Dependability of Service-Oriented Computing: Time-Probabilistic Failure Modelling

Anatoliy Gorbenko[1], Alexander Romanovsky[2]
Vyacheslav Kharchenko[1], and Olga Tarasyuk[1]

[1] Department of Computer Systems and Networks (503),
National Aerospace University, Kharkiv, Ukraine
{A.Gorbenko,O.Tarasyuk}@csac.khai.edu,
V.Kharchenko@khai.edu,
[2] School of Computing Science, Newcastle University, Newcastle upon Tyne, UK
Alexander.Romanovsky@ncl.ac.uk

Abstract. In the paper we discuss a failure and servicing model of software applications that employ the service-oriented paradigm for defining cooperation with clients. The model takes into account a time-probabilistic relationship between different servicing outcomes and failures modes. We put forward a set of measures for estimating dependability of service provisioning from the client's viewpoint and present analytical models to be used for the assessment of the mean servicing and waiting times depending on client's timeout settings.

Keywords: Service oriented computing, dependability, failure modelling, servicing time, optimal timeout settings.

1 Introduction

The Service-Oriented Architecture (SOA) supports rapid, low-cost and seamless composition of globally distributed applications, and enables effective interoperability in a loosely-coupled heterogeneous environment. Services are autonomous, platform-independent computational entities that can be dynamically discovered and integrated into a single service to be offered to the users or, in turn, used as a building block in further composition. The essential principles of SOA and services provisioning form the foundation for various modern and emerging IT technologies, such as service-oriented and cloud computing, SaaS (software as a service), etc.

The service-oriented paradigm of cooperation between clients and providers is now widely used in e-science, critical infrastructures and business-critical systems. Failures of such applications can affect people's lives and businesses (see, for example, the well-known incident at the London Stock Exchange on 8 Sept. 2008 or a spate of recent service outages on the Amazon S3 and Google cloud platforms). Thus, ensuring dependability of SOA-based systems is a must, as well as a challenge.

P. Avgeriou (Ed.): SERENE 2012, LNCS 7527, pp. 121–133, 2012.

Although the SOA and web services technologies have seen significantly improved in recent times, we believe that they have not yet revealed their full potential. In particular, SOA is still in its infancy when it comes to ensuring dependability of large-scale dynamically composed service-oriented systems involving multiple independent web services. Dependability enhancing technologies will thus be essential in supporting mission and business critical application, for personal use or in enterprise, government or military.

There is significant on-going research devoted to dependability and performance in service-oriented computing [1]. Recent related work (e.g. [2, 3, 4]) introduced several approaches to incorporating fault tolerance techniques (including voting, backward and forward error recovery mechanisms and replication techniques) into the web service architectures. There has been work on fault analysis, dependability and performance evaluation and experimental measurements, e.g. [5, 6, 7]. However, coming from dispersed areas of research, the work addresses individual issues but do not yet advance them in unison or offer general solutions. Very often the researchers use simple and, hence, not realistic failure models or do not take into account the interdependency between dependability and performance of service-oriented solutions that is in the very nature of such distributed interacting systems.

To be more effective fault-tolerant techniques incorporated into the service-oriented architecture should distinguish between evident failures of different types, like application exceptions, communication errors or timeouts and should be capable to minimize the probability of non-evident application errors.

Experimental studies [10, 12] show that response time of web services very often can exceed ones minimal value in more than 10 or even 20 times. Moreover, sometimes clients await for the response from a web service for more than two hours instead of reporting an exception or resending a request. Therefore, right timeout setting is a key means improving performance of many distributed systems including web services.

Besides most of the fault tolerance and error recovery mechanisms at the application level also depend on timeout settings [1-4]. However, the existing work mainly focuses on optimizing timeouts used by communication protocols like TCP and HTTP without examining how timeout settings at the application level affect both performance and dependability of web services.

This is why the purpose of the paper is (i) to develop an advanced failure model for the service-oriented architecture taking into account a time-probabilistic interconnection between different servicing outcomes and (ii) to investigate analytical models assessing the average servicing and waiting times under certain timeout in case of probabilistic uncertainty of web services performance characteristics.

The rest of the paper is organised as follows. In Section 2 we describe the proposed failure and servicing model capturing the fundamental principles of the service-oriented architecture from the client's point of view. Section 3 proposes analytical models aimed at estimation of average servicing and waiting time depending on timeout settings in case of known response time probability density function. In Section 4 we present a numeric example of using the proposed analytical solution.

2 SOA Failure and Servicing Model

2.1 Servicing Outcomes and Web Services Failures

Web services as any other complex software may contain faults which may manifest themselves in operation. On every request the web service may succeed, i.e. return a correct response, or fail, i.e. return an incorrect response or not return any response at all within waiting time. Such failure behaviour of the web services is characterised by the probability of failure on demand (*pfd*). This probability can be statistically measured as a ratio between *r* failures observed in *n* demands [8]. It can vary between the environments and the contexts (operational profiles) in which a web service is used.

The various factors, which affect the *pfd* may be unknown with certainty, thus the value of *pfd* may be uncertain as well. This uncertainty can be captured by a probability density series or probability distribution, built by aggregating usage experience of different clients. The response returned to the client by a web service may be of several types:

1. *correct result*;
2. *evident error* – an error that needs no special means to be detected. It concerns exception messages of different types reported to the client and notifying him about denial of the requested service for some reason;
3. *non-evident (hidden) error* – an error that can be detected only by using a multiversioning at the application level (e.g. diversity of web services used).

However, the distributed nature of the service-oriented architectural model does not guarantee that the client receives a response from the web service within the finite time. If this happens we face so-called timing failures when the response is received too late or is not received at all (see Table 1). Thus, the known dependability definition [9] should be extended for service oriented systems as the "ability to deliver service *within the expected time* that can justifiably be trusted".

Table 1. Description of possible servicing outcomes

Result's correctness	Time of receiving	Servicing outcome	Symbolic notation
Correct result	Until timeout	Correct servicing	OK
Non-evident (hidden) error		Hidden error	HE
Exception message		Evident error	EX
Correct result	After timeout	No response during timeout (silence)	TO
Non-evident (hidden) error			
Exception message			

In the Figure 1 we adopt the failure model introduced by Avizienis, et al. in [9] to the distributed nature of service-oriented systems. The model distinguishes between the two main failure domains: (i) *timing failures* when the duration of the response delivered to the client exceeds the specified waiting time – the application timeout (i.e. the service is delivered too late), and (ii) *content failures* when the content (value) of the response delivered to the client deviates from implementing the system function.

Probabilities p^{ok}, p^{he} and p^{ex} are conditional probabilities. They are conditioned on the arrival of some response within the timeout. Probabilities p^{ex} and p^{he} refer to failure modes that in the Avizienis's classification correspond to the *detectability* viewpoint, where they are classified as: *signaled* and *unsignaled* failures, respectively.

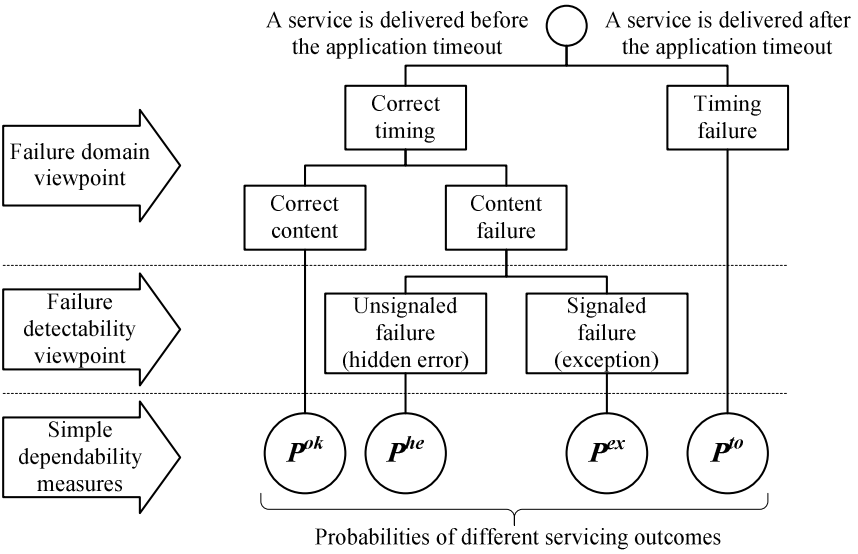

Fig. 1. Service failure modes from the failure domain viewpoint

2.2 Simple and Complex Dependability Measures

Four servicing outcomes form the set of collectively exhaustive events characterized by probabilities:

- p^{ok} – probability of correct servicing within the specified waiting time (i.e. timeout);
- p^{he} – probability of non-evident incorrect servicing within the waiting time;
- p^{ex} – probability of evident incorrect servicing (i.e. exception message reporting) within the specified waiting time;
- p^{to} – probability of timeout.

These probabilities can be combined together to form complex dependability measures characterizing different dependability attributes of a web service or service-oriented system from the client's point of view (see Table 2).

We also introduce the following two measures to estimate the performance of a web service:

- t^{av_srv} – average servicing time (average time estimated for those invocations when a response of any type (OK, HE or EX) was received by a client until the timeout);
- t^{av_wait} – average waiting time (average time estimated for all invocations including those when the timeouts were triggered). It is obvious that $t^{av\_srv} \le t^{av\_wait}$.

Table 2. Complex dependability measures for SOA and Web Services

Dependability attribute	Measure
Accessibility (readiness for the response within the waiting time)	$1-(p^{ex\text{-}con}+p^{to})$, where $p^{ex\text{-}con}$ is the probability of getting an exception message like "TCP connection times out" testifying to inability to establish a network connection with the remote host on the specified TCP port ($p^{ex\text{-}con}$ is a part of p^{ex}); the closer to 1 the better
Availability (readiness for the servicing within the waiting time)	$1-\frac{p^{ex}}{p^{ok}+p^{he}+p^{ex}}=1-\frac{p^{ex}}{1-p^{to}}$, i.e. the probability of getting a response until timeout excluding evident errors, i.e. {OK, HE}; the closer to 1 the better
Trustworthiness (assurance of a correct service within the waiting time)	$1-\frac{p^{he}}{p^{ok}+p^{he}+p^{ex}}=1-\frac{p^{he}}{1-p^{to}}$ or $1-\frac{p^{he}}{p^{ok}+p^{he}}$, i.e. the probability that a web service returns a correct or evident incorrect response (i.e. a signalled failure), given that a response was received before the timeout; the closer to 1 the better

Usually, clients of a remote service have limited possibility to measure its dependability attributes. Most of the time it is only possible to count how many times a response from a web service is not received before the application timeout or how many times exceptional messages of different type are returned instead of the awaited response.

However, as it was shown in [13], typically the exception message does not provide enough information to understand what exactly happened and to distinguish for sure between numerous client, network or service failures. In table 2 we discuss several dependability attributes that can be practically used by a client of a service-oriented systems among those, provided in [9]:

- *Accessibility* – readiness of a service for the response of any type, i.e. {OK, HE, EX}, given that a response is received by a client before the timeout. To account the accessibility attribute client should take out the consideration a client-side exception caused by inability to establish a connection with a web service.

- *Availability* – readiness for the servicing within the waiting time. In the contrast to the classical definition of availability [9] we count here both correct and incorrect servicing (i.e. unsignalled failures) as a client usually cannot distinguish between them without applying application-specific failure detection techniques.
- *Trustworthiness* – assurance of a correct service within the waiting time. This attribute can be measured by applying the voting procedure making use of the redundancy of functionally-equivalent diverse web services provided by different vendors. A more sensitive measure that can be used in addition to the one already presented in Table 2 is the probability of getting an incorrect unsignaled response, given that a response was received (the closer to 0 the better).

2.3 Failure Model Assumptions and Properties

Probabilities p^{ok}, p^{he}, p^{ex} and p^{to} are interconnected and their values depend on the time-out settings used by a client. In our failure model we use the following assumption capturing the time-probabilistic dependence between different servicing outcomes.

Assumption 1. The sum of probabilities of all servicing outcomes {OK, HE, EX, TO} is equal to one as they form the set of collectively exhaustive events: $p^{ok} + p^{he} + p^{ex} + p^{to} = 1$

Assumption 2. Servicing time is a random variable with the known probability distribution function $f_t(t)$ and certain parameters. This function and values of its parameters can be determined by hypothesis checking using experimental data in a way, described in [10].

Assumption 3. Time during which a client waits for the response is limited by the *timeout* parameter.

Assumption 3. Probabilities of the correct (OK), evident and non-evident incorrect (EX and HE) servicing does not depend on the time when the response was received by a client. This assumption can be used if we take out of consideration some specific exceptions mainly caused by networking errors and failures. For example, usually exceptions arisen 21 seconds after a web service has been invoked by a Windows client are caused by inability of client software to establish a TCP connection with the remote host. This time depends on client's OS settings. For example, for Linux it is set at about 180 seconds by default.

The interdependency between probabilities of different servicing outcomes under the specified assumptions is shown in Fig. 2. Based on the analysis of Fig. 1 and taking into account the assumptions used we can formulate the set of properties of the proposed web services failure model.

Property 1. Value of the probability of timeout depends on the application timeout used by a client: $p^{to} = f(timeout)$. The bigger timeout value the bigger the probability of getting a response (the less the probability of timeout):

$$\forall\, timeout_1, timeout_2: timeout_1 > timeout_2 \Rightarrow \int_0^{timeout_1} f_t(t)dt > \int_0^{timeout_2} f_t(t)dt\,.$$

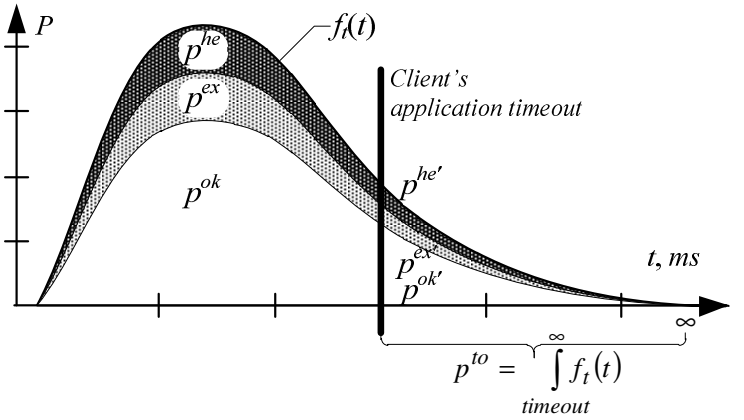

Fig. 2. Time-probabilistic interconnection between different servicing outcomes

Property 2. Changing of timeout value causes changing the probability of timeout and, hence changing (redistribution) values of p^{ok}, p^{he}, p^{ex} as long as the sum of all probabilities must be equal to one. Hence, p^{ok}, p^{he}, p^{ex} are functions of a *timeout* value.

Property 3. Probability of timeout is equal to the integral of $f_t(t)$ over the interval [timeout…∞]:

$$p^{to} = \int_{timeout}^{\infty} f_t(t)dt = 1 - \int_{0}^{timeout} f_t(t)dt \,. \tag{1}$$

Property 4. The sum of probabilities of getting a correct, evident and non-evident erroneous result $p^{ok'}$, $p^{he'}$, $p^{ex'}$ after a timeout is equal to the probability of a timeout p^{to}:

$$p^{ok'} + p^{he'} + p^{ex'} = p^{to} \,.$$

Property 5. The sum of probabilities of getting a correct, evident and non-evident erroneous result before (p^{ok}, p^{he}, p^{ex}) and after ($p^{ok'}$, $p^{he'}$, $p^{ex'}$) the specified timeout is equal to 1:

$$(p^{ok} + p^{ok'}) + (p^{he} + p^{he'}) + (p^{ex} + p^{ex'}) = 1 \,.$$

We hold properties 4 and 5 under the hypothesis that each computation eventually terminates. In practice, the client can face rare cases of a never-ending computation due to some network or service failures. If we use the TCP 'keep-alive' option these connections will be automatically terminate after two hours with the exceptional message "Connection timed out" returned to the client.

Property 6. There is a constant ratio between the corresponding probabilities of getting a correct, evident and non-evident erroneous result before (p^{ok}, p^{he}, p^{ex}) and after ($p^{ok'}$, $p^{he'}$, $p^{ex'}$) the specified timeout:

$$\frac{p^{ok}}{p^{ok'}} = \frac{p^{he}}{p^{he'}} = \frac{p^{ex}}{p^{ex'}}.$$

Property 7. Pairwise sums of the corresponding probabilities of getting a correct, evident and non-evident erroneous result before (p^{ok}, p^{he}, p^{ex}) and after ($p^{ok'}$, $p^{he'}$, $p^{ex'}$) the specified timeout are equal to the constant values independently of the timeout value:

$$(p^{ok} + p^{ok'}) = p^{ok\infty}, \quad (p^{he} + p^{he'}) = p^{he\infty}, \quad (p^{ex} + p^{ex'}) = p^{ex\infty},$$

where $p^{ok\infty}$, $p^{ex\infty}$, $p^{he\infty}$ are the probabilities of getting a correct, evident and non-evident erroneous result with the unlimited waiting time, i.e. when *timeout* $\to\infty$.

Property 8. There are analytic dependencies between probabilities of getting a correct, evident and non-evident erroneous result before (p^{ok}, p^{he}, p^{ex}) and after ($p^{ok'}$, $p^{he'}$, $p^{ex'}$) the specified timeout and a probability of this timeout triggered, i.e. p^{to}:

$$p^{ok} = (1 - p^{to}) \cdot p^{ok\infty}, \; p^{he} = (1 - p^{to}) \cdot p^{he\infty}, \; p^{ex} = (1 - p^{to}) \cdot p^{ex\infty}.$$

Taking into account the third property we can define:

$$p^{ok} = p^{ok\infty} \cdot \int_0^{timeout} f_t(t)dt \tag{2}$$

$$p^{he} = p^{he\infty} \cdot \int_0^{timeout} f_t(t)dt \tag{3}$$

$$p^{ex} = p^{ex\infty} \cdot \int_0^{timeout} f_t(t)dt \tag{4}$$

3 Average Servicing and Waiting Time Assessment Models

The average servicing time can be estimated by using a well-known equation for the mean value of the independent variable defined by its probability density function:

$$t^{av} = E[T] = \int_0^{\infty} t \cdot f_t(t)dt.$$

The expectation of a probability density function $f_t(t)$ truncated from the right by a *timeout* can be defined as [11]:

$$t^{av\_srv} = E^{trunc\_srv}[T] = \frac{\int_0^{timeout} t \cdot f_t(t)dt}{\int_0^{timeout} f_t(t)dt} = \frac{\int_0^{timeout} t \cdot f_t(t)dt}{F_t(t \le timeout)}. \tag{5}$$

However, equation (5) is correct only if we are interested in the average servicing time t^{av_srv} of those invocations in which a response of any type (OK, HE or EX) is received by a client before the specified timeout.

The average waiting time t^{av_wait} estimated for all invocations including those when a timeout is triggered can be defined as a sum of the average servicing time t^{av_srv} under the specified timeout and a product of the timeout value and the probability of timeout:

$$t^{av\_wait} = E^{trunc\_wait}[T] = \int_0^{timeout} t \cdot f_t(t)dt + timeout \cdot \int_{timeout}^{\infty} f_t(t)dt = \\ = \int_0^{timeout} t \cdot f_t(t)dt + timeout \cdot \left(1 - F_t(t \le timeout)\right). \quad (6)$$

This is due to the fact that the waiting time for those invocations for which a timeout is triggered is equal to the timeout value. Hence, the weight of a tail of the probability density function $f_t(t)$ truncated by the timeout is concentrated at the truncation border (see Fig. 3).

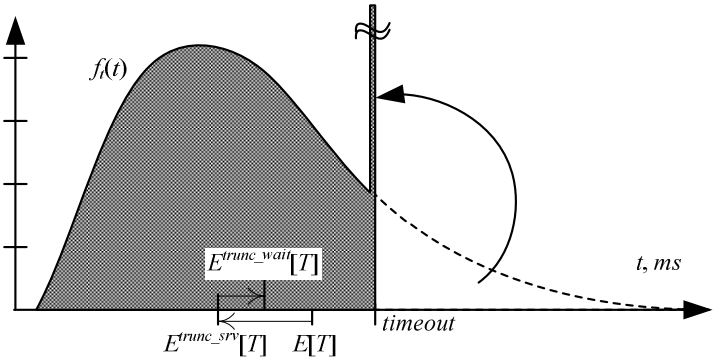

Fig. 3. Concentration of the mass of a probability of a timeout at the truncation border

4 An Example of Average Servicing and Waiting Times Estimation

This section presents an example of estimating the average servicing time t^{av_srv} and the average waiting time t^{av_wait} in case of the exponential distribution of a web service response time. By applying equations (4) and (5) we have:

$$t^{av\_srv}(timeout) = \frac{\int_0^{timeout} t \cdot \mu \cdot e^{-\mu \cdot t}}{1 - e^{-\mu \cdot timeout}} = -\frac{e^{-\mu \cdot timeout} + \mu \cdot timeout \cdot e^{\mu \cdot timeout} - 1}{\mu \cdot \left(1 - e^{-\mu \cdot timeout}\right)}, \quad (7)$$

$$t^{av\_wait}(timeout) = \int_0^{timeout} t \cdot \mu \cdot e^{-\mu \cdot t} + timeout \cdot e^{-\mu \cdot timeout} = -\frac{e^{-\mu \cdot timeout} - 1}{\mu}. \quad (8)$$

Corresponding curves of t^{av_wait} and t^{av_srv}, as well as p^{to} depending on the timeout value are shown on Fig. 4 and Fig. 5. The curves were plot using parameter μ which is equal to 0.01 that corresponds to 100 ms of average response time in case of the unlimited waiting time. Figures 4 and 5 show that, actually, there is no point in waiting for a response from a web service for more than 600 *ms* as the probability of timeouting by that time is less than $2{,}47 \cdot 10^{-3}$. At the same time an average servicing time in this case is equal to 98.5 *ms*. If the response time of a web service is subject to one of the heavy-tailed distributions, which is more realistic in practice, the profit will be more significant.

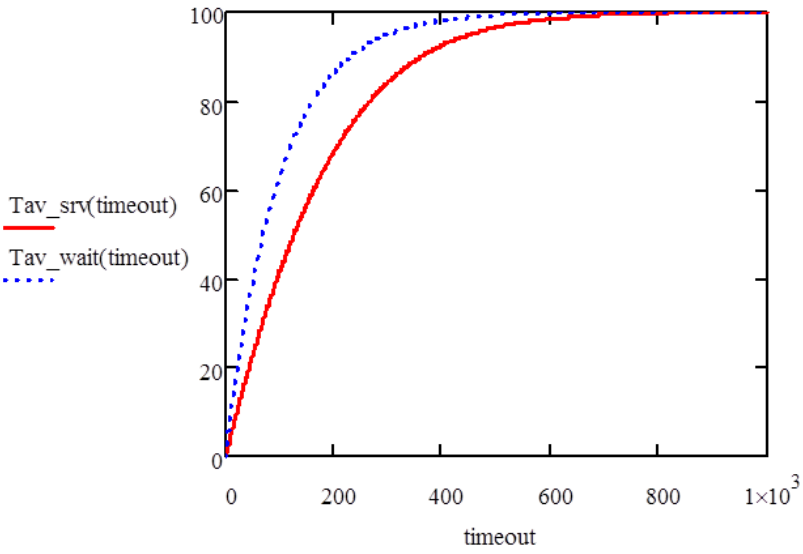

Fig. 4. Curves of the average servicing time t^{av_srv} and average waiting time t^{av_wait} depending on timeout value in case of the exponential distribution of the random response time with the parameter μ=0.01

Finally, we can solve the equation (1) relatively to *timeout* and then put the result into (5) and (6). This will allow us to resolve a trade-off problem between the dependability and performance of a particular Web service. In our example (in case of exponential distribution of the Web service response time) the *timeout* can be expressed by the formula:

$$p^{to}(timeout) = \int_{timeout}^{\infty} \mu \cdot e^{-\mu \cdot t} dt = e^{-\mu \cdot timeout} \Rightarrow timeout(p^{to}) = -\frac{ln(p^{to})}{\mu}. \quad (9)$$

Putting it into (7) and (8) we can get equations estimating t^{av_wait} and t^{av_srv} depending on specified probability of timeout p^{to}. This dependency, shown in Fig. 6, can be used to setup application timeout and to assess performance of a web-service (using t^{av_wait} and t^{av_srv} measures) with regards to dependability requirements expressed in form of maximal allowed value of p^{to}.

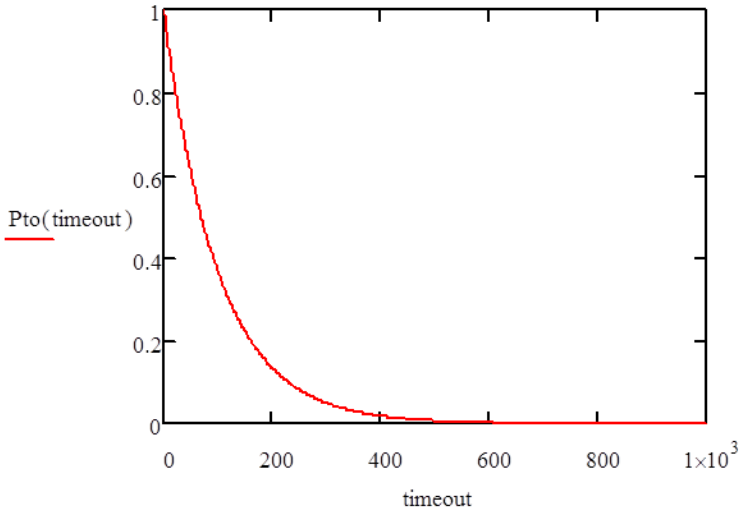

Fig. 5. Curve of the probability of timeout p^{to} depending on timeout value in case of the exponential distribution of the random response time with the parameter μ=0.01

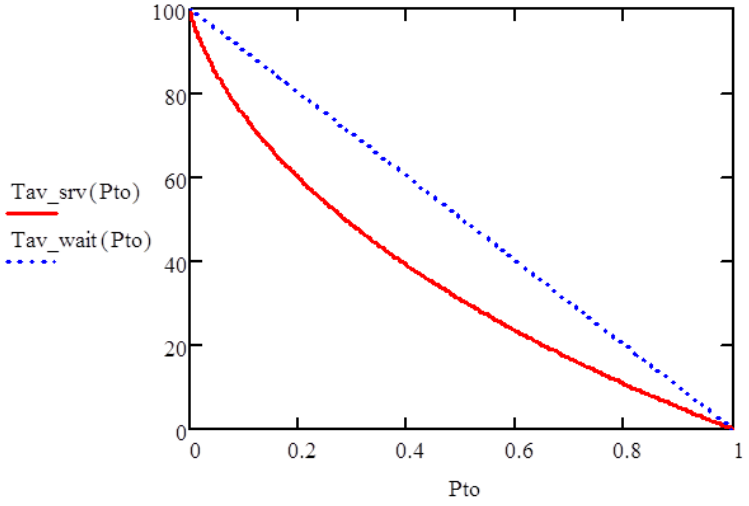

Fig. 6. Curves of the average servicing time t^{av_srv} and average waiting time t^{av_wait} depending on the probability of timeouting p^{to} in case of the exponential distribution of the random response time with the parameter μ=0.01

5 Conclusions

Ensuring and assessing dependability of complex service-oriented systems are complicated when these systems are dynamically built or when their components (i.e. web services) are dynamically replaced by the new ones with the similar functionality but unknown dependability characteristics.

The lack of sufficient evidence about the characteristics of the communication medium, components and their possible dependencies makes it extremely difficult to achieve and predict (composite) service dependability which can vary over a wide range in a very random manner. This uncertainty of services running over the Internet and clouds exhibits itself through the unpredictable response times and complex failure model, resulting from the distributed and loosely-coupled cooperation between the service provider and consumers in the unpredictable Internet-environment.

In the paper we introduce a failure model for service-oriented applications, that can be considered as a specialization for this domain of the general failure model presented by Avizienis, Laprie, Randell and Landwehr in their IEEE TDSC 2004 paper [9]. Based on this failure model, we discuss the relationship between different failure modes and timeout settings, and propose the set of simple and complex measures estimating dependability of SOA solutions from the client's point of view. We also propose analytical models for average servicing and waiting times estimation depending on the application timeout settings used by the client software.

The proposed equations for estimation of a probability of different servicing outcomes and average servicing and waiting times can help in choosing the right application timeouts which are the fundamental part of all fault-tolerant mechanisms working over the Internet used as the main error detection mechanism here. Making use these equations software developers can solve a trade-off problems between maximizing the probability of a correct servicing and minimizing the waiting or servicing time.

Acknowledgement. We are grateful to Aad van Moorsel for his feedback on the earlier version of this work. The work is partially supported by the FP7 KhAI-ERA project and by the TrAmS-2 EPSRC/UK platform grant.

References

1. Cardellini, V., Casalicchio, E., Regina, K., et al. (eds.): Performance and Dependability in Service Computing: Concepts, Techniques and Research Directions. IGI Global (2012)
2. Lyu, M.: Service Reliability Engineering: Performance Evaluation, Fault Tolerance, and Reliability Prediction. In: International Symposium on High Confidence Software (2011)
3. Sargeant, A.: Dependability of Dynamic Binding in Service-Oriented Computing. In: Int. Conf. on Dependable Systems and Networks (2011)
4. Zheng, Z., Lyu, M.A.: QoS-Aware Fault Tolerant Middleware for Dependable Service Composition. In: Int. Conf. On Dependable Systems and Networks, pp. 239–248 (2009)
5. Zheng, Z., Lyu, M.: A Runtime Dependability Evaluation Framework for Fault Tolerant Web Services. In: Int. Conf. On Dependable Systems and Networks (2009)

6. Fahad, A., Saurabh, A., Saurabh, B.: Dangers and Joys of Stock Trading on the Web: Failure Characterization of a Three-Tier Web Service. In: 30th International Symposium on Reliable Distributed Systems (2011)
7. Mendes, N., Duraes, J., Madeira, H.: Evaluating and comparing the impact of software faults on web servers. In: 9th European Dependable Computing Conference (2010)
8. Smith, D., Simpson, K.: Safety Critical Systems Handbook: A Straightforward Guide to Functional Safety, IEC 61508 and Related Standards, 3rd edn. Butterworth-Heinemann, Oxford (2004)
9. Avizienis, A., Laprie, J.-C., Randell, B., Landwehr, C.: Basic Concepts and Taxonomy of Dependable and Secure Computing. IEEE Transactions on Dependable and Secure Computing 1(1), 11–33 (2004)
10. Gorbenko, A., Romanovsky, A., Mamutov, S., et al.: Real Distribution of Response Time Instability in Service-Oriented Architecture. In: 29th IEEE International Symposium on Reliable Distributed Systems, pp. 92–99 (2010)
11. Olive, D.J.: Applied Robust Statistics. Southern Illinois University Press, Illinois (2008)
12. Reinecke, P., van Moorsel, A., Wolter, K.: Experimental Analysis of the Correlation of HTTP GET Invocations. In: Horváth, A., Telek, M. (eds.) EPEW 2006. LNCS, vol. 4054, pp. 226–237. Springer, Heidelberg (2006)
13. Gorbenko, A., Kharchenko, V., Romanovsky, A., Mikhaylichenko, A.: Experimenting with exception propagation mechanisms in service-oriented architecture. In: 4th Int. Workshop on Exception Handling, pp. 1–7 (2008)