Simple and Flexible Way to Integrate Heterogeneous Information Systems and Their Services into the World Data System

Submitted: 8th October 2021; accepted: 17th March 2022

Grzegorz Nowakowski, Sergii Telenyk, Kostiantyn Yefremov, Volodymyr Khmeliuk

DOI: 10.14313/JAMRIS/4-2021/29

Abstract:
The approach to applications integration for World Data Center (WDC) interdisciplinary scientific investigations is developed in the article. The integration is based on mathematical logic and artificial intelligence. Key elements of the approach – a multilevel system architecture, formal logical system, implementation – are based on intelligent agents interaction. The formal logical system is proposed. The inference method and mechanism of solution tree recovery are elaborated. The implementation of application integration for interdisciplinary scientific research is based on a stack of modern protocols, enabling communication of business processes over the transport layer of the OSI model. Application integration is also based on coordinated models of business processes, for which an integrated set of business applications are designed and realized.

Keywords: *research; application integration; business processes; mathematical logic; formal logic; inference mechanisms; multi-agent systems; protocols; software agents*

1. Introduction

In view of the globalization of the economy and social life, National Science must integrate into the world and European organizations that promote the consolidation of research and consequently the development of scientific activity [27]. This is a very important process since there is an urgent need for interdisciplinary research, primarily for the assurance of sustainable development globally and regionally [26]. However, effective implementation of interdisciplinary research requires the creation of appropriate conditions for information exchange in the process of solving scientific problems. The scientific and technical progress that in its time facilitated the creation of information and communication technologies (ICT) nowadays benefits greatly from them. They are developing rapidly, covering new spheres of human activity and enhancing performance. Yet only field specialists can use ICT in a rational way, whereas the need for efficient information exchange within interdisciplinary research can only be met through rational ICT application by specialists with deep knowledge in their areas of expertise [19].

The development of the information technologies (IT) domain that is experiencing qualitative changes related to ICT strengthening has created conditions for distributed computation and the efficient use of information and other resources. Consolidation of resources and the introduction of virtualization technologies are contributing to the process of substituting local solutions with distributed ones that allow the comprehensive use of all computing powers and data storage systems linked into a global network, thus granting access to accumulated information resources. The service approach formed on the basis of communication services has spread to the infrastructure, software development tools, and applications. The emergence of a wide range of new types of services, especially content-based ones, has led to the convergence of services and the formation of a generalized concept of information and communication services (ICS). The number of ICS providers has grown rapidly and convergent providers have emerged. The wide functionality, high quality, and moderate price of new services rendered by providers allow businesses to abandon the development of in-house or IT infrastructure and to use a wide variety of available ICS for component-based design of their information and telecommunication systems (S).

However, unified access to services is becoming a condition for the efficient use of the advantages of distributed systems and the possibilities of service-oriented technologies. At the same time, the formation of a new democratic IT environment, in which even small businesses can render services, has naturally been accompanied by the use of various tools and access technologies [19]. Therefore, historically the IT environment is heterogeneous, hence user access to its resources is to some extent complicated or at least inconvenient.

The same situation is characteristic of scientific activity in particular. For example, the World Data Centers (WDC) system created in 1956 under the aegis of the International Council for Science (ICSU) ensures collection, storage, circulation, and analysis of data obtained in various science areas [22]. During its existence, the WDC system has accumulated a lot of data and applications that may be used to solve the challenging problems of social development. They are one of the most powerful information resources used by hundreds of thousands of scientists, and the demand for it is increasing in proportion to the need for interdisciplinary research related to sustainable development, the solution of urgent environmental protection issues, etc. However, problems related to

the incompatibility of legacy applications caused by architecture differences, the variety of data presentation formats, and other factors prevent the effective use of such WDC resources.

Promising architectural solutions are being developed and gradually implemented in the ICT field, such as Next Generation Network (NGN) and Next Generation Service Overlay Network (NGSON) [24], that ensure the interaction of various transport layer technologies. Yet there is a need for a comprehensive integration of resources from various sources, and ICT developers' efforts should seek to enable scientists working in various domains to use the accumulated resources based on their areas of expertise, and not on the IT particularities. The sources, comprehensive access to which it is reasonable to ensure, include databases [16], websites and portals, various legacy file management systems, and data repositories structured according to various models.

Nowadays there exist more than 50 WDCs that for more than 50 years have created a system for data accumulation, analysis, processing, and international exchange. WDCs' powerful data storage systems retain huge volumes of astronomical, geophysical and other scientific data. Certainly, from the point of view of scientists working on various resource-intensive problems, it is important to have access not to a large disordered system of possibilities, but to an integral complex of data and application sources intended to meet their specific needs, with a user-friendly interface that does not require any special IT knowledge. Nevertheless, the system for WDC data accumulation, analysis, processing, and international exchange does not provide such access. Furthermore, it was not designed for the growing level of scientific society's requirements and is not versatile enough to be used in interdisciplinary research. Therefore, a new interdisciplinary structure was created in 2008 – the World Data System (WDS) – to develop and implement a new coordinated global approach to scientific data, which guarantees omni-purpose equal access to quality data for research, education, and decision making. The new structure will have to solve the accumulated tasks, primarily the unification of formats and data transfer protocols, assurance of convenient access to data, and the organization of scientific data quality control [27].

The related difficulties are emerging because the existing WDCs are accumulating and providing heterogeneous data determined by the specificity of a corresponding science area and country. The possibilities of telecommunication systems are constantly growing, as well as the possibilities of modern computers and data storage, thus making way for new distributed computation technologies – grid systems and cloud-based computation. Integrating formerly independent systems for the accumulation, storage, and processing of WDC data on a new advanced integration basis will allow a considerable enhancement of their overall efficiency. The creation of such a system will provide scientists with convenient centralized access to formerly separate resources, facilitating and quickening scientific and research activity around the world.

To implement such integration solutions that support the continuous lifecycle of research data, it is proposed to apply the well-proven Continuous Integration (CI) and Delivery (CD) software development practices.) [28], [29], [30]. Continuous Integration and Continuous Delivery are often quoted as the essential ingredients of successful DevOps. DevOps is a software development approach which bridges the gap between development and operations teams by automating build, test and deployment of applications. It is implemented using the CI/CD pipeline. DevOps lifecycle contains six phases: Plan, Build, Continuous Integration, Deploy, Operate and Continuous Feedback. Throughout each phase, teams collaborate and communicate to maintain alignment, velocity, and quality.

CI/CD are required for software development using an Agile methodology, which recommends using automated testing to fast debug working software. Automated testing gives stakeholders access to newly created features and provides fast feedback.

The use of CI/CD practices requires the fulfillment of a set of basic requirements for a development project.

In particular, the source code and everything needed to build and test the project should be stored in a repository of the version control system, and operations for copying from the repository, building and testing the whole project should be automated and easily called from external programs. The GitLab environment allows to manage project repositories, document functionality and results of improvements and tests, as well as track errors and work with the CI/CD pipeline [31], [32].

2. Related Work

In this paper, the authors would primarily like to focus on the consideration of urgent among the WDC problems requiring efficient solutions is that of integration. Presently, numerous WDCs use legacy technologies based on program systems with rigid structures, whereas newer solutions have for a long time been based on client- server and web-oriented technologies. The service approach to data processing and analysis has not spread to the required extent. Sometimes connection to geoinformatics systems and even to basic public services such as Google Maps is unavailable. Combined with the lack of a unified standard, this fact shows that the need for data sources integration, including applications integration and databases [16] integration, may be considered sufficiently substantiated. Therewith, the formation of a modern IT environment envisages the following integration aspects:

- technology: integrated data and services should be based on a single stack of interaction protocols, using common data transfer formats in order to render the integration
- possible;
- content: a single data description format in the semantic structure of data sources should be supported;

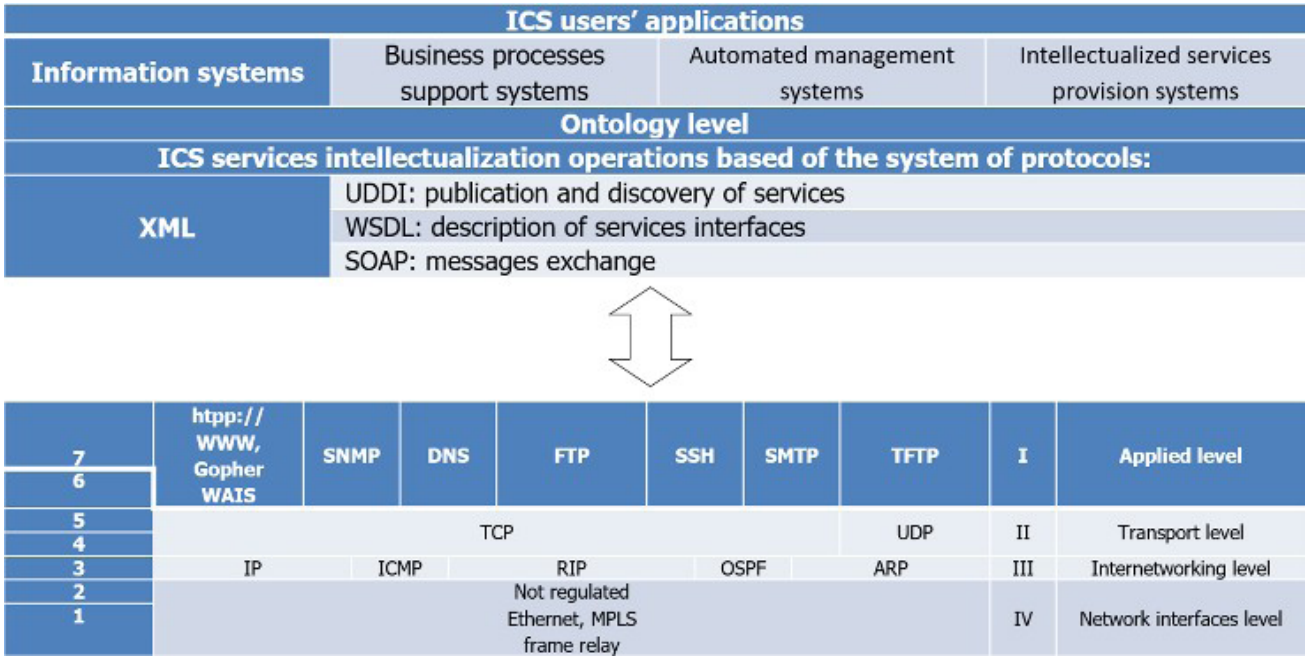

Fig. 1. Stack of services interaction protocols

- functionality: the use of different services in the single structure to solve user-defined problems.

An overview of existing integration solutions in all the mentioned aspects, starting with the integration technology aspect has been done. The existing solutions are powered by technologies for the creation, functioning, and development of distributed systems. Notwithstanding the differences of various technologies for the creation of distributed service-oriented systems, the overall principles and theoretical and methodological approaches are always similar. Independent services should be registered, described, and provided with the possibility to communicate transparently with clients and with each other. Furthermore, networking interaction requires us to determine protocols for all levels of the OSI model. To do so, the corporate ITS standard structure with services intellectualization operations may be used, as suggested in [15], taking into account international, state, and branch standards, and corporate documents, first of all [4]. The structure consists of two parts, where the first covers the traditional four levels of standards of the TCP/IP protocols stack, and the second covers the user applications in accordance with the ITS class, destination, and services intellectualization operations. Fig. 1 presents an example of the corporate standard structure.

For convenience, the figure presents the levels of the international standard for interaction of open OSI systems and their correlation with the corresponding levels of the TCP/IP protocols stack. The ITS classification used is proposed in [15]. It allows the systematization of various ICS by the level of users' requirements and queries, practical needs, and professional training, with no limitations imposed on the ITSs' functionality, attributes, or operations. The changes introduced apply to the first part of the structure and to the detailing of protocols that power the intellectualized interaction of applications in the course of solving users' more complex problems.

The services registry is maintained by the Universal Description, Discovery and Integration (UDDI) technology [6] that allows both people and client-programs to publish information on services and search for a required service.

The Web Service Description Language (WSDL) that, according to the W3C definition, constitutes an XML format for the description of networking services as a set of operations working with document- or procedure-oriented information through messages [11], is used for the unified description of services, which allows them to be used independently from the programming language. WSDL documents, by virtually creating a unified layer that allows the use of services created on the basis of various platforms, describes the service interface, URL, communication mechanisms "understood" by the service, methods provided by it with corresponding parameters (type, name, location of the service Listener), and the service messages structure. Let us mention that the majority of platforms support the formation of WSDL descriptions of services, enabling developers to work with services as simple classes.

To implement the key part of the interaction -messages exchange – one of the widespread technologies may be used, and the selection should be determined by compliance with the requirements of the system for data accumulation, processing, and exchange:

- SOAP: a user-friendly technology that is easy to use with the Business Process Execution Language (BPEL), which ensures interaction of distributed systems irrespective of the object model, operating system, or programming language. Data is transferred as special-format XML documents [5];
- CORBA: a mechanism created to support the development and deployment of complex object-oriented applied systems, which is used for the integration of isolated systems, allowing programs that are developed in different programming lan-

guages and working in different network nodes to interact with the same ease as if they were in the address space of a single process. Such interaction is ensured through unified construction of their interfaces using a special-purpose declarative Interface Definition Language (IDL). At the same time, the interface and the description thereof do not depend on the operating systems or the processor architecture either [1];

- REST (Representational State Transfer) is a style of software architecture for distributed systems, which is used to create web services. A global URL identifier unambiguously identifies each unit of information in an unvarying format. Data is transferred without any layers [17];
- WCF (Windows Communication Foundation) is Microsoft's platform designed to create applications that exchange data through the network and are independent from style and protocol [7].

Although nowadays the most acceptable choice seems to be SOAP, based on which the interaction presented in Figure 1 is performed, all the technologies allow the integration of services of various origin [9,10].

To take into consideration service-based, process-functional, and component-based approaches to the design and maintenance of users' solutions, the following well-known architectural means of organizing shared functionality are used:

Web Service Choreography is the approach that determines the protocols of web services' interaction to perform a single global task by performing parts thereof. The role assumed by the service determines its model of exchanging messages with other services. This method shows high efficiency for small tasks, but with increased complexity of tasks the number of services involved grows rapidly, solutions become too massive, and the efficiency drops quickly [13].

Service orchestration is the approach that uses an orchestrator service to associate services into a single multifunctional business service. The role of orchestrator services may be performed by agents introduced into the system, which allows its hierarchical structure to be maintained, thus combining the advantages of orchestration and multi-agent systems (MAS). To perform orchestration, a special-purpose language was developed to describe the business processes and the protocols of their interaction – BPEL. This language, sometimes called the orchestrating language, provides an orchestrator process (service) to coordinate the work of services subordinated to it. The coordinated interaction is enabled through messages exchange between web services. The problem lies in the fact that, in order to carry out long-term business processes with complex structure, not only messages themselves are important, but also the exchange sequence, account of state, results, time and other aspects of business processes. If the work of basic web services is organized by a program agent as a higher-level orchestrator service, we get the management mechanism at the web services level, whereas organizational aspects (messaging sequence, account of load, etc.) will be dealt with by higher-level agents. As a rule, SOAP is used along with BPEL to implement messages exchange, although there are alternatives [2].

The scientific community is already using systems based on the decomposition of the overall problem (flow) into a sequence of individual subtasks (subflows), the so-called workflow system. The effective interdisciplinary workflow environments worth mentioning are Triana Workflows and Kepler. A researcher user may edit the task-performing scripts, using system automation means, which renders it versatile and widens its application sphere. Those systems' architecture consists of the following levels:

1) client (allows a user to quickly develop new workflows using the graphic script editor);
2) middleware (its flow server ensures workflows performance, authorization and coordination with the heterogeneous resources involved);
3) resource (provides resources to perform separate subflows).

A workflow constitutes a separate ready-to-use component. Components may be parts of other components. Due to rigid typification (internal formats determined for flows and components) and the formation of metadata for each component, which describe the types of their input and output data, flows and components are verified to make sure their input and output data are compatible. These systems were widespread due to their easy-to-use graphic user interface, extensive library of standard components developed for various disciplines, effective script performance manager, and means of monitoring.

Taverna Workflow Management System, a system for the development, design, and performance of complex workflows, which allows the use of different services such as WSDL, BioMoby, SoapLab and other types to be coordinated, has also been widely used lately [8].

Alas, such systems are incompatible at the levels of both the performance manager and the description of flows and components. Furthermore, they lack an intellectual constituent, or such a constituent is not sufficiently developed, and therefore any working sequence of services needs to be built up manually or edited, which substantially limits these systems' versatility and application sphere.

Based on the abovementioned technologies, a few solutions have been worked out, which may be used to create applied systems for the accumulation, processing, and exchange of scientific data in different areas. The best known among them are the ESIMO and GEOSS systems. They are quite widespread, although they have a number of draw- backs in terms of processing heterogeneous information:

- difficulties in compiling the association of heterogeneous types of data from different sources in one query;
- difficulties in enhancing the functionality and degree of automation, caused by drawbacks in these systems' architecture, which does not envisage the possibility of forming a single network;

- no unified metadata model for all the metadata objects;
- no unified standard for data presentation and processing;
- users need special expertise, knowledge, and skills to form a query.

Based on this overview, we can draw a substantiated conclusion regarding the possibility of using the developed approaches, models, technologies, and means for integration of applications, primarily:

- schemes for the implementation of applications interaction based on the stack of protocols UDDI, SOAP (or REST), BPEL, WSDL, XML above the transport level;
- technology of WCF type;
- approach to the integration of services through an orchestrator service into a single multifunctional business service.

Nonetheless, for user-formulated tasks, the level of automation of services interaction schemes should be enhanced. Ultimately, this is a question of developing efficient models and methods of applications integration, based on which a new state-of-the-art applications integration system may be created, capable of providing users with the abovementioned advantages.

Secondly, let us dwell briefly on the content aspect. As a rule, data integration in information systems is viewed as the provision of a single unified interface to access a certain number of generally heterogeneous independent data sources [3,14]. For a user, information re- sources from all the integrated sources should be presented as a new single source, and the system providing such possibilities is called the data integration system [12]. There are several data integration approaches, some of which are successfully implemented and quite widespread in real systems. The analytical review thereof is presented in [12,20]. The data space concept viewed as a new abstraction of data management has been steadily changing its positions lately. In Ukraine and Poland, it was developed in works by N. Shakhovska, for instance [21].

Finally, let us move on to the third aspect of integration. The view on integration as the interaction of applications is not as widespread, yet is extremely important in the con- text of resolving the comprehensive integration issue in the new IT environment. It was mostly formed within the community of specialists in programming automation, notably in relation to the implementation of promising technologies for industrial software production, such as "component-based programming". In terms of counteracting program systems complexity, the focus was transferred from developing versatile, adjustable, and scalable structures, such as application packages, to developing means for the interaction of independent program systems.

The specific systematic research of the application integration began at the turn of the millennium, when the issue of the integration of numerous business applications, primarily of CRM, ERP, SSM classes, etc., abruptly emerged. Under the new economic conditions, business charges IT with new tasks, the fulfillment of which through the purchase of new and the modernization of available business applications does not ensure a return on investment any more. At that very moment, the need arose to build enterprises' information systems based on the integrated complex of business applications, and the first attempts were made at the formalized description of business applications and their interaction with the aim of reaching the business divisions' objectives. Thus emerged the Component Object Model (COM) and Common Object Request Broker Architecture (CORBA) technologies that ensured programs interaction on a single computer, and subsequently such possibilities were applied to the interaction of program systems located on different computers. The abovementioned possibilities implemented in the Distributed COM (DCOM), Electronic Access Interchange (EAI), and CORBA technologies ensured independent program systems integration to solve users' problems that are more complex.

Another solution that is important for applications integration was the Service-oriented Architecture (SOA), whose emergence gradually, through the integration of multi-bases and federated databases, large data storages, and various repositories, promoted the transition of the integration concept onto web applications. Nonetheless, a total renunciation by IT departments of solving efficacy issues through the integration of business applications into the existing IT infrastructure with minimal losses only became possible after the 2008 crisis, with the emergence and implementation of cloud-based computation technologies. At that very moment, there arose the obvious necessity to transparently see the IT infrastructure of an enterprise's entire information system and the means of its architecting, operation, and development, with an account of the new possibilities of the IT environment that was being intensely formed on the basis of consolidation, virtualization, and multi-services.

The industry leaders proposed a solution based on integration platforms ensuring complete functionality through advanced means for scaling and adjustment. However, such a centralized approach requires a respective IT infrastructure and not every enterprise can afford this. Furthermore, it entails a certain dependence of the enterprise on companies producing such integration solutions.

Therefore, the majority of enterprises are more interested in the integration approach related to the wide use of business applications by various developers through proxies, such as Message Oriented Middleware (MOM) programs, combined with modern approaches to cost-cutting in IT-infrastructure ownership, primarily the means of using legacy applications. Another obstacle in the way of implementing the decentralized approach is differences in the architectural components of business applications by different producers, notably in the models of data and business processes, technologies of designing the program system and its basic elements (DBMS serv-

ers and applications), component cooperation means, users' access, etc.

Cloud-based computation allows the provision of software services of a determined quality for a fair price, thus justifying the practicability of software (SaaS), infrastructure (IaaS) and platform (PaaS) services [25]. Nowadays, this activity may be viewed as a theoretical and methodological basis for the migration of the service approach to information systems creation and the wide introduction of component-based development thereof.

In order for legacy applications to become part of a service-based system, their data models and business processes must be in a certain way coordinated with those supported by the new systems, and interaction with the program system's basic elements must be ensured. Therefore, data integration in information systems and integration as the coordination of applications have to be combined. The necessity arises to develop a technical solution that will allow:

- creating mechanisms for efficient information resources management;
- widening areas of protocols standardization and data exchange formats based on the interaction of commodity systems;
- creating cross-platform applications, integrating data and applications;
- facilitating systems deployment and development by concealing software and hardware internal details.

To create solutions that provide the above possibilities, it is required to:

1) develop formal models and methods to supply metadata in order to describe sources, their functionality, and particularities of access;
2) develop models and methods for the automated design and implementation of unified-architecture applications that are capable of efficient interaction in order to reach the objectives of interdisciplinary research, and on the basis thereof to create platform solutions according to the PaaS model;
3) develop models, methods, and means to implement applications interaction at the semantic level;
4) develop models and methods to facilitate users' queries for required data using the terms from topical area(s) of research that are convenient for them;
5) bring the IT infrastructure of the WDC system into compliance with the needs through consolidation, virtualization, and multi-services powered by cloud-based computation models and technologies;
6) develop models, methods, and means for efficient administration of the IT infrastructure of the WDC system, capable of ensuring the determined level of information services quality.

One of the most crucial issues is that of developing mathematical models, methods, and means for the integration of various applications, both legacy ones based on traditional technologies and client-server and web-oriented technologies. The article proposes an approach to applications integration using the mathematical logic instrument and artificial intelligence theory for interdisciplinary research through the example of the WDC system functioning. The specificity of the applications integration for interdisciplinary research is that the coordination of business processes models is not required because it is, in fact, substituted with schemes for performing users' tasks. In general, the applications integration is performed on the basis of coordinated business processes models, and the integrated complex of business applications is intended to support them.

3. Definition of the Application Integration Problem for Interdisciplinary Research

Mathematical models and methods as the basis for a holistic solution should be created to power data storage and processing centres, which will provide users with versatile possibilities at the level of information integration and servers availability. The mentioned models and methods should be devoid of the drawbacks characteristic of the algorithmic approach, related to the need to reprogram data processing algorithms upon the emergence of new data types and changes in the implemented algorithms or the emergence of new ones. At the data processing level, such a solution should ensure [18,22]:

- computations distribution and the use of remote hardware resources;
- versatility and adaptation to the system load;
- system usage space;
- data supply by remote client or service;
- availability of intellectual data processing means directly to the end user with no special knowledge or skills;
- fast and simple integration of data and applications of various global information systems.

4. The Applications Integration System Architecture

The principal tenet for the creation of a distributed system is the method of organizing services interaction. Of the two known main ways of services interaction – orchestration and choreography – the more efficient for the WDC systems is the former. Indeed, orchestration that aggregates basic services into hierarchically integrated systems, subordinating them to administrators – "orchestrator" services – allows services to be unified by attributes that are convenient to WDC (science area, functionality, regional location, etc.), and to provide orchestrator services with powers in accordance with the international data exchange policies. Thus, it is possible to line up a simple and effective system, in which the search for the necessary services or the construction of their composition for the complex queries inherent in interdisciplinary research will not require a lot of time, as every Orchestrator Service has information on the functionality of inferior services, and in addition they

can coordinate their possibilities in the process of the planning and execution of users' queries. Plugging new services into the system will not present a particular problem either, as doing so will only require the service description to be laid down, entered into the registry, and assigned to a certain orchestrator service [18,22].

Such an approach will work when users know exactly their information needs and have corresponding knowledge and skills to compile chains of queries to the known orchestrators. What is more important is that the approach will provide the opportunity to work with the system for users who cannot initiate services, by determining the sequence of their work and specifying execution and interaction parameters. Users only have to know how to formulate their needs in terms of a particular subject domain. In this case, the association of services and the organization of their cooperation require an intellectual constituent. For this purpose, intellectual agents are used, which constitute the system core, implementing its functioning logic, which promotes the formation of queries, plans their execution, and organizes basic services interaction.

The introduction of intellectual agents as orchestrator services into the hierarchical structure forms a two-tier system; the bottom level consists of services performing basic tasks, the top level consists of intellectual agents that orchestrate basic services. By using the registries of subordinated basic services and their functionality and by interacting with each other, interconnected intellectual agents implement methods of logical inference. The inference result is the composition of basic agents' operations that allow user-defined tasks to be performed. This operations composition, or proof, is transferred to the lower level, where the operations necessary to solve the user's task are performed. At this point, control is handed over to the lower-level agents that only return the final result to be sent to the user.

The user's query itself is performed upon formation of the proof containing references to one or several pointers to data sources and IDs of methods that the system services should apply to the data. Some queries do not require pre-processing of data, since the execution of services methods is sufficient upon receipt of information from the data source. Others require a certain sequence of prior data operations, and as a result, the execution of other services methods. The necessary connections are described in the axioms entered into the knowledge base upon registration of respective services in the system. These operations yield the final product of the system – data that are the solution to the user's problem. Since the system operation is determined by the user's query, the system interface should help them make a query in familiar terms.

The proposed solution uses UDDI to create the service registries and WSDL for the unified description thereof. The key part – messaging – is implemented using SOAP, and services orchestration – using BPEL. Agents are described by means of JADE. The system components interaction scheme in presented in Fig. 2.

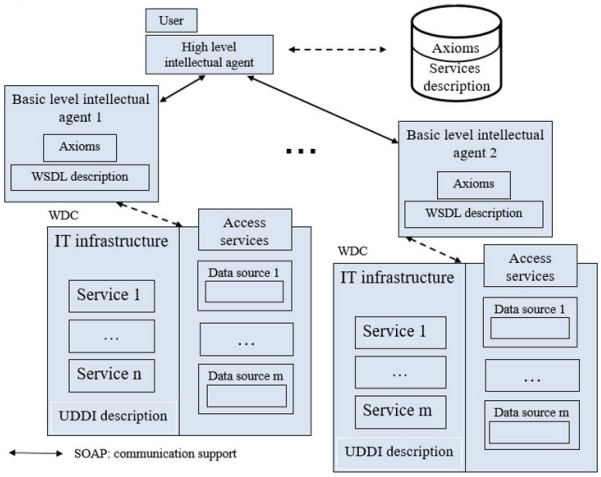

Fig. 2. System components interaction scheme

The description of the interface (WSDL) of every low-level service registered in the system and of related axioms is input into the system knowledge base and directly into the knowledge base of the Controller Agent to which the new service will be subordinated. The Controller Agent is selected in accordance with the policies adopted in the WDC system, for example, by the criteria of the geographical location of the deployed agent, in order to minimize data exchange.

At the agent level, the orchestrator agent (or several), having received the user's query, interrogates the agents in order to search for the necessary services and available resources. The user's query is processed taking into account the services of the system, their functionality (described by the axioms of the system), and the inference rules defined by logical formalism. The resulting proof is transferred to the lower level to be implemented.

At the services level, each agent invokes the required services from its set and sends processed data according to the action chains. This process is carried out in accordance with the task solution tree reconstructed by the solution tree reconstruction mechanism based on the proof.

To realize this interaction of services in the WDC system, the most appropriate is the logical approach, which allows both the level of the abovementioned requirements to be reached and the drawbacks of a traditional algorithmic approach to be eliminated. Indeed, it is only required to develop the inference method and the solution tree reconstruction mechanism that will implement the query formation processes, plan their execution, reconstruct and implement the task solution scheme. The logical approach is the most appropriate one to create and describe these constituents. The logical approach implementation requires one to:

- describe the existing applications and their functional capacities in the formal language;
- formulate the inference rules;
- determine the inference method;
- develop an algorithm of the user's query execution tree reconstruction based on the proof;
- implement these methods and mechanisms in the agents of the system.

5. The Formal Logical System

The principal tenet for the creation of a distributed system is the method of organizing services interaction [18,22,23].

Let us describe the formalism upon which the program system that will ensure solution of the formulated problem will be built. We will take the first order clausal logic for a base and describe the formal system language in accordance with the structural elements determined in [27].

Symbols:

service: (,), [,], {, }, :, <, >, ,;

constant:

1. *individual, of primary types* (int, real, char, bool) – a_1^1, a_2^1, ..., a_1^2, a_2^2, ... where each constant a_i^k pertains to type (primary type) *k; structural type* (construct) – c_1, c_2, ...; procedural type (method) – d_1, d_2, ...; objective type (problem, entity, relation) — e_1, e_2, ...;
2. *functional i-place*, for individuals of type *k* – h_1^1, h_2^1, ..., h_1^2, h_2^2, ...;
3. *predicate i-place,* for individuals of type *k* – A_1^1, A_2^1, ..., A_1^2, A_2^2, ... (this class includes taxonomic, relational and other predicates, as well as traditional relations, at least equality = and order ≥);

variable: for individuals of type *k* – x_1^1, x_2^1, ..., x_1^2, x_2^2, ..., where every variable x_i^k pertains to type *k;*

logical: $\neg, \wedge, \vee, \leftarrow, \exists, \forall, \Leftrightarrow$

Individual terms of type *k:*

1. each individual constant a_i^k of type *k* is an individual term of type *k*
2. each free variable x_i^k for individuals of type *k* is an individual term of type *k*;
3. if h_i^j is a certain functional constant for individuals of type *k* and τ_1, ..., τ_j are terms for individuals of type *k,* then h_i^j (τ_1, ..., τ_j) is an individual term of type *k;*
4. there are no other individual terms of type *k*.

The terms obtained by applying construction rules 1 or 2 of the definition will be called primary, and all the others—complex.

Formulas for individuals:

1. if A_i^j is a predicate constant for individuals and τ_1, ..., τ_j are terms for them, then A_i^j (τ_1, ..., τ_j) is the atomic formula for individuals;
2. the atomic formula for individuals is the formula for them;
3. there are no other formulas for individuals.

Hereinafter we consider that the system contains an omni-purpose transformer of formulas into the traditional for the clausal form (we are talking about the Horn clauses) view with a single → symbol, atomic formulas to its left and right, and an implicit quantifier $\forall$.

Specifiers are constructions of type τ1, where τ1 is a term for an individual object. The construct specifiers are:

if $e_1^e ... e_i^e$ are individual terms of type entity, and $e_1^r ... e_i^r$ are individual terms of type relation, and $A_1^j(a_1^1 ... a_j^1) ... A_i^j(a_1^k ... a_j^k)$ are atomic formulas for the individuals of primary types, and τ is an individual term of type construct, then τ: $(e_1^e ... e_i^e, e_1^r ... e_i^r, A_1^j(a_1^1 ... a_j^1) ... A_i^j(a_1^k ... a_j^k))$ is a construct specifier.

4. there are no other construct specifiers.

Preconditions:

1) if c_1 is a specifier of construct, and Π is a sequence of atomic formulas, then $<\tau_1:(c_1,\Pi)>$ is the elementary precondition;
2) elementary precondition – precondition;
3) if $<\tau_1>$ is a precondition, and τ_2 is an elementary precondition, then $<\tau_1, \tau_2>$ is the precondition;
4) there are no other preconditions.

Post-conditions:

1) if τ_1 is a construct specifier, and Π is a sequence of atomic formulas, then $<\tau_1:(c_1,\Pi)>$ is the elementary post-condition;
2) elementary post-condition – post-condition;
3) if $<\tau_1>$ is a post-condition, and a τ_2 is an elementary post-condition, then $<\tau_1, \tau_2>$ is the post-condition;
4) there are no other post-conditions.

Specifiers of methods:

1) if τ is an individual term of type method, and $<\tau_1>$ is a precondition, and $<\tau_2>$ is a post-condition, then τ: ($<\tau_1>$, $<\tau_2>$) is a method specifier;
2) the arity by constructs of method precondition must not be lower than the arity by constructs of method post-condition;
3) there are no other method specifiers.

Specifiers of problems:

1) if $<\tau_1>$ is a precondition, and $<\tau_2>$ is a post-condition, and τ is a term for an individual object of type Problem, then τ: $<<\tau_1>,<\tau_2>>$ is the problem specifier;
2) there are no other problem specifiers.

Clause is an expression of type Π →Λ, where Π is a sequence of atomic formulas; Λ is a single atomic formula. Depending on their type, atomic formulas will be distributed into the following clauses:

1) individual (contains only atomic formulas for individuals);
2) typified (contains only atomic formulas for types);
3) general type (contains atomic formulas for individuals and types).

The system's knowledge base consists of the ontology of axioms depicting methods of system services and the ontology of data sources described in the OWL language based on RDF. The knowledge base also includes the inference rules necessary to obtain

the desired result when it is required to combine methods of a single service or of different services [18, 22, 23].

Inference rules:

1) If $\mathbf{d}_1$: ($<\boldsymbol{\tau}_1>$,$<\boldsymbol{\tau}_2>$) and $\mathbf{d}_2$: ($<\boldsymbol{\tau}_3>$,$<\boldsymbol{\tau}_1>$) then $\mathbf{d}_1$: ($<\mathbf{d}_2>$,$<\boldsymbol{\tau}_2>$)
2) If $\mathbf{d}_1$: ($<\boldsymbol{\tau}_1>$,$<\boldsymbol{\tau}_2>$) and $\mathbf{d}_2$: ($<\boldsymbol{\tau}_3 \wedge \tau_4>$,$<\boldsymbol{\tau}_1>$) then $\mathbf{d}_1$: ($<\mathbf{d}_2>$,$<\boldsymbol{\tau}_2>$)
3) If $\mathbf{d}_1$: ($<\boldsymbol{\tau}_1 \wedge \tau_2>$,$<\boldsymbol{\tau}_3>$) and $\mathbf{d}_2$: ($<\boldsymbol{\tau}_4>$,$<\boldsymbol{\tau}_1>$) and $\mathbf{d}_3$: ($<\boldsymbol{\tau}_5>$,$<\boldsymbol{\tau}_2>$) then $\mathbf{d}_1$: ($<\mathbf{d}_2 \wedge d_3>$,$<\boldsymbol{\tau}_3>$)
4) If $\mathbf{d}_1$: ($<\boldsymbol{\tau}_1>$,$<\boldsymbol{\tau}_2>$) and $\mathbf{d}_2$: ($<\boldsymbol{\tau}_3 \vee \tau_4>$,$<\boldsymbol{\tau}_1>$) then $\mathbf{d}_1$: ($<\mathbf{d}_2>$,$<\boldsymbol{\tau}_2>$)
5) If $\mathbf{d}_1$: ($<\boldsymbol{\tau}_2 \wedge \tau_2>$,$<\boldsymbol{\tau}_1>$) then $\mathbf{d}_1$: ($<\boldsymbol{\tau}_2>$,$<\boldsymbol{\tau}_1>$)
6) If $\mathbf{d}_1$: ($<\boldsymbol{\tau}_1,\boldsymbol{\tau}_2>$,$<\boldsymbol{\tau}_3>$) and $\mathbf{d}_2$: ($<\boldsymbol{\tau}_4>$,$<\boldsymbol{\tau}_1>$) and $\mathbf{d}_3$: ($<\boldsymbol{\tau}_5>$,$<\boldsymbol{\tau}_2>$) then $\mathbf{d}_1$: ($<\mathbf{d}_2, \mathbf{d}_3>$,$<\boldsymbol{\tau}_2>$)
7) If $\mathbf{d}_1$: ($<\boldsymbol{\tau}_1>$,$<\boldsymbol{\tau}_2,\boldsymbol{\tau}_3>$) and $\mathbf{d}_2$: ($<\boldsymbol{\tau}_2>$,$<\boldsymbol{\tau}_4>$) and $\mathbf{d}_3$: ($<\boldsymbol{\tau}_3>$,$<\boldsymbol{\tau}_5>$) then $\mathbf{d}_2$: ($<\mathbf{d}_1>$,$<\boldsymbol{\tau}_4>$) and $\mathbf{d}_3$: ($<\mathbf{d}_1>$, $<\boldsymbol{\tau}_5>$)

6. The Inference Method

We use the method proposed in [27] based on analogy and types of assertions. Its idea is to build a proof in abstract space and subsequently use it to control the inference process in the initial solution search space. This will increase the inference effectiveness by cutting off most of the unpromising proof branches in the initial space.

Hereinafter a multiset of literals (atomic formulas) will be called a multiclause (m-clause), and literal L will be recorded in the m-clause as many times as it is repeated. An ordinary clause will be considered as an m-clause in which every literal's multiplicity is 1. The operations $\cup$ (unification), $\cap$ (intersection), — (difference), $\cdot$ (concatenation) and relation $\subseteq$ (occurrence) for multisets are naturally performed.

Suppose $A_1 \in C_1$, $A_2 \in C_2$ and α_1, α_2 are such substitutions that some literal L *on the right side of* A_1 and on the left side of A_2, $A_1\alpha_1 = \{L\}$, $A_2\alpha_2 = \{L\}$, and if $|A_i|>1$, i = 1, 2, the corresponding substitution α_i is the most common unifier of literals from A_1. *Then the* clause *obtained from the* unification *of* clauses $C_1\alpha_1$ and $C_2\alpha_2$ *by removing L of the left and right parts is called the m-resolvent of m*-clauses C_1 *and* C_2. Suppose all the m-clauses in the original set S are ordered, and C_1, C_2 are two of them. Suppose $A_1 \in C_1$, $A_2 \in C_2$ and α_1, α_2 are such substitutions that for some literal L *on the right side of* A_1 and on the left side of A_2, $A_1\alpha_1 = \{L\}$, $A_2\alpha_2 = \{L\}$, and if $|A_i|>1$, i = 1, 2, the corresponding substitution α_i is the most common unifier of the literals from A_i. *Then* the *clause* obtained from the *unification* of *clauses* $C_1\alpha_1$ *and* $C_2\alpha_2$ by removing L of the right part *and the last L of the left part* and by eliminating any non-*underlined* literal that is not followed by any other literal, *is called* the ordered linear m-resolvent of ordered m-*clauses* C_1 and C_2. If the last literal on the left side of the ordered m-*clause* is unified with the underlined literal on the right side of the same *clause*, the ordered linear m-resolvent can be obtained by removing the last literal from the left part of the ordered m-*clause*, i.e., *m-clause* reduction.

The next pair whose first component is a set of proof vertices and the second is a set of triples of vertices (to select components, a pair of the function-selectors s-N(Tm) and s-M(Tm) will be respectively used) is called the m-resolution proof and denoted *Tm*. Each vertex of the m-resolution proof is characterized by a mark and the depth in the proof, so that if $n \in$ *s-N(Tm), then s-L(n)* is the mark of vertex *n* and *s-D(n) is its* proof depth. If $<n_1, n_2, n_3> \in$ *s-M(Tm),* then *s-L*(n_3) is the *m-resolvent of s-L*(n_1) and s-L(n_2). Each triple of such a form is called an m-resolution. If $n \in$*s-N(Tm)* and vertex *n is* not the third component of any of the triples from *s-M(Tm),* then *n is called* the initial vertex of proof *Tm. The mark of such* vertex *is the initial* m-clause. If $n \in$*s-N(Tm)* and vertex *n is neither the first nor the second* component of any of the triples from *s-M(Tm),* then n *is called* the terminal vertex of proof and its mark *is called* the terminal m-clause. In general, *Tm* must satisfy the following restriction: if $< n_1, n_2, n_3> \in$ *s-M(Tm), then* $< n_2, n_1, n_3> \in$ *s-M(Tm).* The depth (in the proof) of each vertex n, $n \in$*s-N(Tm),* by definition:

1) *s-D(n) = 0* for any initial vertex *n;*
2) *s-D(n) = 1* + $\min\{\max\{\text{s-D}(n_1), \text{s-D}(n_2)\} | <n_1, n_2, n> \in \text{s-M(Tm)}\}$ for any non-initial vertex *n.*

If the marks of the initial vertices *Tm* belong to the set of m-clauses *S*, then m-resolution proof *Tm* is *called the* proof from *S.* C is inferred from *S* if *Tm* is a proof from *S,* while *C is* the mark of one of the *Tm* vertices. Suppose the value of function Result() is defined for *Tm* and is equal to C, if terminal clause C of proof *Tm* is the only one, i.e. Result(Tm) = *C*. Suppose Tm_1 and Tm_2 *are m*-resolution proofs. Proof Tm_1 *is called the sub*-proof of Tm_2 (and denoted $Tm_1 \subseteq Tm_2$), if the following conditions are fulfilled: *s-N*(Tm_1) $\subseteq$ *s-N*(Tm_2) and *s-M*(Tm_1) $\subseteq$ *s-M*(Tm_2). Moreover, if all the initial vertices of Tm_1 *are* the initial vertices of Tm_2, then Tm_1 *is called the* initial *sub*-proof of Tm_2.

Suppose *Tm is the m*-resolution proof from *S and all its m*-clauses *are ordered*. If for any triple $< n_1, n_2, n_3>$ from *s-M(Tm) s-L*(n_3) is *the ordered linear m-resolvent of s-L(n1)* and *s-L*(n_2), then *Tm is called the ordered linear* proof from *S. Then, by the m*-resolution proof *definition, the following condition must be satisfied*: $\max\{\text{s-D}(n_1), \text{s-D}(n_2)\} = \text{s-D}(n_3) - 1$.

In addition to the general properties of m-resolution proofs, *the ordered linear* proofs have their own features. For example, for *the ordered linear* m-resolution proof *the following conditions are fair*:

1) if $< n_1, n_2, n_3> \subseteq$ *s-M(Tm),* then $< n_2, n_1, n_3> \notin$ *s-M(Tm);*
2) for each non-initial vertex n_3 *with proof* depth r *there exists* a triple $< n_1, n_2, n_3>$, in which n_1 has the proof depth r-1, and n_2 corresponds to the initial vertex or is absent;
3) *Tm* contains a single terminal clause.

To manage an ordered linear proof, we will use the typification abstraction proposed in [26]. Suppose *f* is such mapping from the set of *m*-clauses into the set of *m*-clauses that:

1) if *m*-clause C_3 is the *m*-resolvent of *m*-clauses C_1 and C_2, while $D_3 \in f(C_3)$, then there *exist* such

$D_1 \in f(C_1)$ and $D_2 \in f(C_2)$ that the result of the substitution of some *m*-resolvent D_1 and D_2 belongs to D_3;
2) f(∅)=(∅);
3) if the result of some substitution of *m*-clause C_1 belongs to *m*-clause C_2, then for any abstraction D_2 for C_2 there exists such abstraction D_1 for C_1 that the result of D_1 substitution belongs to D_2.

Such mapping *is called f m*-abstraction mapping, while any *D* from *f(C) is called m*-abstraction. The typification mapping is understood as a certain mapping ϕ from a set of literals into a set of literals that reflects each atomic formula into the formula whose terms have the type closest to the basic types in the hierarchy. Work [8] determines that typification mapping φ is *m*-mapping.

Work [14] determines that typification mapping φ is *m*-mapping. A relationship between the *ordered linear* proofs *Tm* and *Um is called the* relationship of improvement (and denoted $Tm \rightarrow^{f} Um$), where *f* is *m*-abstraction mapping, if the vertices of proofs *Tm* and *Um are in such a* relationship of *accordance R* that:
1) the following conditions are fulfilled simultaneously: $\forall n(n \in s\text{–}N(Tm))\ \exists n'(n' \in s\text{–}N(Um))(nRn')$ and $\forall n(n \in s\text{–}N(Um))\ \exists n'(n' \in s\text{–}N(Tm))(nRn')$;
2) if nRn', where $n \in s\text{–}N(Tm)$, $n' \in s\text{–}N(Um)$, then *n* and *n′* may be initial or terminal vertices only simultaneously;
3) for terminal vertices *Tm* and *Um*, relationship *R* is a one-to-one relationship;
4) if $<n_1, n_2, n_3> \in s\text{-}M(Tm)$, $<n'_1, n'_2, n'_3> \in s\text{-}M(Um)$ and $n_3Rn'_3$, then $n_1Rn'_1$ and $n_2Rn'_2$;
5) if *n* and *n′* are initial vertices of proofs *Tm* and *Um* respectively**,** and *nRn′*, then $s\text{-}L(n') \in f(s\text{-}L(n))$;
6) if *n* and *n′* are non-initial vertices of proofs *Tm and Um respectively*, and *nRn′*, then example $s\text{-}L(n')$ belongs to *f(s-L(n)).*

So the ordered linear proofs have the property of minimality, i.e. there is exactly one terminal vertex, and if triple $<n_1, n_2, n_3>$ belongs to the minimal ordered linear deduction, then no other triple can contain vertex n_3 as the third component.

On this property of the ordered linear proofs, the authors have built the inference mechanisms implemented by agents. Once a user has formulated the problem, the system builds a proof – in fact, by checking the possibility of performing the query with existing resources. This task is performed by Orchestrator Agents as an important part of the system's intellectual kernel. With access to the list of services, their descriptions and axioms that specify services application capabilities, these agents initiate the inference process. The inference process constructs a chain of services capable of providing the user with the desired result, starting with obtaining data from relevant sources.

The ordered linear *m*-resolution proof is specified by a pair of vertices sets V = $\{k_1, k_2, k_3 \ldots k_n\}$ and vertices triples T$\{<k_1, k_2, k_3>, <k_3, k_4, k_5> \ldots <k_{n-2}, k_{n-1}, k_n>\}$, where k_i is the vertices of the tree, k_n is the terminal vertex, the result of the algorithm execution. The proof is formed based on the post-condition of the formulated problem as the initial proof vertices, and the compatible axiom from the knowledge base as the lateral vertex. The axiom is compatible if the sequence of atomic formulas matching the given proof vertex or any part thereof is the axiom post-condition. The third vertex of the triple is obtained by application of the axiom to the post-condition formula, taking account of the inference rules. The third vertex of the first triple becomes the first vertex of the second triple, and the process is repeated until the terminal vertex that corresponds to the formulated problem precondition is reached. If the atomic formula corresponding to the third vertex of the triple can be reduced by using one of the inference rules, it is reduced in the following triple that will only have two vertices – the first corresponding to the formula to be reduced, and the second corresponding to the reduced formula. To search for compatible axioms, the constructs of the currently processed post-condition are compared to the constructs of the ontology axioms. Thus, the set of axioms that describe services processing the objects of the desired type and format is selected from the ontology. The inference mechanism deducts and selects axioms only in this set. If, while searching for axioms for another vertex, several compatible axioms are found in the knowledge base, each of these axioms is used for further construction of their "parallel" versions of the proof. As a result, the inference mechanism can form several proof sets with various lengths and complexity, depending on the number of axioms in the knowledge base and combinations of their application to each vertex.

After the inference mechanism's work is over, the proof with the fewest vertices triples and the fewest applied axioms is selected from among the set of obtained proofs. Longer, cyclic, or dead-end proofs are discarded.

This set of formulas and clauses, corresponding to the proof triples vertices and reflecting the application of axioms and the rules of their construction, is the result of the inference mechanism work.

The inference mechanism work algorithm is presented in Figure 3.

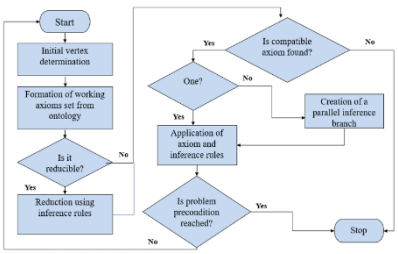

Fig. 3. Inference mechanism work algorithm

The example of inference mechanism work and solution tree reconstruction for problem P of type problem

P: <<A>,<K>>, where A and K are a precondition and a post-condition is presented in Figure 4.

Proof, a set of vertices triples: <*A*,*A*∧*A*>, <*A*∧*A*, *B*←*A*,*B*∧*A*>, <*B*∧*A*,*C*←*A*,*B*∧*C*>, <*B*∧*C*,*B*∧*C*∧*B*>, <*B*∧*C*∧*B*, *B*∧*C*∧*C*∧*B*>,<*B*∧*C*∧*C*∧*B*,*E*←*B*∧*C*',*E*∧*C*∧*B*>,<*E*∧*C*∧*B*,*D*←*B*, *E*∧*C*∧*D*>, <*E*∧*C*∧*D*,*F*←*E*∧*C*,*F*∧*D*>, <*F*∧*D*,*K*←*F*∧*D*,*K*>

7. The Mechanism of Solution Tree Reconstruction

To obtain from the proof generated by the inference mechanism a functional sequence of actions to be used by our system, taking account of services features and nature, the solution scheme reconstruction mechanism should be initiated [18,22].

The solution scheme constitutes a connected directed graph with no oriented cycles with parallel directed paths from the root to the vertices; it has three types of vertices, and is specified by triple $G=<V,E,\Theta>$, where $V=V_1\cup V_2\cup V_3$, V_1 is a set of method vertices, V_2 is a set of precondition vertices and post-condition vertices, V_3 is a set of data vertices in which data is merged or split; E is a set of edges; Θ is a subset of Cartesian product E×V×V, which determines the correspondence of edges to pairs of vertices. The scheme determines the sequence and the correspondence according to the data of actions to be performed by the data processing system's executive mechanisms to obtain the result desired by the user [18,22].

Algorithm for vertex scheme reconstruction was presented on Figure 5.

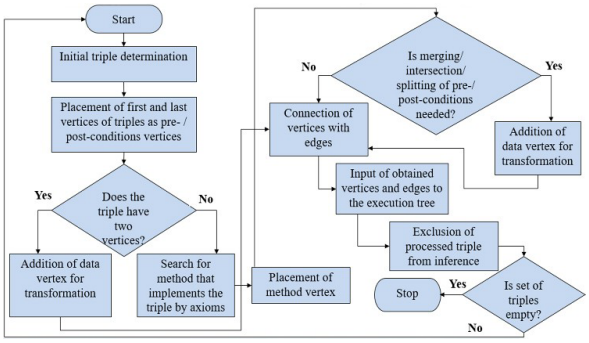

Fig. 5. Work algorithm of the solution tree reconstruction mechanism

Example of the solution tree reconstruction mechanism work for problem. Tree reconstruction by triples:

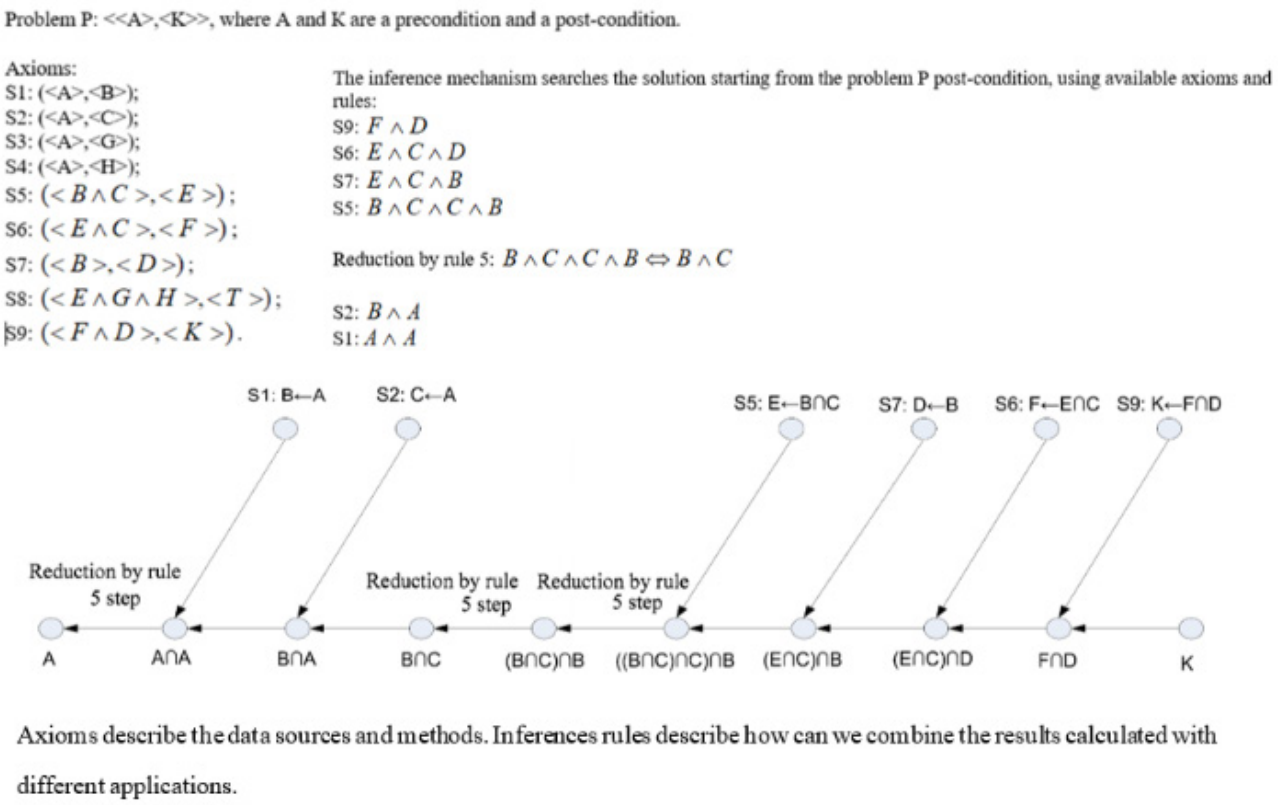

Fig. 4. The example of inference mechanism work and solution tree reconstruction

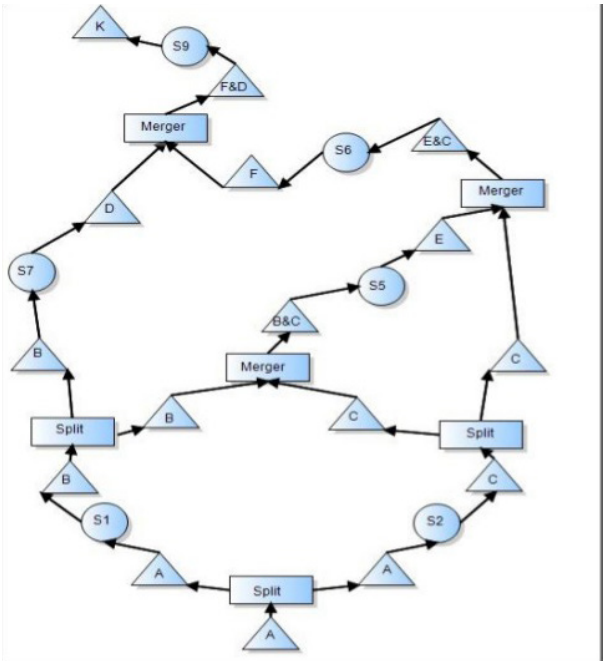

Fig. 6. The example of the solution tree reconstruction mechanism work for problem

The reconstructed solution tree is fed to the actuator input. The implementation of a particular method specified in the solution tree is represented by a construct that describes the input data (entities, connections, relations between them), method preconditions, method post-conditions, and output data. The solution tree branches downstream of the data-splitting vertex can be executed in parallel until they reach the data-merging vertex, where, after completing all the parallel branches involved in merging, they continue to be executed consecutively. The work algorithm of the solution tree reconstruction mechanism is presented in Figure 6.

The determined sequence of services performance:
S1 (<A>, <B>)
S2 (<A>, <C>)
S5 (<C∧B>, <E>)
S7 (<B>, <D>)
S6 (<E∧C>, <F>)
S9 (<F∧D>, <K>)

8. Application of Logical Approach to Problem Solution

This section is not mandatory, but can be added to the manuscript if the discussion is unusually long or complex.

The approach efficiency on the real scientific task of calculating the component of life safety and indicating critical values of threat indicators for analyzing the sustainable development of the regions of Ukraine has been demonstrated.

The life safety component formula:

$$\sqrt[3]{\sum_0^n Threat^3},$$

where the threat value is normalized by formulas (2) or (3).

The purpose of indicating critical values of threat indicators is to determine the priority in consideration thereof in the decision-making process on the level of a single region and the entire country in order to mitigate the impact of threats on sustainable development.

Suppose that for every administrative unit $i=\overline{1,n}$ there is a set of values $\langle x_{i,1},x_{i,2},...,x_{i,m}\rangle$ of indicators $X_j, j=\overline{1,m}$, which characterize the negative impact of certain phenomena on the sustainable development processes in the economic, social, and ecological spheres. Such indicators, whose essence and composition are determined by experts, will be called threat indicators.

Given the content of the critical values indication problem, the following characteristic function must be determined:

$$\Psi(x_{i,j}) = \begin{cases} 0, if\ value\ of\ x\ is\ not\ critical \\ 1, if\ otherwise \end{cases}$$

where $i=\overline{1,n},\ j=\overline{1,m}$.

It is clear that the determination of function $\Psi(x_{i,j})$ must be based upon certain criteria that take into consideration exceeding by $x_{i,j}$ a hazardous limit, the relative position of region i in the indicators rating X_j compiled for the comparison groups and for the entire country, and the degree of "hazard" of value $x_{i,j}$ in comparison to values of other indicators for region i.

To account for the relative position of the region in the entire country's ratings, the following criterion is used:

$$R_{i,j} = \left(1 + e^{\frac{a-x_{i,j}}{b}}\right)^{-1}$$

if higher values of indicator X_j correspond to a higher impact of the respective threat on the sustainable development, and:

$$R_{i,j} = 1 - \left(1 + e^{\frac{a-x_{i,j}}{b}}\right)^{-1}$$

if lower values of indicator X_j correspond to a higher impact. In formulas (2)-(3), parameters a and b are calculated by the following formulas:

$$a = \overline{X_j} = \frac{1}{n}\sum_{i=1}^{n} x_{i,j},\ b = \sigma(X_j) = \sqrt{\frac{\sum_{i=1}^{n}(x_{i,j} - \overline{X_j})^2}{n}}$$

Criterion $R_{i,j}$ is a dimensionless number that assumes values within [0,1]. Values of around 0.5 correspond to average values of X_j in the selection, and values higher than 0.75 correspond to values that exceed the average ones by more than a standard deviation. In this case, the characteristic function (1) taking account of one criterion $R_{i,j}$ may be expressed as follows:

$$\Psi_R(x_{i,j}) = \begin{cases} 0, R_{i,j} < 0,75; \\ 1, R_{i,j} \geq 0,75. \end{cases}$$

Criterion $P_{i,j}$ that takes into account a region's relative position in the comparison group may be calculated by formulas (2)-(3) taking account of the fact

that parameters a and b are calculated by formula (4) independently for each comparison group.

Criteria $R_{i,j}$ and $P_{i,j}$ are dimensionless numbers and are of similar nature and can therefore be aggregated through a weighted sum:

$$K_{i,j}=w_R R_{i,j}+w_P P_{i,j};\ w_R+w_P=1,$$

where weighting factors w_R and w_P are determined by experts.

Thus, for region $i=\overline{1,n}$ we have a set $\langle K_{i,1},K_{i,2},...,K_{i,m}\rangle$ of values of the aggregated criterion that accounts for a region's relative position in ratings compiled for comparison groups and for the entire country. Now, among values $K_{i,j}$, $j=\overline{1,m}$, the worst must be determined, which may also be performed by formula (2):

$$I_{i,j}=\left(1+e^{\frac{a-K_{i,j}}{b}}\right)^{-1}$$

where parameters a and b are calculated in selection $K_{i,j}$, $j=\overline{1,m}$.

For values of criterion $I_{i,j}$ the same remarks apply as for criterion $R_{i,j}$. Therefore, characteristic function (1) may be expressed as follows:

$$\Psi_I(x_{i,j})=\begin{cases}0, I_{i,j}<0,75;\\ 1, I_{i,j}\geq 0,75.\end{cases}$$

Thus, values $\Psi_I(x_{i,j})$ correspond to the highest priority of attention to be paid to the value of indicator X_j in the administrative decision-making process on the level of single region i.

Let us pass to the example.

Task: to calculate the indication of critical values of threat indicators for the entire country, in comparison with the group of a macro-district, and regarding a specific region.

Given: $X_{i,j}$ – a set of threat indicators values

Problem: $<<X_{i,j}>,<R_{i,j},R'_{i,j},P_{i,j},P'_{i,j},I_{i,j},I'_{i,j}>>$ where $R'_{i,j}$,$P'_{i,j}$, $I'_{i,j}$ are criteria that underwent indication.

Axioms:

Linear normalizing for the entire country: $(<D_{i,j}>,<R_{i,j}>)$

Linear normalizing for a macro-district: $(<D_{i,j}>,<P_{i,j}>)$

Aggregation of criteria $R_{i,j}$, $P_{i,j}$: $(<R_{i,j},P_{i,j}>,<K_{i,j}>)$

Linear normalizing for a specific region: $(<K_{i,j}>,<I_{i,j}>)$

Indication of critical threats values for the country: $(<R_{i,j}>,<R'_{i,j}>)$

Indication of critical threats values for a macro-district: $(<P_{i,j}>,<P'_{i,j}>)$

Indication of critical threats values for a region: $(<I_{i,j}>,<I'_{i,j}>)$

Search for a solution:

Axiom 7: $(<I_{i,j}>,<I'_{i,j}>)$

Reduction by rule 5:

$R_{i,j},R'_{i,j},P_{i,j},P'_{i,j},I_{i,j},I_{i,j}\Leftrightarrow R_{i,j},R'_{i,j},P_{i,j},P'_{i,j},I_{i,j}$

Axiom 4: $(<K_{i,j}>,<I_{i,j}>)$

Axiom 3: $(<R_{i,j},P_{i,j}>,<K_{i,j}>)$

Reduction by rule 5:

$R_{i,j},R'_{i,j},P_{i,j},P'_{i,j},R_{i,j},P_{i,j}\Leftrightarrow R_{i,j},R'_{i,j},P_{i,j},P'_{i,j},P_{i,j}$

Reduction by rule 5: $R_{i,j},R'_{i,j},P_{i,j},P'_{i,j},P_{i,j}\Leftrightarrow R_{i,j},R'_{i,j},P_{i,j},P'_{i,j}$

Axiom 5: $(<R_{i,j}>,<R'_{i,j}>)$

Reduction by rule 5: $R_{i,j},R_{i,j},P_{i,j},P'_{i,j}\Leftrightarrow R_{i,j},P_{i,j},P'_{i,j}$

Axiom 1: $(<D_{i,j}>,<R_{i,j}>)$

Axiom 6: $(<P_{i,j}>,<P'_{i,j}>)$

Reduction by rule 5: $D_{i,j},P_{i,j},P_{i,j}\Leftrightarrow D_{i,j},P_{i,j}$

Axiom 2: $(<D_{i,j}>,<P_{i,j}>)$

Reduction by rule 5: $D_{i,j},D_{i,j}\Leftrightarrow D_{i,j}$

The proof constitutes a set of vertices triples:
<D; D, D>, <D, D; (<D>,<P>); D, P>, <D, P; D, P, P>, <D, P, P; (<P>,<P'>); D, P, P'>, <D, P, P'; (<D>,<R>); R, P, P'>, <R, P, P'; R, R, P, P'>, < R, R, P, P'; (<R>,<R'>); R, R', P, P'>, < R, R', P, P', R, R', P, P', P>, < R, R', P, P', P; R, R', P, P', R, P>, < R, R', P, P', R, P; (<R, P>, <K>); R, R', P, P', K>, < R, R', P, P', K; (<K>,<I>); R, R', P, P', I>, < R, R', P, P', I; R, R', P, P', I, I>, < R, R', P, P', I, I; (<I>,<I'>); R, R', P, P', I, I'>

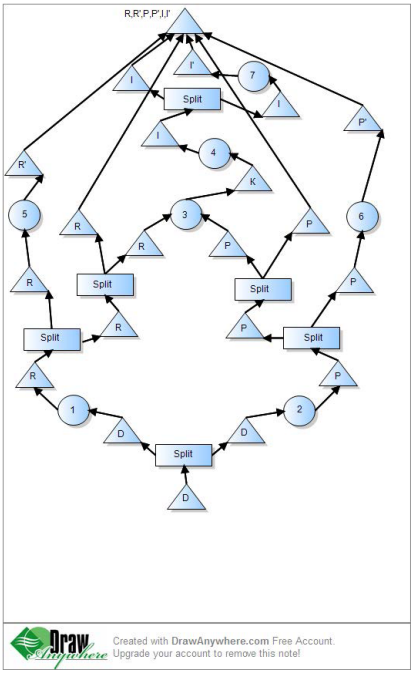

Fig. 7. The example of the solution tree reconstruction mechanism work

The execution sequence is found:
Axiom 2: $(<D_{i,j}>,<P_{i,j}>)$
Axiom 6: $(<P_{i,j}>,<P'_{i,j}>)$
Axiom 1: $(<D_{i,j}>,<R_{i,j}>)$
Axiom 5: $(<R_{i,j}>,<R'_{i,j}>)$
Axiom 3: $(<R_{i,j},P_{i,j}>,<K_{i,j}>)$
Axiom 4: $(<K_{i,j}>,<I_{i,j}>)$
Axiom 7: $(<I_{i,j}>,<I'_{i,j}>)$

9. Conclusion

Based on the analysis of existing data centres, their equipment and software, a high-quality solution is offered that provides a simple and flexible way to integrate heterogeneous information systems and their services into the World Data System. One of the key features of the proposed solution is the automation of algorithm construction of the actions sequence that executes users' queries. A logical formalism has been created to describe this solution, and on its basis an inference method and a solution tree reconstruction mechanism have been developed.

The analysis of available technologies used for the implementation of distributed systems allowed the use of such a set of software solutions for practical implementation of this solution: UDDI for creating a registry of services entered into the system; WSDL for a unified description of services; SOAP for exchanging notifications between services; BPEL for the overall coordination of services. Intellectual agents can be implemented using JADE.

The implementation of the proposed solution will provide an opportunity to use all the integrated computing capacities and data storage systems of the World Data System in a comprehensive manner. Thus, users will be able to easily gain access to all the necessary resources and services available to the system.

ACKNOWLEDGEMENTS

The research was conducted at the Faculty of Electrical and Computer Engineering, Cracow University of Technology and was financially supported by the Ministry of Science and Higher Education, Republic of Poland (grant no. E-1/2022).

AUTHORS

Grzegorz Nowakowski* – Faculty of Electrical and Computer Engineering, Cracow University of Technology, Cracow, Poland, e-mail: gnowakowski@pk.edu.pl.

Sergii Telenyk – Faculty of Electrical and Computer Engineering, Cracow University of Technology, Cracow, Poland, e-mail: stelenyk@pk.edu.pl.

Kostiantyn Yefremov – World Data Center for Geoinformatics and Sustainable Development, Kyiv, Ukraine, e-mail: k.yefremov@wdc.org.ua.

Volodymyr Khmeliuk – National Technical University of Ukraine "Igor Sikorsky Kyiv Politechnic Institute" Kyiv, Ukraine, e-mail: hmelyuk@gmail.com.

* Corresponding author

REFERENCES

[1] "About the Common Object Request Broker Architecture Specification Version 3.4," The Object Management Group (OMG), https://www.omg.org/spec/CORBA Accessed on: 2022-08-30.

[2] M. Juric, "BPEL and Java," 2021, https://www.theserverside.com/news/1364554/BPEL-and-Java Accessed on: 2022-08-30.

[3] "Data Integration Definition," https://www.heavy.ai/technical-glossary/data-integration Accessed on: 2022-08-30.

[4] "About the OMG System Modeling Language Specification Version 1.6," The Object Management Group (OMG), https://www.omg.org/spec/SysML/About-SysML/ Accessed on: 2022-08-30.

[5] "SOAP Version 1.2 Part 1: Messaging Framework (Second Edition)," World Wide Web Consortium (W3C), https://www.w3.org/TR/soap12-part1/ Accessed on: 2022-08-30.

[6] "UDDI Version 3.0.2," http://www.uddi.org/pubs/uddi_v3.htm Accessed on: 2022-08-30.

[7] "Windows Communication Foundation Architecture Overview," Microsoft Corporation, http://msdn.microsoft.com/en-us/library/ aa480210.aspx Accessed on: 2022-08-30.

[8] "Taverna – Apache Incubator," https://incubator.apache.org/projects/taverna.html Accessed on: 2022-08-30.

[9] S. A. El-Seoud, H. F. El-Sofany, M. A. F. Abdelfattah and R. Mohamed, "Big Data and Cloud Computing: Trends and Challenges", *International Journal of Interactive Mobile Technologies*, vol. 11, no. 2, 2017, 10.3991/ijim.v11i2.6561.

[10] S. Graham, *Building Web services with Java: making sense of XML, SOAP, WSDL, and UDDI*, Sams Publishing, 2005.

[11] J. Greer, *Web Services Description Language: 55 Most Asked Questions – What You Need To Know*, Emereo Publishing, 2014.

[12] M. R. Kogalovsky and L. A. Kalinichenko,"Conceptual and ontological modeling in information systems", *Programming and Computer Software*, vol. 35, no. 5, 2009, 241–256, 10.1134/S0361768809050016.

[13] J. Laznik, *BPEL and Java Cookbook: Over 100 Recipes to Help You Enhance Your SOA Composite Applications with Java and BPEL*, Packt Pub., 2013.

[14] A. Y. Levy, "Logic-Based Techniques in Data Integration". In: J. Minker (eds.), *Logic-Based Artificial Intelligence*, 2000, 575–595, 10.1007/978-1-4615-1567-8_24.

[15] P. Maslianko, "Fundamentals of the Methodology of System Design of Information and Communication Systems", *Research Bulletin of the National Technical University of Ukraine "Kyiv Polytechnic Institute"*, vol. 6, 2007, 54–60.

[16] G. Nowakowski, "Open source relational databases and their capabilities in constructing a web-based system designed to support the functioning of a health clinic", *Czasopismo Techniczne*, vol. Automatyka Zeszyt 1-AC (2), 2013, 53–65.

[17] G. Nowakowski, "Rest API safety assurance by means of HMAC mechanism", *Information Sys-*

tems in Management, vol. 5, no. 3, 2016, 358–369.

[18] G. Nowakowski, S. Telenyk, K. Yefremov and V. Khmeliuk, "The Approach to Applications Integration for World Data Center Interdisciplinary Scientific Investigations". In: *2019 Federated Conference on Computer Science and Information Systems*, 2019, 539–545, 10.15439/2019F71.

[19] S. Telenyk and O. Pavlov, "Algorithmization and IT in management," Kyiv: Technics, 2002.

[20] N. Shakhovska, M. Medykovskyj, V. Lytvyn, "Dataspace Class Algebraic System for Modeling Integrated Processes," *Journal of Applied Computer Science*, vol. 20, no. 1, 2012, 69–80, https://it.p.lodz.pl/file.php/12/2012-1/JACS-1-2012-Shakhovska.pdf Accessed on: 2022-09-11.

[21] N. Shakhovska, "Methods of Processing Data Using Consolidated Data Space", *Problems of Development, National Academy of Sciences of Ukraine, Institute of Software of NAS of Ukraine*, vol. 4, 2011, 72–84.

[22] S. Telenyk, G. Nowakowski, K. Yefremov and V. Khmeliuk, "Logics based application integration for interdisciplinary scientific investigations". In: *2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, 2017, 1026–1031, 10.1109/IDAACS.2017.8095241.

[23] S. Telenyk, G. Nowakowski, E. Zharikov and J. Vovk, "Conceptual Foundations of the Use of Formal Models and Methods for the Rapid Creation of Web Applications". In: *2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, 2019, 512–518, 10.1109/IDAACS.2019.8924416.

[24] M. Ulema, B. Wu, J. Hwang, F. Lin and J.-H. Yi, "Next generation service overlay networks (NGSON)", *IEEE Communications Magazine*, vol. 50, no. 1, 2012, 52–53, 10.1109/MCOM.2012.6122532.

[25] H. G. Miller and J. Veiga, "Cloud Computing: Will Commodity Services Benefit Users Long Term?", *IT Professional*, vol. 11, no. 6, 2009, 57–59, 10.1109/MITP.2009.117.

[26] M. Zgurovsky, O. Akimova, A. Boldak, S. Voitko, O. Havrysh, O. Gluhanyk, I. Dzhygyrey, K. Yefremov, A. Ishchenko, A. Kovalchuk, M. Kokorina, S. Lazareva, I. Makodym, T. Matorina, Y. Matsuki, A. Melnychenko, I. Pyshnohrayev, A. Prakhovnik, G. Statyukha and S. Tulchinska, "Part 2. Ukraine in Indicators of Sustainable Development (2011-2012)". In: *Analysis of Sustainable Development – Global and Regional Contexts*, 2012.

[27] M. Z. Zgurovsky, A. D. Gvishiani, K. V. Yefremov and A. M. Pasichny, "Integration of the Ukrainian science into the world data system", *Cybernetics and Systems Analysis*, vol. 46, no. 2, 2010, 211–219, 10.1007/s10559-010-9199-9.

[28] V. Ivanov and K. Smolander, "Implementation of a DevOps Pipeline for Serverless Applications". In: M. Kuhrmann, K. Schneider, D. Pfahl, S. Amasaki, M. Ciolkowski, R. Hebig, P. Tell, J. Klünder and S. Küpper (eds.), *Product-Focused Software Process Improvement. PROFES 2018, Lecture Notes in Computer Science*, 2018, 48–64, 10.1007/978-3-030-03673-7_4.

[29] C. Vassallo, S. Proksch, A. Jancso, H. C. Gall and M. Di Penta, "Configuration smells in continuous delivery pipelines: a linter and a six-month study on GitLab". In: *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2020, 327–337, 10.1145/3368089.3409709.

[30] N. Siegmund, N. Ruckel and J. Siegmund, "Dimensions of software configuration: on the configuration context in modern software development". In: *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2020, 338–349, 10.1145/3368089.3409675.

[31] S. P. Gilroy and B. A. Kaplan, "Furthering Open Science in Behavior Analysis: An Introduction and Tutorial for Using GitHub in Research", *Perspectives on Behavior Science*, vol. 42, no. 3, 2019, 565–581, 10.1007/s40614-019-00202-5.

[32] M. Auch, M. Weber, P. Mandl and C. Wolff, "Similarity-based analyses on software applications: A systematic literature review", *Journal of Systems and Software*, vol. 168, 2020, 10.1016/j.jss.2020.110669.