UDC 004.7

Yu.D. Svystunov[1], N.V. Lukova-Chuyko[2]

[1] *National Technical University "Kharkiv POLITECHNIC Institute", Kharkiv*
[2] *Taras Shevchenko Kyiv National University, Kyiv*

# USE OF THE CONCEPT OF FUNCTIONAL RESERVE IN ORDER TO PROVIDE QUALITY ASSURANCE OF WEB SERVICES

*Nowadays, it is already not enough for enterprise information system to provides simple automation of informational and computational business problems, a corporate information system must change as quickly as quickly company's business requirements and business processes change. It was one of the main preconditions for the appearance of service oriented architecture, a.k.a. SOA. It is construed as a paradigm of distributed set of functionality, where each of them is represented as a service. The paper describes the main parameters of the quality of such services determined by different standards and offers the concept of functional reserve in order to provide guaranteed services quality when creating an application represented by a composition of services.*

***Keywords:*** *Web service, SOA, quality, conceptual model, web-services provision.*

## Introduction

Currently, the success of a business depends strongly on how it is automated and how quickly the company can offer a new service or product to market.

Almost every IT division of a company must have had a task to provide business with uninterrupted IT services. Implementation of traditional solutions for application integration is a challenge that requires significant investment. In addition, during the installation, it is often needed to write the code. In this regard, a problem of developing technology faster and less expensive application integration appeared. SOA, or service-oriented architecture, provides placement of services in the network in the runtime, that allows automating these intensive processes, thereby significantly decreasing all the costs for integration [1].

Also, the basic preconditions for the emergence of SOA include high dynamics of modern business and steadily increasing demands on a constant adaptation of information systems in relation to this dynamic. It is already not enough that information system provides simple automation of informational and computational business problems. It is necessary to strive for the situation when rapidly changing conditions of a business arising from increased competition find the full reflection in an information system, in other words, a corporate information system must change as quickly as quickly company's business requirements and business processes change.

The integration of heterogeneous and distributed data cannot solve all the issues of enterprise management [4]. According to the process approach, the greatest value is not the data itself, but the information use in various business processes of a company. In advanced information systems, it is accepted to consider as "atomic" unit not the raw data, but some service that meets some elementary business process. In particular, such a service can give some data being analogous to "atomic" unit of classic information systems.

SOA is construed as a paradigm of organization and use of distributed set of features that could be controlled by different owners. The basic concept of this architecture is a notion of information service. Information service is an atomic function of an automated system that is suitable for use while developing applications that implement the business logic of processes that are being automated, both for the system itself and for other automated systems' applications.

The basis of service-oriented architectures is distributed software components (services) provided or used by independent parties. Since access to these components is not limited by the organizations' bounds, it must be supported by explicit contracts of components and by generally accepted standards. This is equally important for the user that service provides not only essential functionality but also other non-functional requirements (speed, security, etc.). Thus, currently, more and more attention is given to the policy of ensuring the quality of service (QoS) [7]. Indicators of the quality of Web-services include providing the necessary level of security, reliability and fault tolerance.

To ensure the security of Web-services the numerous standards and supporting their methods are directed especially: WS-Security, WS-Trust, Extensible Access Control Markup Language (XACML), Security Assertion Markup Language (SAML), etc. [2]. For example, it is safe to use banking services, pay bills and make purchases on the Internet.

Researches in the areas of Web-services and SOA are now focused on the protocols, functionality, transactions, ontologies, composition, semantic Web, and interoperability, but insufficient attention is given to reliability of services.

SOA allows developers to search for services and use services provided by different suppliers. Traditional software reliability is proof of correctness, fault tolerance, formal verification based on models (model checking), testing, evaluation, and etc., that can increase the level of trust for some services. However, developers should recycle these methods to enable their application to provide their use in dynamic applications consisted of linked services at runtime.

Therefore, it is important to develop methods and models of integration and managing services in order to provide guaranteed quality of service in SOA systems.

## 1. Quality of Web services

Considering the main existing SOA systems in terms of management services, it can be concluded that they provide great opportunities monitoring services and alerts the user in different situations. However, they do not provide the quality of service policy management and do not allow end-users to obtain data of services' quality.

The main objective of SOA is ensuring cooperation of various weakly-connected applications. These applications can vary greatly in functionality as well as in non-functional characteristics (performance, reliability, cost, etc.) that must be valid during the time of providing service.

The quality of service (QoS) is one of the main criteria when choosing service instance by the user. In this case, users pay equal attention to both functional and non-functional requirements.

Currently, there are several methods to determine the level of QoS, which can provide a copy of the service. Here are some of them.

WSDL (Web Service Definition Language) is a standard mechanism for WS-Policy Framework identified by the OASIS consortium. Just for this purpose, the UDDI (Universal Description, Discovery, Integration) is widely used.

The use of WS-component is only possible for newly established services. Primarily due to the fact that these components should be embedded in the implementation of the service. At the same time, there are quite a number of services that are based on the technology of SOA but do not use QoS components. Similarly, while using WS-components, the question regarding service modification and changing QoS parameters remains, because the standard does not regulate this process.

An important issue is the control of the stated parameters of QoS, since over time these parameters may change under different kinds of factors. Consequently, a situation may arise when a user requests and receives a service with specific parameters of QoS and in process of use of service these parameters will change.

Each copy of service publishes information about themselves in a single place named UDDI registry using XML-document. This document defines the meta-types that reflect different aspects of this service (area of destination, the owner, terms of use and different technical parameters (address, the protocol used, the claimed QoS performance)). If there is a necessity of accessing service of a specified type, scanning of all the services registered in UDDI registry is performed in order to choose the service that meets user's requirements. In this case, the question regarding the control of current QoS parameters for compliance with stated remains open as well. There are guidelines under which the owner of the of service should on their own monitor the status of an instance of a service and make changes to the UDDI registry. However, these recommendations have not found practical application.

Thus, it is quite important to identify and control the level of QoS that the specified service instance can provide. Consequently, there is a need to develop methods that will perform monitoring of the state of service instances and pass it to the network management systems.

Terms and concepts in the area of control and evaluation of the quality exactly web-services are not defined by national and international standards. Given the focus of industrial services provided by web services using data-processing systems for various purposes that are part of sophisticated hardware and software, the closest in this subject area and related areas of standardization are:

1) GOST 27.002-89 "Reliability engineering. Basic concepts. Terms and definitions";

2) GOST 34.003-90 "Information technology. The complex of standards for automated systems. The automated systems. Terms and definitions";

3) ISO / IEC 9126-93 "Information technology. Evaluation of software products. Characteristics of the quality and guidance on their application" [3];

4) A series of international standards ISO / IEC 25000 - Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE).

The consortium OASIS (Organization for the Advancement of Structured Information Standards) also works on developing the standards regarding information systems. OASIS is the leader in the number of issued standards pertaining to the web services [7].

### 1.1. Indicators by the consortium OASIS standards

Contract of service covers both functional and non-functional aspects of the behavior of a service component. Functional aspects consist of the business semantics of component operations, including its interface and protocol used. Non-functional aspects include technical features of interaction, such as data serialization and protocols QoS.

QoS protocols contain information about the quality of service provision. The level of quality of service by OASIS standard has four quality indicators: response time, maximum capacity, availability, and reliability.

Response time means the length of time after sending a request to the moment of receiving a response. Response time can vary, depend on the three types of delays: client delays, network delays, and server latency. Client latency (CL) is a time spent by the client part of an application during the processing of the request. This is the time from request to the client until the sending request to the server ($CL_1$), and after receiving the response by the client until the completion of its processing ($CL_2$). Network latency (NL) is a time spent on the request and response messages transfer over the network. This is the amount of time between the event "client sends a request" and the event «web-service receives the request» ($NL_1$), and the time between event "server sends a response" and the event "client receives a response» ($NL_2$).

Server latency (SL) is a time of request processing and response composing on server-side. This is the amount of time between the event "server sends a request" and the event «web-service receives the request» ($SL_1$), the time of request processing ($SL_2$), and the time between the event «web-service sends a response" and the event "server receives a response» ($SL_3$).

Three types of latency and response time can be calculated using following formulas:

$$CL = CL_1 + CL_2, \quad (1)$$

$$NL = NL_1 + NL_2 \quad (2)$$

$$SL = SL_1 + SL_2 + SL_3, \quad (3)$$

$$\text{Response Time} = CL + NL + SL, \quad (4)$$

Maximum throughput (*MT*) is defined as the maximum number of requests that service provider can process for a certain period of time. Throughput can be calculated as follows:

$$MT = \max\left(\frac{\text{number of requests}}{\text{measured time}}\right), \quad (5)$$

where *num of requests* is the number of requests processed by the server for a *measured time*; *measured time* is the time period during which the measurements were carried out.

System's availability is a measure that determines the accessibility of Web-service. Availability can be expressed by the following formula:

$$\text{Availability} = 1 - \frac{\text{down time}}{\text{measured time}}, \quad (6)$$

where *down time* is a time during when web-service was not available; *measured time* is the time period during which the measurements were carried out.

Reliability is a measure that determines the probability of returning the response after successful execution of a web-service request. Reliability can be expressed by the following formula:

$$\text{Reliability} = \frac{\text{num of responses}}{\text{num of requests}}, \quad (7)$$

where *num of responses* is the number of responses from the service; *num of requests* is the number of requests to the service.

**1.2. Indicators by ISO standard**

According to the ISO / IEC 9126-93 standard, software quality (QoS) is the entire number of features and characteristics of software products, which refers to its ability to satisfy established or foreseeable needs. The standard defines six basic characteristics that describe software quality: functionality, reliability, usability, efficiency, maintenance and mobility. According to the standard, these characteristics can be applied to any type of software, including applications and data within the software and hardware, and form the basis for further refinement and describe software quality. The standard provides guidance on the application of model specifications and assessment process that reflects the basic steps needed for evaluating software quality. The standard does not define specific performance characteristics of quality (characteristics that determine the properties of software products that can be classified as quality characteristics), methods of measurement, ranking and evaluation, and results in only an illustrative qualitative model that determines the characteristics of quality in terms of the recommended comprehensive indicators (Fig. 1).

Apart from determining the terms and concepts of quality software above-mentioned standards allow the ability to change the definitions by adding their original attributes, revealing important terms used, indicating objects within the scope of the definition. In our case, these objects are Web services of enterprise information systems and weakly-bounded systems, that are a combination of these web services.

Regarding web services, features of maintenance and mobility are important for a service provider. For the initiator of the interaction of corporate information systems characteristics associated with the operation of web services are important. These characteristics include reliability and efficiency of their use in the weakly-bounded enterprise systems. Description of functionality can be used to classify web services by necessary function provided by them and determine functionally similar subsets of web services (e.g. supporting program of the SIP-components development and delivery). When using these characteristics to describe and evaluate web services it is needed to identify relevant indicators, set equal ranking and evaluation criteria for each company or the target web-service application. Metrics, ranking levels, and evaluation criteria should be agreed at the evaluation results exchange level [4, 6].

## 2. The concept of functional reserve of web services

Conceptually, web services can be divided into atomic services and composite services. Atomic ser-

vices are web services that on their own provide certain services to users. Atomic services are self-contained and do not depend on any other web services. Composite services provide services to users calling other web services (forming web service composition) and/or being web services of corporate information systems.

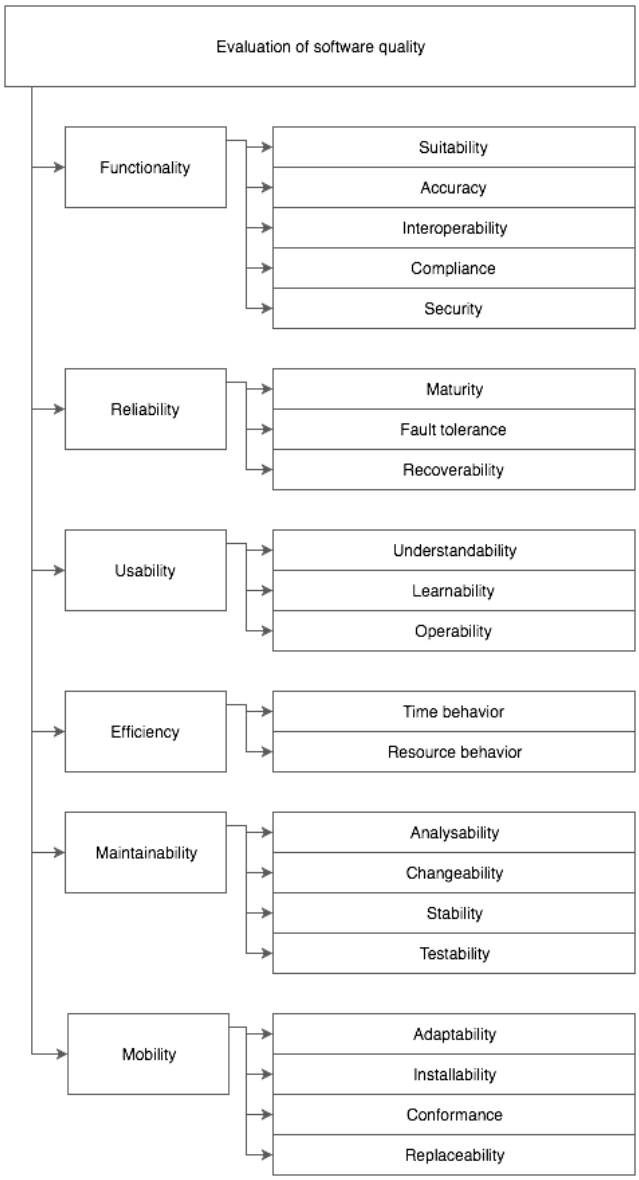


Fig. 1. The characteristics and complex indicators of quality

Functionality and interfaces defined by the language describing web services become more complex. To identify the web service with similar or identical functionality in scientific and technical literature there are proposed several methods of machine learning. However, the efficiency and accuracy of these methods are insufficient for their application in practice. Functionally equivalent Web services developed independently by different manufacturers may use different function names, input parameters and data types. With machine methods, it is really difficult to recognize that these services actually provide the same functionality. It is necessary to solve this problem primarily at the conceptual level of describing the common informational space of web services. For this, the conceptual model defined in a single namespace should be developed, which providers and developers of web service will adhere (Fig. 2).

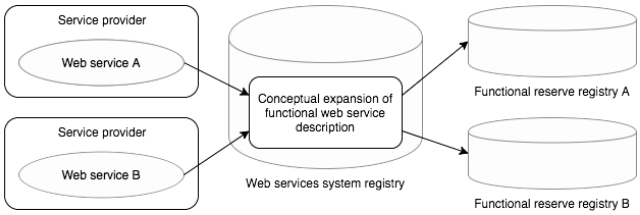


Fig. 2. The common informational space of web services

To solve the above-mentioned problems of search and recognition identical (similar) web services at the conceptual level, let's introduce the concept of functional reserve and define common terminology, which will follow when developing the conceptual model and description of web services. Then, web services developed by different manufacturers can be described in the same interface. Following common terminology when describing web services, it is possible to develop automated procedures for registration of web services in the relevant registers of the functional reserve of web services when publishing them in the system registry of web services. To maintain a registry of functionally similar web services it is necessary to expand the functional descriptions of web services [5].

Conceptually expansion of functional descriptions of Web services is a multilevel descriptions model, which establishes the correspondence between the request to perform the necessary functions on the side of corporate information system application and set of functionally similar web services, in other words, potential candidates to perform the requested function. Let's define three main conceptual levels: domain level, the level of a common information space of web services and the level of functional reserve.

Let's define the domain level as the set of relations R, presented in the namespace, understandable by users of corporate information systems and expressed by attributes of objects in a subject area. The form of the request of the domain level to perform the desired function P (A), where A is a set of function arguments, in terms of domain relations can be defined as follows:

$$P(A) = \bigcap\nolimits_i R_i(A_i), \qquad (8)$$

where $R_i$ is a domain level relation; $A_i$ is a set of attributes; $C_k$ is a condition like $C_k = a_i \theta c$ , where $a_i$ is an attribute of the relation $R_i$, c is a constant, and $\theta \in \{=, \neq, <, >, \leq, \geq\}$.

The level of a common informational space of web services is a description of all functional reserves $\forall F : F \subseteq U_w$, each of which is represented by a unique name of the desired function F in a single namespace of web services and can be defined as follows:

$$R_i(a_1, a_2, ..., a_n) = \bigcap_k C_k , \quad (9)$$

where $a_i$ is an attribute of relation $R_i$, $x_i$ are the formal parameters of correspondent function $\psi_j$;

It means that for getting the relations $R_i$ it is needed to call web services that perform related functions $\psi_j$. And the relation, which are in different domains can be defined in the same formal functions' namespace, whose composition can change if the definition of new relations for existing or new domain is necessary.

Let's call a set of functionally similar web services F, a functional reserve where $\forall F : F \subseteq U_w$, satisfying the formal description of the subject area.

$$F = (I_1, I_2, ..., I_n, \Lambda); I_i = (\Psi_i, \Upsilon_i, P_i), i = 1, ..., n , \quad (10)$$

where $I_i$ is an instance of service that is part of the functional reserve, which is represented as a directed acyclic graph; $\Psi_i = \{\psi_{ij} \mid 1 \le j \le m\}$ is a set of functions granted by the i-th instance of functionally similar web-service; $P_i$ is a necessary function that is granted by the i-th instance of the Web service (root element of functions' graph, in other words, is an entry point through which it is possible to access features of i-thinstance); $\Upsilon_i = \{\gamma_{ij} \mid 1 \le j \le l\}$ reflects the relationship between the functions within the i-th instance of the service within the same graph; $\Lambda = \{\lambda_{ij} \mid 1 \le i \le n \wedge 1 \le j \le n \wedge i \ne j\}$ shows the relationship between the two non-root functions provided by various instances of web services.

Dependencies between the two functions of the same service determine the order in which they should be called to perform the desired function by the same instance of service. Dependencies between features of different web services define the sequence of calling these functions when performing the required function by composite web service or by several web services belonging to different corporate information systems. Dependencies between graphs root elements or necessary functions determine the sequence of calling Web services as part of the composition, which is web services calling plan.

## 3. Conceptual model of software infrastructure

The conceptual model of web services integration's software infrastructure is illustratively shown in fig. 3, 4. It reflects the general principles of mediated interaction between providers and consumers of web services that formed the basis of program infrastructure of the adaptive fault-tolerant system of access organization to functionally-similar web services. Web services' providers register web services in functional registries of common informational space based on conceptual expansion of functional description of web services, supported by three-tired distributed model of web services description in user namespace of corporate systems, in other words, in the namespace of necessary functions, that is the formal namespace of common informational space in general, and namespace of specific instances of functionally-similar web services, in other words, in namespace of internal functions (operations) of web service, needed for executing of necessary function. The mechanisms of displaying namespaces are hidden from providers and consumers of web services and are the part of a functional model of program infrastructure of web services integration.

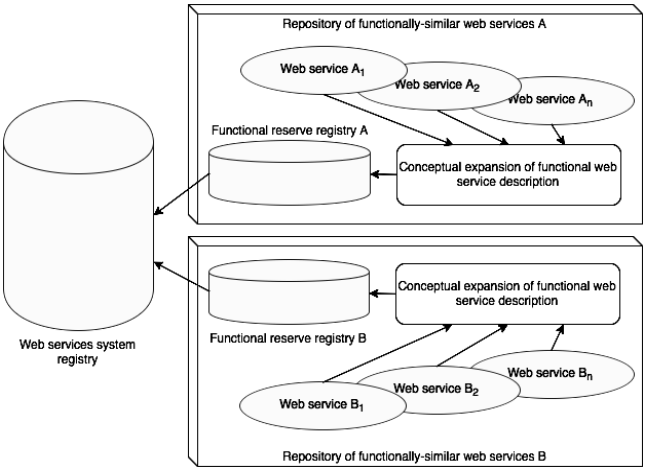


Fig. 3. The conceptual model of program infrastructure

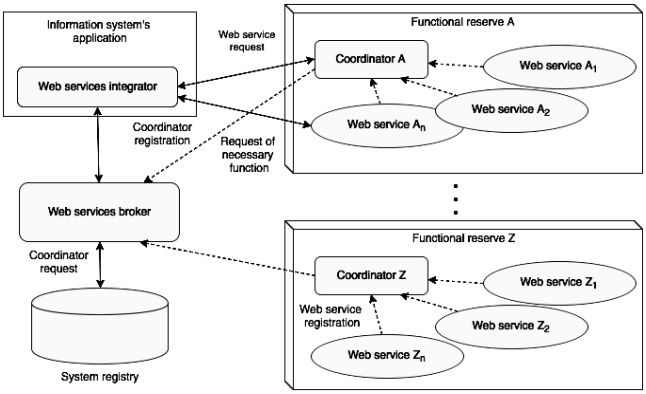


Fig. 4. Components of the infrastructure of web services integration

Apart from the register, each functional reserve is represented by a repository, providing space for the formation of the pool of web services instances by adaptive algorithms for fault-tolerant systems providing access to functionally similar web services.

The system of access organization for functionally-similar web services is defined over the components of software infrastructure of web services integration and conceptually is a failover strategy that is a combination of methods, algorithms, parameters and settings, required for adaptive and dynamic management of web services composition. The main components of software infrastructure of web services integration are web services integrator, web services broker, and web services, coordinator. Conceptually, web services integrator is represented by developed exemplary solutions of design: dynamic selector, quality monitor, and communicator, that allows performing web services integrator implementation both as a separate software component and as a part of middleware software or developed new applications of information systems.

The conceptual model of program infrastructure of web services integration reflects typical components (Fig. 4) and the order of their interaction without implementation details. During the operation, web services integrator receives from the web services broker address of functional reserve coordinator. It creates a pool of instances including a list of addresses and values of QoWS of functionally-similar web services based on which and considering user preferences the initial optimal of choice of fault-tolerant web services composition will be defined.

After fulfillment of the plan of web services call at the stage of selecting fault tolerant web services composition, integrator begins to locally accumulate new values of QoWS obtained while calling of each instance, returning them sometimes to coordinator and using them for dynamic correction of a strategy of choice fault-tolerant web services composition. Coordinator, in its turn, receives refreshed values of QoWS from integrators of geographically distributed users that call instances of this functional services reserve, thereby adding public data sets of QoWS for all users.

Services integrator consists of three components, functionally corresponding to the names used for the development of standard solutions design. Dynamic selector reconstructs the plan of web services calling chosen initially of fault-tolerant composition based on users requirements and information on the QoWS parameters. Quality monitor calculates values of instances parameters that are called and exchange them with a coordinator of a web service functional reserve, that in its turn provides the quality monitor with data sets of QoWS, obtained from other web services users. Communicator calls to web services instances according to current web services composition configuration.

## Conclusion

The paper describes the building of a conceptual model of web services integration to provide guaranteed quality of service. In order to understand how to make the control and estimation of services quality, the common web services quality indicators were considered in two independent factors: by the standards of global consortium OASIS and by ISO / IEC 9126-93 standard.

Taking into account above-mentioned standards the conceptual model of software infrastructure was built based on the concept of function web services reserve. The obtained results allow stating that by developing of Web services and adding them to the respective functional reserves registers, companies will expand their businesses, which in turn will attract more users and developers and improve service quality.

## References

*1. Channabasavaiah K., Holley K., Tuggle E. Migrating to a service-oriented architecture. IBM, 2004.*

*2. Linthicum D. Cloud Computing and SOA Convergence in Your Enterprise. Boston: Addison-Wesley, 2010.*

*3. GOST 27.002-89. Nadezhnost v tekhnike. Osnovnye ponyatiya. Terminy i opredeleniya [State standard 27.002-89 Reliability engineering. Basic concepts. Terms and definitions]. Moskow: Standart inform, 2010. 33 p.*

*4. Svystunov Y. D. (2016) Metody orhanizatsii vzaiemodii rozpodilenykh kompiuternykh system na osnovi servis-oriientovano I arkhitektury [Methods of interaction of distributed computer systems based on service-oriented architecture]. Information processing systems: scientific research journal, vol. 5, no. 142, pp. 142-147.*

*5. Elements of Service Oriented Analysis and Design: an ointer disciplinary modeling approach for SOA project. Available at: http://www128.ibm.com/developerworks/library/wssoad1. (accessed 28.11.2016).*

*6. Introduction to Service Oriented. Available at: http://searchdatamanagement.techtar-get.com/feature/Introduction-to-service-oriented-architecture. (accessed 24.11.2016)*

*7. OASIS Web Services Quality Model. Available at: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsqm. (accessed 26.11.2016).*

*Надійшла до редколегії 15.04.2017*

**Рецензент:** д-р техн. наук, проф. І.В. Рубан, Харківський національний університет радіоелектроніки, Харків.

### ВИКОРИСТАННЯ КОНЦЕПЦІЇ ФУНКЦІОНАЛЬНОГО РЕЗЕРВУВАННЯ ДЛЯ ЗАБЕЗПЕЧЕННЯ ЯКОСТІ ЗАБЕЗПЕЧЕННЯ ВЕБ-ПОСЛУГ

Ю.Д. Свистунов, Н.В. Лукова-Чуйко

*В даний час вже недостатньо того, щоб корпоративна інформаційна система забезпечувала простоту автоматизації інформаційних і обчислювальних бізнес-задач, корпоративна інформаційна система повинна змінюватися так само швидко, як швидко змінюються бізнес-вимоги і бізнес-процеси компанії. Це одна з основних передумов появи архітектури, орієнтованої на обслуговування, сервісів. У статті описуються основні параметри якості таких послуг, певні різними стандартами, і пропонується концепція функціонального резервування для забезпечення якості гарантованих послуг при створенні заявки.*

***Ключові слова:** веб-сервіс, SOA, якість, концептуальна модель, веб-сервіс.*

### ИСПОЛЬЗОВАНИЕ КОНЦЕПЦИИ ФУНКЦИОНАЛЬНОГО РЕЗЕРВИРОВАНИЯ ДЛЯ ОБЕСПЕЧЕНИЯ ОБЕСПЕЧЕНИЯ КАЧЕСТВА УСЛУГ WEB

Ю.Д. Свистунов, Н.В. Лукова-Чуйко

*В настоящее время уже недостаточно того, чтобы корпоративная информационная система обеспечивала простоту автоматизации информационных и вычислительных бизнес-задач, корпоративная информационная система должна меняться также быстро, как быстро меняются бизнес-требования и бизнес-процессы компании. Это одна из основных предпосылок появления архитектуры, ориентированной на обслуживание, сервисов. В статье описываются основные параметры качества таких услуг, определенные различными стандартами, и предлагается концепция функционального резервирования для обеспечения качества гарантированных услуг при создании заявки.*

***Ключевые слова**: веб-сервис, SOA, качество, концептуальная модель, веб-сервис.*