, 2019, Mumbai, India

ACM ISBN 978-1-4503-6286-3/19/04…$15.00
https://doi.org/10.1145/3302541.3311525

1 Introduction

Our society is increasingly relying on digital services composed of a number of IT system components such as hardware, software, storages and networks. Assuring dependability of IT systems is essential for performance engineering, as failures or outages of digital services running on the systems degrade the performance resulting in critical impacts on our society. However, dependability cannot be achieved solely through a single method or technique. Total and continuous efforts to improve the dependability are required, since the sources of errors and failures can reside in any system components in almost all the phases of the system lifecycle.

To assure the dependability of IT systems consisting of various system components across the lifecycle, model-based approach discussed in this paper have been widely studied and effectively used in practice. In model-based approach, to analytically assess the dependability of a system, the system configurations and behaviors are modeled in an abstract manner taking into account the internal or external uncertainties such as component failures or workload changes. The approach does not require any system tests or experiments on a real production system that are usually very expensive. In a system design phase, the target system even does not exist, therefore the model-based analysis is an essential means to dependability evaluation from its design. The approach is also effective in the system operation phase. When any system operation need to be changed, the results of the change can be easily estimated by model-based analysis without changing the current operations in the production system. For the technical details of model-based dependability analysis, the reader may refer to the book by Trivedi and Bobbio [1].

A practical issue when employing the model-based analysis is how to efficiently compose a good dependability model which precisely represents the target system configuration and behaviors. Even by analytic experts, composing a precise dependability model is a hard task especially when the target system is large and complex. It is one of the important challenges in dependability engineering to provide efficient modeling techniques and methodologies for assisting engineers to deal with the models for dependability assessment.

In this paper, we overview the recent studies and practices of model component reuse for dependability analysis. The means of model component reuse allows us to construct the whole system model more efficiently than making it from scratch. Past experiences and knowledge can be utilized in the form of reusable model components. The approach can also be helpful to reduce modeling errors caused by inherent system complexity or insufficient skill level of engineers, and thereby it improves the quality of dependability evaluation. First, in section 2 we review the studies related to model reuse technique for dependability analysis that includes reliability, availability, maintainability and safety analysis. As an example of model-based availability analysis, we introduce the component-based availability modeling framework that can help constructing the analytic model for system availability analysis from reusable model components. The framework has been applied for availability analysis of cloud service systems and data backup operations. Next, in section 3 we explain the importance of dependability modeling process that can leverage the reusable model components and overview the asset-based dependability analysis methodology. The methodology is associated with the software development practices in organizations where various artifacts created through software development projects are archived as organizational assets in anticipation of reuse in future similar projects. Dependability model components can also be archived together in the project assets so that any future dependability analysis can benefit from the asset. Finally, in section 4 we discuss the potential extension of these approaches to apply dependability analysis for recent IoT systems and services. IoT systems often connect cyber spaces with real world using a number of IoT devices such as sensors and actuators. Evaluating the actual impacts of such cyber-physical systems on the real-world applications becomes more important. We present some open issues for research and practice for the future study.

2 Reuse of dependability models

Availability, reliability, safety and maintainability are commonly known attributes of dependability [2]. Dependability models are used for abstracting system configurations, functions, and behaviors so as to analyze the quality or quantity of the dependability attributes. In order to effectively compose dependability models, various modeling formalisms and tools have been developed. SHARPE [3], SPNP [4], and Möbius [5] are the representative examples of such tools that are extensively used both in research and practice. The tools can automate some steps of modeling and analysis process with user-friendly interfaces and effective solution techniques. After composing the model for analysis of a particular system using the tools, a part of the model or the knowledge gained during the process can be reused for any future analysis. In this section, we review some existing studies on the techniques of model component reuse for efficiently analyzing system reliability, availability, maintainability and safety.

For reliability analysis, models like fault tree, reliability block diagram and their variants are used to represent logical structure of system components. The composed models can be used to derive the qualitative properties like a single point of failure as well as the quantitative measures such as the probability of system failure. The composition of these models is relatively easy since they are composed in a combinatorial manner using combinatorial logics that connect fundamental elements such as basic events or reliability blocks. Since a part of the model represents a fault event or reliability of a specific component, it can be reused repeatedly wherever the corresponding system component is used in systems. Component fault trees [6] aim to reuse sub-trees of a fault tree to construct a new one efficiently. The approach has been applied to reliability analysis of real systems [7]. Similar concept was also presented in hierarchical models that associate the output of different models with a higher-level model in a hierarchical manner with combinatorial logics. In [8], a hierarchical model was used to compose a fault tree for blade server system that was consisted of common system components such as CPUs, memories and disks. The component models for such common system components can be reused in other systems. For example, availability models for CPU and memory subsystems are commonly used in the different literature [8][9].

For availability and maintainability analysis, we often require state-space models to capture the state transitions of the system resulting from failure-recovery operation. The composition of state-space models is more cumbersome in comparison to non-state-space models because all the possible states and their transitions need to be carefully investigated. Even with domain experts, sometimes it is difficult to enumerate all the possible state transitions especially when the system has a number of inter-dependent components. Higher-level formalism such as stochastic Petri nets (SPNs) [10][11] and stochastic activity network (SAN) [12][13] give powerful solutions to complex modeling processes by automating the generation and analysis of state-space models. However, even with the aid of the automated composition approach, reusing the parts of state-space models faces another type of difficulty. For example, stochastic reward nets (SRNs) [4][14], a variant of SPN, has the concept of subnet that is a part of the whole model and can be inter-connected with other subnets. Dependencies among subnets need to be specified by the guard functions that define the conditions to fire the associated transitions. However, it is not always easy to specify a guard function because it requires a clear understanding of the target system behavior and relevant knowledge for specifying the dynamics of Petri nets by guard functions. In some cases, places or transitions in different subnets need to be merged or removed in a composition process so that the behavior of the Petri net correctly capture the real system behavior [15].

To mitigate the difficulty, the component-based availability modeling framework Candy [16] was presented, in which SysML models are used to specify the system configurations and they are transformed into SRNs for availability analysis. SysML is a semi-formal modeling language [17] inherited from UML for the purpose of specifying system engineering processes. OpenMADS also provides an open-source implementation version of the framework [18]. In this framework, the difficulty of guard

function assignments among SRN subnets are automatically resolved by the stereotyped associations defined among SysML model elements. In other words, types of system component dependencies are reused in SysML level models and the associated availability models are automatically generated from the SysML models. Although there has been many related work proposing automated generation of SPNs from system modeling languages such as UML, AADL and SysML [19][20][21], reusability of model components and the way to resolve component dependency are less discussed. Candy was applied to compose the availability models for web application system hosted on a cloud service infrastructure [18] and the defined dependency resolution patterns are reused in the analysis of data backup system [22].

For safety analysis, fault trees, failure mode and effect analysis (FMEA), hazard and operability study (HAZOP) are commonly adopted methodologies in practice. For fault tree analysis, as explained in the reliability analysis case, reuse of component fault tree enables effective composition of large-scale fault tree. FMEA is a bottom-up approach to analyze failure modes of system components and their consequences to the whole system, while HAZOP is a team-based process to identify the potential hazard situations using standardized guide words. Either FMEA or HAZOP do not need mathematical models since the main objective of the analysis is to find out potential hazard situations. There are, however, some practices to reuse the intermediate artifacts of the process that can be used for improving the efficiency of safety analysis in the future projects [23][24]. In [23], SysML models were used to specify the system functions with their failure modes and they were automatically translated into the corresponding FMEA. Similar to Candy, the information necessary for dependability analysis are reused in SysML level models. Whenever a common function is used in the design of another system, the result of corresponding FMEA can be reused so that the repeated manual FMEA process is omitted. Although HAZOP heavily relies on team discussion where automated generation is not an appropriate solution, it has been presented that the previous experiences of conducting HAZOP can be reused as knowledge for assisting other HAZOP analysis [24]. It is based on case-based reasoning which is the method to use previously obtained knowledge to solve new problems. The presented technique was implemented in a prototype tool KROSA and evaluated through three domain-specific cases with industrial experts. The application of case-based reasoning to HAZOP have also been studied in chemical industrial domain [25].

Most of the above-mentioned techniques for model component reuse in dependability analysis mainly focus on how to make models reusable and how to synthesize them together to an integrated model more efficiently. Nevertheless, dependability analysis in practice cannot be separated from the associated system development process that is a significant factor to determine the success of reuse approach. It is important to bridge the gap between dependability modeling techniques and actual system engineering process. The next section discusses the process perspectives where asset-based development process and software product lines are presented as examples.

3 Asset-based dependability analysis

To reuse dependability model components properly in real projects, the contextual information where each component is built plays a very important role. Software reuse in system development project likely to fail if any contextual information or specification of a software component is not intelligibly provided as it may lead to misuse of the software component. Similarly, without any contextual information, dependability model reuse studied in research does not work well in real projects. To preserve contextual information of software components to be reused, organizations can employ an asset-based development process. In an asset-based development project, all the artifacts created in software/system development processes are packaged to an organizational asset that is stored in a repository with metadata and is able to be retrieved easily by queries. The artifacts can include requirement specifications, design documents, source codes, test cases and associated data. Users can figure out the contextual information by traversing these artifacts in the repository and judge if the part of software can be reused in a new project.

Supported by asset-based development process, asset-based dependability analysis has been presented as a model-based dependability analysis methodology [26]. In many system development projects, system safety and availability analysis are required in early stages of the project to meet the system requirements. Under the asset-based dependability analysis, dependability models are associated with other software artifacts and are packaged to a project asset so that developers can effectively reuse the models in other projects in compliance with the usage context of the model. Dependability models may include reliability model, availability model, safety model and associated parameter values. Such reusable model components are linked to other software artifacts in the same asset following to the information scheme of the asset-repository [26].

The primal benefit of this approach is the enhanced efficiency of the dependability analysis by means of model component reuse. The organized repository and structured query interface provided by asset-based development process can effectively support discovering relevant model components that may be the part of the asset archived in the past projects. The approach also brings the benefits to the quality of dependability evaluation since the risk of misuse of model components is reduced by checking the contextual information. Moreover, the statistical confidence of reliability or availability estimation can be improved if the data for parameter values of model components is accumulated by the repetitive reuse of the asset [26].

Despite the benefits mentioned above, whenever asset-based dependability analysis is employed, the following aspects need to be considered as well in practice.

- **Systems thinking**: The concept of model component reuse is rooted in reduction in a sense that a system is assumed to be

Table 1: Techniques and methodologies for model component reuse to dependability analysis

	Reliability	***Availability***	***Safety***
Dependability models	Fault tree, Reliability block diagram, etc.	Markov chains, stochastic Petri nets, etc.	Fault tree, FMEA, HAZOP, etc.
Component-based modeling techniques	Component fault tree [6][7], hierarchical model [8][9].	Candy [16], OpenMADS[18], hierarchical model [8][9].	From SysML to FMEA [23], KROSA [24].
Methodologies for assisting model component reuse	Asset-based analysis [26].	Asset-based analysis [26].	Asset-based analysis [26], safety-critical software product line [29][30].

composed of the combination of common components. However, safety analysis often requires a system view as safety is an emergent property of the system which cannot be reduced into components' properties [27]. Without careful attention to the entire system behavior, asset-based safety analysis may fail to identify important hazard conditions.

- **Prospect for similar projects**: Compared to single independent project which does not necessitate to create the asset, asset-based development process requires the additional effort to create the asset in anticipation of the future component reuse. Such efforts may not be necessary if the asset is never used in the future. In terms of cost-effectiveness, it is imperatively important to properly estimate the volume of future similar projects which can benefit the asset reuse.
- **Asset maintenance cost**: In addition to asset generation cost, it is also important to take into account the cost of asset maintenance. All the artifacts in an asset is created in a specific software/system development project. Whenever any changes or updates of the project occur, the affected artifacts studied in the repository also need to be updated accordingly. For example, an availability model component should be revised if the behavior or function of corresponding system component is changed. In asset maintenance phase, it requires maintenance cost to keep all the artifacts consistent with the actual software system.

Although the asset-based approach requires such an initial investment and continuous maintenance efforts as well as a view of system thinking, when it aligns with the organizational strategy (e.g., to build a competitiveness in a target market), similar software products or systems are developed considerably faster with lower cost that can build a significant competitive advantage of the organization.

Software product line [28] is another important methodology to support software development process which attempts to promote software reuse. Software product line is typically employed in the organizations who have similar projects with specific needs of a particular market or mission area. For safety-critical software systems such as medical devices or automotive systems, the artifacts generated through safety-analysis can also be incorporated with the asset defined in the software product lines. However, since each project has some variable parts, safety analysis assets may not be directly reused to other projects. To address this issue, a state-based modeling approach was introduced to capture the variations of software product line so that safety models such as fault tree can be generated automatically across the product line [29]. More recently, a variability management tool is further integrated with model-based safety analysis technique so that the safety analysis artifacts can be reused through the software product lines [30]. These approaches also provide a systematic way to assist dependability model reuse by resolving the contextual dependence.

To summarize, component-based modeling techniques often aims to reuse model components, while they are not always beneficial in practice without adapting them to the relevant development process. Asset-based dependability analysis and safety-critical software product line can effectively accommodate model component reuse in the system development process. Table 1 summarizes the techniques and methodologies described in Section 2 and 3. Note that this is not a comprehensive survey result of existing literature about component-based modeling techniques and methodologies for assisting model component reuse.

4 Discussion and future challenges

In the final section, we discuss the future extensions of model component reuse techniques and methodologies for dependability analysis in the context of cyber physical systems. Recent advances of IoT services further increase the dependence of real-world to software systems. IoT systems monitor real world data and make decisions to control the world using advanced data analytics. As a result, dependability of software systems providing IoT services severely impacts on our lives and societies. The consequences of malfunctions or unavailability of software systems need to be carefully assessed in view of real world impacts.

For qualitative aspect, traditional safety analysis methods are capable to analyze the impacts of faults in software systems on its users or environments, thereby the methods can also be applied in design for future IoT systems. Nevertheless, compared to traditional IT systems which mainly run in cyber space, IoT systems may have multiple and continuous interaction to real world. In order to explore hazard situations across different domains (i.e., IT system domain and real world application domain), more advanced methodology might be necessary. System theoretic process analysis (STPA) was presented as a new methodology for safety analysis that looks into a control structure of a system instead of investigating component failures [27]. To explore potential hazardous situations of real world

applications, an analysis with control structure can be helpful. STPA has been applied for safety and security analysis of a power grid system as a case study of cyber-physical system [31]. How to assist such a human-intensive safety analysis process is still an important challenge.

For quantitative aspect, since reliability and availability analysis mainly applied to product or system level, quantitative impacts on the social environment cannot be evaluated adequately. To quantify such impacts, we may need more application-level or service-specific dependability measures on top of reliability and availability analysis. Performability [32], service availability [33], defects per millions [34], are the examples of such measures that have been studied in the research of dependability analysis. In fact, there are still gaps between the application-level dependability measures and the measures of social impacts that will be characterized in different time-scale and abstraction level. To bridge the gap, we will require more higher-level consideration that can be captured by a social system model representing dynamic interactions among social entities in real-world. A representative example of such dependability analysis was found in the study of safer city solution provided by a distributed video surveillance system [35]. In this study, system dynamics [36] are used as a social system model in order to characterize the impacts of performance of surveillance function on the crime risk in the city. While system dynamics only capture the causal relationships among the concerned variables in a rough sketch, they are still very useful to roughly estimate the real-world consequences of IoT services. The estimated results help IoT service providers to have a better understanding of their value propositions and to determine where to invest more.

One of the important challenges of system dependability analysis in IoT era is the development of model component reuse approach for social impact analysis. As dependability of IoT system impacts on real-world, many IoT applications need to be assessed its stoical impacts as a result of system dependability. Although model-based approach is a clue to the solution, it does not scale if social models with system dependability model needs to be constructed manually for every project. IoT systems sharing a common objective or common components can exploit the component reuse approach and asset-based development process as studied and experienced in dependability engineering. To step forward to this direction, the followings are considered as new challenges.

- What kind of higher-level measures in a social context need to be quantified which were not directly addressed in system level dependability analysis (e.g., crime risk, safety level, traffic congestion, customer satisfaction, etc). The measures are highly context-dependent even using the same function or service. To clarify the context and determine the measures of interest, we may require the communications among several stakeholders including engineers, modeling experts, domain experts and actual users.
- To quantify the measures of interests, how we can reuse the part of socio-ICT models. Two or more different IT systems may be connected with the same social system model in a part. In contrast, there is also the case a new IoT system may have connections to different social models. Since social model and system model can probably be generated by different users, a systematic way to connect those models will be required.
- To encourage model component reuse, how we should extend the asset structure for dependable IoT system development. For instance, information scheme for social model needs to be defined so that it can be traversed in the asset repository by query.
- It is also important to consider how to educate engineers to understand the model-based approach and reuse the relevant model components for dependability analysis in asset-based development process. In practice, any development process does not work well without educated users. A sort of framework or development environment that can assist engineers to build experiences through practices might be required.

Finding answers to these questions could be interesting future research avenues.

ACKNOWLEDGMENTS

This work has been based on the practical experience in model-based dependability analysis acquired through the previous research projects. The author would like to thank all the collaborators in my past projects building this experience.

REFERENCES

[1] K. S. Trivedi and A. Bobbio, Reliability and availability engineering : modeling, analysis and applications, Cambridge University Press, 2017.
[2] A. Avizienis, J.C. Laprie, B. Randell and C. Landwehr, Basic concepts and taxonomy of dependable and secure computing, IEEE Trans. on Dependable and Secure Computing, vol. 1, no. 1, 2004.
[3] K. S. Trivedi and R. Sahner, SHARPE at the age of twenty two, SIGMETRICS Perform. Eval. Rev., vol. 36, no. 4, pp.52-57, 2009.
[4] G. Ciardo, J. Muppala, and K. S. Trivedi, SPNP: Stochastic Petri Net Package, In Proc. of the Third International Workshop on Petri Nets and Performance Models, pp. 142–151, 1989.
[5] D. Deavours, G. Clark, T. Courtney, D. Daly, S. Derisavi, J. M. Doyle, W. H. Sanders, and P. G. Webster, The Möbius framework and its implementation, IEEE Trans. on Softw. Eng., vol. 28, no. 10, pp.956-969, 2002.
[6] B. Kaiser, P. Liggesmeyer, and O. Mäckel. A new component concept for fault trees, In Proc. of the 8th Australian workshop on Safety critical systems and software, pp. 37-64, 2003.
[7] K. Hofig, A. Joanni, M. Zeller, F. Montrone, M. Rothfelder, R. Amarnath, P. Munk, A. Nordmann, Model-based reliability and safety: reducing the complexity of safety analyses using component fault trees, In Annual Reliability and Maintainability Symposium (RAMS), pp. 1-7, 2018.
[8] W. E. Smith, K. S. Trivedi, L. Tomek, J. Ackeret, Availability analysis of multicomponent blade server systems, IBM Systems Jornal, 2008.
[9] D. Kim, F. Machida, and K. S. Trivedi, Availability modeling and analysis of a virtualized system, In Proc. of IEEE Int'l Symp. Pacific Rim Dependable Computing (PRDC 2009), 2009.
[10] M. K. Molloy, Performance Analysis Using Stochastic Petri Nets, IEEE Trans. on Computers, vol. 31, no. 9, pp. 913-917, 1982.
[11] G. Florin, and S. Natkin, Evaluation based upon stochastic Petri nets of the maximum throughput of a full duplex protocol, Application and Theory of Petri Nets, Springer, pp. 280-288, 1982.
[12] J. F. Meyer, A. Movaghar, and W. H. Sanders, Stochastic activity networks: Structure, behavior, and application, In Proc. International Workshop on Timed Petri Nets, pp. 106–115, 1985.
[13] W. H. Sanders and J. F. Meyer, Stochastic activity networks: For- mal definitions and concepts, In Lectures on Formal Methods and Performance Analysis, First EEF/Euro Summer School on Trends in Computer Science, ser. LNCS, no. 2090, pp. 315–343, 2001.
[14] K. S. Trivedi, Probability and Statistics with Reliability, Queuing, and Computer Science Applications, John Wiley, New York, 2001.

[15] F. Machida, D. Kim, and K. S. Trivedi, Component-based availability modeling for cloud service management, In Supplemental Proc. of 21st International Symposium on Software Reliability Engineering, 2010.
[16] F. Machida, E. Andrade, D. Kim, K. S. Trivedi, Candy: component-based availability modeling framework for cloud service management using SysML, In. Proc. of Int'l Symp. on Reliable and Distributed Systems (SRDS), pp. 209-218, 2011.
[17] S. Friedenthal, A. Moore, and R. Steiner, A practical guide to SysML: systems modeling language, Morgan Kaufmann, 2014.
[18] E. Andrade, M. Alves, R. Matos, B. Silva, P. Maciel, Openmads: an open source tool for modeling and analysis of distributed systems, In Proc. of International Conference on computer safety, reliability, and security. pp 277–284, 2013.
[19] S. Distefano, M. Scarpa, and A. Puliafito, From UML to Petri nets: the PCM-based methodology, IEEE Trans. on Soft. Eng., vol. 37, no. 1, pp. 65-79, 2010.
[20] A. E. Rugina, K. Kanoun, and M. Kaâniche, A system dependabiliy modeling framework using AADL and GSPNs, Architecting Dependable Systems IV, vol. 4615, LNCS, R. de Lemos, C. Gacek, and A. Romanovsky, Eds.: Springer-Verlag, pp. 14-38, 2007.
[21] E. Andrade, P. Maciel, G. Callou and B. Nogueira, A methodology for mapping sysML activity diagram to time Petri net for requirement validation of embedded real-time systems with energy constraints, In Proc. of the Third International Conference on Digital Society, pp. 266-271, 2009.
[22] R. Xia, X. Yin, J. Alonso, F. Machida and K. S. Trivedi, Performance and Availability Modeling of IT Systems with Data Backup and Restore, IEEE Trans. on Dependable and Secure Computing, vol. 11, no. 4, pp. 375-389, 2014.
[23] P. David, V. Idasiak, F. Kratz, Reliability study of complex physical systems using SysML, Reliability Engineering and System Safety, vol. 95, pp. 431 – 450, 2010.
[24] O. Daramola, T. Stalhane, G. Sindre and I. Omoronyia, Enabling hazard identification from requirements and reuse-oriented HAZOP analysis, In Proc. of 4th Int'l Workshop on Managing Requirements Knowledge (MARK), pp. 3-11, 2011.
[25] J. Zhao, L. Cui, L. Zhao, T. Qui, and B. Chen, Learning HAZOP expert system by case-based reasoning and ontology, Computer and Chemical Engineering, vol. 33, no. 1, pp. 371-378, 2009.
[26] F. Machida, J. Xiang, K. Tadano, and S. Hosono, An asset-based development approach for availability and safety analysis on a flood alert system, In International Workshop on Recent Advances in the DependabIlity AssessmeNt of Complex systEms (RADIANCE), pp. 51-56, 2015.
[27] N. G. Leveson, Engineering a safer world: Systems Thinking Applied to Safety, MIT Press, 2012.
[28] P. Clements and L. Northrop, Software product lines: practices and patterns, SEI series in software engineering, Addison–Wesley, 2001.
[29] J. Liu, J. Dehlinger, and R. Lutz, Safety analysis of software product lines using state-based modeling, Journal of Systems and Software, vol. 80, no. 11, pp. 1879–1892, 2007.
[30] A. L. Oliveira, R. Braga, P. C. Masiero, Y. Papadopoulos, I. Habli, T. Kelly, Model-based safety analysis of software product lines, International Journal of Embedded Systems, vol. 8 no. 5/6, 2016.
[31] I. Friedberg, K. McLaughlin, P. Smith, D. Laverty, and S. Sezer. STPA-SafeSec: Safety and security analysis for cyber-physical systems. Journal of Information Security and Applications, vol. 34, part 2, pp. 183-196, 2016.
[32] J. F. Meyer, On evaluating the performability of degradable computing systems, IEEE Transactions on Computers, vol. 29, no. 8, pp. 720-731, Aug, 1980.
[33] D. Wang, and K. Trivedi, Modeling user-perceived service availability, In Proc. of International Service Availability Symposium, pp.107-122, 2005.
[34] S. Mondal, X. Yin, J. Muppala, J. Alonso Lopez, and K. Trivedi, Defects per million computation in service-oriented environments, IEEE Transactions on Services Computing, vol. 8, no. 1, pp. 32–46, 2015.
[35] F. Machida, M. Fujiwaka, S. Koizumi, and D. Kimura, Optimizing resiliency of distributed video surveillance system for safer city, In Supplemental Proc. of International Symposium on Software Reliability Engineering (ISSRE), pp. 17-20, 2015.
[36] J. D. Sterman, Business Dynamics: Systems thinking and modeling for a complex world, New York: McGraw, 2000.