Article citation info:
WALKOWIAK T. Simulation based availability assessment of services provided by web applications with realistic repair time. Eksploatacja i Niezawodnosc – Maintenance and Reliability 2014; 16 (2): 341–346.

Tomasz WALKOWIAK

SIMULATION BASED AVAILABILITY ASSESSMENT OF SERVICES PROVIDED BY WEB APPLICATIONS WITH REALISTIC REPAIR TIME

SYMULACYJNA OCENA GOTOWOŚCI USŁUG SYSTEMÓW INTERNETOWYCH Z REALISTYCZNYM MODELEM CZASU ODNOWY*

Paper presents a numerical method for determining changes of availability of services provided by web applications in time. It takes into account the reliability and functional aspects. The reliability analysis allows determining the probability that the system is operational at a given time. The analysis includes the structure of a computer system, random times to failures (hardware or software – related to security breaches) and the detailed repair time model. The repair time model takes into account working hours of administrators and a time associated with delivering components to exchange. Functional analysis allows the determination of the probability that the user will be served in less than a specified time limit. It is based on modelling the interaction between a user and a server as a sequence of tasks on one or more hosts. It takes into account the variability of workload over a week and web application parameters such as the choreography of service, allocation of tasks on hosts and technical parameters of hosts and tasks. The described method was the basis for development of a Monte-Carlo simulator that allows variability of service availability over a week to be calculated. The paper contains the numerical results of the sample analysis.

Keywords*: web application, availability, reliability, Monte-Carlo simulation.*

W artykule przedstawiono metodę numerycznego wyznaczania zmian gotowości usług internetowych w czasie. Bierze ona pod uwagę aspekty niezawodnościowe i funkcjonalne systemu komputerowego świadczącego usługi. Analiza niezawodnościowa pozwala na wyznaczenie prawdopodobieństwa, że system będzie zdatny w danym momencie. Uwzględnia ona strukturę systemu komputerowego, losowe czasy do uszkodzeń (sprzętowych jak i oprogramowania związanych z naruszeniami zabezpieczeń) oraz szczegółowy model odnowy biorący pod uwagę godziny pracy administratorów oraz czas związany z dostarczeniem elementów do wymiany. Analiza funkcjonalna pozwala na wyznaczanie prawdopodobieństwa, że użytkownik zostanie obsłużony w czasie mniejszym niż zadany. Oparta jest ona na modelowaniu procesu realizacji usługi jako sekwencji zadań wykonywanych na jednym lub kilku komputerach. Bierze ona pod uwagę zmienność intensywności napływu użytkowników w ciągu tygodnia oraz parametry takie jak: sekwencję zadań, alokację zadań na komputerach oraz parametry techniczne komputerów i zadań. Opisana metoda była podstawą do stworzenia aplikacji komputerowej wyznaczającej techniką symulacji Monte-Carlo zmienność gotowości systemu w ciągu tygodnia. Artykuł zawiera numeryczne rezultaty przykładowej analizy.

Słowa kluczowe*: usługa internetowa, gotowość, niezawodność, symulacja Monte-Carlo.*

1. Introduction

Services provided on Internet are nowadays the basis of many enterprises.Their correct and uninterrupted operation is an important aspect of business success. Users of services provided by web applications can very easily move to another portal. Therefore, essentials are methods that allowassessing the service availability. Especially important is to investigate the influence of various parameters related to the configuration and maintenance of the web application on service availability.

Availability in case of repairable systems, to which web systems belong to, is understood as the probability that the system is operating at a specified time. Therefore to define availability of web application we need to understand what it means that service is operating. We will follow definition from [29]: "the system is operational when required tasks are performed correctly within the defined time limits". It includes two important aspects analysed in this paper.

In the first, this is the reliability aspect. The system must operate. It cannot be failed. Secondly, the functional aspect must be considered. The user who will be waiting for a response for longer than a few seconds will treat service as failed, and possibly will look for another website with similar functionality.

There are a large number of studies examining the reliability of websites or web applications. However, in most cases the reliability analysis is based on Markov processes [3] and the analysed system is modelled as a simple serial one [18]. In this paper, we propose to focus on the more realistic modelling of how repairs of computer systems are carried out. It will take into account working hours and weekends of system administrator. Analysis of the functional aspect is based on the commonly used solution in the analysis of complex technical systems: modelling and simulation [4]. Modelling is focussed on a process of execution of a user request that is described as a sequence of tasks realised on technical services provided by the system [26]. The computer simulation is responsible for calculation of service availability. It is based on a time event simulation with Monte Carlo analysis [8]. The simulator allows to calculate the probability that the user will receive an answer in a time less than the given threshold under given workload.

(*) Tekst artykułu w polskiej wersji językowej dostępny w elektronicznym wydaniu kwartalnika na stronie ***www.ein.org.pl***

2. State of art and related works

Reliability analysis of complex technical systems (to which web systems belong to) is a subject of researches for many years. It is described in a number of good textbooks, for example in a book by Barlow and Proshan [3]. The problem of repair time modelling is most often solved by the use of the exponential distribution [1, 3, 18]. This is due to popularity of Markov processes in reliability analysis. In relatively fewer cases researches are using other distributions for repair time modelling. For example, in [13, 14] authors assume that the repair time could be described by the geometric process(proposed by Lama in [16]). Moreover, most often it is assumed [1, 3] that the repair time begins immediately after the failure. However, the other scenario is also analysed in the literature [7, 30]. It assumes that the time from the component's failure to the completion of the repair is composed of two different periods: waiting and real repair. The waiting period is modelled as a random variable independent of the lifetime of the component. There are also works [14] that apply a mix approach, in which the repair time with a certain probability consists of the waiting and the real repair time or only of the real repair time.

The analytical approaches (mentioned above) have significant limitations [21] which includes the limited distributions (mostly exponential), narrow applicability (only to a special class of systems) and poor compliance with reality. To overcome these limitations the simulation approach is applied [4]. The use of computer simulation [5], in particular the Monte-Carlo simulation [8], to analyse the reliability of complex systems is also described by many researches, for example in the monograph [15] or in papers [6, 21].

Methods of availability analysis of web systems that are described in the literature are based on the analytical models [9, 12] or on real system monitoring [2, 11, 22]. In contrast, the method described in this paper is based on the use of computer simulation. The simulation is used for determining the user response time as a function of the user request rate (chapter 5).

The response time can be determined experimentally by performing load tests of real systems (using tools as: Funkload, Apache JMeter, Rational Performance Tester or Developer Tools from Microsoft), estimated by analytical models[19, 25] (in most cases based on queuing networks [17]) or by simulation techniques [24] (also very often based on queuing models).

3. Service availability

One of the quality metrics used for web systems is availability, defined as the probability that a user will be properly serviced by a system at a specific time, i.e.:

$$A(t) = \Pr(U_t), \tag{1}$$

where U_t is an event that system provides correct responses.

Let S_t be an event that the computer system on which the web application is deployed is operational (there are no failures). Using an equation for conditional probabilities the service availability could be calculated as:

$$A(t) = \Pr(U_t \mid S_t)\Pr(S_t) + \Pr(U_t \mid \overline{S_t})\Pr(\overline{S_t}). \tag{2}$$

In case of analysed class of systems a user will benot correctly serviced if the computer system is failed. Therefore, $\Pr(U_t \mid \overline{S_t}) = 0$. It simplifies the above equation to:

$$A(t) = \Pr(U_t \mid S_t) \cdot \Pr(S_t). \tag{3}$$

The above equation is the basis of a numerical method that allows to calculate the value of the service availability. For readability, we will introduce names for individual factors of the product (3). Let $\Pr(S_t)$ be calledthe system availability, which is understood as the probability that the computer system, on which the web application is deployed is operational, is not failed. Let's $\Pr(U_t \mid S_t)$ call the functional availability defined as the probability that the user will receive an answer in a time less than a given threshold (t_{max}) under assumption that computer system is operational. Therefore, the functional availability is given by the equation:

$$\Pr(U_t \mid S_t) = \Pr(responsetime(T) < t_{\max} \mid T = t \wedge S_t). \tag{4}$$

4. Reliability model

The paper considers a very wide class of web systems. In general some business services are accessed by the user using web interactions. The service responses are dynamically computed by the service components, which also interact with each other using the client-server protocols. On the low level the system consists of interconnected hosts with installed software (technical services) responsible for providing business service. In the considered approach to web system modelling, the hosts are the basic components of the system infrastructure. Thus, all the failures are attributed to them (of course they origin from hardware or software components faults).

There are numerous sources of failures in web systems. These encompass hardware malfunctions (transient and persistent), software bugs, human mistakes, viruses, exploitation of software vulnerabilities, malware proliferation, drainage type attacks on system and its infrastructure (such as ping flooding, DOS). We propose to model all of them from the point of view of resulting failure.

The most common source of host failures are caused by some exploitation of security vulnerability, often a proliferation of bogus service requests that lock up all the server resources. This is a common consequence of insufficient security metrics (for example caused by wrong firewall rules or a lack of antivirus software, etc.). It may result in inoperational host failure or in a lost in performance. In the first case the host cannot process services that are located on it, these in turn do not produce any responses to queries from the services located on other hosts. In the second case the host can operate, but it cannot provide the full computational resources, causing some services to fail or increasing their response time above the acceptable limits.

A web system from the point of reliabilityis a repairable system. The reliability analysis requires to model failures occurrence, their results and the repair time. For simplicity, we will assume that a failure of any host (caused by hardware or software component fault) will result in a web systemfailure [18].

Moreover, we propose to model system failures by a set of independent host failures.The occurrence of a host failure is described by a random process. Where, the time to failure is modelled by the exponential distribution (like in the Markov model). In case ofa repair time we propose to model it in details taking into consideration repairman working hours and a time required to replace the failed component or a time needed to react on a software failure.

Let's assume that the system administrator works five days a week in selected hours. And only during these hours any failure could be discovered and any repair operation could be started (for example software reinstallation). Moreover, a time required to deliver new equipment by courier will be taken into consideration. We assume that the system component to be replaced will be available on the next working day at a given hour.

Let's divide host failures into two groups: failures that require delivery of a new component and ones that don't (like failures con-

nected with software or cases when required component is available onsite). Let's mark the probability of the first type failures as *pf*.

Taking into account all these assumptions we could distinguish following elements of period from the component's failure to the completion of the repair (withsystem administrator working hours from 8 am to 5 pm with one hour lunch break):

- time to failure discovery– equal to zero if a failure happens during working hours, in other cases it is a time to next working period: the finish of lunch break (2 pm) or next day (8 am), for example if a failure will occur just after 5 pm on Friday, it will be equal to 63 hours.
- failure analysis time – a time needed to discover a failure reason, we propose to model it by a random variable with a truncated normal distribution, if the failure analysis overlaps with not working hours, it is enlarged by a time to next working period (1 hour in case of overlapping witch lunch time, 15 hours during working days evenings and nights or even 63 hours in case of Friday afternoon);
- delivery time – with a probability of 1-*pf* it is equal to zero (it allows to model an operating system or software component failure), in other cases equal to a time to next working day (it models the delivery of a new system component by courier);
- component replacement time –time required to replace a failed hardware component or to reinstall failed software component, it is modelled by a random variable with the truncated normal distribution plus extra value in case of overlapping with not working hours(like in the case of failure analysis time).

Presented reliability model includes random values with exponential, truncated normal and discrete distributions. Therefore, it is hard to calculate the system availability by analytical methods (like Markov or semi-Markov chains). But it could be done using the numerical approach, strictly speaking by Monte-Carlo simulation [8]. Therefore, authors developed the reliability simulator. It was implemented in C++ within the Scalable Simulation Framework (SSF) [20]. SSF is an object–oriented API, a collection of class interfaces with prototype implementations. For the purpose of simulating reliability states the Parallel Real-time Immersive Modeling Environment (PRIME) [20] implementation of SSF was used. The main reason of using SSF was the fact that it was also used by authors for the development of the functional simulator of web system described in the next chapter.

Operation of the simulator is based on repetition of the failure and repair process simulation. Following simulations differ in the values of random variables occurring in the analysed system. Knowing the distribution of these variables (input data for simulation), one can get information about the behaviour of the system in the most probable cases. By observing changes of some metric values, the information about their distributions and any other statistics could be easily gathered. A single simulation last for a fixed length (in the analysed case a multiplication of seven days).

The reliability model contains some arbitrary assumptions, such as working hours of administrator or a number of shifts. The used method allows changing the assumptions and thus different scenarios could be also simulated. However, it could require changes in the source code of the simulator. Sample numerical results are presented in chapter 6.

5. Functional model

5.1. Response time prediction

As it was presented in chapter 3, the service availability depends on the system reliability (described in the previous chapter) and on the web system performance measured by the user response time. To calculate the user response time we need to analyse the process of user request execution. It is performed from the point of view of business service realised by web system [10].

The user initiates the communication requesting some tasks on a host, it could require a request to another host or hosts, after the task execution a host responds to requesting server, and finally the user receives the respond. Requests and responses of each task give a sequence of a user task execution, according to a given choreography. It could be described as a sequence of requests:

$$choreography(u) = \left(c(task_{b_1}), c(task_{b_2}), ..., c(task_{b_n}) \right), \quad (5)$$

where $c(task_{b_i})$ represents a request ($\Rightarrow$) to $task_{b_i}$ or a response ($\Leftarrow$) from a given task. Some tasks after completing the calculation return-reply to the sender. Other tasks before sending the response can call other tasks. It is important to mention that each task is deployed on one of hosts and it cannot be changed during running of the system.

Based on the above model (a detailed description can be found in [26]) the response time is equal to the time required for communication between hosts, on which tasks are deployed and the time required to perform each of tasks. For following choreography:

$$\begin{aligned} &choregraphy(u) = \\ &u \Rightarrow task_1 \Rightarrow task_2 \Leftarrow task_1 \Rightarrow task_3 \Leftarrow task_1 \Leftarrow u \end{aligned} \quad (6)$$

the user response time could be calculated as a sum:

$$\begin{aligned} resoponsetime(u) = delay(h_0, h_1) + pt(task_1) + delay(h_1, h_2) + pt(task_2) + \\ delay(h_2, h_1) + delay(h_1, h_3) + pt(task_3) + delay(h_3, h_1) + delay(h_1, h_0) \end{aligned}, \quad (7)$$

where $delay(h_0, h_1)$ is a time of transmitting a request from host h_i to h_j, and $pt(task)$ is a task processing time (on a host on which the task is deployed). For contemporary web systems not associated with media broadcasting the time of transmitting a request can be modelled by a random variable with a truncated normal distribution [26].

The processing time of each task depends on the type of task (its computational complexity), the type of a host (its performance) and of its load (number of other requests being handled concurrently). The last of these parameters is not a constant value over time. It depends on the workload (number of users) and its changes. Processing time is difficult to be determined in an analytical way, that is why the computer simulator based on the Monte- Carlo technique [8] was developed [27]. It was implemented, as the simulator discussed in the previous chapter, in the Prime SSF [20]environment. For a given choreography (for example described in WS-CDL), deployment of tasks to hosts, configuration of hosts (processor performance and number of cores), technical service parameters (types and configuration of web servers), computational complexity of each task, and model of a user (like a number of users) the developed software allows to determine the user response time.

5.2. Functional availability

Mentioned simulator software allows to determinate in a numerical way the probability that the user will receive an answer in a time less than a given threshold in a function of workload (described by intensity of user requests):

$$Af(\lambda) = \Pr(responsetime < t_{\max} \mid workload = \lambda). \quad (8)$$

This metric is a numerical representation of client's perception of particular business service quality. It measures the probability that the user will not resign from active interaction with the service due too long service response time.

If we know how the intensity of user requests varies over a time ($workload(t)$) we could calculate the functional availability as:

$$\Pr(U_t \mid S_t) = Af(workload(t)) . \quad (9)$$

The values of the workload function depend on the type of offered service. For example, websites used by students have the greatest influx of users in the period just before exams and mostly at night. In the sample analysed in the next section, it is assumed that the workload is periodic with a period equal to one week. Daily and weekly variability is often observed in public services. This can be noticed analysing the variability of traffic over the networks of large Internet service providers.

6. Availability analysis of a case system

6.1. System availability

All the simulation results that illustrate the presented method were computed using a real-life example of the case-study web system. The system consists of six hosts. There were five hosts with business components communicating according to selected choreography and one router with a firewall. Since, today's computer devices are characterised by high reliability parameters, the intensity of failures was set to one year per year. The repair model parameters were set as follows:

- mean failure analysis time – 3 hours,
- mean component replacement time – 1 hour,
- probability of hardware failure *pf* (requiring a component to be delivered by courier) – 0.2,
- working hours of administrator – 8 am to 17 pm, with one hour break for lunch,
- standard deviation of truncated normal distributions – 20% of its mean values.

The achieved results for simulating 1,000 weeks 50,000 times are presented in Fig. 1. The changes of system availability during a week could be noticed. The lowest value is achieved on Monday. It agrees with authors experience, that Monday is a day that there isthe most work to be done by system administrator. It is caused by the fact that the administrator is not working during weekends and failures that occurred during that time are maintained on the next working day.

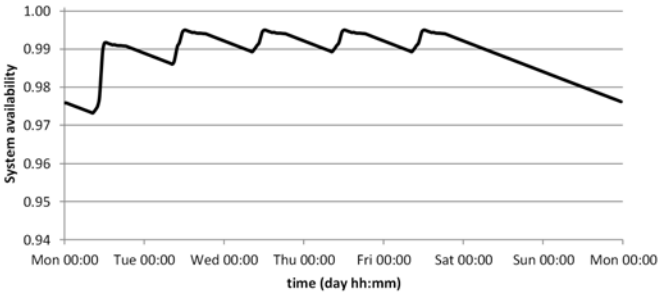

Fig. 1. System availability changes over a weekachieved by computer simulation of the case system

6.2. Functional availability

Another factor contributing to the service availability is the functional availability, defined as the probability that the user will obtain a response in time less than the given limit, assuming that the computer system running the service is not failed. As already mentioned the analysed system consists of six hosts and performs selected choreography. The time limit was set to 12 seconds. The results obtained from simulation, i.e. functional availability in a function of the user request rate, are shown in Fig. 2. Assuming weekly workload variations shown in Fig. 3, the functional availability in a function of time is presented in Fig. 4.

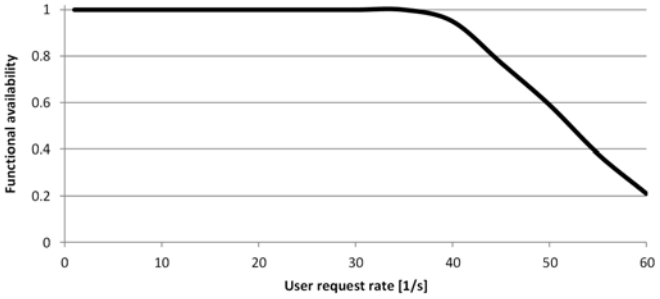

Fig. 2. Functional availability in a function of user request rate

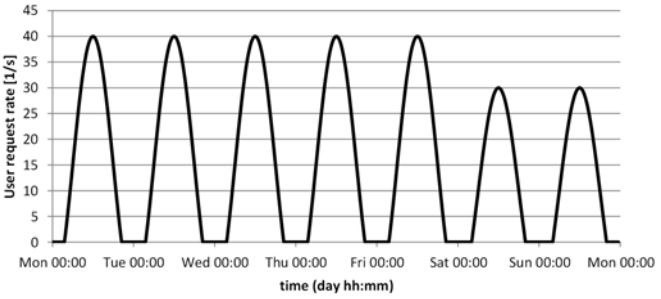

Fig. 3. Assumed changes in workload over a week

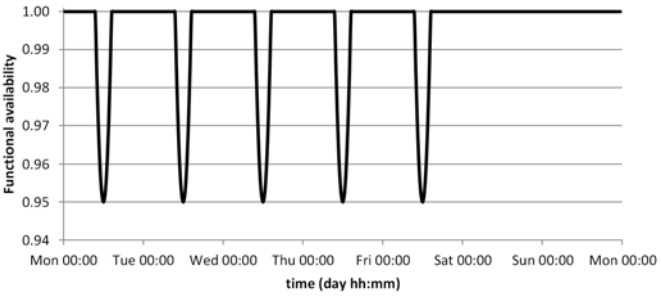

Fig. 4. Functional availability changes over a week for thecase system

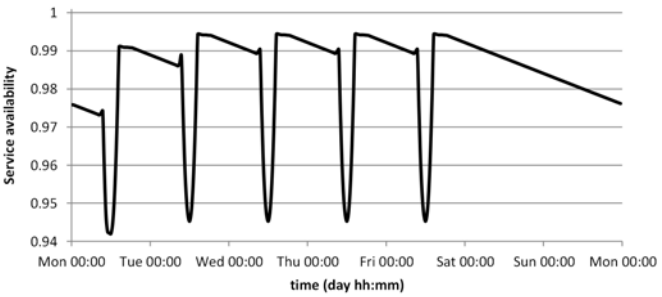

Fig. 5. Service availability changes over a week for thecase system

6.3. Service availability

According to equation (3) the product of the system (Fig. 1) and functional (Fig. 4) availability gives the services availability (Fig. 5) for the case web system. The daily variations of service availability are results of the diurnal variation in the workload and administrators working hours. In addition, there is a noticeable decline of availability in the

weekend reaching its minimum on Monday around noon. The minimum value on Monday is linked to the accumulation of failures from Saturday and Sunday and daily maximum of the user request rate.

7. Final remarks

7.1. Conclusions

This paper presented a method for determining changes of availability of service provided by web applications in time (over a week). The method considers reliability and functional aspects. The reliability analysis takes into account the structure of the system (number of hosts), random failure occurrence (modelled by exponential distributions), failure analysis time (modelled by truncated Gaussian distribution), component replacement time (next working day) and working hours of system repairman (administrator).

Functional analysis allows to determinate the probability that the user will receive an answer in a time less than the given threshold. Functional model parameters include the web system choreography (interaction between tasks), system structure (number of hosts, host performance, task allocation on hosts and task execution time) and workload (changes in the user request rate over a week).

Numerical calculations were performed using the simulator developed by authors. The use of Monte-Carlo simulation allowed a flexible functional and reliability model of web system to be created. However, the presented approach has some significant drawbacks. Firstly, the simulation time associated with multiple repetitions (the basic principle of Monte-Carlo simulation) in case of the functional simulator could be large. Secondly, changes in the structure of the model (but not in its parameters) may require changes in the source code of the simulator.

The presented approach provides an easy way to explore the impact of changes in the system maintenance (such as working hours of administrators) or changes in functional parameters of the system (e.g. other allocation of tasks, increased level of security or performance of hosts) on the significant from the point of view of user's parameter: the service availability.

7.2. Future work

The presented method includes some assumptions that limit its applicability. First of all, the web system is modelled as a serial reliability system, i.e. a failure of any system component (hosts) results in a failure of the system. It is not true for web systems that useload balancing techniques or systems deployed in computing cloud. In such cases the failure of one of hosts (hardware or software) results in degradation of system performance not in a failure of the whole system.

The second assumption is to model failure results as an inoperation of a single host. However, some of security breaches, like viruses or malware, could result in degradation of hosts performance. So the web system is able to answer to user requests, only response time in longer. It is possible to extend the proposed two–state (operational – failed) reliability model to a multistate one. Each state will be described by a state of each system component (host). Whereas, each component could be in one of three states: operational, degraded performance and failed. The state probability for each of *N*-states (let mark them as S_i) could be calculated by a simple extension to simulator described in chapter 4. Using simulator described in chapter 5, it is possible to calculate functional availability

($\Pr(responsetime(T) < t_{\max} \mid T = t \wedge S_i)$) in each of states and finally the functional availability of service as:

$$A(t) = \sum_{i=1}^{N} \Pr(responsetime(T) < t_{\max} \mid T = t \wedge S_i) \cdot \Pr(S_i, t) \qquad (10)$$

The main problem of this solution is a number of states. For *n* independent elements, which could be failed, it gives 2^{2n} states [28]. So for the case system analysed in this paper it gives 4096 states. For each of states a time consuming functional simulation has to be performed (chapter 5). The problem could be solved by limiting the sum in (10) to the most probable states and skipping less probable states.

Another area of future work concerns the application of the proposed method in the analysis of websites deployed in computing cloud. In such case, the hardware failures playa much smaller role. This is due to the use of virtualization techniques and migration of virtual machines in the cloud in case of failures. However, the most frequently occurring software failures caused by security breaches and the process of recovery of the system after such failures can be modelled in the same way as described in this work.

The presented work was supported by the Polish National Science Centre under grant number N N516 475940.

References

1. Aven T, Uwe J. Stochastic Models in Reliability. New York: Springer, 1999.
2. Banerjee S, Srikanth H, Cukic B. Log-based reliability analysis of Software as a Service (SaaS). Proceedings of the 21st International Symposium on Software Reliability Engineering 2010; 1: 239-248.
3. Barlow R, Proschan F. Mathematical Theory of Reliability. Philadelphia: Society for Industrial and Applied Mathematics, 1996.
4. Birta L, Arbez G. Modelling and Simulation: Exploring Dynamic System Behaviour. London: Springer, 2007.
5. Clymer J. Simulation-based engineering of complex systems. Wiley Series in Systems Engineering and Management, Wiley-Interscience, 2009.
6. Faulin J, Juan A, Serrat C, Bargueño V. Predicting availability functions in time-dependent complex systems with SAEDES simulation algorithms. Reliability Engineering & System Safety 2008; 93(11): 1761-1771.
7. FinkelsteinM. A scale model of general repair, Microelectron. Reliab 1993; 33(1): 41-44.
8. Fishman G. Monte Carlo: Concepts, Algorithms, and Applications, Springer-Verlag, 1996.
9. Gokhale SS, Jijun L. Performance and Availability Analysis of an E-Commerce Site. Computer Software and Applications Conference, 2006; 1: 495-502.
10. Gold N, Knight C, Mohan A, Munro M. Understanding service-oriented software. IEEE Software 2004; 21: 71-77.
11. Goseva-Popstojanova K, Singh AD, Mazimdar S, LiF. Empirical characterization of session-based workload and reliability for Web servers. Empirical Software Engineering; 11(1): 71-117.
12. Janevski N, Goseva-Popstojanova K. Session Reliability of Web Systems Under Heavy-Tailed Workloads: An Approach based on Design and Analysis of Experiments. IEEE Transactions on Software Engineering 2013; 99(PrePrints): 1.

13. Jia J, Wu S. A replacement policy for a repairable system with its repairman having multiple vacations. Computers & Industrial Engineering, 2009; 57(1): 156-160.
14. Jishen J,ShaominW. Optimizing replacement policy for a cold-standby system with waiting repair times. Applied Mathematics and Computation Journal 2009; 133–141.
15. Juan AA, Faulin J, Ramirez-Marquez J E, MartorellAlsina S S. Simulation Methods for Reliability and Availability of Complex Systems, Springer 2012.
16. LamY. A note on the optimal replacement problem, Adv. Appl. Prob. 1998; 20(2): 479-482.
17. Lavenberg S. A perspective on queueing models of computer performance. Performance Evaluation 1989; 10:53-76.
18. Lipinski Z. State Model of Service Reliability. International Conference on Dependability of Computer Systems. IEEE Computer Society 2006; 35-42.
19. Liu X, Jin H, Sha L, Zhu X.Adaptive control of multi-tiered Web application using queueing predictor. Proc. IEEE/IFIP Netw. Operations Management Symp. (NOMS2006) 2006; 106-114.
20. Liu J. Parallel Real-time Immersive Modelling Environment (PRIME), Scalable Simulation Framework (SSF), User's manual, Colorado School of Mines Department of Mathematical and Computer Sciences. [Available online: http://prime.mines.edu/], 2006.
21. Malinowski J. A simulation model for complex repairable systems with inter-component dependencies and three types of component failures. Advances in Safety, Reliability and Risk Management - Proceedings of the European Safety and Reliability Conference, ESREL 2011, 2012; 128.
22. Merzbacher M, Patterson D. Measuring end-user availability on the Web: practical experience. Proceedings of the International Conference on Dependable Systems and Networks, 2002; 473 – 477.
23. Michalska K, Walkowiak T. Simulation approach to performance analysis information systems with load balancer. Information systems architecture and technology: advances in Web-Age Information Systems,Wrocław: Oficyna Wydawnicza Politechniki Wrocławskiej 2009; 269-278.
24. Rahmawan H, Gondokaryono YS. The simulation of static load balancing algorithms. International Conference on Electrical Engineering and Informatics 2009; 640-645.
25. Urgaonkar B, Pacificiy G, Shenoy P, Spreitzery M, Tantawi A. An analytical model for multitier internet services and its applications. Proc. of the ACM SIGMETRICS'05 Conference on measurement and modeling of computer systems, 2005; 291-302.
26. Walkowiak T. Information systems performance analysis using task-level simulator.International Conference on Dependability of Computer Systems. IEEE Computer Society Press, 2009; 218-225.
27. Walkowiak T. Simulation approach to Web system dependability analysis. Summer Safety and Reliability Seminars, SSARS 2011; 1: 197-204.
28. Walkowiak T, Michalska K. Functional based reliability analysis of Web based information systems. Dependable computer systems / Wojciech Zamojski [et al.] (eds.). Berlin ; Heidelberg: Springer, 2011; 257-269.
29. Zamojski W. Redundancja sprzętowa współczesnych systemów technicznych. Nadmiarowość w inżynieri i niezawodności. XXXII Zimowa Szkoła Niezawodności, Szczyrk. Radom: Instytut Technologii Eksploatacji 2004; 398-411.
30. ZhangY L, Wu S. Reliability analysis for a k/n(F) system with repairable repair-equipment, Appl. Math. Model. 2009; 33(7): 3052-3067.

Tomasz WALKOWIAK
Institute of Computer Engineering, Control and Robotics
Wrocław University of Technology
Wybrzeże Wyspiańskiego 27, 50-370, Wrocław, Poland
E-mail: Tomasz.Walkowiak@pwr.wroc.pl