

Methods of Creating Service-oriented Software Systems

Andrii Kozyriev ¹, Svitlana Litvin ¹

¹ Kharkiv National University of Radioelectronics, Nauky ave. 14, Kharkiv, 61166, Ukraine

Abstract

The object of research is service-oriented architectures and ontological models.

The aim of the work is to create a formalism for describing the interaction of services and algorithms for building web service interfaces for the implementation of an effective SOA-system using a paradigm based on ontologies of data access.

The result is algorithms for automated construction of web service interfaces for SOA-architecture, which allow to automatically build web-service interfaces.

Keywords

Ontologies, service-oriented architectures, intellectual analysis, semantic analysis, data access.

1. Introduction

The growing popularity of technologies such as cloud computing, distributed systems, and Semantic Web in recent years on the Internet shows the relevance of the task of organizing interaction of Internet resources among themselves. Service-oriented architecture (SOA) provides such interaction of software development.

At present there are many systems for creation of web applications with SOA architecture: IBM, Microsoft .NET, Oracle SOA Suite. The necessity of such systems is caused by problems of EAI (Enterprise Application Integration) [1].

EAI is an integration architecture that consists of a set of technologies and methods for integrating systems and applications across corporate architectures to simplify and automate business processes.

To implement the SOA system, it is necessary to create formalism for describing system processes by ontology and effective algorithms or to improve existing OBDA (Ontology Based Data Access) algorithms.

To achieve the goal of work, the following tasks must be solved:

1. Carry out analysis of basic types of cooperation structures of SOA-services;
2. Develop algorithms of description and analysis of conceptual model of SOA-system, oriented on ontological approach, which allows to adequately describe interaction of web-services;
3. Design algorithms of automated construction of interfaces of web-services for SOA-architecture, which allow to build interfaces of web-services based on ontology;
4. Develop the software implementation of SOA-system, which uses received algorithms of automated construction of interfaces of interaction of web-services.

COLINS-2022: 6th International Conference on Computational Linguistics and Intelligent Systems, May 12–13, 2022, Gliwice, Poland.

EMAIL: andrii.kozyriev@nure.ua (A. Kozyriev); svitlana.litvin@nure.ua (S. Litvin).

ORCID:0000-0001-6383-5222 (A. Kozyriev); 0000-0002-7183-6345 (S. Litvin).



© 2022 Copyright for this paper by its authors.

Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

2. Analysis of the problem resolution and substantiation

2.1. General concepts of SOA-systems

With the growing popularity of technologies such as cloud computing, distributed systems, and Semantic Web on the Internet, the task of organizing interaction of Internet resources among each other is urgent. To provide such interaction allows service-oriented architecture (SOA) of software development [3].

The central concept of SOA architecture is service [3]. The name *service* is defined in the dictionaries as *performance of works (functions) for each other*.

The services are the central concept of SOA architecture because they implement the concept of scale because of transparency and easy integration. Transparency is expressed through a description of the service, which contains information necessary for interaction with the service, and describes it in such terms as incoming data of services, source data of services and related information.

The following are three main structures that characterize SOA systems in terms of internal composition and interaction with computing environment [3]:

- Composite service – a parent service with a complex structure that covers one or more simple subsidiaries, which are approved by the outside world – this formation can be presented in the form of a hierarchical structure;
- Service cluster is a group of services distributed on the purpose and technology-related services, which interact to solve the task.

SOA-architecture provides creation of application software systems, built by means of distributed interaction services [4].

2.2. Web services as modern tools of global networks

When developing a system for Internet the main object of the SOA concept is a web-service – a web-address program system that can be identified by a web-address, which is equipped with standardized interfaces for interaction on standardized protocols [5].

There are many implementations of SOA-architecture: Rest, RPC, DCOM, CORBA, SOAP. Currently, most software solutions developed in accordance with service-oriented architecture often use a complex of web services interacting with the protocol SOAP.

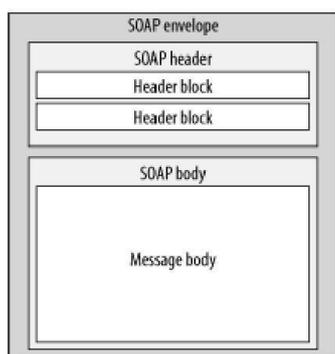


Figure 1: The SOAP protocol structure

SOAP (Simple Object Access Protocol) is a protocol for exchanging structured information in a distribution computing environment for the implementation of web services in computer networks. The SOAP protocol is used to exchange XML messages and to call remote procedures (RPC). As a result, the XML-RPC protocol is extended. SOAP interacts with all application layer protocols and is most used over HTTP [5]. The SOAP [6] protocol structure is shown in Fig. 1.

WSDL (Web Services Description Language) – this XML language specification is intended to create documents that describe web services in detail and how they are accessed. Each WSDL document contains the following basic information about web services:

- Types, sent and received by the XML message service;
- Messages sent by the web service;
- Binding – the way the message will be delivered.

As shortcomings it is worth noting that application of SOAP increases volume of messages and processing time, necessary time to build web-service based on SOAP or other tools [7].

2.3. Analysis of semantic web technologies

Semantic Web (semantics) is a promising industry in the field of Internet development [8]. The main task of Semantic Web is to present information in a form suitable for machine processing. The main tools of semantics web are semantics networks, universal resource identifiers (URIs) and ontology [9].

The following is a detailed summary of the major Semantic Web tools [10]. The Semantic Web structure consists of many language descriptions approved by the W3C Consortium. This set, among other things, contains URI, XML, RDF, RDF-S, OWL [11] (see Fig. 2).

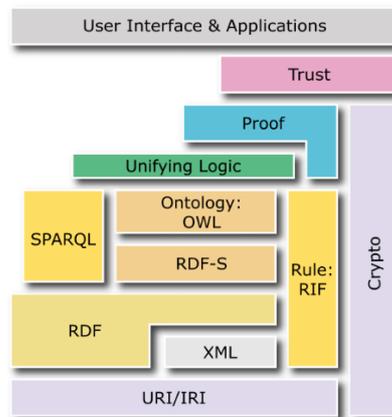


Figure 2: Understand Semantic Web [11]

URI is a universal resource identifier, usually used to reference an object using HTTP on the Internet. Because the URI is globally unique, they allow you to identify the same objects in different locations in the Semantic Web environment [10].

XML is an extended markup language that does not carry semantics. The structure of the XML document is defined by the XML Schema specification [12].

RDF is a standard that describes the graphs in which arcs and nodes contain URI. The format of the RDF-data description is a "subject-relation-object" triplet.

The W3C Consortium created an XML schema for the RDF document description. The following is an example of an RDF-document that describes part of the business processes of the Internet-shop [10] (see Fig. 3).

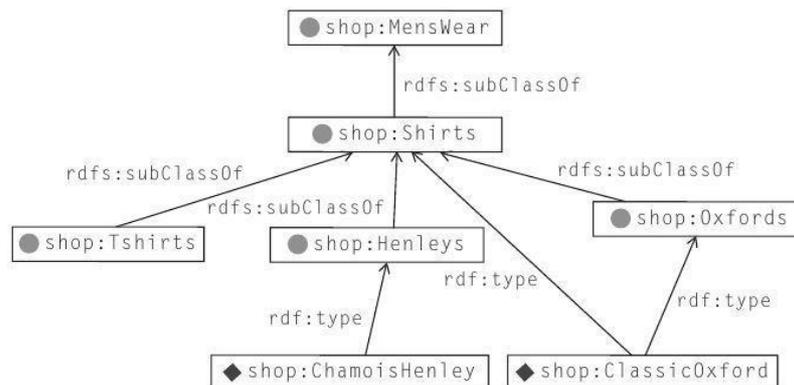


Figure 3: example of RDF-document graph

RDF graphs can be used for ontology [13], described by RDF-S and OWL.

OWL (Web Ontology Language) is an ontology description language for Semantic Web technology. OWL allows you to describe web documents and reality objects in the format of descriptive logics. A URI is associated with each OWL ontology object.

A new version of the OWL 2 web ontology language has been developed [27]. OWL 2 has a similar structure and is compatible with OWL 1. These factors make it possible to integrate OWL 1 ontologies into OWL 2 ontologies, but not vice versa. At the same time, OWL 2 adds new functionality and several new profiles. OWL 2 extends expressive power to the SROIQ descriptive logic format. The emergence of this factor has led to the weakening of restrictions on OWL DL. The following are the new features of OWL 2 compared to OWL 1 [28]:

- Advanced set of data types and ranges;
- Keys;
- More elaborate descriptions of annotations;
- Chain of properties;
- Outstanding properties;
- Asymmetric properties;
- Reflexive properties.

2.4. Knowledge bases and descriptive logic for description of conceptual models

Knowledge Base (KB) is a description that covers a specific subject area. Knowledge bases are most often used for building intellectual systems [14].

Intellectual system is a technical or software system that can solve tasks traditionally considered creative, belonging to a specific subject area, knowledge about which is stored in memory of such system [14].

Ontology is the hierarchical way of representation in the knowledge base of the set of understanding and their relations [16].

Concepts or classes are abstract sets of objects. They contain copies or subclasses [17].

Indices or instances are specific material or ideal objects.

Classes and indices in ontology may contain attributes that are used to store information.

Attributes define the relationship between ontology objects. In most cases, the ratio becomes the attribute of the object with the value as another object.

Ontology is divided into general and applied. General or basic ontologies contain universal concepts for most subject areas. This type of ontology means the presence of a basic set of terms. There is a practical implementation of technologies of unification of ontology, where general ontology is used [18].

Description logics (DL) [18] are used as a description of ontology. Descriptive logic was created to expand semantics and frame structures by mechanisms of formal logic.

In terms of DL knowledge of KB is divided into:

- TBox is a set of terms of a general kind called a term that can describe a subject area or conceptual data model, such as an entity-relationship (ER) model [19]. ER-data model is used for designing relational databases [19];
- ABox is a set of statements about the indicators of private kind.

Completeness is a property of a finite model property [20]. The presence of the property of completeness in descriptive logic means that for this logic it is easy to perform logical analysis based on solved procedures. Computational complexity of logical analysis of descriptive logics is a characteristic that determines the dependence of the required amount of memory and time to perform the algorithm on the size of the input data.

Computational complexity is one of the main problems in the development of descriptive logics. The study of algorithm optimization begins with determining the class of computational complexity of the problem that the algorithm must solve [27]. Understanding the principles of complexity

reduction in descriptive logics allows not only to choose the appropriate logic to describe ontologies, but also to use this knowledge in designing a knowledge base [21].

The task of building an SOA-system should be solved by applying the paradigm based on ontologies of data access (OBDA – Ontology-Based Data Access). OBDA is access to data through a high-level ontology interface [7].

The implementation of the OBDA system should allow obtaining data from ABox stored in databases. The efficiency of data access is achieved by displaying requests in a relational database environment, which has effective means for processing requests and other benefits. TBox should be used to provide the necessary semantic information at a higher conceptual level. The proposed scheme allows reasoning on TBox and ABox independently of each other [24].

3. Description of theoretical research carried out

3.1. Description of SOA models

As the basic theoretical concept, service-oriented architecture (SOA) is used, which is a paradigm of software development by application of distributed weak-connected components [1]. When developing a system for Internet the main object of the SOA concept is a web-service – a software system that can be identified by a web address, equipped with standardized interfaces for interaction on standardized protocols.

To build an interface ISOA between two web services you need to have the following information in the form of three values:

$$I_{SOA} = \langle C_s, Q_s, S_s \rangle$$

where C_s – parameter containing the address of the web client requesting the information (client address), Q_s – many requested information resources (queried resources), S_s – a parameter that contains the address of the web service responding with the information (server address).

3.2. Analysis of descriptive logic methods

A set of characters (alphabet) is specified to formulate the syntax of any DL language. This set creates expressions within a specific language. In addition, each language of descriptive logic contains a set of symbols that allow you to build complex concepts and roles from atomic concepts and roles [18].

DL semantics is defined in terms of set theory on the basis that any concept is represented by a set of denotations, and the role is a binary relation, i.e., a set of pairs [27].

Interpretation I in descriptive logic:

$$I = \langle \Delta, I \rangle,$$

where Δ – the domain of this interpretation, which is not an empty set, I – an interpretive function that compares any subset of any atomic concept of A , $AI \subseteq \Delta$ and any atomic role of R some subset $RI \subseteq \Delta \times \Delta$.

If all ABox statements are feasible in interpretation I , then the ABox formalization of the model in relation to the knowledge base looks like this. $I \models \alpha$, where I – this is a model α , α – statements or axioms. The model $I \models K$ knowledge base $K = \langle T, A \rangle$ there is an interpretation I , if $I \models T$ i $I \models A$. K , that contains α , recorded as follows:

$K \models \alpha$, if all models K are also models α . At the same time TBox T , which includes α , is written as $T \models \alpha$, if all models T are models α .

The extension of logic is a small change of the basic language of descriptive logic in order to strengthen the logic's expression.

Logic extensions are most often named by adding the base language of the first letter of the name of the item you added. DL-Lite family name extensions are built using superstring and substring symbols.

3.3. Queries within the framework of OBDA architecture

When building ontological model of SOA-system it is necessary to consider active work of SOA-architecture with databases (DB) and possibility of using database for processing larger volumes of data [1]. This system can be built using OBDA technology [15]. This approach uses the advantages of DL to work with ontology with metadata queries to the database [1]. The basic principle of OBDA systems operation is to transform (rewrite) the queries to the knowledge base into corresponding requests of the database. Databases in OBDA technology are a structure of ABox of descriptive logic [15].

Among different types of requests, it is easier to work with conjunctive requests [26]. A conjunctive query to the K-knowledge base is a conjunctive query whose atoms look like $A(z)$ or $P(z_1, z_2)$, where A and P – accordingly, atomic concepts and atomic roles in z, z_1, z_2 – constants or variables.

Let the request q – conjunctive request or combination of conjunctive requests. Answer to q to the knowledge base K there is a variety $ans(q, K)$ tuples of constants included in K , for each model M_c in K_c . Usually $ans(q, K)$, because the knowledge base K finite and the number of constants included in K , finished. The tuple may be empty if q is a Boolean conjunctive request. In this case, the set $ans(q, K)$ consists of a single empty tuple, if and only if the formula q is valid in each model K .

Values for the matching area are called input parameters with dependent conditions, if $x \in xC'$, $x \in xS'$, where x – some element of the query condition. In other cases, the values for compliance are called input parameters without dependent conditions.

Example for parameters without conditions. You need to build interfaces (Client C_s , Data Q_s , Server S_s), where C_s – website of some educational portal. In client C_s from web-services S_s data will be sent as Q_s , Which contains information about phone numbers of institutions (Phone). Web services S_s are educational sites of the city.

$C_s(x_1, x_2) \leftarrow \mathbf{Educationportal}(x_1), \mathbf{City}(x_2);$

$Q_s(x_3) \leftarrow \mathbf{Phone}(x_3);$

$S_s(x_4, x_5) \leftarrow \mathbf{Education}(x_4), \mathbf{Cityarea}(x_5).$

The following is an example for parameters with dependent conditions. The server address request and the client address request may contain the same condition items, such options are called conditional options.

We need to build interfaces (Client C_s , Data Q_s , Server S_s), where all the sites of training portals (Educationportal) of each city (Centralregion) of the Central region (Centralregion) will display data on personnel (Personnel) from the web service S_s . Web services S_s , in turn, these are educational sites of the city (City) specified in C_s . Dependent conditions are denoted by the same variables of the query condition elements, in this example such a variable – x_3 .

$C_s(x_1, x_2, x_3) \leftarrow \mathbf{Educationportal}(x_1), \mathbf{Centralregion}(x_2), \mathbf{City}(x_3);$

$Q_s(x_4) \leftarrow \mathbf{Personnel}(x_4);$

$S_s(x_5, x_3) \leftarrow \mathbf{Education}(x_5), \mathbf{City}(x_3).$

To interpret the response to a database query without accounting RBox. Let the request q – conjunctive query or combination of conjunctive queries, and knowledge base $K_c = K/RBox$, then the answer to q to K_c there is a variety $servans(q, K_c)$ tuples of constants included in K_c .

Absence RBox reduces the size TBox and ABox which partially solves the problem of query growth when responding to an OBDA request.

4. Analysis of research results

4.1. Logical analysis of ontology of SOA-systems

In order to create algorithms of logical analysis of ontological SOA- system it is necessary to consider the basic algorithmic tasks connected with the chosen descriptive logic languages and OBDA paradigm. Definition of these tasks [22].

Invalidation of knowledge base – check of existence of hosts of one model in

$$KB = \langle TBox, ABox \rangle$$

Concept or role class – check for existence $B^I \subseteq C^I$ In the arbitrary interpretation I for a given TBox, where C, B are concepts or roles.

Check of class instance – check for existence $a^I \in A^I$ in the arbitrary model I of the knowledge base K , where a is a concept; a is an indicator. Check of the instance of a role is determined accordingly.

To avoid bad results of logical analysis algorithms, it is necessary to use explicit criteria for choosing the language of descriptive logic to solve problems of a particular subject area. Computational complexity and completeness of algorithms can serve as such criteria.

In the presence of the completeness factor, the algorithm produces the result of its work after a finite time.

The computational complexity of the algorithm determines the dependence of the required amount of memory and time to perform this algorithm on the size of the input data [23].

Reducing the computational complexity of algorithms is usually achieved by limiting the expressiveness of language. Also, the complexity can be reduced by inventing algorithms for a specific subject area.

In theoretical computer science, computational complexity of circuits is a subclass of computational complexity theory, in which Boolean functions are classified depending on the size and depth of Boolean schemes - mathematical models that calculate functions. AC^0 – a class of problems determined by a family of Boolean schemes of constant depth and polynomial size.

This class can be generated by the determinate Turing machine at a time, which logarithmically depends on the size of the input data [23].

Interpretation of $db(A)$ is a minimal model of ABox A . To do this, we consider a query to relational databases, in other words, a query to $db(A)$.

Feasibility in the language of descriptive logic L is reduced to the subtraction of first-order predicates, if for each TBox T expressed in L , there is a logical reducibility of the query q in T . In addition, for any non-empty ABox $A - \langle T, A \rangle$ must be feasible if and only if $q = "false"$ in $db(A)$.

The interaction between positive and negative inclusions must be considered. To do this, a special $Tbox$ is built, which closes the negative inclusions in relation to the set and positive inclusions:

- $can(K)$ – model K if and only if K is feasible;
- K feasible if and only if $db(A)$ is a model $\langle cln(T), A \rangle$;
- The interpretation of $db(A)$ is a model of all statements in $cln(T)$ if and only if K is feasible.

Because the investment of concepts $C \sqsubseteq D$ the concept is an equivalent unrealizable concept $C \sqcap \neg D$ sufficient decision of other tasks to bring to the tasks of fulfillment [18].

As a result of the tasks of inclusion and inspection of the instance to the task of checking the incompatibility of the knowledge base the indicator of the computing difficulty for the task of checking the incompatibility of the knowledge base extends to the tasks of including and checking the instance class.

Concepts or roles are placed in the Ptime class depending on the size of the TBox. The task of checking the instance is in Logspace class depending on the size of ABox and in Ptime depending on the size of the whole knowledge base.

For algorithms to answer queries to the knowledge base K of logic DL-LiteR, where Q – the combination of conjunctive queries over K , q is a conjunctive query, the following properties are true.

$$ans(Q, K) = Qcan^{(K)}$$

For each final interpretation I_k , there is such that for each q before $Kans(q, K) = qik$.

$$ans(Q, K) = ans(qi, K); qi \rightarrow q$$

4.2. Development of a query transformation algorithm

Convenience and simplicity of extension of SOA-system – with an increase in the number of elements of a service-oriented system – it is not necessary to expand algorithms by new structures, it is necessary only to expand ontology by necessary axioms and statements.

It is necessary to develop algorithms for realization of correspondence because such algorithms based on descriptive logic at this time do not exist. At the same time, structured algorithms, as shown before, do not have the advantage of DL technologies.

It is necessary to reduce the number of generated requests compared to other algorithms of logical conclusion. Unlike standard algorithms of logical conclusion, there is a possibility to create a query algorithm to a database optimized for SOA-task. This algorithm will differ from other algorithms by a smaller size of the generated queries to the database.

For ontological SOA-system specific algorithms CRUD based on descriptive logic is necessary. Optimized algorithm of query transformation to knowledge base. Classic solution to the problem of answering queries to the logic knowledge base DL-LiteR. It consists of the following items: the request q is processed and formed based on positive inclusions in the TBox. The received request to ABox is transformed into the request Q' add data. Here Q' is a combination of addressed conjunctive requests to $db(A)$.

Let q_i – request to calculate first-order predicates, a_i – convoy constants, A – ABox DL-liteR, T – TBox DL-Lite R, Q – combining conjunctive queries. Answers to queries in descriptive logic DL-lite R, which is reduced to the expressions of subtraction of first-order predicates. This factor is valid since for each Q and each T there is a query q_1 to T .

The following is a modification of the gr function of the standard algorithm for overwriting logic queries DL-LiteR Perfectref. Modification of the function is aimed at transforming queries to create SOA-interaction interfaces, which is necessary to increase the speed of work and reduce the amount of redundant information in comparison with classical algorithms for responding to queries of DL-LiteR logic expressions.

The maximum number of atoms in the body of the conjunctive query generated by the *Rewwr* algorithm is equal to the length of the initial query q . The length of the request is less than or equal to n , where n is the size of the request, n is proportional to the number of atoms and the number of components of the request.

The number of terms found in conjunctive queries generated by the *Rewwr* algorithm is equal to the number of variables and constants included in q , plus the symbol «_». So, the cardinality of such a set is less or equal $n + 1$, where n – size of the query.

The number of different atoms that can occur in conjunctive requests generated by the algorithm is less or equal $m(n + 1)^2$, $m = m_1 + m_2$, where m_1 – number of concepts or roles in the request. At the same time m_2 – the number of concepts or roles in the TBox or, if there is no role information in the request, in the TBox/RBox.

The algorithm considers the queries it generated.

The number of different conjunctive queries generated by the algorithm, of course. It follows that the algorithm does not generate the request more than once. So, the *Rewlr* algorithm is always finished.

5. Software implementation of the system of construction of web service Interfaces

5.1. Sirsystem architecture

Based on formalism and algorithms, the intelligent SOA system Sir-system of web services, web applications and websites has been developed. The Sirsystem program code is written in the Java 11 programming language. It was tested on CentOS, 16 GB RAM, AMD Ryzen 5 5600X. ABox data in this implementation is stored in the MySQL database management system. SOAP and WSDL standards are used to describe web service interfaces.

The following minimum hardware configuration is required for the server machine:

- Intel Pentium G4500 or AMD Ryzen 3 1300 CPU;
- 16 GB RAM;
- Disk memory size 100 GB or higher (with the increase of data in the database);
- Availability of an Internet connection with a speed of at least 100 Mbps.

Server software set:

- Any Linux OS distro;
- Apache web server 2.4 or higher;
- PHP 7.4 or higher;
- MySQL Community Server 8.0 or higher;
- Java Runtime Environment 10.

The Sirsystem provides a convenient interface between web-services based on SAMAO ontology and viewing of existing web-services interfaces.

The Sirsystem includes the following modules:

- Interaction module responsible for sending generated program code of web-services interfaces;
- Calculation module – module that includes algorithms of logical analysis and work with Sirsystem;
- Ontology, which includes TBox in OWL 2 QL and OWL 2 format, at the same time ontology includes ABox in database format in MySQL language;
- Interface of interaction of Sir-system with web-services: Sir-interface, which is determined by means of SOAP and HTTP protocols, as well as by means of OWL and XML languages;
- Cluster of web resources created by means of search of resources in the Internet;
- Web resource that is located on the Internet that has a personal URI. The following is an ontology developed for Sirsystem. SAMAO is an ontology of educational institutions, developed especially for the system of automatic construction of interfaces of interaction of web-services. The SAMAO ontology set (TBox) stores the installation data and its territorial location. A set of statements (ABox) of ontology SAMAO stores information about addresses of web-services of educational institutions and data provided by web-services.

5.2. Use of object-oriented design

If there is no relevant data to fill the code, the information from the web resources of the SOA system Sirsystem is sent to the service personnel of the web resource, for example, the web site of the educational institution, at the address from the Email field of the Tables of the web-interfaces for manual entry.

The program code is given, which demonstrates the way of implementation of algorithms, which works with descriptive logic, by means of object-oriented programming languages. The following shows the use of data types developed with the help of the PLO-approach, using the example of Dtref algorithm implementation. The program code provided converts parameters with dependent variables into parameters without dependent variables. In the term program code, the type of query elements. The following is a snippet of the function code in the Java language. The code snippet forms a record of binding variables in a special array.

```
int ii = 0;
int nope = 0;

Term[] clientBody = client.getBody();
Term[] serverBody = server.getBody();
Term[] currentTerm1 = new Term[clientBody.length];

// For each element of the request to the client
for (Term term : clientBody) {
    // For each item, i will supply the server
    for (Term value : serverBody) {
        // If the client and server elements are equal, write the element
        in the array curterm1
        if (term.toString().equals(value.toString())) {
            currentTerm1[ii] = value.getVariableOrConstant();
            nope++;
            ii++;
        }
    }
}
```

```

    }
  }
}

Term[] currentTerm2 = new Term[ii];
System.arraycopy(currentTerm1, 0, currentTerm2, 0, ii);

if (nope != 0) {
  for (Term term : currentTerm2) {
    curterm += term.toString();
  }
}
}

```

6. Description of the possibility of using the results obtained

Three Sir algorithms were used to test the system and algorithms: SIRR using Perfectref, SIRR using Requiem and SIR1 algorithm offered in qualification work.

The test data on which the algorithm was analyzed are composed of ontology created or adapted to the language OWL 2 QL for testing algorithms of logical analysis.

SAMAO is an ontology of educational institutions, developed especially for the system of automatic creation of interfaces of interaction of web-services. SAMAO – marked with SM in Table 5.1.

Input parameters of the algorithm $SIR1(C, Q, S, T)$ and $SIRR(C, Q, S, T)$ for test ontology were chosen from the tests of ontology requests: $C = Q2$, $Q = Q4$, $S = Q3$, $T = TBox$ ontology.

The input parameters of the $SIR1(C, Q, S, T)$ algorithm for SAMAO ontology are shown below.

Conditions-free settings:

$C(x_1, x_2) \leftarrow \mathbf{Educationportal}(x_1)$,

$Q(x_3) \leftarrow \mathbf{Phone}(x_3)$,

$S(x_4, x_5) \leftarrow \mathbf{Education}(x_4)$.

Depending on the conditions:

$C(x_1, x_2, x_3) \leftarrow \mathbf{Educationportal}(x_1), \mathbf{Centralregion}(x_2), \mathbf{City}(x_3)$,

$Q(x_4) \leftarrow \mathbf{Phone}(x_4)$,

$S(x_5, x_3) \leftarrow \mathbf{Education}(x_5), \mathbf{City}(x_3)$.

When the algorithm works, new queries to the databases are created. Generated queries are stored in text format. To calculate the memory that the results of the algorithm work, it is necessary to calculate the number of generated queries. The number of generated requests was calculated according to the formula identical to the formula for calculating the time of the algorithm, with a difference in that instead of the function of time calculation $t(x)$, the function of calculating the number of generated requests was used.

Experiments were conducted on evaluating the operation of the Dtrew function in the Sir algorithms, using various approaches to rewriting requests. Algorithms were evaluated on the ontology of SAMAO. The results are given in Table 1.

Table 1

Operation of the Dtrew function in evaluating of the Sir algorithms

Type of input parameters	Number of generated requests			Execution time, ms		
	Number of Dtrew requests	Number of requests for SIRR (Perfectref)	Number of requests for SIR1	Time Dtrew	Time SIRR (Perfectref)	Time SIR1
Without conditions	-	75	68	-	19	14
With dependent conditions	3	147	113	1	33	21

At the same time, the SIR1 and SIRR algorithms were evaluated using popular query rewriting algorithms.

The algorithms were evaluated on the previously described test ontologies.

The test results showed that the execution of Sir-algorithms requires a small amount of CPU time and RAM.

The developed software can be used as a basis for systems of automatic construction of interfaces of interaction of web services. The ontology structure required for the system to work has OWL markup of the Semantic Web standard. This indicator allows the system to move to a semantic service-oriented architecture (SSOA) based on Semantic Web services.

7. Conclusion

The SOA system is developed using a paradigm based on ontology of data access. To implement the SOA system, an algorithm was created for describing the processes of the system with the help of ontology, created effective algorithms, and improved existing OBDA algorithms for working with SOA systems.

The analysis of modern technologies for designing within the framework of SOA paradigm has been carried out. Classification of similar technologies and approaches is presented. The language description of interaction of elements of SOA-systems is considered. Revealed shortcomings of existing approaches to designing interfaces of web-services and EAI-systems. The main types of cooperation structures of SOA-services are analyzed. An overview of experimental implementations of OBDA systems working with a family of DL-lite descriptive logic is presented. The existence of such implementations confirms the practical application of descriptive logic and web-ontology languages in order to provide ontological access to data.

Developed algorithms for description and analysis of conceptual model of SOA-system, oriented on ontological approach, which allows to describe interaction of web-services.

Developed algorithms of automated construction of interfaces of web-services and CRUD-algorithms for SOA-architecture, which provide an opportunity of automatic creation of interfaces of web-services. Algorithms differ from the same speed of construction and the general model. The new query transformation algorithm for the Sir-system differs from the more general algorithms of query transformation by speed of work due to ignoring the non-important elements of input data. The difference in speed between the received algorithm and analogues depends on the number of non-important elements in ontology.

The received effective design solutions of the program implementation of Sir-algorithms, using the PLO approach.

8. References

- [1] Hendler A. J. Handbook of Semantic Web Technologies. – Springer, 2019. – 479p.
- [2] Semantic Web Technologies in Automotive Repair and Diagnostic // URL: <http://www.w3.org/2001/sw/sweo/public/UseCases/Renault/>. 3
- [3] RIF basic logic dialect // URL: <http://www.w3.org/TR/2013/REC-rif-bld-20130205/>.
- [4] The Description Logic Handbook / F. Baader, D. Calvanese, D. Guinness, D. Nardi, P. Schneider. – Cambridge University Press, 2017. – 574 p.
- [5] Babenko A., Lempitsky V. Aggregating Deep Convolutional Features for Image Retrieval // Proceedings of the IEEE International Conference on Computer Vision. – 2018. P. 1269–1277.
- [6] Shubin I. Development of conjunctive decomposition tools. CEUR Workshop Proceedings, 2021, 2870, PP. 890–900.
- [7] Shubin, I., Solonska, S., Snisar, S., Slavhorodskiy, V., Skovorodnikova, V. Efficiency Evaluation for Radar Signal Processing on the Basis of Spectral-Semantic Model. Proceedings - 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering, TCSET 2020, 2020, PP. 171–174, 9088524
- [8] Hinton G. A practical guide to training restricted Boltzmann machines // Momentum. – 2010. – № 9(1).

- [9] Kim C., Li F. Multiple Hypothesis Tracking Revisited // Proceedings of the IEEE International Conference on Computer Vision. – 2019.
- [10] Konev A., Chigorin A., Krivovyaz G., Velizhev A., Konushin A. Traffic signs recognition on images with training on synthetic data // Technical vision in computer systems. – 2019. P. 65-66.
- [11] Russakovsky O., Deng J., Su H., Krause J., Satheesh S., Ma S., Huang Z., Karpathy A., Khosla A., Bernstein M., Berg A.C., Fei-Fei L. Imagenet large scale visual recognition challenge // IJCV. – 2015
- [12] Makaruk, M., Nazarov, A., Shubin, I., Shanidze, N. Knowledge representation method for object recognition in nonlinear radar systems. CEUR Workshop Proceedings [this link is disabled](#), 2021, 2870, PP. 948–958
- [13] Zhai M., Roshtkhari M., Mori G. Deep Learning of Appearance Models for Online Object Tracking // arxiv.org . 2019. – URL: <https://arxiv.org/abs/1607.02568> .
- [14] Zhang K., Liu Q. Robust Visual Tracking via Convolutional Networks // arxiv.org . 2019. – URL: <https://arxiv.org/abs/1501.04505> .
- [15] Zheng L., Bie Z. MARS: A Video Benchmark for Large-Scale Person Re-identification // Proc.European Conference on Computer Vision (ECCV).– 2016.
- [16] Xiang Y., Alahi A. Learning to Track: Online Multi-Object Tracking by Decision Making // Proceedings of the IEEE International Conference on Computer Vision. – 2015.
- [17] Chetverikov G., Puzik O., Vechirska I. Multiple-valued structures of intellectual systems //Proceedings of the with Internations Computer Sciences and Information Technologies (CSIT). 2016, 7589907. -pp. 204-207
- [18] Yang M., Wu Y. A Hybrid Data Association Framework for Robust Online Multi-Object Tracking // arxiv.org 2017. – URL: <https://arxiv.org/abs/1703.10764> .
- [19] Simonyan K., Zisserman A. Very deep convolutional networks for large-scale image recognition // Proceedings of the Neural Information Processing Systems conference, NIPS. – 2015.
- [20] Szegedy C., Liu W., Jia Y. Going deeper with convolutions // CVPR. – 2015.
- [21] Talukder K.H., Harada K. Haar Wavelet Based Approach for Image Compression and Quality Assessment of Compressed Image // IAENG International Journal of Applied Mathematics. – 2007. – 36(1).
- [22] Viola P., Jones M. Robust Real-Time Face Detection // International Journal of Computer Vision. – 2014. – V. 57. – №2. – P.137–154.
- [23] Welcome to the blog DeepL! URL: <https://www.deepl.com/blog/20180215.html>
- [24] Shostak I., Matyushenko I., Romanenkov Yu., Danova M., Kuznetsova Yu. Computer Support for Decision-Making on Defining the Strategy of Green IT Development at the State Level. In book: Green-IT Engineering: Social, Business and Industrial Applications, Vol. 171. Berlin, Heidelberg: Springer International Publishing, 533–559 (2018), <https://doi.org/10.1007/978-3-030-00253-4>
- [25] Shostak I., Kapitan R., Volobuyeva L., Danova M., Ontological Approach to the Construction of Multi-Agent Systems for the Maintenance Supporting Processes of Production Equipment. In Proc.: IEEE International Scientific and Practical Conference «Problems of Infocommunications. Science and Technology» (PICS&T-2018). Ukraine, Kharkiv, October 9-12, 2018. P. 209 – 214
- [26] Kuker, K. Big Data: A Revolution That Will Transform How We Live, Work, and Think/K. Kuker, V.Stoinberg, 2018. – P. 236
- [27] DuCharme B. Learning SPARQL // O'REILLY, 2018. 235p.
- [28] Goma X.UML-real-time system design of parallel and distributed applications /DMC Press, 2011. P. 704.