DOI: 10.15439/2014F47
ACSIS, Vol. 3

Service Design and Distributed System Reliability in Intelligence Information System Based on Service-Oriented Architecture

Jugoslav Achkoski
Military Academy “General Mihailo Apostlski” – Skopje, associate member of “Goce Delchev” University - Shtip str. Vasko Karangelevski NoN, 1000 Skopje, Macedonia
Email: jugoslav.ackoski@ugd.edu.mk

Vladimir Trajkovik
Ss. Cyril and Methodius University, Faculty of Computer Science and Engineering str. Ruger Boskovik 16, 1000 Skopje, Macedonia
Email: vladimir.trajkovik@finki.ukim.mk

***Abstract*—This paper presents the model of Intelligence Information System (IIS) based on a Service-Oriented Architecture. In this paper we propose the new service’s model, based on the Intelligence cycle and other systems which are necessary for gathering Intelligence information and data.**

The paper is mostly focused on the system architecture and services design as a mainstream for definition of the services. Furthermore, additional attention is dedicated on the Distributed System Reliability (DSR).

I. Introduction

INTELLIGENCE Information System Model gives contribution in Homeland Security and Civil Military Emerging Risks assessment through the possibility of providing information in an appropriate way, by implementing pushing and pulling mechanisms into information systems, selection of data and creation of information from raw data that can be used in creating intelligence products and dissemination reports for the authorities.

In the Intelligence Information System, which is based on SOA, are applications written in different programming language. The service design should provide interoperability between applications. It indicates that the services written in different programming languages are capable to communicate. In this connotation, processing elements (web servers, application’s servers, sensors for collecting data and so on) can create distributed environment for sharing information. SOA based multi-tier approach provides legacy systems to be hooked up in a new infrastructure where the new systems and legacy systems can communicate without complexity in communication protocol (SOAP messages).

In the phase of creating distributed systems, it is crucial to have metric for distributed system reliability. It provides appropriate distribution of the system’s components, because introduced algorithm for distributed system reliability shows where the gaps in the systems are. Also, the designers of a system can achieve higher level of system reliability using the distributed system reliability metric. In the distributed system, each node can present service and each service can present system, subsystem or processing element. Consequently, each service in the architecture has certain value of reliability. These values of reliability are very important for system designers.

The paper is organized as follows. Section 2 presents related work about the research presented in the paper. Section 3 is dedicated to architecture of the system. In the Section 3 are presented different levels of the system architecture and how they are connected. Section 4 presents service design where it is mostly focused on service interface. Section 5 demonstrates the algorithm for computing distributed system reliability and we implement GEAR as an algorithm for the system reliability. Finally, in the Section 6, concluding remarks of the paper are presented.

II. Related work

In [3], the quality attributes of loose coupling and autonomy for services in the context of service-oriented architecture are given. In order for services to be influenced by these quality attributes, an evaluation should be done during the phase of development of service design. According to [3], the recent research is focused on the textual description of the desired quality attributes and the thereby resulting formalized metrics require more information than the already available, or are based on theoretical models that hamper their applicability. In this paper, we present quality indicators for unique categorization, loose coupling, discoverability and autonomy. Formalized metrics is created for each quality indicator, in order to measure service candidates and service design in the Service oriented architecture Modeling Language (SoaML) [4], the standardized language for modeling service – oriented architecture. To illustrate the metrics and to verify their validity, service candidates and service designs of a campus guide system as developed at the Karlsruhe Institute of Technology, are evaluated.

In [6], a study of service reliability and availability for distributed systems is presented. The study gives an application example in order to explain usefulness of the GEAR algorithm. Furthermore, in the paper is presented research about reliability of modeled centralized heterogeneous distributed system (CHDS). Also, in the paper is studied implementation of availability function of virtual machine.

III. Architecture of Intelligence Information System

General architecture of Intelligence Information System prototype is presented on Figure 1. As a result of system complexity, the solution is presented as a layer model of architecture.

On the lowest level, IIS prototype has distributed system which consists of heterogeneous databases. In this case, most important database for IIS is database which holds data for users who use it. Intelligence center has responsibility for this database.

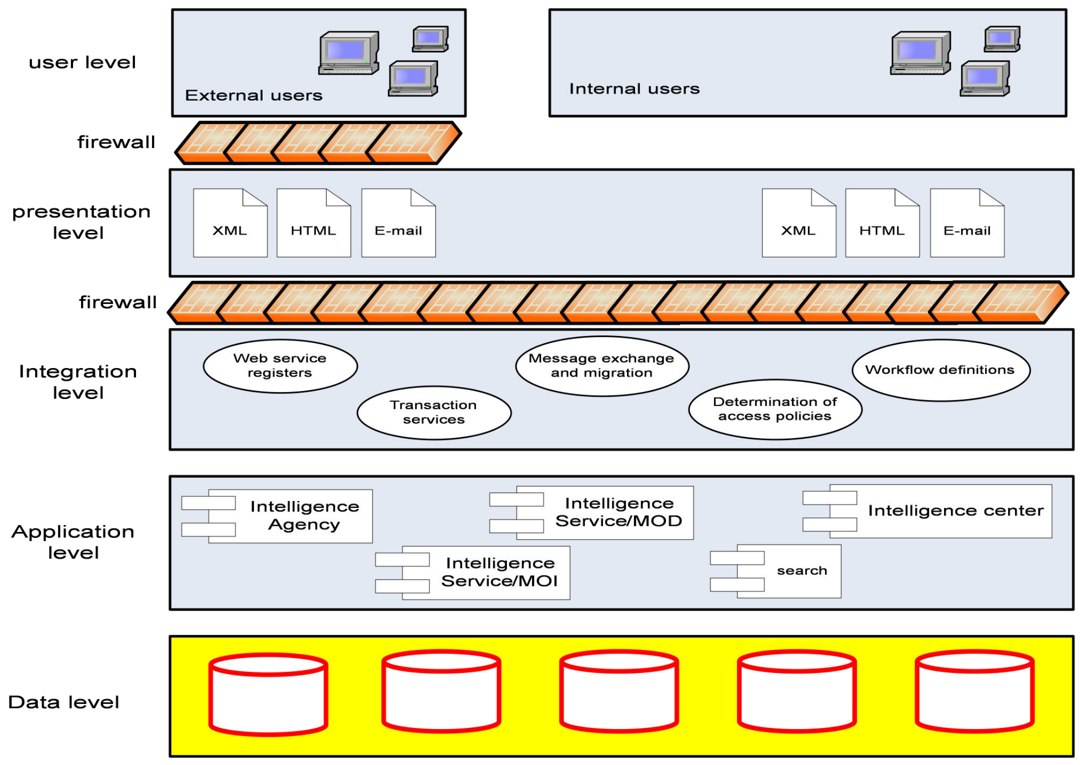

Fig 1. Prototype of Intelligence Information System Architecture

Access to a separate database will be made with application logic of module, which is part of internal information systems on government institution. This application should provide interfaces to the integration logic level [8], [9].

Integration level is a key level for our IIS model. That level should provide services through workflow which will be connected with modules of internal information systems and their transformation into web services. As a result of provided web services, integration level should exposed them into appropriate web services registers depending of security level. This level also govern security polices and polices for exchanging and adopting messages from different sources, in case of usage in comparable format. Finally, this level is taking care for governance of the services offered by IIS in a way of transactions when it is needed. With one sentences, this level is providing the functionality of the services in IIS.

The services should be available for different categories of users. For the purposes of protecting Intelligence Information System, firewall should be installed behind this level, which is followed by the level of presentational logic.

The presentation level can be implemented in a form of portal, which can offer: list of web services over approach to service registries, integration of web services with e-mails or directly as far procedure call of the applications (RPC) in a standard format (XML), but also as a ordinary HTML text for separated union of services – users. Exchanging information with external information systems is achievable through communication network, where IIS model is protected with another additional firewall. In this way, maximum protection from unexpected system failures is accomplished [8], [9].

IV.Service Design

In the process of service design it is possible services to be implemented in an existed platform or in the new platform, which will be created as a state-of-the art solution with straightforward purpose. This distinction is important, because our model allows web service to be implemented in both of these cases. Exploiting the services in such a way allows easy building of novel modules within information systems architecture and additionally, allows taking a pace with a contemporary ICT technology.

We propose coupling as a way for measuring service design. In the Intelligence Information System achieved desired level of coupling allows integration of subsystems and sensors as service providers with minimum number of connections between services. For example, if new sensor is added to the system, the communication that should be established between sensor and application server or other processing elements in the system does not imply that every server should establish communication with the new sensor [11].

In terms of service granularity (scope of functionality exposed by the services), it is the most convenient to create coarse-grained interfaces that implement a complete business process.

The coarse-grained interface should provide access to the data from different software artifacts and processing elements in the system depending of the user requirements [7]. It indicates that in the IIS sensors and other hardware components, which should be connected, have to exchange information in order to provide information for the senior decision makers or other end users. These components are based on different programming language (C++, JAVA, C and so on) and if the service's interface is not implemented in the applications, they could not exchange their data types. For instance, if application is written in JAVA and interface for this application is cre-

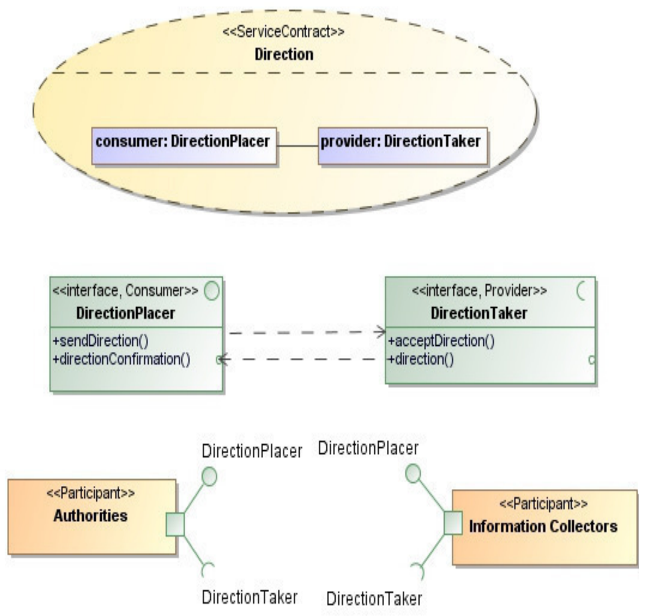

Fig 2. Specification of the Direction service, consisting of two roles, their respective consumer and provider interface type, and the corresponding ports on the participants

ated in JAVA, then application written in C++ could not use this JAVA interface, because data types (string, integer, float and so on) in JAVA and C++ are not treated in the same way. In our approach, the service's interface is based on XML, because applications written in different languages can exchange their data using WSDL (Web Service Definition Language). WSDL provides interoperability between applications.

A.Service contract based approach

The Service oriented architecture Modeling Language (SoaML) specification defines UML profile and metamodel for designing services within service-oriented architecture. Goals of SoaML refer to support activities at the stage of modeling and designing services and invoke them in model-driven development approach (MDA). It should support SOA in business and IT perspectives [1], [2].

SoaML specification defines three different types of approaches for specifying services:

- The simple interface based approach uses an UML interface to specify a one-way service interaction [1] [2].
- The service contract based approach extends an UML collaboration to specify a binary or n-ary service interaction [1], [2].
- The service interface based approach extends a UML class to specify a binary or n-ary service interaction [1], [2].

Different SoaML approaches recommend usage of divided UML parts which mean that reading SoaML specification is not understandable. Because of the reasons previously mentioned, problems in designing information systems emerge in software engineering [1].

A service contract based approach defines service specifications that define functions of service stakeholders (consumer and provider) and interface that implements these functions. In order to fulfill services' tasks interfaces must implemented services' function. Interfaces are types of ports in service-oriented architecture that requires each stakeholder to accomplish its task in the appropriate service contract [1].

The service contract based approach increases the UML collaboration in the model that present structured part of services' interactions. It can be used for specifying services that include contractual obligation, i.e. an agreement between two or more parties, which is relevant for circumstances of already established interaction patterns between the participants. These interaction patterns are used for exchanging messages and specifying interfaces between participants [1].

In order to demonstrate service contract based approach we are using services that make part of Intelli-

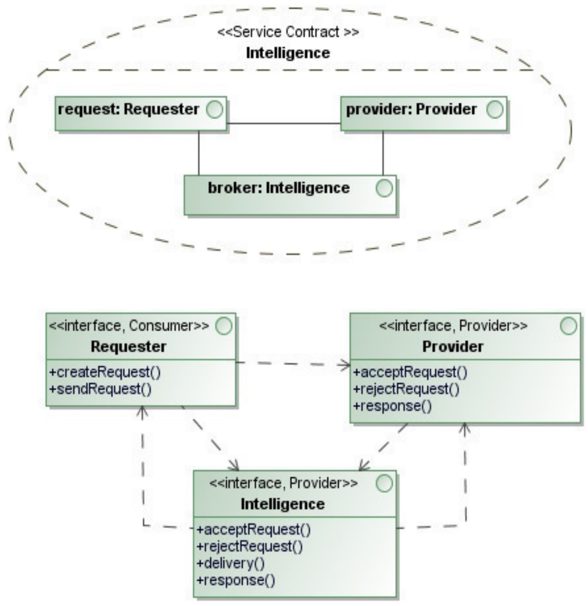

Fig 3. Specification of the Intelligence service contract, consisting of three roles, the respective consumer interface and the two provider interface types

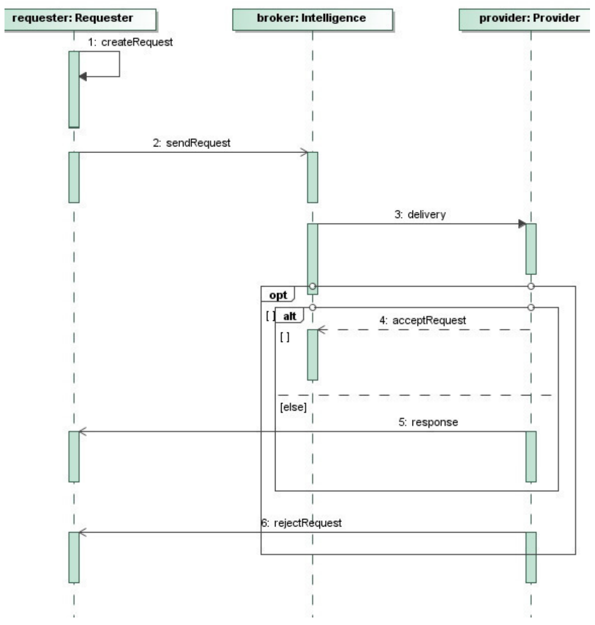

Fig 4. Specification of the Intelligence service choreography

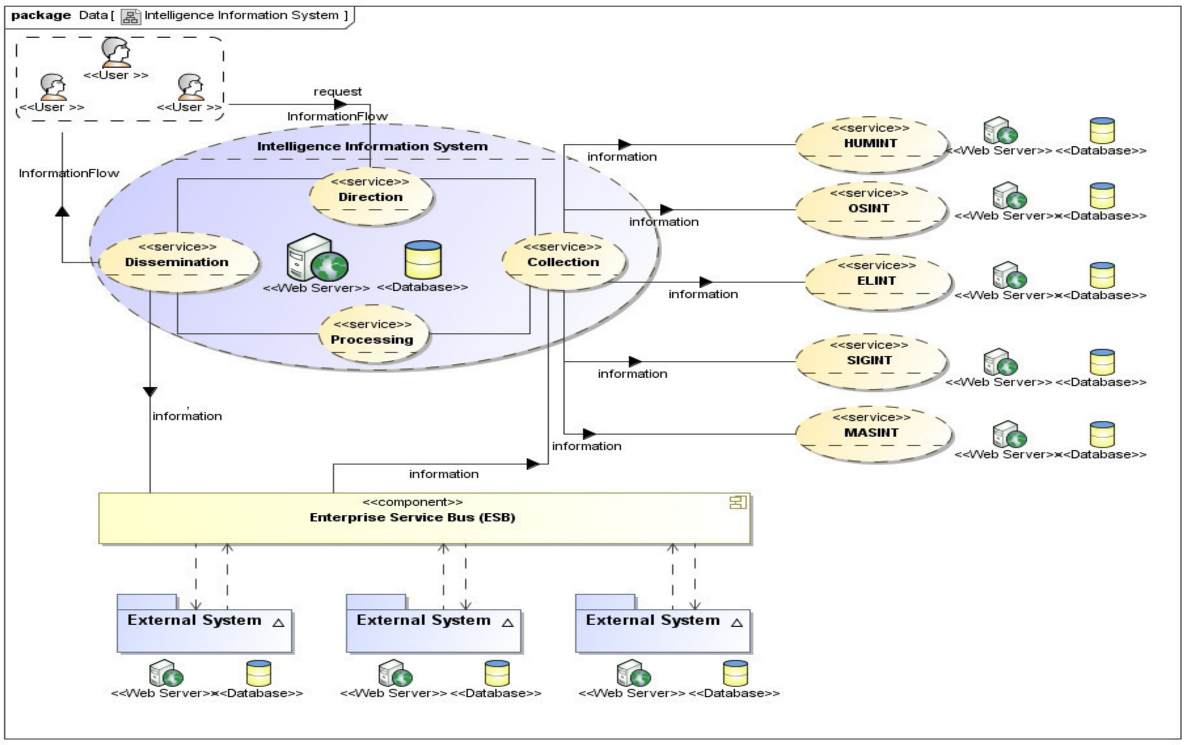

Fig 5. The dataflow in the Intelligence Information System - based on SOA

gence Information System. Services should contribute in defining a binary service contract, a multi-party service contract and a compound service contract that contributes to explaining service contract based approach. First, we suppose that Direction service contract can be modeled as two independent service contracts. One of them should specify an interaction for placing directions and another one should specify an interaction for taking directions in the process of intelligence information collection. Figure 2 shows specification of Direction service contract, consisting of two roles, namely their respective consumer and provider interface: DirectionPlacer and DirectionTaker [1].

The service contract shows that there is dependency between these two types of interfaces and they have to be modeled with UML dependencies. Participants use interaction in service contract and fulfill their tasks through appropriate interface. The binding between these interfaces is established by ports. From the role bindings in the services architecture we deduce that the Authorities have a request port typed by the DirectionPlacer interface, and the Information Collector has a service port typed by the DirectionTaker interface.

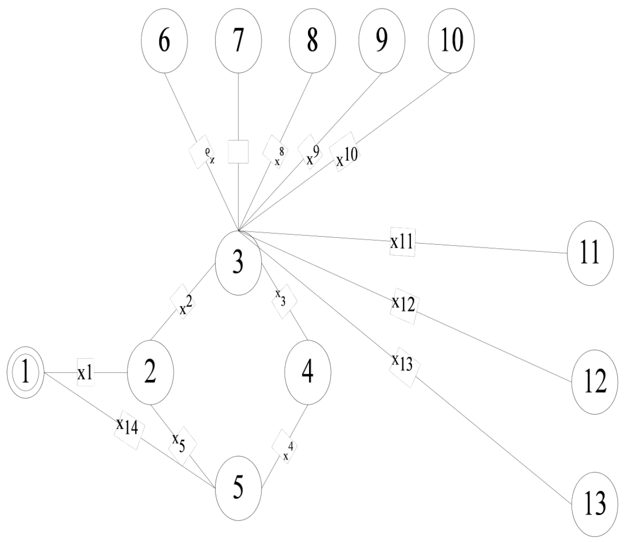

Fig 6. The graph of the Intelligence Information System - based on SOA

In this example service contract presents packing of two interfaces, providing that two interfaces are part of one service specification and not specified as a two independent service specification as two separate interface. Furthermore, it is recommended that a behavior on service contract is specified, i.e. a service choreography or a service protocol. Actually, there is a disagreement on whether a specification of service choreography should be used for understanding design of service interface in order to support exchanging message. SoaML is agnostic with regards to behavioral modeling and basically states that any UML behavior, e.g. interaction models, activity models or state machines, can be used [1].

The service contract based approach is convenient for specifying interaction between two or more roles that are introduced for establishing an agreement such as, for example, a message exchange. Service contract can be also applied as a reusable specification element, which can be re-used during the design time for connecting different stakeholders. In addition, this approach supports modeling of multiparty service contracts including three or more participants, as well as modeling of compound service contracts where the existing service contract can be used for defining several granular service contracts [1].

Figure 5 can be used to elaborate multiparty service contract. In our service model, we use Intelligence service contract where the interaction between the requester and

Table I.
Relations b/w an edge, a service name and vertices

A Node	A service name	A link	node vectors
1	end - user	x_1	1-2
2	Direction	x_2	2-3
3	Collection	x_3	3-4
4	Analyzing	x_4	4-5
5	Dissemination	x_5	5-2
6	HUMINT	x_6	3-6
7	OSINT	x_7	3-7
8	ELINT	x_8	3-8
9	SIGINT	x_9	3-9
10	MASINT	x_{10}	3-10
11	External System	x_{11}	3-11
12	External System	x_{12}	3-12
13	External System	x_{13}	3-13
14	Dissemination	x_{14}	5-1

the provider of information is mediated by Intelligence broker.

Figure 3 shows the specification of Intelligence service contract with three roles: information requester, information provider and intelligence information broker. These three roles have independent types of consumer and provider interfaces called Requester, Provider and Intelligence. The dependencies between the interfaces are explicitly modeled using UML dependencies. Stakeholders also have ports whose function is to connect services in service-oriented architecture.

Figure 4 shows the specification of the service choreography using UML interaction. Here we should notice that this is multiparty service contract, since the requester interacts directly with the information provider through delivering of messages. Except for the direct message delivering interaction, all other interactions pass through Intelligence broker. Service interaction starts with requirements for intelligence information toward the intelligence broker. At a later time a delivery is made which is either accepted or a grievance is sent to the broker and forwarded to the provider, who may file a justification in order to clarify whether to accept or ignore the requirements.

V. The Distributed System Reliability of the Intelligence Information System

The purpose of the GEAR algorithm is to compute accuracy of distributed computer system, which actually is composed of memory units, processing elements, and other hardware and software. Probability of application or service to be accurately executed in a distributed system is called availability of the distributed system.

In one distributed system, the nodes can present memory units, processing elements and programs (see Figure 5). The nodes can exchange data through a communication network in order to execute a program from separate nodes. Failures of the communication links or failures of the services in the distributed computer system, decrease the level of system performance and availability of the system. The level of success of one program in one node in distributed computer system depends on availability (successful program execution) of all other nodes, which indicates that the node have to be accessible from all other programs required for the appropriate program execution and also, communication links should be available without failures.

To compute DSR for IIS, we select GEAR, which is dedicated for computing reliability of links in computer networks. We introduce following assumptions:

- The RV maintains the information about links in the computer network [6];
- If link is operational, it has value 1 and if not, it has value 0 (faulty) in reliability expression;
- The “d” represents the link when we do not know whether link is operational or faulty and it is not computed in reliability expression;
- The LV has information about the edges that are traversed in the subnetwork [6];
- Each service presents self-contained processing element, and we can assume that they are self-contained node;
- The structure of the IIS is modeled as a graph and the graph does not have any loops [6], [9];

On Figure 6 is presented graph topology of distributed system, which is based on schema of Figure 5.

According to above mentioned assumptions, the dataflow in Figure 5 and the graph in Figure 6, in order to compute DSR of Intelligence Information System, we have to combine these pieces as in Table I. To be more precise, we can use the table to update RV and LV, because it gives a clear explanation how services, links and nodes are connected b/w each other.

In order to be computed reliability of computer distributed system, the GEAR algorithm requires both vectors RV and LV to be updated in the every iteration at each node. In order to be computed these two vectors in the GEAR algorithm are implemented simple rules without complexity.

To be updated Reliability Vector, we have to follow next two rules [5], [6]:

1. The updated RV value about new edge is obtained from the value of parent node and the value of vertices where the link is traversed from its parent edge.

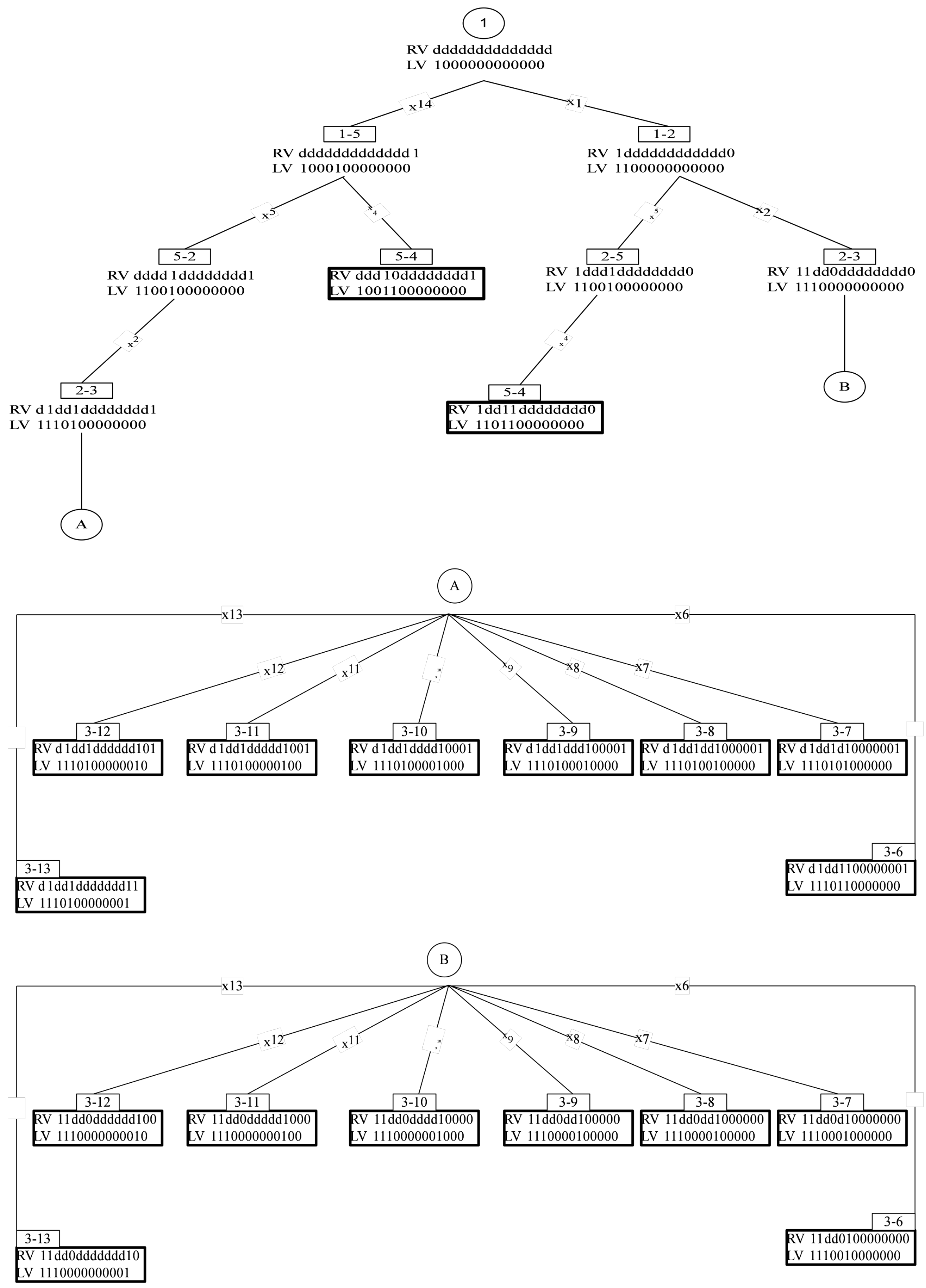

Fig 7. A complete tree for evaluating DSR in IIS

2. The edges, which are on the left side of the first updated edge (previous statement), are updated with the value of the edge, which is on their left side, and it is value 0 about the vertices and value 1 about vertices, which is traversed from its parent edge.

The intention of the LV is to avoid loops in the algorithm, which implies that one node is not traversed more than once [5], [6]. The updating of this vector is simple, which means that every node has value from the parent node in the tree and its value is represented with 1. Other nodes have value 0 and these nodes are not connected to the parent nodes and the vertices of the updated node.

In order to show, where in the tree the GEAR algorithm stop, we set up bold rectangles (Figure 7). The tree shows that the vertices in the graph from the Figure 5 have ending edges with the following numbers: 4, 6, 7, 8, 9, 10, 11, 12, 13. The starting edge is number 1.

In the each ending edge (bold rectangle), we can note the symbols as 1, 0 or d. In the ending edge where the value is 1, we replaced p and everywhere in the edge where the value is 0, we replaced q. The symbol d is not concerned with the computation expression because it is not involved in the reliability expression. It allows the expression (1) to be created for DSR.

$$\begin{aligned}\mathbf{DSR} = {} & p_4q_5p_{14} + p_1p_4p_5q_{14} + p_1p_2q_5p_{13}q_{14} + \\ & p_1p_2q_5p_{12}q_{13}q_{14} + p_1p_2q_5p_{11}q_{12}q_{13}q_{14} + \\ & p_1p_2q_5p_{10}q_{11}q_{12}q_{13}q_{14} + p_1p_2q_5p_9q_{10}q_{11}q_{12}q_{13}q_{14} + \\ & p_1p_2q_5p_8q_9q_{10}q_{11}q_{12}q_{13}q_{14} + \\ & p_1p_2q_5p_7q_8q_9q_{10}q_{11}q_{12}q_{13}q_{14} + \\ & p_1p_2q_5p_6q_7q_8q_9q_{10}q_{11}q_{12}q_{13}q_{14} + p_2p_5p_{13}p_{14} + \\ & p_2p_5p_{12}q_{13}p_{14} + p_2p_5p_{11}q_{12}q_{13}p_{14} + \\ & p_2p_5p_{10}q_{11}q_{12}q_{13}p_{14} + p_2p_5p_9q_{10}q_{11}q_{12}q_{13}p_{14} + \\ & p_2p_5p_8q_9q_{10}q_{11}q_{12}q_{13}p_{14} + \\ & p_2p_5p_7q_8q_9q_{10}q_{11}q_{12}q_{13}p_{14} + \\ & p_2p_5p_6q_7q_8q_9q_{10}q_{11}q_{12}q_{13}p_{14}\end{aligned} \tag{1}$$

In Figure 7, DSR is calculated with the computation of p, and q. These two coefficients formulate the probability of every computer network link to be available for transferring data between services with probability p=0.9 (q=0.1) [6]. According to the expression (1) and replacement of the values p and q, we can obtain the result for DSR of Intelligence Information System.

$$DSR=0.8898$$

In conclusion of this section, we have to stress that obtained result about DSR is not on an appropriate level for the above mentioned system. Our intention is to create the system based on Service Oriented Architecture where we can obtain results of 0.977 out of 100% [10].

Furthermore, it is possible to increase the level of distributed system reliability, but we have to pre-plan the graph of computer network infrastructure with different ways of connections between edges and vertices. In addition, we have to pre-plan the ending edges where algorithm stops.

VI. CONCLUSION

The implementation of service-oriented architecture in Intelligence Information System increases the intelligence efficiency. Establishing a developmental methodology and designing the model can serve as a basis for building efficient information system.

The System architecture is explained in order to show the general concept of the system in terms of connectivity between system components, and relationship between layers. About the system architecture, we can firmly conclude that level of integration logic is a basis of the Intelligence Information System. Although, in the system architecture is diverse levels, only the level of integration logic is most significant for Service-Oriented Architecture. At the level of integration logic are set up most important services for appropriate system functioning.

The service design will contribute to the building Intelligence Information System based on an SOA platform because software artifacts can achieve a certain level of interoperability. Therefore, diverse hardware and software can exchange their data types in order to satisfy Intelligence functions. As a conclusion about service design, we can stress that core of interoperability relies on XML because WSDL and SOAP are based on XML.

The DSR provides the system's metric for reliability where many information systems rely on this metric. This metric provides reliable values for connecting nodes (processing elements, applications, I/O devices etc.) in distributed systems and it can be exploited in the early stage of information system development. Furthermore, DSR could be used for gathering testing data on further stage of system development. The obtained results from the tests will be taken from the equations for general metric about quality of service (QoS).

REFERENCES

[1] B. Elvesæter, A.-J. Berre, A. Sadovykh, "Specifying Services using the Service oriented architecture Modeling Language (SoaML): A baseline for Specification of Cloud-based Services," in Proc. *1st International Conference on Cloud Computing and Service Science (CLOSER 2011)*, 7-9 May 2011. http://closer.scitevents.org/

[2] M. Gebhart, M. Baumgartner, S. Oehlert, M. Blersch, and S. Abeck, "Evaluation of Service Designs based on SoaML," in Proc. 5th *International Conference on Software Engineering Advances (ICSEA)* pp. 7-13, 2010, doi: 10.1109/ICSEA.2010.8

[3] M. Gebhart, S. Abeck, "Metrics for Evaluating Service Designs based on SoaML," *International Journal on Advances in Software*, vol. 4(1&2), 2011, pp. 61-75. http://iariajournals.org/software/

[4] OMG, "Service oriented architecture modeling language (SoaML) – specification for the UML profile and metamodel for services (UPMS)," Version 1.0 Beta 1, 2009

http://www.uio.no/studier/emner/matnat/ifi/INF5120/v10/undervisningsmateriale/09-12-09-SoaML.pdf

[5] A. Kumar, D.P. Agrawal, "A generalized algorithm for evaluating distributed-program reliability," *IEEE Trans. Reliability*, vol.42, Issue 3, pp. 416 – 426, Sep. 1993. doi: 10.1109/24.257825.

[6] Y.S Dai, M. Xie, K.L. Poh, G.Q. Liu "A study of service reliability and availability for distributed systems," *Elsevier, Reliability Engineering & System Safety*, vol. 79, Issue 1, 1 January 2003, pp. 103–112, http://dx.doi.org/10.1016/S0951-8320(02)00200-4

[7] M. P. Papazoglou, W.-J. van den Heuvel "Service-Oriented Design and Development Methodology," *International Journal of Web Engineering and Technology (IJWET)*, vol. 2 Issue 4, July 2006, pp.412-442.

[8] J. Achkoski, V. Trajkovik, and D. Davcev, "Service-Oriented Architecture Concept for Intelligence Information System Development," in Proc. *3rd international conferences on advanced service computing service computation 2011 (IARIA)*, Rome, Italy, September 25 - 30, 2011.

[9] J. Achkoski, V. Trajkovik,"Intelligence Information System (IIS) with SOA-based Information Systems," in Proc. *33rd International Conference on INFORMATION TECHNOLOGY INTERFACES, IEEE*, Cavtat/Dubrovnik, Croatia, June 27 - 30, 2011.

[10] J. Hurwitz, R. Bloor, C. Baroudi, and M. Kaufman, "Service Oriented Architecture (SOA) For Dummies," Hoboken, NJ 07030-5774: John Wiley & Sons. 2007.

[11] Priyantha, Nissanka B., et al. "Tiny web services: design and implementation of interoperable and evolvable sensor networks." Proceedings of the 6th ACM conference on Embedded network sensor systems. ACM, 2008.