УДК 004.052

A. BOYARCHUK, V. KHARCHENKO

National Aerospace University "KhAI", Ukraine

MAKING WEB-SERVICES FAULT-TOLERANT: METHODS AND TECHNICS

With emergent growing use of Internet, Web services become increasingly popular and their growth rate surpasses even the most optimistic predictions. Services are self-descriptive, self-contained, platform-independent and openly available components that interact over the network. They are written strictly according to open specifications and/or standards and provide important and often critical functions for many business-to-business systems. The analysis of failure stages of web-services is presented. The matrix of correspondence between fault-tolerance stages and methods for their ensuring is developed and explored.

SOA, Web service, fault-tolerance (FT), FT stages, FT methods, FT techniques.

Introduction

The use of Web services in very different areas is growing; there is an increasing demand for dependability. Service-oriented Architectures (SOA) are based on a simple model of roles. Every service may assume one or more roles such as being a service provider, a broker or a customer.

The use of services, especially Web services, became a common practice. In Web services, standard communication protocols and simple broker-request architectures are needed to facilitate an exchange (trade) of services, and this model simplifies interoperability. In the coming years, services are expected to dominate software industry. As services begin to permeate all aspects of human society, the problems of service dependability, security and timeliness are becoming critical, and appropriate solutions need to be made available.

Several fault tolerance approaches have been proposed for Web services in the literature [1 – 8], but the field still requires theoretical foundations, appropriate models, effective design paradigms, practical implementations, and in-depth experimentations for building highly dependable Web services.

Related works. The objective of the paper

There are several scientific works devoted to the definition of principles of "dependable computing" [9] and "secure fault tolerance" [10]. In this researches the way for realization of fault tolerance for different computer systems. But the mentioned paper do not include the mechanisms and techniques used in service-oriented architectures. Currently, there is no experimental investigation to evaluate the reliability and availability of Web services systems. The paper [11] may be considered as one of the most important paper in this area. In this paper, the authors identify parameters impacting the Web services dependability, describe the methods of dependability enhancement by redundancy in space and redundancy in time and perform a series of experiments to evaluate the availability of Web services. To increase the availability of the Web service, they use several replication schemes and compare them with a single service.

At the other hand, it is worth mentioning the important work [12] which provides the comprehensive analysis of the concept of dependability and description of the most important tasks of analysis and synthesis which might be solved in the process of development the dependable system, including systems of service-oriented infrastructure. The *specific objective* of the proposed article is analysis of methods and techniques for achieving web-services fault-tolerance by means of development and survey of the two-dimension matrix for correspondence between the sets of fault-tolerance stages and methods/techniques used for implement these stages. The paper includes the description of the failure response stages of the web-services (section 1), major fault classification (section 2), analysis of the methods for ensuring fault-tolerance (section 3) and developed matrix which might be considered as correspondence between the above-mentioned categories (section 4). Conclusions as well as outlining of the roadmap for future works are presented at the section 5 of the paper.

1. Failure Response Stages of Web Services

Web services go through different operation modes, so when failures occur and the failure response of Web services can be classified into different stages. When a failure occurs, the Web service should confine the failure by applying fault detection techniques to find out the failure causes and the failed components should be repaired or recovered. Then, reconfiguration, restart and reintegration should follow. The flow of the failure response of a Web service is proposed below and the details of each stage are described as follows [12]:

Fault confinement. This stage limits the fault impact by attempting to contain the spread of fault effects in one area of the Web service, thus preventing contamination of other areas. Fault-confinement can be achieved through the use of fault detection within the Web services, consistency checks, and multiple requests/confirmations.

Fault prevention. This stage includes the mechanisms for preventing the possible faults forecasted by special techniques.

Fault detection. This stage recognizes that something unexpected has occurred in a Web service. Fault latency is the period of time between the occurrence of a fault and its detection. Techniques fall here into two classes: off-line and on-line. With off-line techniques, such as diagnostic programs, the service is not able to perform useful work while under test. On-line techniques, such as duplication, provide a real-time detection capability that is performed concurrently with useful work.

Diagnosis. This stage is necessary if the fault detection technique does not provide information about the fault location. Typically, fault diagnosis encompasses both fault detection and fault location.

Isolation. This stage occurs when a fault is detected and located. The Web service can be composed of different components. When providing the service, there may be a fault in individual components. The system may reconfigure its components either to replace the failed component or to isolate it from the rest of the system.

Tolerating. This stage utilizes techniques to eliminate the effects of faults. Three basic recovery approaches are available: fault masking, retry and rollback. Faultmasking techniques hide the effects of failures by allowing redundant information to outweigh the incorrect information. Web services can be replicated or implemented with different versions (NVP). Retry undertakes a second attempt at an operation and is based on the premise that many faults are transient in nature. Web services provide services through a network, and retry would be a practical approach as requests/replies may be affected by the state of the network. Rollback makes use of the fact that the Web service operation is backed up (checkpointed) at some point in its processing prior to fault detection and operation recommences from that point. Fault latency is important here because the rollback must go back far enough to avoid the effects of undetected errors that occurred before the detected error.

Restart. This stage occurs after the recovery of undamaged information. – Hot restart: resumption of all operations from the point of fault detection and is possible only if no damage has occurred.

– Warm restart: only some of the processes can be resumed without loss.

– Cold restart: complete reload of the system with no processes surviving. The Web services can be restarted by rebooting the server.

Repair. At this stage, a failed component is replaced. Repair can be off-line or on-line. Web services can be component-based and consist of other Web services. In off-line repair, either the Web service will continue if the failed component/sub-Web service is not necessary for operation or the Web services must be brought down to perform the repair. In on-line repair, the component/sub-Web service may be replaced immediately with a backup spare or operation may continue without the component. Reconfiguration may be included at this stage if it come directly after repair.

Reintegration. At this stage the repaired module must be reintegrated into the Web service. For on-line repair, reintegration must be performed without interrupting the Web service operation.

2. The faults

One of the important part of the taxonomy of the dependability [13] is threats. First of all we have to consider the faults which may be caused by different reasons.

As a result the set of faults might be classified into three big groups [12]: development faults (DF), physical faults (PF) and interaction faults (IF). The first group (DF) are usual for software products and become visible in case of special conditions occurred (i.e. incorrect input data). The second group (PF) are usual for hardware components and appeared due to the natural reasons (depreciation of the equipment). The last but not the least group (IF) are the consequence of external intrusions (informational attacks, human errors, short-term physical forces led to hardware faults).

As known the faults cause the failures of computer systems. The sequence of the actions occurred for the different faults is the following:

1. For DF: The mistake actions or decisions during the project development lead to fault injection to the project. The fault appears during the functioning of the computer system (web-service) in case of invalid input data. The next step is the system failure during the delivering of the requested service.

2. For PF: The fault is incurred by natural reason. The computing error is appeared and system failure will be the result.

3. For IF: after the external intrusion of the informational, physical or other character the fault is appeared; the computing error is appeared and system failure will be the result.

3. The methods for ensuring fault-tolerance

The set of the methods for ensuring fault-tolerance in the systems of service-oriented architecture may be distributed to the three groups: majority reservations, diversity approach, specific methods and their combinations.

The each method for ensuring fault-tolerance can be studies taking into consideration the presence or absence of the elements of the set described at the Section 3 (confinement, prevention, detection, diagnosis, isolation, tolerating, restart, repair, reintegration) [12].

Majority reservation (2 of 3). The majority reservation ensures the fault-tolerance by simple tolerating the fault.

Adaptive majority reservation (AMR). This method includes the actions “fault detection” (detection of the fault at the second channel), “fault identification” (identification of the working channel), “tolerating the fault” (tolerating the fault of first and then second channel), “reconfiguration” (reconfiguration from majority structure to single-channel one).

AMR with rating voting of the channels. This method may be considered as one of the most effective ones. It allows to tolerate and, in some cases, to prevent the possible fault. This method may be implemented, for example, in service-oriented architectures, which integrate the target web-services with the special Middleware, mechanisms for rating voting and exception handling [14,15,16].

Diversity approach. Using of the method allows to solve the tasks aimed at the development of software products stable against the FF and DF. It became possible due to decreased probability of fault by general reason caused by defects of software tools. The different versions of software are in operation in different reserve channels. Beside this, the usage of diversion allows to increase the security of the product thanks to adaptive choice of diverse configuration of web-servers, operation systems and applications. This method is operating on the basis of aggregation and dynamical updating of the information containing the data of different types of component vulnerabilities for different types of attacks [17].

Single service without retry and reboot. The Web service is provided by a single server without any replication. No redundancy technique is applied to this Web service.

Single service with retry. The Web service provides the service and the client retries another Web service when there is no response from the original Web service after timeout.

Single service with reboot (restart). The Web service provides the service and the Web service server will reboot when there is no response from the Web service. Clients will not retry after timeout when there is no response from the service.

Spatial replication (SR) with failover. We use a generic spatial replication: The Web service is replicated on different machines and the request is transferred to another machine when the primary Web service fails (failover). The replication manager coordinates among the replicas and carries out a failover in case of a failure. Clients will only submit the request once and will not retry.

SR with failover and retry. This is a hybrid approach. Similar to the point 4 where the Web service is replicated on different machines and the request is transferred to another one (failover) when the primary Web service fails. But the client will retry if there is no response from the Web service after timeout.

4. Development of the integrated matrix

On the basis of the described stages of fault tolerance on the one hand and the methods and techniques for ensuring fault tolerance on the other one we try to develop the matrix 9 x 9 for exploring the correspondence between the stages and methods (table 1).

As a result we have to mention that method of adaptive majority reservations with rating voting of the channels is the most effective method among given three methods of majority reservations. This is confirmed by the results of survey [12] stated that this integrated method might be used in the complex systems of web-services managed by Middleware subsystem.

Table 1

Correspondence between fault-tolerance stages and methods

Stage/ method	Majority reservation	AMR	AMR with rating voting of the channels	Diversity approach	Single service without retry and reboot	Single service with retry	Single service with reboot	SA with failover	SA with failover and retry
Confinement			+	+				+	+
Prevention			±	+			+		
Detection		+	+	+	+	+	+	+	+
Diagnosis		+	+	+		+	+	+	+
Isolation		+	±			+		+	+
Tolerating	+	+	+	+		+			+
Restart		+	+	+					
Repair (+ reconfig)		+	±						
Reintegration		±	±	+				+	+

The diversity approach occupies the special status at the matrix: it may consist of the diversity approaches to the development of the target web-services, middleware subsystem, web-servers platforms, special environments used in application of service-oriented architectures. It is worth mentioning that diversity approach may be effectively completed with principles of multi-parametrical adaptation (MA) and multi-level managed degradation (MD) and thus form the joint concept called 3M [18]. MA is used for implementation of different software contours of reconfiguration management and MD is responsible for redistribution of redundant and irredundant resources.

The last but not the least group of methods based on different implementations of redundancy has presented the key fact that the method of spatial replication with failover and retry is the method with the biggest covering of fault-tolerance stages in spite of its time and resource inefficiency. The system failover requires additional time but no resources needed for its execution; nevertheless the replication mechanisms are in extreme need of replication points and system resources.

5. Conclusions. Future works

The article is aimed at analysis of methods and techniques for achieving web-services fault-tolerance by means of development and survey of the two-dimension matrix for correspondence between the sets of fault-tolerance stages and methods/techniques used for implement these stages.

We have explored 9 stages of fault tolerance: fault confinement, prevention, detection, diagnosis, isolation, tolerating, restart, repair and reintegration and make a brief description of each stage. We surveyed the 9 different methods and techniques for ensuring fault-tolerance at the systems of service-oriented architecture and analyzed their impact on the each stage of fault-tolerance. We have to conclude that adaptive majority reservation with rating voting of the channels may be considered as the method which covers (fully or partially) all set of the fault-tolerance stagers. Besides, we have to underline that spatial replication with failover and retry has the most potential and is considered as the most effective method for support of fault-tolerance in complex system of web-services.

In the future, we plan to extend the proposed scheme with a wide variety of systems and environments and analyze the impact of the used method (or its combination) of fault-tolerance to heterogeneous systems. The complexity of the realization of each method might be also considered for applications of service-oriented architecture. Secondly we are going to make the assessment of each method and technique taking into consideration the set of criteria such as estimated time, required resources and impact on usability.

References

1. Looker N., Munro M. WS-FTM: A Fault Tolerance Mechanism for Web Services // University of Durham, Technical Report. – 19 Mar. 2005.

2. Liang D., Fang C., Chen C. FT-SOAP: A Fault-tolerant Web Service // Institute of Information Science, Academia Sinica. – Technical Report 2003.

3. Liang D., Fang C. Yuan S. A Fault-Tolerant Object Service on CORBA // Journal of Systems and Software. – 1999. – Vol. 48. – P. 197-211.

4. Townend P., Groth P., Looker N., Xu J. Ft-grid: Fault-tolerance system for e-science // Proc. of OST e-Science 4 All Hands Meeting. – 2005. – Vol.1. – P.12-20.

5. Merideth M. Byzantine-Fault-Tolerant Middleware for Web-Service Application // Proc. of IEEE Symp. on Reliable Distributed Systems, Orlando, FL, 2005. – P.91.

6. Erradi A., Maheshwari P. A broker-based approach for improving Web services reliability // Proc. of IEEE International Conference on Web Services, 11-15 Jul. 2005. – Vol. 1. – P. 355-362.

7. Tsai W., Cao Z., Chen. Y. Web services-based collaborative and cooperative computing // Proc. of Autonomous Decentralized Systems, 2005. – P. 552-556.

8. Leu D., Bastani F., Leiss E. The effect of statically and dynamically replicated components on system reliability // IEEE Transactions on Reliability. – 1990. - Vol .39, Issue 2. – P.209-216.

9. Харченко В.С. Исследование гарантоспособных структур УВС // Проектирование многомашинных комплексов РВ. – М.: Знание. – 1990. – С. 58-61.

10. Харченко В.С. Модели и свойства отказоустойчивых многоальтернативных систем // Автоматика и телемеханика. – 1992. – № 12. – С. 140-147.

11. Pat. P.W. Chan, Michael R. Lyu, Malek M. Making Services Fault Tolerant // ISAS 2006, Springer-Verlag Berlin Heidelberg. – 2006. – P 43-61.

12. Харченко В.С. Концепция гарантоспособности и ее приложения для ИУС АЭС // Ядерная и радиационная безопасность. – 2007. – № 1. – С. 35-41.

13. Avizienis A., Laprie J.-C., Randell B., Landwehr C. Basic Concepts and Taxonomy of Dependable and Secure Computing // IEEE Trans. on Dependable and Secure Computing. – 2004. – Vol. 1, № 1. – P. 11-33.

14. Харченко В.С., Горбенко А.В. и др. Инструментальная платформа для создания гарантоспособных композитных (нтегрированных) web-сервисов "IN-DECS" // Информационные технологи в науке, производстве, образовании. – Х.: ХНУРЭ, 2005. – С. 35-37.

15. Tartanoglu F. and s.o. Coordinated Forward Error Recovery for Composite Web Services // Proc. of the 22th Symposium on Reliable Distributed Systems (SRDS), Florence, Italy, 2003. – P. 167-176.

16. Kharchenko V., Popov P., Romanovsky A. On Dependability of Composite Web Services with Components Upgraded Online // Proc. of Workshop on Architecting Dependable Systems, Italy, 2004. – P. 14-20.

17. Лысенко И.В., Халин М.В., Харченко В.С. Анализ и разработка методов и средств защиты информации с использованием многоверсионных технологий. – Отчет по НИР № 0104U003502 "Разработка научно-методических основ и информ. технологий оценки и обеспечения отказоустойчивости и безопасности КС аэрокосмических комплексов КП" / Научн. рук. Харченко В.С. – НАКУ "ХАИ", 2003. – С.415-445.

18. Харченко В.С., Токарев В.И. Проектирование отказоустойчивых и живучих КС управления на основе концепции "3М" // Вісник Технологічного університету Поділля.– 2003. – № 3. – С.29-32.

Поступила в редакцию 22.02.2007

Рецензент: д-р техн. наук, проф. В.А. Краснобаев, Харьковский национальный технический университет сельского хозяйства им. П. Василенко, Харьков.