ECIS 2014 Proceedings

PREDICTING AVAILABILITY AND RESPONSE TIMES OF IT SERVICES

Sascha Bosse
Otto von Güricke University Magdeburg, Magdeburg, Saxony-Anhalt, Germany, sbosse@ovgu.de

Christian Schulz
Otto von Güricke University Magdeburg, Magdeburg, Saxony-Anhalt, Germany, christian.schulz@ovgu.de

Klaus Turowski
Otto-von-Güricke-University Magdeburg, Magdeburg, Germany, klaus.turowski@ovgu.de

Sascha Bosse, Christian Schulz, and Klaus Turowski, 2014, "PREDICTING AVAILABILITY AND RESPONSE TIMES OF IT SERVICES", Proceedings of the European Conference on Information Systems (ECIS) 2014, Tel Aviv, Israel, June 9-11, 2014, ISBN 978-0-9915567-0-0
http://aisel.aisnet.org/ecis2014/proceedings/track20/5

PREDICTING AVAILABILITY AND RESPONSE TIMES OF IT SERVICES

Complete Research

Bosse, Sascha, Otto von Guericke University Magdeburg, Germany, sascha.bosse@ovgu.de

Schulz, Christian, Otto von Guericke University Magdeburg, Germany, christian.schulz@ovgu.de

Turowski, Klaus, Otto von Guericke University Magdeburg, Germany, klaus.turowski@ovgu.de

Abstract

When IT service providers adapt their IT system landscapes because of new technologies or changing business requirements, the effects of changes to the quality of service must be considered to fulfill service level agreements. Analytical prediction models can support this process in the service design stages, but dependencies between quality aspects are not taken into account. In this paper, a novel approach for predicting availability and response time of an IT service is developed, which is simulation-based to support dynamic analysis of service quality. The correctness of the model as well as its applicability in a real case can be evaluated. Therefore, this work presents a step towards an analytical framework for predicting IT service quality aspects.

Keywords: Availability, Response Time, IT Service Management, Simulation.

1 Motivation

The service-orientation of IT gives organizations the opportunity to define their business process support as a set of IT services, which can be provided by a combination of internal or external IT service providers (Cannon, 2011). However, when outsourcing IT support to external enterprises, the service consumer needs guarantees about the service quality due to the lack of control over the external provider (Stahlknecht and Hasenkamp, 2002). Therefore, service level agreements (SLA), which are negotiated between IT provider and consumer, include guarantees for the quality of service.

On the one hand, the violation of these assurances may lead to penalty costs and a loss of reputation for the IT service provider (Emeakaroha et al., 2012) and should thus be avoided. On the other hand, the technical basis for hosting IT services – the IT system landscape – has to be adapted permanently in order to face new business requirements and innovative technologies (Eckert et al., 2007). This leads to the challenge of considering effects on the quality of service when developing the IT system landscape. Due to the high complexity and heterogeneity of these landscapes, this is not a trivial problem (Terlit and Krcmar, 2011).

In order to cope with this challenge, the IT Infrastructure Library (ITIL) recommends the usage of prediction models to support decision processes in the management of IT system landscapes (Hunnebeck, 2011). To this end, a range of different approaches dealing with the prediction of single quality attributes have been developed in recent years. However, methods that are considering several quality metrics and their inter-dependencies are still lacking (Milanovic and Milic, 2011).

Two crucial quality aspects to be considered in SLAs are availability and response time (Keller and Ludwig, 2003) due to the fact that only a service that is available and fast responding can support the service consumer's business processes effectively. Predicting these criteria separately ignores their relation: a service with very high response times will be recognized by the customer as being unavailable (Hunnebeck, 2011). Furthermore, the reliability of redundant resources, for instance, has an effect on the response times of the whole service if capacity is affected. In order to consider the relation between the quality criteria of availability and response times in the field of IT services, a simulation-based prediction approach is developed within this paper according to the design science research methodology, cf. e. g. (Hevner et al., 2004; Österle et al., 2010; Peffers et al., 2008), since simulation approaches combine scalability as well as flexibility and support dynamic analyses.

The remainder of this paper is organized as follows: in the state of the art section, different approaches for availability and response time prediction of IT services are presented and discussed. On the basis of selected approaches, the simulation model is introduced in the availability and response time model section. Its verification and application to a real case is described in the evaluation section. The conclusion section summarizes the paper as well as its results and provides an outlook to future research.

2 State of the Art

2.1 Availability Prediction for IT Services

"Availability is the ability of a service [...] to perform its agreed function when required" (Hunnebeck, 2011)

Although reliability analysis is a research field with history, the specific requirements of IT system landscapes led to new approaches in this area. Methods for predicting the availability of IT services can be distinguished into qualitative, quantitative and analytical approaches (Bosse, 2013). Qualitative methods provide informal predictions about future system behavior on the basis of expert analyses. However, the output of these methods strongly depends on the involved experts, which makes the results rather subjective (Milanovic and Milic, 2011).

Quantitative approaches use data from system monitoring to apply data analysis methods for predicting IT service availability. Since these methods do not consider the structure of the system, they can be seen as black box approaches. On the basis of a suitable system monitoring, future predictions are easy to apply and very fast. Nevertheless, the effects of changes on the system landscapes cannot be estimated since new monitoring data is gathered, which reduces the applicability of these approaches in high-dynamic IT system landscapes (Immonen and Niemelä, 2008).

This disadvantage can be avoided if the system is not considered to be a black box, but a combination of its components, which is done in analytical approaches. These methods can be further classified as combinatorial, state-space-based and hybrid models (K. Trivedi et al., 2008). In combinatorial approaches, components can be related in a serial or parallel way. This enables the very fast computation of IT service availability using simple probability calculations. However, these approaches are limited to independent components and thus, cannot map the complex dependencies in IT system landscapes (Callou et al., 2012). State-space-based approaches are able to inherit these dependencies, since each possible state in the system and the transition probabilities between them are modeled. Nonetheless, in very complex systems such as IT system landscapes, these approaches are affected by the problem of state-space explosion, induced by the exponential growth of the number of states (Sachdeva et al., 2008). The alternative is to encode the state-space, for example, in a Petri net and to evaluate it by using Monte Carlo simulation (Zille et al., 2010). This approach has the advantage that not only the mean values, but also the variance of a service's availability can be computed. As this is a measure for the probability of SLA violations, this information can support SLA management more effectively than steady-state approaches can (Franke, 2012). In order to reduce model complexity without losing modeling power, hybrid methods combining the advantages of combinatorial and state-space-based approaches have been developed. In these methods, component-level availability is modeled with state-space approaches while system-level availability is evaluated in a combinatorial way (Bosse, 2013; Callou et al., 2012).

Depending on means and goals for availability prediction, qualitative, quantitative or analytical methods can be applied successfully. Nevertheless, analytical and especially state-space-based or hybrid methods are most suitable to predict availability in a high-dynamic and complex environment such as IT system landscapes. At the same time, simulation methods provide a better scalability than Markov models and are able to model arbitrary random distributions which are more realistic than exponential distributions, especially when modeling recovery times (Chellappan and Vijayalakshmi, 2009).

2.2 Response Time Prediction for IT Services

The response time of an IT service is a measure for the performance of this service (Balsamo et al., 2004), hence for the "speed with which a system accomplishes the tasks it was designed to" (Jewell, 2008). Other related metrics are utilization and throughput (H. H. Liu and Crain, 2004). The response time of a service can be divided into network delay, queuing and processing time (Rud, 2009). The most popular methods to predict the response times of an IT service quantitatively are prototyping or benchmarking, but the application of these methods can be difficult due to system complexity (Y. Liu et al., 2005) and the lack of measurement data (Balsamo et al., 2004). Therefore, prediction models should be applied in early design stages in order to support the process of performance management before wrong decisions are made (Terlit and Krcmar, 2011).

In the literature review from (Balsamo et al., 2004), the authors identify five classes of methods for performance prediction in early design phases: process algebra-, Petri net-, simulation-, stochastic process- and queuing network-based approaches. The latter approaches can be further distinguished into architectural-pattern-, trace-analysis- and UML[1] for performance-based methodologies. They conclude that techniques based on UML models are very popular in all classes, since UML is the standard framework in software development. Queuing networks are the preferred class of performance models (Balsamo et al., 2004). However, simulation-based methods are recommended for dynamic performance analysis, since other methods do only provide a steady-state-result (Jewell, 2008).

Since most approaches for quality prediction in the design stages target monolithic systems, the Palladio Component Model (PCM) – a meta-model for quality analysis of component-based software systems – was developed to address heterogeneous business information systems (Reussner et al., 2011) which can be found in IT system landscapes. The PCM consists of four sub-models (Rathfelder et al., 2011): the component model describes the system's elements, their behavior and resource demands. The dependencies between single components are specified in the composition model. How physical resources are allocated to components is defined by the deployment model, while the usage model describes the workload that is induced on the system. The PCM thereby includes analytical models such as Petri nets or queuing networks as well as simulation methods (Rathfelder et al., 2011). However, disadvantages such as the lack of availability models for physical resources, hierarchical

[1] Unified Modeling Language - http://www.omg.org/spec/UML/

structures and internal component states do exist in the meta-model and need to be eradicated (Reussner et al., 2011). Nonetheless, the PCM provides a good basis for developing a new prediction approach for the availability and response time of IT services since it considers the architectural patterns that can be found in modern IT system landscapes.

3 Availability and Response Time Prediction Model

Since none of the considered prediction methods is able to predict the availability and response time of IT services considering the whole IT system landscape (including physical, hardware, software and human resources), a novel approach is developed in this section. As simulation models are more suitable for SLA analysis in the field of availability prediction as well as in the field of response time prediction, a simulation-based approach is chosen. It is based on the availability model in (Bosse, 2013) which considers inter-component dependencies as well as operator interaction and is extended with a trace-based performance model. In the following sub-sections, first, the data model for response time and availability is presented before the behavior of the simulation model is described.

3.1 Data Model

All entities, attributes and functions that have to be defined to instantiate the prediction model are described in this section and are illustrated in the figures 1 to 3. Availability and response time can be computed for an atomic service, characterized by a random distribution for the time to request and a value for the maximum processing time before a request is rejected. A so-called operation graph is assigned to each atomic service which includes the trace of operations for a service. In this graph, consecutive connections as well as exclusive and parallel paths of operations can be modeled, so e.g. UML sequence diagrams can be used as an input. For exclusive paths, the execution probability of each path has to be defined. An operation represents a function that is performed during the service request process that relies on certain resources provided by technical components.

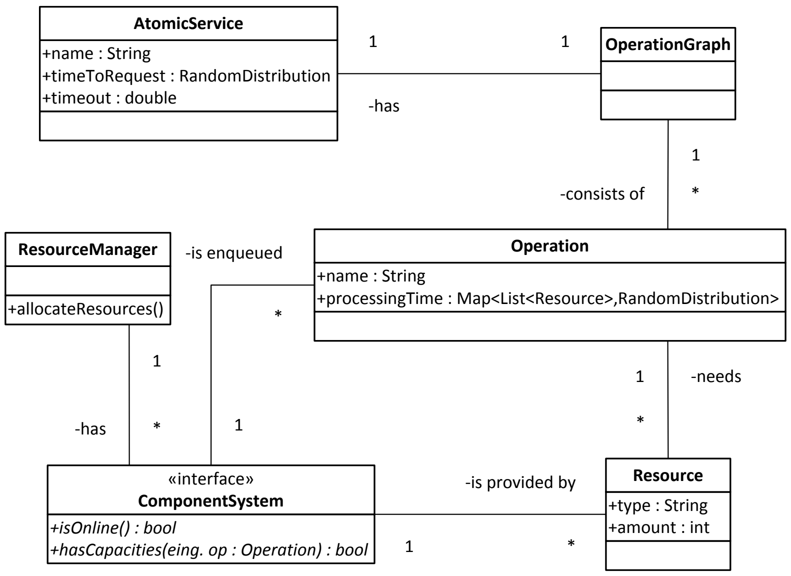

Figure 1. Entities, attributes and functions that have to be defined for the usage, component and deployment model.

For an operation, its processing time is defined as a function that maps a list of resources to a random distribution of the execution time. This models different execution times depending on the amount of resources that are allocated for the operation on the component. Resources are provided by so-called component systems which can be single components or more complex arrangements such as serial (each sub-system is needed for an operation) or parallel systems (redundant sub-systems providing the same resources). A resource manager, which is responsible for the allocation of resources, is assigned to each component system. The standard resource manager allocates the minimum resources required by an operation using a first fit algorithm, more complex managing strategies need to be implemented by the model creator. This is supported by the realization of the software engineering concepts of modularization and encapsulation in the data model.

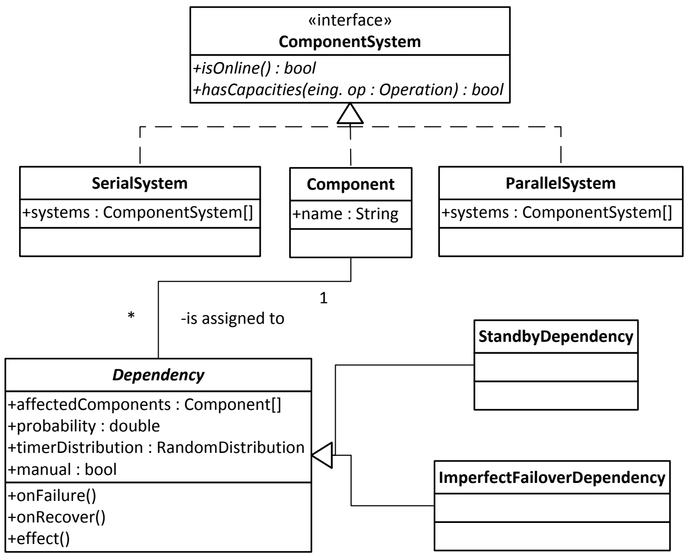

Figure 2. Entities, attributes and functions that have to be defined for the composition model.

Single components can be defective, so different failure types can be defined for them. Each failure is specified by random distributions for the time to failure and the time to recover. In addition to that, it is defined whether the failure needs manual recovery by an operator or not. Failure and recovery of components can have effects on other components. This can be modeled by using dependencies which are characterized by a causing component, a list of affected components, a probability of occurrence, a Boolean value for manual interaction, a random distribution for the effect to happen and the effect itself as a function. For example, a standby dependency can be defined for a scenario where a redundant system will only be started after a primary system fails. In case of cold-standby (manual activation of the redundant system), an operator is needed before the redundant component can take over. Another example is the imperfect failover dependency where a failure of one component affects all components of the same type.

Operators are limited resources collected in an operator pool. If failures or dependencies need operator interaction, a task is created which is stored in a priority queue sorted by task priority and timestamp. If an operator is available, it is assigned to the affected component/dependency and is no longer available until the component is recovered or the dependency is finished. After that, the operator may repeat the task because of human error, which is determined by an error probability value.

In difference to the PCM, this model includes architectural patterns, but as passive model elements in the context of resources since components do only provide resources. These resources are demanded by the operations in this model, which corresponds to a trace analysis-based technique. However, the four sub-models of the PCM can also be found in this approach: the entity *AtomicService* with its attribute *timeToRequest* defines the usage model. The component model is built by the entities *Operation* and *OperationGraph* which define the system behavior and the resource demands on physical components while the entity *ResourceManager* is corresponding to the deployment model (figure 1). The abstract entities *ComponentSystem* and *Dependency* describe the composition model (figure 2). Since the PCM does not provide human interaction, the operator model (figure 3) has no counterpart in the PCM. The different sub models are connected by the entities *ComponentSystem*, *Component* and *Dependency*.

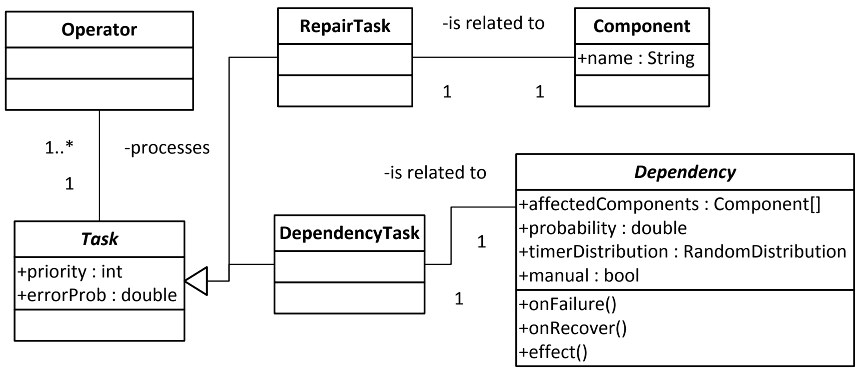

Figure 3. Entities, attributes and functions that have to be defined for the operator model.

3.2 Simulation Model

Once the data model for a specific scenario has been instantiated, a generalized stochastic Petri net-based simulation model (cf. Ciardo et al., 1989) is constructed automatically, combining Petri net models of the operation graph and of each component. For the operation graph, first of all a transition is created that generates tokens in a succeeding place according to the defined time to request distribution. For each defined operation a transition and a succeeding place are constructed. These transition/place constructs are connected corresponding to the defined operation graph, using single edges for consecutive operations as well as fork/join and parallelization/synchronization templates for exclusive and parallel operations (cf. figure 7 presenting a Petri net operation graph).

When a request arrives in the system, a token is created at the beginning of the Petri net operation graph, the request's entry time is stored and the processing of the request starts by calling the first operation. When an operation has been finished, the token in front of the transition is destroyed and succeeding places are filled with tokens. Once a token is created in places without succeeding transitions, the request processing is finished and its response time can be computed. If the request's timeout is reached before it is completely processed, the request token is deleted from the model and the corresponding request is rejected.

In order to finish a called operation, a certain amount of resources has to be available during the processing time. Therefore, the operation is queued in the corresponding component system. If enough resources are available on the component system, the defined resource manager will be called in order to allocate an amount of resources for the operation. Depending on the allocated amount, a processing time can be computed according to the defined random distribution. After this time has elapsed, the completion of the operation is reported and the allocated resources are released again. However, component failures can affect the available resources and may lead to the abortion of a called operation. In this case, the service request is rejected.

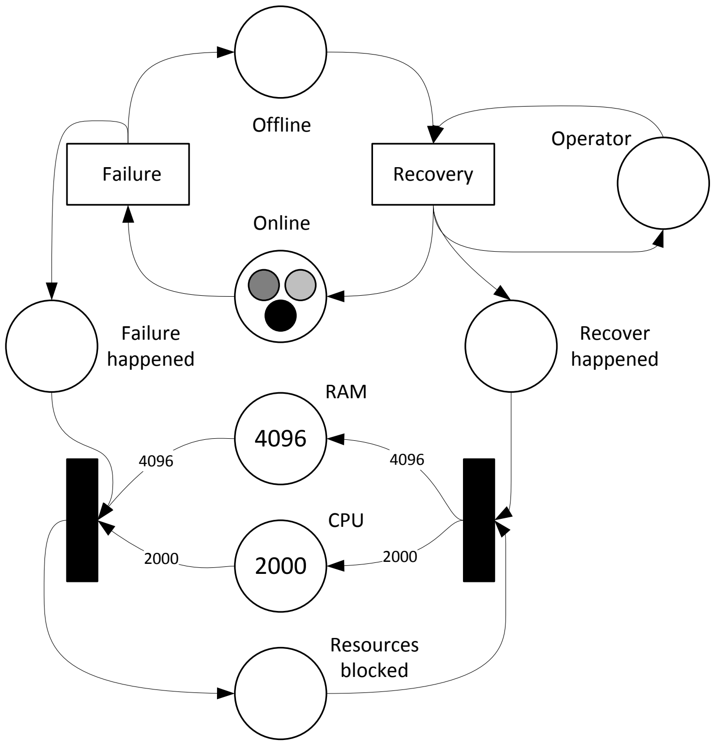

Figure 4. Exemplary component Petri net model.

Component failures are determined by the component model. In figure 4, an exemplary component model is illustrated. For each resource type, a place is created holding an amount of tokens equal to the amount of resources provided, in the example 4096 units of memory and 2000 units of processing power. The two places “offline” and “online” represent the current state of the component. Different failure types are represented by differently colored tokens. Once one failure token is fired from the “online” to the “offline” place, all resources of the affected component are blocked, so that they are not available for operations anymore. In a single component or a serial system, this will lead to operation abortion. In case of a parallel connected system, the resource manager tries to allocate resources on other redundant systems which increases the likelihood of timeouts. Firing a token from the “offline” to the “online” place may need an operator token from the operator pool. If the failure token is in the “online” place again, resources are unblocked immediately. By the time of the failure/recovery, possibly defined dependencies are activated.

When the simulation starts, all components are online. Request arrivals and component failures are determined randomly in each single simulation run. As soon as the simulation is finished, the service requests can be analyzed: the ratio of answered requests to total requests gives an estimate of the probability that the requested service performed its function and hence for the service’s availability (Bosse, 2013). For the response time of the service, the mean value can be computed by analyzing response times of all answered requests. Analyzing these values over a sufficient number of simulation runs results in confidence intervals for the service’s availability and response time.

4 Evaluation

In this section, the evaluation of the created artefact is described. For that purpose, first the verification of the simulation’s correctness is explained before a validation is presented that shows the applicability of the approach in a case-study.

4.1 Verification Experiments

In order to verify the correctness of the conceptual model, the implemented simulation model is tested by using extreme condition tests to build special scenarios, in which the behavior of the simulation should be comparable to other approaches. If not, sensitivity tests are applied, cf. (Sargent, 2010).

In the first scenarios, a service is modeled consisting of three operations, combined serially, parallel and exclusive. These operations have a zero resource demand, neither timeouts nor failures are defined. Therefore, the response time results should fit the ideal expectations. In table 1, expected values for the response time from probability theory and confidence intervals (α = 0.99) from the simulation with 100 replications are presented. The execution times of the operations are uniformly distributed and path probability is given in the case of exclusive paths. The results indicate that the computation of response times is correct.

Scenario	Execution Time	Expected Response Time	Response time 0.99-confidence interval
3 serial operations	~U(13,17)	45.0	[44.939;45.080]
3 parallel operations	~U(13,17)	16.0	[15.999;16.003]
3 exclusive operations	~U(13,17) [0.5], ~U(14,18) [0.3], ~U(15,19) [0.2]	15.7	[15.684;15.707]

Table 1. Comparison of expected and computed values for response time in the first three verification scenarios.

Following the tests for response times without resource demands and timeouts, the next scenario introduces resource demands and thus, limiting capacities. A service is modeled with a single operation (execution time ~ U(13,17)) that demands one resource unit. Since ten resource units are available in this scenario, the system has a maximum throughput of 10/15 = 2/3 (Buzen, 1976). This scenario was simulated once without timeouts and once with a timeout of 50.

Figure 5 presents the results of a sensitivity analysis for different values of the request arrival rate. The grey curve shows the mean values for the service's response time without timeout and the black curve the mean values with the timeout of 50. For arrival rates lower than 0.5, both curves are identical at a mean response time of 15. For values higher than 0.6, the system is more and more unstable and response times increase. Without a defined timeout, all requests are further processed, even if the response time will be unacceptably high for a customer. For values above the maximum throughput of 2/3, response times grow exponentially. If a timeout is defined, the response times are stable again if the arrival rate is higher than the maximum throughput. On the other hand, more and more requests are rejected because timeout is exceeded. While the percentage of successfully answered requests in the scenario without timeout is always 100 %, the availability in the scenario with timeout decreases in the same extend as both curves diverge in the diagram, from mean availability of 99.4 % at an arrival rate of 0.6 to 1.5 % at a rate of 0.9. Since the behavior of the model and its result is plausible, this test verifies the correct implementation of resource demands and timeouts.

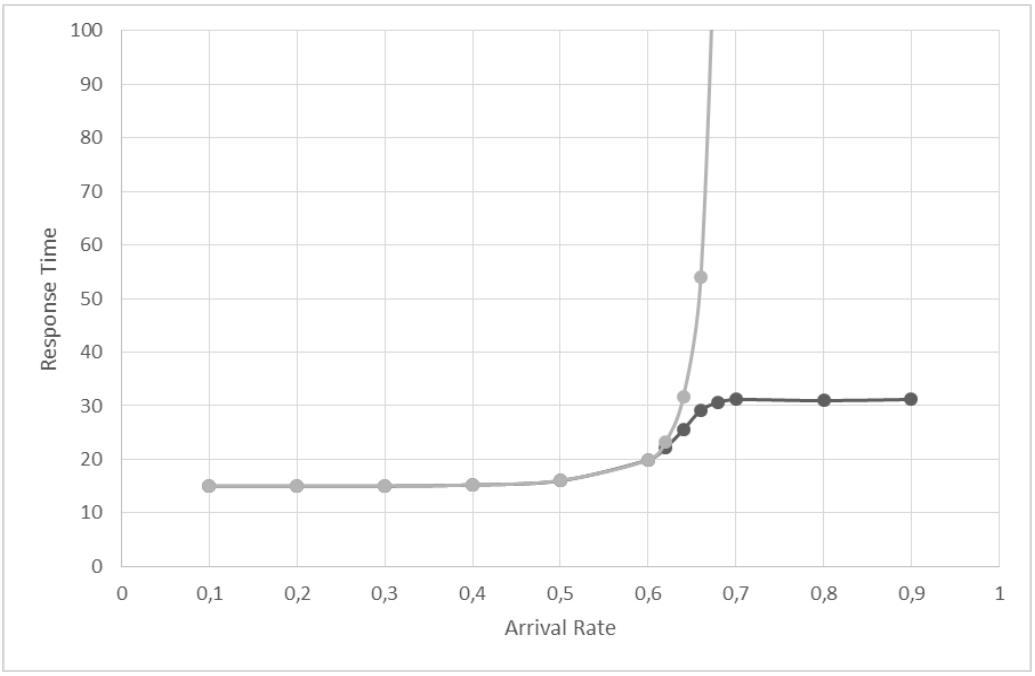

Figure 5. Sensitivity analysis of response time values depending on arrival rate.

In the next verification scenarios, the correctness of component failures is tested. For that purpose, the availability of a single component as well as of a serial and a parallel connection of three components are computed. The expected values are derived by combining analytical formula for availability in time (Shooman, 2002) and the average number of requests currently processed due to the fact that these requests are automatically rejected in case of system failure. Table 2 shows the expected and the simulated values for the availability. Since expected values are included in the confidence intervals, the simulation of the service availability is verified.

Scenario	**MTTF/MTTR**	**Expected Availability**	**Availability 0.99-confidence interval**
1 component	8760/10	0.99715	[0.9970;0.9975]
3 serial components	8760/10	0.99145	[0.9913;0.9917]
3 parallel components	8760/1000	0.99875	[0.9986;0.9992]

Table 2. Comparison of expected and computed values for availability in the last three verification scenarios.

Besides the fact that the major functions of the simulation model are successfully verified, also the relation between availability and response time can be investigated in the verification experiments. The sensitivity analysis with timeout illustrates the dependency of availability on the response time: When response times increase, more and more requests cannot be answered and hence the availability of the service to the customer is affected. Another relation is revealed by the last verification scenario, where three parallel connected components provide three redundant resource pools. The 0.99-confidence interval for the response time in this scenario is [15.011;15.012], so the value is differing significantly from theoretical value of 15. This is the case, since failures of redundant components may not lead to outages, but affect capacity and throughput of the service, so response times increase. These facts demonstrate that both quality aspects are correlated and a prediction model should consider their relation.

4.2 Case-Study: Website Delivery Service

In order to demonstrate the ability of the developed approach to support availability and response time analysis, it is applied in a case-study of a real-world service, which delivers a HTML webpage to a client. For that purpose, a *TYPO3 4.7.6* content management system (CMS) running on an *Apache 2.2.14* web server, a *memcached 1.4.2* memory caching system and a *MySQL 5.1.72* database are operated on an *Ubuntu 10.04 LTS* server. These systems are hosted on a computer with an *Intel(R) Xeon(R) X5650* @ 2.67 GHz and 4096 MiB of RAM. The operation graph of this service is presented as an UML sequence diagram in figure 6. In order to execute these operations, 256 MiB of RAM have to be allocated. The more processing power is assigned to the single operations, the faster they are executed. A resource manager is defined that allocates the same amount of processing power for each current operation, so that the CPU is fully utilized.

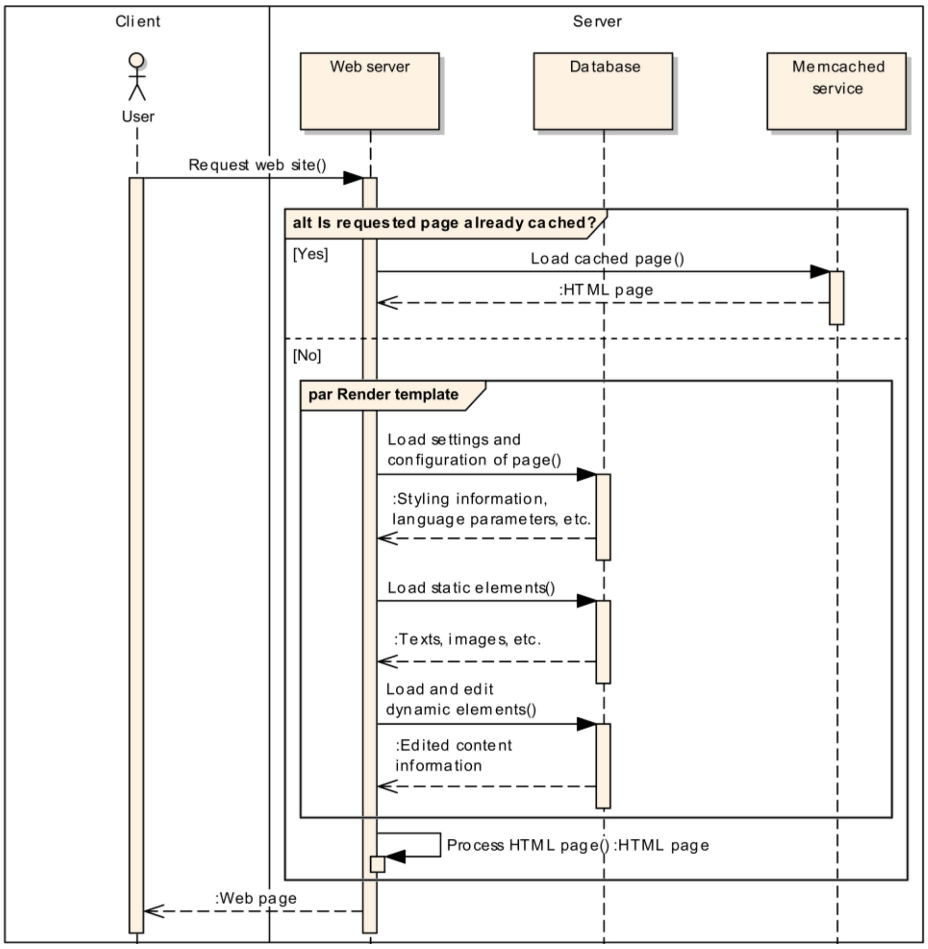
Client
Server
User
Web server
Database
Memcached service
Request web site()
alt Is requested page already cached?
[Yes]
Load cached page()
:HTML page
[No]
par Render template
Load settings and configuration of page()
:Styling information, language parameters, etc.
Load static elements()
:Texts, images, etc.
Load and edit dynamic elements()
:Edited content information
Process HTML page() :HTML page
:Web page

Figure 6. Operation graph of the website delivery service.

When a client requests a HTML page, it is first tested if the *memcached* service has already cached the requested page. In this case, the cached page can be sent as the response to the requesting client. Otherwise, the content of the requested page has to be rendered from pre-defined templates. These templates include the CMS settings and configuration from the *Apache* and *MySQL* server, the static content elements such as texts and images as well as dynamic content elements from the database, which are all processed in a parallel way.

As soon as all information has been loaded, the HTML page can be created and sent to the requesting client. After that, a copy of the requested page is put into the cache. The execution time distributions –

triangular distributions created from real data – are presented in table 3, failure parameters for modeled components can be found in table 4. All failures need manual interaction for recovery and one operator is assigned to these components. Timeout is set to ten seconds, since longer response times lead to waiting users (Meyer et al., 2005).

Operation	Execution time triangular distribution in ms		
	Minimum	Mode	Maximum
Load cached page	855	957	1,249
Load settings and configuration	16	31	118
Load static content	56	210	1,117
Load dynamic content	1,010	1,719	2,255
Process HTML page	69	74	87

Table 3. Execution time distributions for each operation of the website delivery service.

Component	MTTF in h	MTTR in h
Web server/caching system (Milanovic and Milic, 2011)	8,760	1.44
Database server (Milanovic and Milic, 2011)	8,760	1.73
RAM (Schroeder et al., 2011)	38,672	1.45
Physical server (Anon, 2012)	5,463	1.97

Table 4. Mean time to failure (MTTF) and mean time to recover (MTTR) for the components in the website delivery service.

Based on the defined operation graph, the Petri net operation graph illustrated in figure 7 could be created. The simulation model was parameterized with a request arrival rate of 0.123 requests per minute and a probability of 90 % that the requested page was cached before, which was computed from historical data. In this scenario, the simulated results can be compared to the mean value of measured response times of the whole service from reality which is 0.016649 minutes or 0.99894 seconds. After 200 replications, the 0.99-confidence interval for the response time is (in minutes) [0.019041; 0.019056]. The mean value of this interval has a relative difference to the real value of 14.417 %. Since a difference of 30 % is tolerable in response time prediction (H. H. Liu and Crain, 2004), the validity of the simulation is improvable, but cannot be denied.

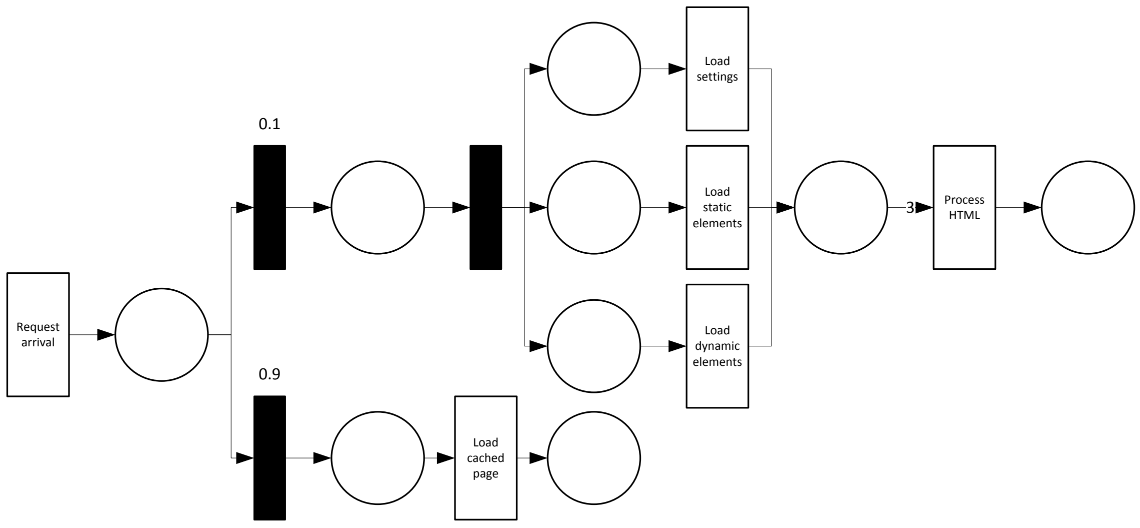

Figure 7. Petri net operation graph of the website delivery service.

After the simulation is tested with the arrival rate from reality, this rate is increased to predict the service's availability and response time under high load. The mean values of availability and response time depending on the arrival rate are displayed in figure 8. With respect to the desired service-level, the maximum arrival rate can be determined before the available resources have to be extended. For example, if an availability of 98 % and a response time of two seconds should be guaranteed, the arrival rate of requests should not exceed 30 requests per minute. If the arrival rate is above 50 requests per minute, the system is unstable and availability decreases dramatically. So the prediction model is able to support decision making in this scenario.

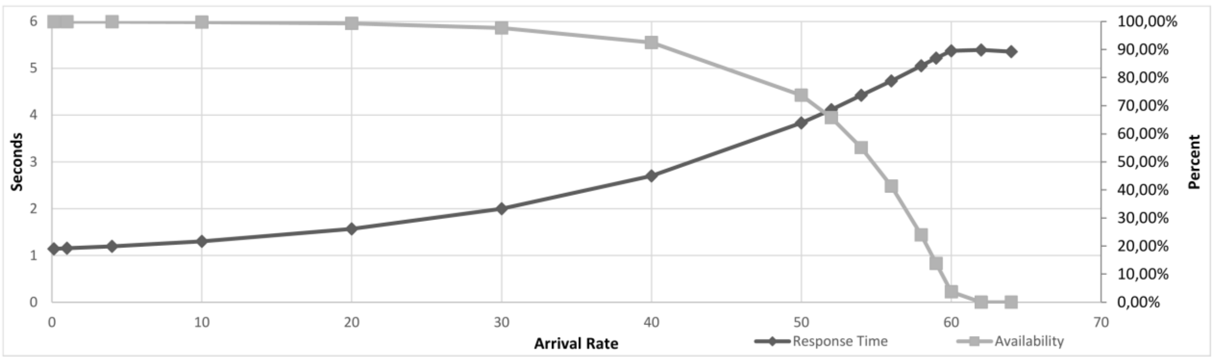

Figure 8. *Availability (primary vertical axis) and response time (secondary vertical axis) of the website delivery service depending on the request arrival rate.*

5 Conclusion

In the context of service-orientation, the prediction of service quality attributes can support decision making in IT service management. However, analytical prediction methods do not consider the inter-dependencies between single attributes. Therefore, a novel approach for predicting availability and response time of IT services was developed in this paper that is taking their dependency into account. For this purpose, suitable availability and response time prediction models from the literature were adapted and combined. The synthetized approach was implemented and verified, before it was applied in a real-world case study of a website delivery service to show its applicability.

Although the developed approach is simple and has to be improved in future research, the dependency between availability and response time could be integrated into the model and evaluated. High response times lead to non-availability from a customer's perspective. In addition to that, unavailability of technical systems can affect the capacity of the service and, thus, the service's response time. The approach developed in this article is the only one found in the literature that considers a response time model based on operation traces as well as physical component availability, inter-component dependencies and operator interaction. Thus, this concept presents a step towards a holistic framework for model-driven prediction of quality attributes of IT services.

In addition to the extensions of the availability and response time prediction model in order to strengthen the accuracy of the results, two major problems of analytical prediction approaches have to be addressed in future research: the model creation process and the model parameterization. Without automatic support, models have to be created and kept up to date by experts, which is a time-consuming and error-prone process. Using information from existing landscape models such as configuration management databases can support the automation of the model creation process and, therefore, the applicability of these approaches in IT service management. For availability models, this problem was addressed in (Milanovic and Milic, 2011) and their approach could be transferred to other prediction models. The problem of model parameterization is crucial to the result's accuracy, but difficult to solve, since manufacturers' data is often too optimistic (Pinheiro et al., 2007) and benchmarks are not available in early design stages (Balsamo et al., 2004). In this respect, an integrated approach combining both real measured data and analytical prediction models seems promising, but needs to be conceptualized in future research. If these problems are solved effectively, the proposed model can be developed further to inherit more service quality aspects and their relationships in order to provide a holistic and consistent prediction model for service quality levels in the design stage.

References

Anon (2012). Records of cluster node outages, workload logs and error logs. Available at: http://cfdr.usenix.org/data.html.

Balsamo, S., Di Marco, A., Inverardi, P. and Simeoni, M. (2004). Model-Based Performance Prediction in Software Development: A Survey. IEEE Transactions on Software Engineering, 30, 295–310.

Bosse, S. (2013). Predicting an IT Service’s Availability with Respect to Operator Errors. In Proceedings of the 19th Americas Conference on Information Systems (AMCIS) Chicago, IL, USA.

Buzen, J.P. (1976). Fundamental Operational Laws of Computer System Performance. Acta Informatica, 7, 167–182.

Callou, G., Maciel, P., Tutsch, D., Araújo, J., Ferreira, J. and Souza, R. (2012). A Petri Net-Based Approach to the Quantification of Data Center Dependability. In Petri Nets - Manufacturing and Computer Science (Pawlewski, P. , Ed.), pp. 313–336, InTech.

Cannon, D. (2011). ITIL Service Strategy 2011 Edition, The Stationery Office.

Chellappan, C. and Vijayalakshmi, G. (2009). Dependability modeling and analysis of hybrid redundancy systems. International Journal of Quality & Reliability Management, 26, 76–96.

Ciardo, G., Muppala, J.K. and Trivedi, K.S. (1989). SPNP: Stochastic Petri Net Package. In Proceedings of the 3rd International Workshop PNPM pp. 142–151, IEEE Computer Society.

Eckert, J., Repp, N., Schulte, S., Berbner, R. and Steinmetz, R. (2007). An Approach for Capacity Planning of Web Service Workflows. In Proceedings of the 13th Americas Conference on Information Systems (AMCIS)

Emeakaroha, V.C., Netto, M.A.S., Calheiros, R.N., Brandic, I., Buyya, R. and Rose, C.A.F.D. (2012). Towards autonomic detection of SLA violations in Cloud infrastructures. Future Generation Computer Systems, 28, 1017–1029.

Franke, U. (2012). Optimal IT Service Availability: Shorter Outages, or Fewer? IEEE Transactions on Network and Service Management, 9, 22–33.

Hevner, A.R., March, S.T., Park, J. and Ram, S. (2004). Design science in information systems research. MIS Quarterly, 28, 75–105.

Hunnebeck, L. (2011). ITIL Service Design 2011 Edition, The Stationery Office, Norwich, UK.

Immonen, A. and Niemelä, E. (2008). Survey of reliability and availability prediction methods from the viewpoint of software architecture. Software and Systems Modeling, 7, 49–65.

Jewell, D. (2008). Performance Modeling and Engineering. In (Liu, Z. and Xia, C. H. , Eds.), pp. 29–55, Springer US.

Keller, A. and Ludwig, H. (2003). The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services. Journal of Network and Systems Management, 11, 57–81.

Liu, H.H. and Crain, P.V. (2004). An Analytic Model for Predicting the Performance of SOA-based Enterprise Software Applications. In Proceedings of the 30th International Computer Measurement Group Conference (CMG)

Liu, Y., Fekete, A. and Gorton, I. (2005). Design-Level Performance Prediction of Component-Based Applications. IEEE Transactions on Software Engineering, 31, 928–941.

Meyer, H.A., Vogt, P. and Glier, M. (2005). Performance – (k)ein Thema für Usability Professionals?, Usability Professionals.

Milanovic, N. and Milic, B. (2011). Automatic Generation of Service Availability Models. IEEE Transactions on Service Computing, 4, 56–69.

Österle, H., Becker, J., Frank, U., Hess, T., Karagiannis, D., Krcmar, H., Loos, P., Mertens, P., Oberweis, A. and Sinz, E.J. (2010). Memorandum on design-oriented information systems research. European Journal of Information Systems, 20, 7–10.

Peffers, K., Tuunanen, T., Rothenberger, M.A. and Chatterjee, S. (2008). A Design Science Research Methodology for Information Systems Research. Journal of Management Information Systems, 24, 45–78.

Pinheiro, E., Weber, W.-D. and Barroso, L.A. (2007). Failure Trends in a Large Disk Drive Population. In Proceedings of the 5th USENIX Conference on File and Storage Technologies (FAST)

Rathfelder, C., Kounev, S. and Evans, D. (2011). Capacity Planning for Event-based Systems using Automated Performance Predictions. In Proceedings of the 26th IEEE/ACM International Conference on Automated Software Engineering (ASE) (Alexander, P. , Pasareanu, C. S. and Hosking, J. G. , Eds.), pp. 352–361, IEEE, Lawrence, KS, USA.

Reussner, R., Becker, S., Burger, E., Happe, J., Hauck, M., Koziolek, A., Koziolek, H., Krogmann, K. and Kuperberg, M. (2011). The Palladio Component Model, Karlsruhe Institute of Technology, Faculty of Informatics.

Rud, D. (2009). Performancebewertung und -sicherung von orchestrierten Serviceangeboten. Otto von Guericke University Magdeburg.

Sachdeva, A., Kumar, D. and Kumar, P. (2008). Reliability analysis of pulping system using Petri nets. International Journal of Quality & Reliability Management, 25, 860–877.

Sargent, R.G. (2010). Verification and validation of simulation models. In Proceedings of the 2010 Winter Simulation Conference (Johansson, B. et al., Eds.), pp. 130–143.

Schroeder, B., Pinheiro, E. and Weber, W.-D. (2011). DRAM Errors in the Wild: A Large-Scale Field Study. Communications of the ACM, 54, 100–107.

Shooman, M.L. (2002). Reliability of Computer Systems and Networks – Fault Tolerance, Analysis, and Design, John Wiley & Sons New York, New York, NY, USA.

Stahlknecht, P. and Hasenkamp, U. (2002). Einführung in die Wirtschaftsinformatik, Springer Berlin Heidelberg.

Terlit, D. and Krcmar, H. (2011). Generic Performance Prediction for ERP and SOA Applications. In Proceedings of the 18th European Conference on Information Systems (ECIS)

Trivedi, K., Ciardo, G., Dasarathy, B., Grottke, M., Matias, R., Rindos, A. and Vashaw, B. (2008). Achieving and Assuring High Availability. In 5th International Service Availability Symposium (ISAS) (Nanya, T. et al., Eds.), pp. 20–25, Lecture Notes in Computer Science, Springer Verlag Berlin Heidelberg, Tokyo, Japan.

Zille, V., Bérenguer, C., Grall, A. and Despujols, A. (2010). Simulation of Maintained Multicomponent Systems for Dependability Assessment. In Simulation Methods for Reliability and Availability of Complex Systems (Faulin, P. et al., Eds.), pp. 253–272, Springer, Berlin, Heidelberg.