New Generation Computing, 30(2012)165-188
Ohmsha, Ltd. and Springer

Service Composition in Knowledge-based SOA Systems

Paweł ŚWIĄTEK, Paweł STELMACH, Agnieszka PRUSIEWICZ
and Krzysztof JUSZCZYSZYN
Institute of Computer Science
Faculty of Computer Science and Management
Wrocław University of Technology
Wyb. Wyspiańskiego 27, 50-370 Wrocław, POLAND
{pawel.swiatek;pawel.stelmach,agnieszka.prusiewicz;
krzysztof.juszczyszyn}@pwr.wroc.pl

Received 29 November 2011
Revised manuscript received 16 December 2011

Abstract In this paper, we present the solution of the problem of Service Composition in Knowledge-based SOA (Service Oriented Architecture) Systems. SOA based systems are built of services - atomic or complex ones. Each service may have a different functional and non-functional description. The methods how to retrieve atomic service and compose complex services according to the user's needs are discussed here.

Keywords: SOA, Service Retrieval, Service Composition.

§1 Introduction

SOA (Service Oriented Architecture) makes it possible to create systems, which help to work out independent business solutions from technological constraints. Information systems designed according to the SOA paradigm are composed of services, which may be executed on user's demand. Each service is an implementation of the specified business functionality which can be used in different applications or, in general, in different (business) processes.[28] Another very important feature of the SOA paradigm is its ability to compose the atomic services into complex services.[24, 38, 55]

Hence, in this work, we propose a framework for specifying complex services and interaction between them, which approach follows the transition of SOA paradigm into Service Oriented Knowledge Utilities (SOKU) that requires development of the methods for acquisition, processing and integration of knowl-

edge about services and their functionality. This approach calls for developing special representation methods of descriptive and procedural knowledge. Nowadays, ontologies are the most popular way for the representation of knowledge about services and their users. The main issue, discussed in this article, is the composition of complex services in SOA/SOKU architecture with respect for the communication and system resources available.[20, 22, 25, 26, 60]

Therefore, to solve this problem, we consider user modeling, service representation, and service composition as well as the application of ontologies in the service composition. The number of atomic and complex services increases rapidly, so users may have a great difficulty in identifying the appropriate services, especially when there are no exact matches in the service specification and we need domain ontology to specify them.[61] The same concerns the identification of communication resources which must be provided in order to execute complex services in distributed environment.

In literature, we can find many approaches to the service composition problem.[30, 41] In work[2], it is mentioned that, the composition system should start with a basic set of requirements rather than with a complete business process definition and in the first step, the whole process should be built, whereas many approaches[32] require a well-defined business process to compose a complex service. The current work often raises the problem of the semantic analysis of user requirements, service discovery (meeting the functional requirements) and the selection of specific services against non-functional requirements (i.e. execution time, cost, security). However, the presented solution has some disadvantages, i.e. these methods have not yet been successfully combined to solve jointly and comprehensively the problem of the composition of complex services that satisfy both functional and non-functional requirements. In many cases, only one aspect is considered. For example, the work[35] focuses on the service selection based only on one functional requirement at a time. Other works[14, 32, 34, 63] show that non-functional requirements are considered to be of key importance, however many approaches ignore the aspect of building a proper structure of a complex service which is the key to optimization of i.e. execution time. Many Artificial Intelligence Planning-based approaches[40] focus on functionalities of the complex service but leave no place for the required non-functionalities satisfaction.

To this date, researchers have approached the service composition from different perspectives. Some have presented specialized methods for service selection or composite service QoS-based optimization.[32] However, despite the importance of their contribution, those solutions are not widely used by other researchers. Some propose complete end-to-end composition tools introducing a concept of two-staged composition:[1] logical composition stage to prune the set of candidate services and then composing an abstract workflow. METEOR-S[2] presents a likewise concept of binding web services to an abstract process and selecting services fulfiling the QoS requirements. Notions of building complete composition frameworks are also clear in SWORD,[46] which was one of the initial attempts to use planning to compose web services. However, the proposed ap-

proaches are closed and do not support implementation of other methods therefore, it is difficult to call them frameworks. And a framework-based approach is what is currently needed in the SOA field in order to create composition approaches that are fitted to different domains and problems characteristic of them.

In this work, we propose a software framework that makes it possible to incorporate various composition approaches and the use of different knowledge repositories such as ontologies, social networks, rule engines, etc. It is open to expansion via web service enabled composition methods and aids developers in building, testing and executing elastic composition scenarios. Our aim was to enable service providers share knowledge with their customers and to support automatic composition of services into business processes within a unified software framework.

This article is organized as follows, in the next section the problems of the user modeling in knowledge-based service composition are discussed. The section three deals with the composition of services and the section five shows the application of the service composition in the telecommunication domain. The last section summarizes the presented ideas.

§2 User Modeling in Knowledge-based Service Composition

2.1 User Modeling

A user modeling is a term denoting the activity of building and updating of a user model in any computer system. The user modeling has especially been developed for the last two decades by many researchers on the different levels of a granularity.[10,19] Hence, the user model is used in many areas of our life:[11] e-learning, e-commerce, e-tourism, e-banking, etc. The user model can be achieved in a simple way, directly by asking the user some questions and fitting the user profile or by applying the complex data mining methods of knowledge discovery based on the user activity observation.[31] The systems that adapt their functionality, content, interface are called in short user-adaptive one.[3–5,43] The goal of the user modeling may be to predict the user behaviour in order to fit the application functionality and interface to the user customized requirements. Hence, the user model typically includes user properties such as preferences, interests, behaviour, knowledge and goals to deliver the adapted content, the way of application usage and the interface (Fig. 1).[9,13,31,37]

Another group of the popular systems that adapt their functionalities is called recommender systems. The recommender systems implement the methods of the product recommendation on the basis of the user model (stored in the user profile) containing the user preferences (gained from the direct questionnaire or analysis of the user interactions with the system) and the information about the user execution environment (user hardware, software, etc.).[33] The user model and the way of its obtaining, is a criterion of the recommender system classification. The taxonomy and the examples of the recommender system classification taking into account the way of building and the structure of the user profile are

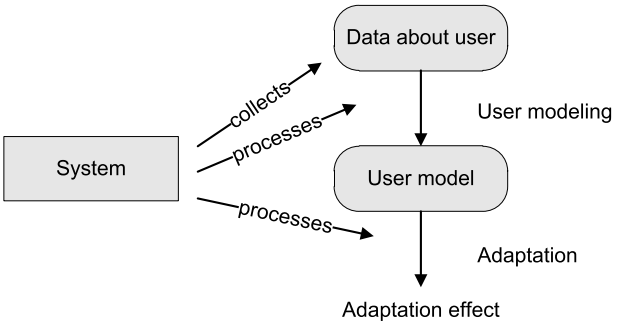

Fig. 1 User Model[9)]

given in the following works.[42, 50, 59)]

In the seventies and eighties, three main approaches of the user modeling based on natural language systems were developed: GOMS (Goals, Operators, Methods, and Selection rules) the family of models, cognitive architectures and user interface prediction models. The GOMS family of models is based on four concepts: goals, operators, methods and selection rules. The user interacts with a computer to accomplish a task that is decomposed into goals, by selecting a method, which consists of a sequence of the actions. If more than only one method is available, the user applies the selection rules. In the GOMS model, the knowledge of the user is represented as a pair: goal-action.[6)] While in the cognitive architectures originally developed by Newell in SOAR (State, Operator And Result) architectures[45)] the user model represents the user's mental models and consists of the rules that model the user's behaviour (the approach very similar to expert systems). And finally the third group of the models encompass the interaction between the user and the computer.[6)]
More recently, the researchers have proposed the user model that includes:

– user data: the information about personal user's characteristics such as: demographic data (name, address, age, sex, education etc.), user's knowledge (especially useful in any education systems), user's skills and capabilities, user's interests and preferences, user's goals and plans (specification of the information the user is interested in),
– usage data: may be 1) directly observed (actions performed by the user that indicate the user's interests and preferences), here we have: temporal viewing behaviour, ratings (products, web pages, etc.), purchase-related actions (in e-commerce systems) or 2) discovered (usage frequency, pattern behaviour - situation and action correlations, action sequences),
– environment data: the user's hardware, software, localization.[37)]

Building the model of the user of SOA based system, we use the chosen elements of the user model proposed in work,[37)] see the following section 2.2.

2.2 SOA-based User Modeling

There are many Web-based adaptive systems that use a user model to adapt their features. The most popular are the educational ones. Among them there are the systems that are based on: adaptive hypermedia (adaptive navigation support, adaptive presentation), adaptive information filtering (content based filtering, collaborative filtering), intelligent class monitoring, intelligent collaborative learning (adaptive group formation and peer help, adaptive collaboration support - coaching and supporting), intelligent tutoring (learning content selection, problem solving support).[9] The SOA based systems are the examples of the Web-based systems. In such systems, it is necessary to apply the user modeling and take into account the user preferences, with respect to the service functionalities and the execution environment or other computer system features. The problem of the user modeling in a SOA based system is quite new. In works[14,15], the authors take out the problem of the user model interoperability in the SOA based systems. We concentrate on the user modeling in one domain (not exchanging or distributing of the user model among the open, web systems) by applying the technologies compatible for the SOA. Then the user model is used in the process of the service composition and the user servicing. We assume that the service composition takes place if there is no service in a service repository that fulfils the user's functional and non-functional requirements. We base on the proposal of the user model, given in work[36] and assume that the user model consists of the following elements:

- environment user data,
- the user interests represented as a set of the concepts from the domain ontology - aggregated characteristics of the user requests,
- the user groups.

Below we discuss the problem of the user interest acquisition. The other elements of the user model are discussed in the following works.[47–49]

The user interests acquisition

We assume that the user requests a service specifying his functional and non-functional requirements. The user interests (the preferences) are obtained on the basis of the historical user request retrieval. They are divided into two categories: functional interests and non-functional ones. The user's *i-th* request (the user Service Level Agreement - SLA) at the *t-th* consists of the functional user requirements (denoted as $SLA_f(t,i)$). To obtain the aggregated user's interests, we use the Web service that as an input takes all the user's requests. In this work, we do not concentrate on the way of the user profile acquistition. To eliminate old requests, each user SLA is weighted using a linear function:

$$SLA(t,i)^{+} = f(t) \cdot SLA(t,i) \tag{1}$$

where:

$f(t) = -t + a$ is a linear function, a is a coefficient,
$f(t) \cdot SLA(t,i) = \langle f(t) \cdot SLA_f(t,i), f(t) \cdot SLA_{nf}(t,i) \rangle$.

To obtain the *i-th* user's functional interests only the SLAs for which $f(t) \geq 0$ (denoted as $SLA(t,i)^{+}$) are taken into account. At the time t_{now} the *i-th* user's interests are represented by the profile UP_f that equals:

$$UP_f(t_{now}, i) = \cup SLA_f(t, i) \tag{2}$$

2.3 Building the User Model with the Web Services

To achieve the user model, that represents any kind of the user's knowledge we propose to use the Web services. The Web services are currently seen as a solution for integration of the heterogeneous resources and making heterogeneous systems interoperable. They are self-contained, self-describing and modular applications that can be published and invoked across the Web.[62] Implementing SOA using the Web services has the advantages such as: the Web services are pervasive, simple, and platform-neutral. They provide: "a) open standards for distributed computing interface descriptions and document exchange via messages, b) independence of the underlying execution technology and application platforms, c) extensibility for enterprise qualities of service such as security, reliability, and transactions."[44] The main advantage of applying the Web services for the user model building is that building the user model becomes very flexible task and depends on the user model structure. The Web Services can be easily exchanged.

For the user modeling, we use the data mining Web services for the user classification, clustering, behavioral pattern extraction (Fig. 2). The recommendation and the personalization services are used during a user service as a result of the user service request. Generally, we assume three scenarios of the user service: a service selection from the service repository that fulfils all the user requirements, the service selection from the service repository that almost

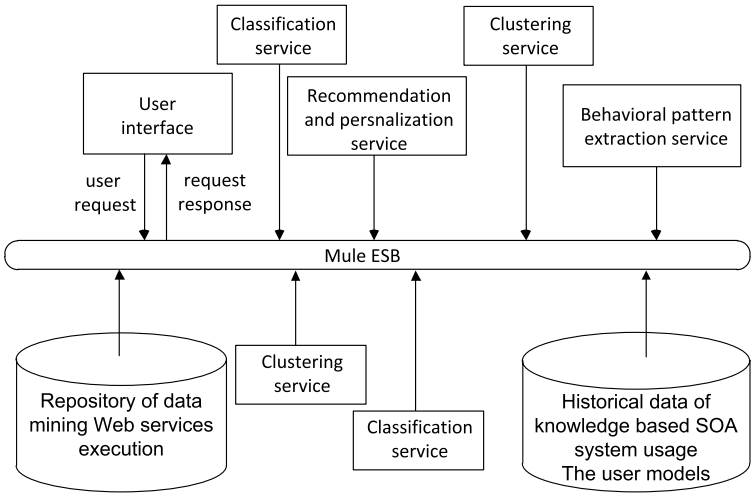

Fig. 2 The Web Services Used to Determine the User Model

satisfies the user requirements (but is acceptable by the user), or the custom-made service composition. The Web services are integrated by the Enterprise Service Bus (ESB).

The main advantage of applying the Web services to the user modeling is that they are: 1) accessible by different web clients that use the Web services via SOAP, 2) interchangeable depending on the method they implement, 3) conformable - for example if we want to group the users and we do not specify the number of the resulted group, the suitable Web service that implements k-means method will be found. Moreover the Web services may be automatically fit into the input data that must be retrieved.[49)]
For the communication between the Web services and applications (in this case responsible for the user model building) a Simple Object Access Protocol (SOAP) is used. SOAP is a protocol intended to exchange structured information in a decentralized, distributed environment.[56)]

§3 Knowlegde-based Service Composition

3.1 Service Composition Approaches

Many organizations use information technology to manage their business processes. Almost all have expanded their applications to make use of the Internet to achieve better communication with their clients and within the company. However, for many of them, this transformation was a costly step and the need for a more distributed and elastic architecture became fact determining their existence on the market. In this context, Service Oriented Architecture (SOA) is the application framework that enables organizations to build, deploy and integrate these services independent of the technology systems on which they run.[46)]
In SOA, applications and infrastructure can be managed as a set of reusable assets and services. With the use of SOA, business can respond faster to market opportunities and get more value from their existing technological assets.[35)]

However, the final success of the SOA concept can only be obtained if those service enabled applications could be effortlessly developed and integrated by many groups, both internal and external to the organization. The *composition of web services* enables SOA architects to build composite services performing advanced tasks. In this context, the tools aiding in that process are necessary to manage such complex systems with the least effort. Nowadays, in organizations, the composition process is mostly manual, however with the increasing number of available services, it is unavoidable that architects will need automated methods of selecting services that fulfil their functional requirements and work well together. Besides the obvious software and message compatibility issues a good service composition should be done with respect to the Quality of Service (QoS) requirements. For the client, preserving the non-functional requirements (availability, performance and security, etc.) is a key factor.[32, 40)]

3.2 Composition Scenario

To compose a complex service means to find a set of atomic services and

bind them together so that they, as a new service, satisfy the user's all functional and non-functional requirements. A typically automated composition process requires a semantic query (description of a complex service to be generated as an output, often referred to as an Service Level Agreement - SLA) and consists of three stages, however not all of them are necessary in every situation. It is especially important for the SLA to be defined with the use of the same domain ontology which the atomic services are described with. The three abovementioned stages are:

– building of a composite service structure,
– building of a composite service scenario,
– finding an optimal execution plan of a composite service.

As the domain knowledge is represented in the form of ontology, and the services are semantically described (in terms of associating their inputs, outputs and functionalities with the concepts, or: sets of concepts, from the ontology) the composition is a process of transforming the user's requirements defined in their SLA into a fully defined composite service that fulfils them. In the first stage of the composition process, the user's requirements are analyzed and assembled into a single structure that represents the structure of a final composite service by a graph where the nodes contain the user's requirements and the edges connect those requirements determining the order in which the final composite service will be executed. Then, with the use of engineering knowledge, this structure is enhanced so it defines a scenario of execution of particular atomic services in a composite service. In this stage, no requirement can be disconnected from other requirements. Also in this stage, the scenario is filled with candidate services that fulfill each of the requirements according to required functionalities. However, they may differ in non-functional properties (as execution time or cost) therefore in the last stage for each functional requirement, a single atomic service is selected, so that all services, that build a composite service, jointly satisfy non-functional requirements.

3.3 Service Composition Scenario

This section presents a general composition scenario and indicates that each of the stages could be performed using different methods - beginning with AI Planning methods to produce a complete scenario; then one could use different semantic selection methods when searching for services fulfiling each requirement (or even propose different distance measures for the concepts in the ontology); finally, various optimization techniques could be used to produce the composite service satisfying the non-functional requirements, not to mention that a variety of non-functional parameters could be demanded and optimized. It is possible to choose from the set of functionally equivalent complex services with respect to the non-functional requirements. The non-functional properties of the complex services are estimated on the basis of their past executions or the information about the performance of their component atomic services is available.

This all led to the necessity of designing a composition framework with

an elastic architecture that would allow composition service designers to incorporate various approaches, test them and deploy in a form of service enabled composition tools well fitted to different domains and problems. The tool consists of two main parts: one is a front end of the application and allows a business client to define their domain by connecting to external service and knowledge repositories (here: ontologies), which will be used to construct composite services; the second part is a layer of Service Composer engine services which is extensible and can provide a variety of services but mainly composition ones. Both layers of the application have a similar architecture and consist of three main elements:

- repository of composite services,
- repository of atomic services,
- knowledge repository for provided services (usually domain ontology).

The two layer Service Composer architecture is founded on following assumptions:

- the first layer is responsible for a definition of composite services and triggering their execution and composition in the second layer,
- the second layer consists of supporting engine services - similarly as the services responsible for composition, execution, finding the best instances of services in the distributed system, searching for services composed earlier, personalization etc.,
- all services (from both layers) can be directly called from an external application through SOAP protocol,
- for most of the time the second layer services should be hidden from the business user and called in various situations: directly i.e. after the composition request, by a graphical user interface while composing a composite service, by hand or by the Workflow Engine configured to call them when executing parts of a composite service.

The composite services are stored as Smart Services, described in SSDL language - it is an internal language designed specially for the composition purposes. Its expressive syntax allows for simultaneous definition of atomic services and requirements of different types in a composite service. It is developed in parallel with the Workflow Engine that supports its broad functionality and knows how to "execute" functionalities defined in the Smart Service. More information on Smart Services and the Workflow Engine can be found in the following sections.

3.4 Smart Service

The idea of Smart Services is intended to extend the concept of a composite service. A Smart Service is in fact a type of a composite service, however, its elements do not necessarily have to be atomic services. A Smart Service is represented by a graph of interconnected nodes, some of which can represent

concrete services, some sets of services and some various types of requirements (like logical or decision nodes). Many different classes of nodes can be defined in Service Composer as long as Workflow Engine is configured to interpret and "execute" them. In this respect, a concept of Smart Services fully supports the Service Composer tool as a framework for the service composition.

The SSDL language (Smart Service Description Language - Fig. 3), used to describe Smart Services, together with the Workflow Engine, allows for the complete definition of a composition scenario for a particular requirement of a composite service. The composition can also be delayed by executing the requirement nodes by the Workflow Engine (instead of composing first and executing services later), which will lead to a dynamic composition at runtime. This will be explained further in more detail.

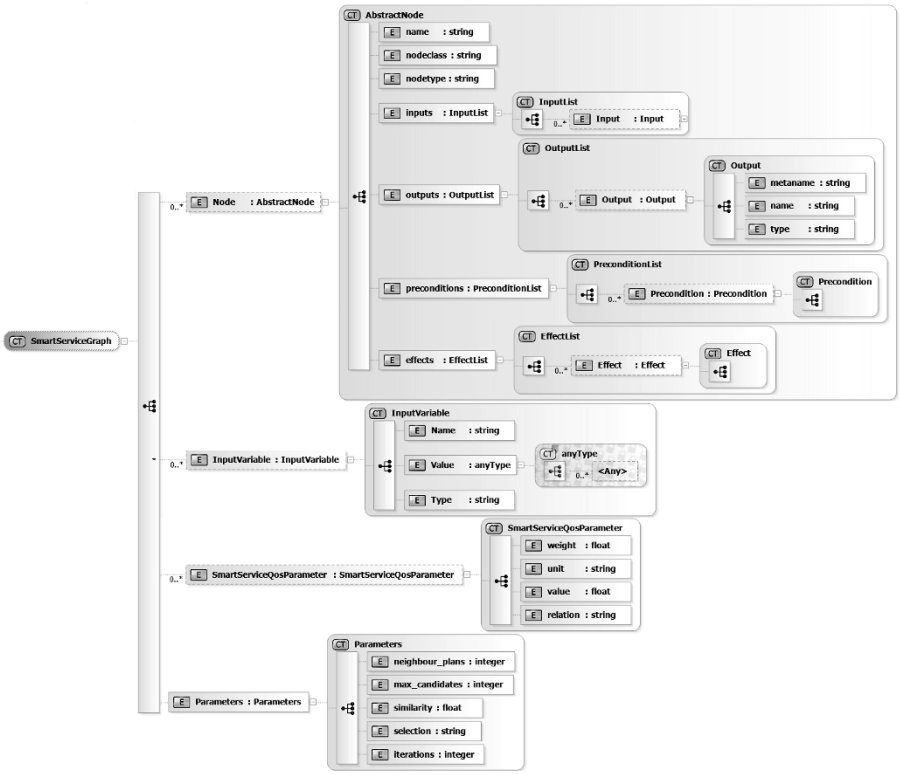

Fig. 3 SSDL Node Types Defined in XSD

The basic class describing a node of SSDL defined composite service is an AbstractNode class:

```
<xs:complexType name="AbstractNode">
    <xs:sequence>
        <xs:element name="name" type="xs:string" />
        <xs:element name="nodeclass" type="xs:string" />
```

```
        <xs:element name="nodetype" type="xs:string" />
        <xs:element name="inputs" type="InputList" />
        <xs:element name="outputs" type="OutputList" />
        <xs:element name="preconditions" type="PreconditionList" />
        <xs:element name="effects" type="EffectList" />
    <xs:sequence>
<xs:complexType>
```

The above example shows that the abstract node already contains much information about the required input and output parameters of services and the language constructs defining preconditions and effects of a service. Other types of nodes are inherited from AbstractNode class - they can describe different kinds of requirements, services, a broker agent requests etc. In fact, the list is open and limited only by the ability of the Workflow Engine to recognize those types of nodes and execute them appropriately. However, some basic types of nodes were predefined and the user defining a request in SSDL can be certain that the Workflow Engine will be able to interpret node classes described in following subsections.

Currently, functionalities (FunctionalityNode) are not much different from the basic Abstract-Node, however, for the Workflow Engine, this is sufficient information that this kind of node should be executed differently from a basic abstract node. Generally, the execution of a functionality node means that some kind of selection service should be run and then its result should be executed. Also interconnected functionality nodes make up a basic service composition request and can be an input to a variety of end-to-end composition services.

Those two types of approaches to functionality node should not be confused, because executing a FunctionalityNode is in fact a greedy approach to the composition (yet it can have its own benefits - such as the composition at runtime or using a broker service) and a typical end-to-end composition can apply completely different methods, before sending a composite service to the Workflow Engine.

Executing a service can mean various things - from a typical execution of a remote web service to executing a broker agent (as a web service) that, given an URL of a desired web service, will try to find and execute an instance of that service that is at the best fit moment for our system (least used, in vicinity in the network, etc.). The service node can also preserve the abstract description of user's requirements, and, after the error occurrence, an error handling procedure can be executed and a selection service can find another candidate to be executed in the original place.

3.5 Workflow Engine

The Workflow Engine can be configured to interpret and execute different kinds of node classes defined in SSDL. By the execution we understand performing a series of actions on the nodes content leading to obtain finally a web service to be executed in its place. Those actions can be defined by Smart Services, which are assigned to specific node classes. This again shows an elastic nature of SSDL language and framework itself, because services defined in the

Service Composer engine can serve for the composition purpose while defining a composite service with the user's graphical interface and can perform composition (as well as other execution scenarios) at runtime during the execution of a requested composite service.

Figure 4 shows that the Workflow Engine works in two phases. First, in the $\langle init \rangle$ phase, it performs validation of the SSDL service execution request and can run services transforming the whole SSDL: such as the broker agent request (which can indicate the all service instances at once) or even the end-to-end composition. Then, in the $\langle process \rangle$, the engine executes each of the nodes of the SSDL. Both in the $\langle init \rangle$ and the $\langle process \rangle$ phase, all actions, if not implemented in the Workflow Engine, can be performed by appropriate web services indicated by the Workflow Engine configuration.

The Workflow Engines' architecture allows for further extension of its capabilities. It is possible that a central repository of execution scenarios would be maintained on a local workflow engine would execute classes of nodes. Of course this would need more control over the process and in this section, is merely signaled as a potential capability to interpret various types of the user's requirements and a method for extending Service Composer capabilities with its

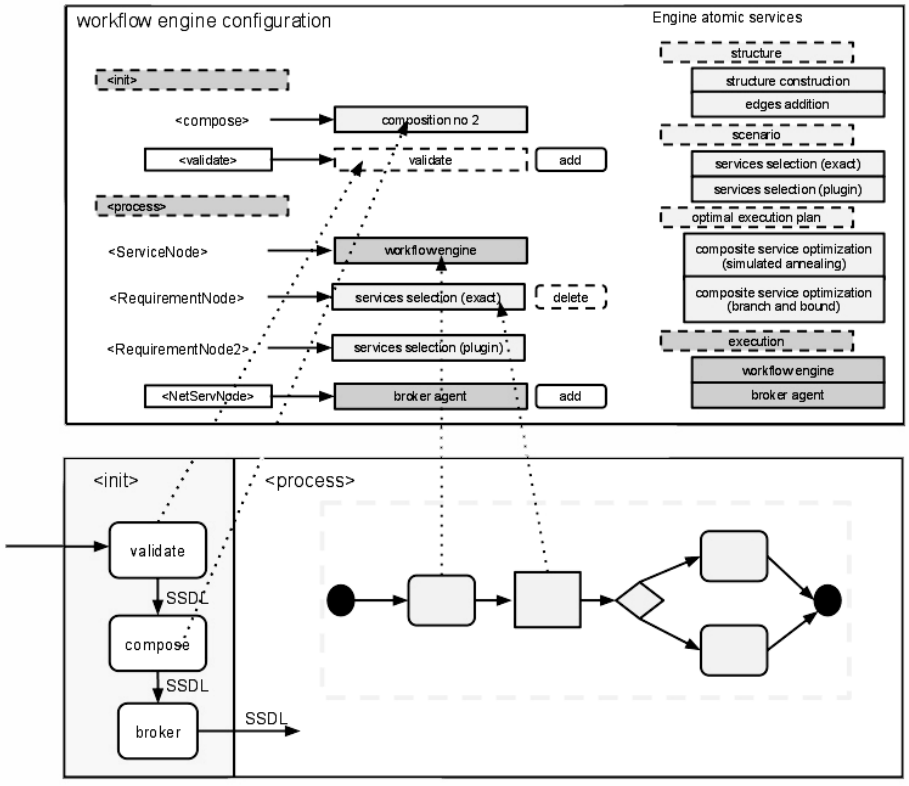

Fig. 4 SSDL Executed by the Service Composers' Workflow Engine

use by various researchers.

§4 Application of the Service Composition in the Telecommunication Domain

Recently, developing information and communication technologies (ICT) have enabled entrepreneurs to develop monolithic architectures into distributed ones. The entrepreneurs could focus on translating business processes into complex services, thus Service Oriented Architecture (SOA) becomes a crucial paradigm in designing service-oriented systems (SoS).[18)]

In SoS, the key element is a service which provides certain and well-defined functionalities and is characterized by the parameters describing the quality of a required and delivered service.[22)] Furthermore, services may be instantiated and assembled dynamically, which leads to changing structure, behavior and location of software application at run-time.[18)] However, to ensure high satisfaction of clients and profits of service providers, the services should be provided in order to ensure high quality of service (QoS).[8)]

The service composition in the telecommunication domain requires to aggregate, update, maintain and process knowledge about telecommunication services in order to analyze, plan and manage in an optimal way telecommunication resources. Moreover, the frequent behavioral patterns in business processes must be discovered for the purpose of optimization of telecommunication service delivery scenarios. The workflow of the service composition and mapping process is presented in Fig. 5.

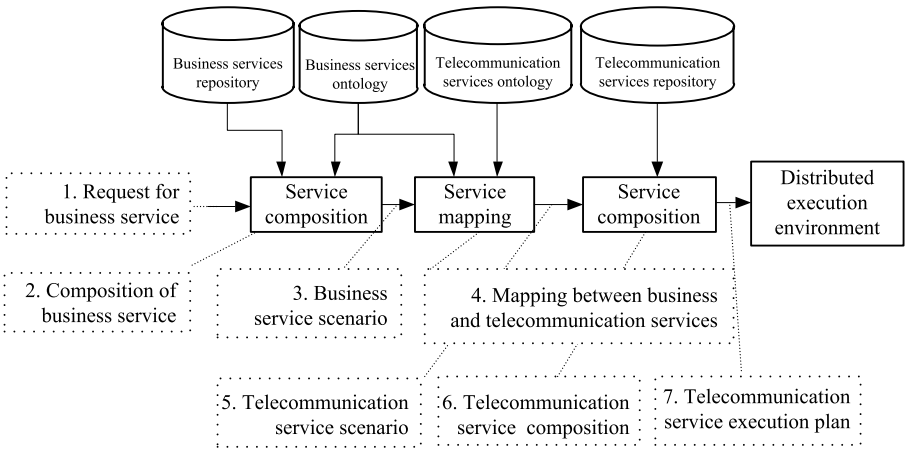

Fig. 5 Workflow of the Services Composition and Mapping Process

To compose the telecommunication services, we use three main modules in our framework:

– *Business service composition* - based on the request a business service is composed;

– *Service mapping* - business scenario is mapped with telecommunication services;
– *Service composition* - in a telecommunication scenario telecommunication services are composed and final execution plan is returned

and repositories:

– business services ontology;
– telecommunication services ontology;
– business services repository;
– telecommunication services repository.

Ontologies are representations for a seamless cooperation and knowledge flow between business processes and formal service descriptions in service-oriented systems. They play a crucial role in service mapping in translating business scenario to telecommunication service scenario.

Furthermore, two kinds of services could be distinguished, namely, computational and communication services. Ontologies, which are implemented in Protégé, are based on SIMS Ontology developed within a project of *Sixth Framework Programme*.[17] SIMS has been extended for wireline telecommunication systems, QoS concepts and parameters.

Generally speaking, computational services are used to process data streams, while communication services facilitate data transmission and end-to-end QoS guarantees. In the proposed system, there are multiple atomic or complex computational services, e.g.:

– signal merging and splitting;
– data compression;
– signal encoding/decoding;
– video/audio stream encoding;
– change detection in datastream;

Telecommunication services ontology is defined by two main types of wired telecommunication systems: access and core segments of the wired networks. The main ontology concepts are:

– Core Network Segment which consists of technologies such as ATM, Frame Relay, X.25 and multiplexing SDH and PDH,
– Access Network Segment which describes access technologies such as Ethernet, ISDN and all xDSL technics,
– Transmission medium defines wired and wireless medium for data, voice and video transmission,
– Connection and communication establishment parameters such as: bandwidth, latency, availability and BER,
– Communication service which has been extended from SIMS by all ISDN and other services.

The task is to compose complex telecommunication services which can deliver functionalities given in a business process describing all required use-case scenarios for a particular business domain. Resulting complex telecommunication services are composed basing on available atomic telecommunication services (computational and communication) provided by telecommunication service providers and/or operators. Additionally, the above complex services must conform with the users' non-functional requirements concerning, among others, quality, security and availability of the assumed business process.

The process of the complex telecommunication service composition consists of four stages: transformation of the business process into a set of complex services, complex business service composition, business and telecommunication service mapping and complex telecommunication service optimization (see Fig. 5).

In the first stage, the business process provided by the user is transformed into a set of complex services, which cover fully all use-case scenarios defined by the business process. Each complex service is defined by functionalities (atomic business services) necessary to be performed in order to fulfil the corresponding use-case scenario. Moreover, a complex service description includes non-functional requirements (e.g.: QoS, security, etc.) concerning the whole complex service and/or some of its functionalities. Each complex service is defined by a service level agreement (SLA) including, among others, functionalities and values of non-functional parameters required to be fulfiled in the corresponding use-case scenario.

Next, each complex service defined by functional and non-functional requirements is composed of available atomic business services. The result of this stage is a set of atomic business services and the order in which they must be executed in order to provide the user with the requested functionality and conforming with non functional requirements.[1)] Note that in order to deliver the requested complex functionality, some of the atomic functionalities may have to be delivered in parallel and/or in sequence. Therefore, in general, the order in which the atomic business services are performed is defined as a directed graph the nodes of which represent atomic business services and the edges represent the precedence relationship between them.[29)] Such a graph, called a complex service scenario, is passed to the next stage of the service composition.

Each node of the complex service scenario graph represents a single atomic business service which provides a well defined functionality. In order for this functionality to be delivered certain telecommunication (i.e. communication and/or computational) services must be executed. The task of the third (services mapping) stage of service composition is to substitute each of the nodes (atomic business services) of the complex service scenario graph for a subgraph representing the execution order of atomic telecommunication services necessary to deliver the requested atomic business functionality.

As an example, let us consider the atomic business functionality consisting in sending a billing information record for a certain time period being the response to a query to CRM application. This atomic business functionality is

delivered by execution of three atomic telecommunication services (see Fig. 6a), i.e.: transmitting a query to the database (communication service), execution of the query in a database (computational service) and returning the result to the user (communication service). Moreover, if a user requires a secure data transfer two additional computational services (cipher and decipher) for each communication service should be executed (see Fig. 6b).

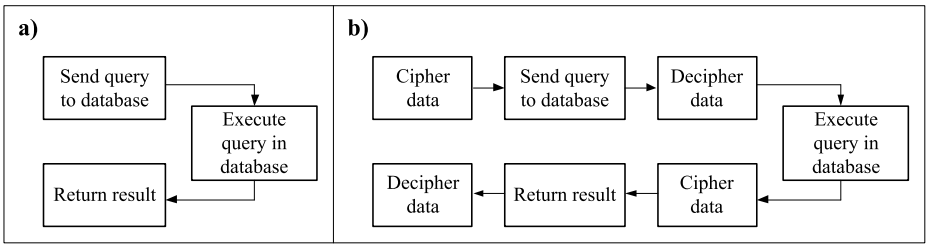

Fig. 6 Exemplary Mapping of a Billing Information Acquisition Atomic Business Service: a) Unsecure and b) Secure Data Transmission

Note that in general, two communication services, taking part in the delivery of the above exemplary atomic business service, are functionally and non-functionally different. The first one consists in sending small message (e.g. SQL query), while the other one may require the transmission of a large volume of data.

The result of the service mapping stage is a complex telecommunication service scenario graph, where the nodes represent atomic telecommunication services and the edges define the precedence relationship between services.

There are multiple approaches to accomplish the mapping stage. First of all, the mapping for each atomic business service may be predefined, which is a simple and efficient but not flexible solution. On the other hand, the mapping task can be treated as a special case of a composition task. In this case, some known methods (e.g. semantic composition[57] or GraphFold algorithm[54]) can be used. Another approach is to use predefined mapping rules and a reasoning engine to find the best possible mapping conforming with non-functional requirements.

The last composition stage is an optimization stage. The task performed at this point consists in finding such a version of atomic telecommunication service available in the system, that non-functional requirements for the whole complex service are met.[58] Additionally, certain complex service optimization tasks may be performed at this stage. These tasks may include, among others, finding the least expensive complex telecommunication service satisfying non-functional requirements or finding such a composition for which the usage of the telecommunication resources is balanced. The first optimization task may be stated by the user in the complex service request, while the latter one is rather the concern of the telecommunication resource operator or service provider. The result of this stage which is passed to the execution environment is a complex

service execution plan defining exactly which versions of telecommunication services will be used to fulfil each particular complex service request.

The first stage of service composition has to be accomplished during the ICT infrastructure design phase for each business process. The following stages of the services composition and delivery may be performed once along with the first stage during ICT system development or they can be performed on-demand each time when a certain use-case scenario is launched.

In the first approach, all complex services are predefined in the design phase and only these services can be performed during the lifetime of the system. This approach is very conservative, it does not make it possible to change or add new complex services and is very inefficient in terms of resource consumption, service reusability and composition flexibility. It allows, however, for a very quick response to new incoming complex services requests, since time-consuming process of complex service composition is performed only once.

In the second approach, on the other hand, each new incoming service request is handled separately, namely the best complex service conforming with non-functional requirements is composed taking into account a current state of the system (e.g. usage of communication and computational resources). This methodology fully draws from the service-oriented architecture paradigm allowing for the efficient resource usage, service reusability, on demand composition of new services, scalability and flexibility. Additionally, it is possible to store predefined complex service execution plans or execution plans resulting from the compositions performed during the system lifetime and apply them to the incoming complex service request when the response time is critical.[51,52)]

The composition process of an exemplary complex telecommunication service is presented in Fig. 7. The considered example concerns handling a physical alert in a laboratory monitoring a business process.

The business process of monitoring a laboratory consists of four use cases (see Fig. 7a): two concerning monitoring of environmental security and two concerning monitoring of physical security. Each use case is handled by the execution of a separate complex business service, which is predefined in a business process. In Fig. 7b an exemplary complex business service for a use case scenario concerning handling a physical alert is presented. The definition of such a service is a result of the first stage of complex service composition consisting in translating the functional and non-functional requirements of a business subprocess of physical alert handling into a sequence of atomic business tasks which have to be performed to meet these requirements.

In the next stage (see Fig. 7c), each atomic business service is mapped to a complex telecommunication service which guarantees the fulfilment of all the required functionalities. In this case, the atomic business service *stream video to monitoring center* consists of a sequence of four atomic telecommunication services: video capturing, video compression, transmission of the compressed video stream and video decompression.

In the last (optimization) stage (see Fig. 7d), particular versions of atomic telecommunication services are chosen. The choice is made based on given non-

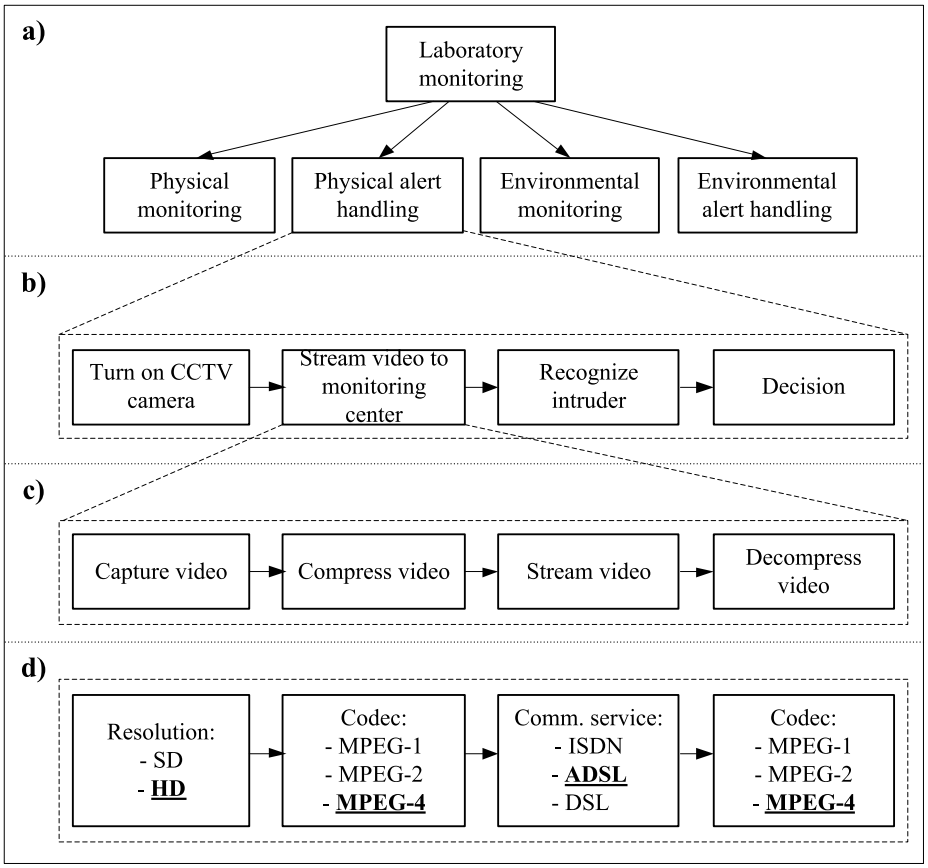

Fig. 7 Exemplary Composition of a Telecommunication Service

functional re-quirements and services availability. In this example, the streamed video is passed to the face recognition service. Therefore, a high definition (HD) video quality is chosen. Moreover, MPEG-4 codec with the highest compression rate was chosen for video encoding. Since HD video encoded with MPEG-4 results in 16Mbps one-way data stream, Asymetric DSL (ADSL) with 16Mbps upstream guarantees was chosen for a communication service. Finally, for the video decompression, the same codec as for compression must be used.

§5 Summary

Many aspects of service composition are done by hand but researchers have always tried to develop more automated approaches which would help to select atomic services, connect them accordingly and watch over data passage between them.

To compose a complex service means to find a set of atomic services and bind them together so that they, as a new service, satisfy user's all functional and non-functional requirements.

The service composition system should focus on formalization and interpretation of the functional and non-functional requirements in order to find services that satisfy them at a given time and are not hardcoded into the business logic. In this work, we presented the framework based on service oriented architecture for the service composition. This framework consists of the methods aiding to construct and deploy the end-to-end composition service. Additionally, we have introduced the concept of Smart Services which, using the SSDL language, describes its functionalities and, along with the Workflow Engine included in the Service Composer framework, interprets itself at execution. To our best knowledge, it is the first framework for service composition that implements this kind of elastic approach. Some composition frameworks described in literature were in fact complex composition methods with well-specified components, not composition frameworks to aid in composition application construction. The proposed architecture and mechanisms allow designers to incorporate a range of composition approaches into the Service Composer, personalize them for specific domains and deliver them as web services. In future work, we intend to utilize this important aspect of Service Composer by developing various methods showing the flexible nature of the framework and encouraging researchers to include composition methods of their own. The service composition is executed based on the user's functional and non-functional requirements that are represented by the SLA. The SLAs are then used to obtain the user's profile - the aggregated characteristics of the user's interests. The problem of the user modeling and the example of the user's profile discovery was discussed in section 2.

We assumed two scenarios of the user's service: 1) based on the SLA the services from the repository are selected. (see section 3), and 2) if there are no services fulfiling the user requirements - the service composition (see section 4). Then (in section 5) we presented the application of the service composition in the telecommunication domain.

The future research aims to extend the interoperability of the system and to make a flexible composition of services provided by many vendors possible. The key aspects of this issue are:

– Providing specialized services for translation of services' descriptions in OWL-S and WSDL to SSDL, in order to automate the updates of service repository.
– Developing modules for ontology management, which is inherently connected with the previous point. This will ensure the consistency of domain knowledge representation which is a key factor of successful service composition.

The first application scenario (building monitoring) will also provide the necessary data coming from living application and service repository, especially data about the service security estimates, measured QoS parameters and user's activity. All these data will be gathered in order to tune the service composition and execution processes and will serve as a basis for developing algorithms which merge semantic and non-functional (QoS-based) service composition.

Acknowledgements

The research presented in this paper has been partially supported by the European Union within the European Regional Development Fund program No. POIG.01.03.01-00-008/08.

References

1) Agarwal, V., Chafle, G., Dasgupta, K., Karnik, N., Kumar, A., Mittal, S. and Srivastava, B., "Synthy: A system for end to end composition of web services, Web Semantics," in *Science, Services and Agents on the World Wide Web In World Wide Web Conference 2005* (Semantic Web Track), *3*, *4*, pp. 311–339, 2005.
2) Aggarwal, R., Verma, K., Miller, J. and Milnor W., "Constraint Driven Web Service Composition in METEOR-S," in *Proc. of the IEEE International Conference on Services Computing 2004*, pp. 23–30, 2004.
3) Benyon D.R., Murray, D.M. and Milan, S., *Modelling users' cognitive abilities in an adaptive system*, Teddington, Middlesex, UK : National Physical Laboratory, 1988.
4) Benyon, D. and Murray, D., "Applying User Modeling to Human Computer Interaction Design," *Artificial Intelligence Review, 7, 3-4*, pp. 199–225, 1993.
5) Benyon, D., Innocent, P. and Murray, D., "System adaptivity and the modelling of stereotypes," in *Proc. of the INTERACT '87, Second IFIP Conference on Human-Computer Interaction 1987* (Shackel, B. and Bullinger, H-J. eds.), 1987.
6) Biswas, P. and Robinson, P., "A brief survey on user modelling in HCI," in *Proc. of the International Conference on Intelligent Human Computer Interaction (IHCI) 2010*, 2010.
7) Blanco, E., Cardinale Y., Vidal, M. and Graterol J., "Techniques to Produce Optimal Web Service Com-positions," in *IEEE Congress on Services 2008*, pp. 553–558, 2008.
8) O'Brien, L., Merson, P. and Bass, L., "Quality Attributes for Service-Oriented Architecture," in *Proc. of the IEEE SDSOA'07 2007*, Ohmsha and Springer-Verlag, pp. 3–9, 2007.
9) Brusilovsky, P., "Methods and techniques of adaptive hypermedia," *User Model. User-Adapt, Interact., 6, 2-3*, pp. 87–129, 1996.
10) Brusilovsky, P. and Peylo, C., "Adaptive and Intelligent Web-based Educational Systems," *International Journal of Artificial Intelligence in Education. 13*, pp. 156–169, 2003.
11) Brzostowski, K., Drapała, J., Świątek, J., "System analysis techniques in eHealth systems: a case study," *LNCS. LNAI, 7196*, pp. 74–85, 2012.
12) Brzostowski, K., Rekuć, W., Sobecki, J., Szczurowski, L., "Service discovery in the SOA system," *LNCS. LNAI, 5991*, pp. 29–38, 2010.
13) Carmagnola, F., Cena, F. and Gena, C., "User Modeling in a Social Web," *LNCS*, pp. 745–752, 2007.
14) Cena, F. and Furnari, R., "Discovering and Exchanging Information about Users in a SOA Environment," *Communication of SIWN - Systemics and Informatics World Net, 4, 3*, pp. 34–38, 2008.

15) Cena, F. and Furnari, R., "A SOA-based Framework to Support User Model Interoperability," in *Proc. of the International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems 2008, LNCS 5149* (Nejdl, W., Kay, J., Pu, P., Herder, E. eds.), Springer, Heidelberg, Hannover, Germany, pp. 284–287, 2008.

16) Charif, Y. and Sabouret, N., "An Overview of Semantic Web Services Composition Approaches," *Electronic Notes in Theoretical Computer Science, 146*, pp. 33–41, 2006.

17) Cieślak, P. and Floch, J., "SIMS - Semantic Interfaces for Mobile Services," *Priority 2.3.2.3 Specific Targeted Research or Innovation Project*, http://www.ist-sims.org/.

18) European Commission, "SIMS - Semantic Interfaces for Mobile Services," *Priority 2.3.2.3 Specific Targeted Research or Innovation Project*, ftp://ftp.cordis.europa.eu/pub/ist/docs/grids/soku-brochure en.pdf, 2006.

19) Fischer, G., "User Modeling in Human-Computer Interaction," *Proc. of the 10th Anniversary Issue of the Journal of User Modeling and User-Adapted Interaction, 11, 1/2*, pp. 65–68, 2001.

20) "From Grids to Service-Oriented Knowledge Utilities. A critical infrastructure for business and the citizen in the knowledge society," *European Commission brochure, Luxembourg: Office for Official Publications of the European Communities*, 2006.

21) Fraś, F., Grzech, A., Juszczyszyn, K., Kołaczek, G., Kwiatkowski, J., Prusiewicz, A., Sobecki J., Świątek P., and Wasilewski, A., "Smart Work Workbench : integrated tool for IT services planning, management, execution and evaluation," *LNCS. LNAI 6922*, pp. 557–571, 2011.

22) Gąsior, D., "QoS rate allocation in computer networks under uncertainty," *Kybernetes, 37, 5*, pp. 693–712, 2008.

23) Grzech, A. and Rygielski, P., "Translations of service level agreement in systems based on service oriented architecture," *LNCS. LNAI, 6277*, pp. 523–532, 2010.

24) Grzech, A. and Świątek, P., "Modeling and optimization of complex services in service-based systems," *Cybernetics and Systems, 40, 8*, pp. 706–723, 2009.

25) Grzech, A. and Świątek, P., "Parallel processing of connection streams in nodes of packet-switched computer communication systems," *Cybernetics and Systems, 39, 2*, pp. 155–170, 2008.

26) Grzech, A. and Świątek, P., "The influence of load prediction methods on the quality of service of connections in the multiprocessor environment," *Systems Science, 35, 3*, pp. 7–14, 2009.

27) Huang, A.F.M., Lan, C., and Yang S.J.H., "An optimal QoS-based Web service selection scheme, An optimal QoS-based Web services selection scheme," *Information Sciences, 179, 19*, pp. 3309–3322, 2009.

28) Hurtwitz, J., Bloor, R. and Baroudi, C., *Service Oriented Architecture FOR DUMmIES*, Wiley Indiana, 2007.

29) Janiak, A., Kozik, A., Lichtenstein, M., "New perspectives in VLSI design automation: deterministic packing by Sequence Pair," *Annals of Operations Research, 179, 1*, pp. 35–56, 2010.

30) Jinghai, R. and Xiaomeng, S., "A Survey of Automated Web Service Composition Methods," in *Semantic Web Services and Web Process Composition International Workshop 2004*, USA, pp. 43–54, 2004.

31) Johnson, A. and Taatgen, N., "User modeling," in *Handbook of human factors in Web design 2005* (Robert Proctor ed.), Lawrence Erlbaum Associates, pp. 426–453, 2005.

32) Jong Myoung, K., Chang Ouk, K. and Ick-Hyun, K., "Quality-of-service oriented web service composition algorithm and planning architecture," *The Journal of Systems and Software, 81, 11*, Elsevier, pp. 2079–2090, 2008.

33) Juszczyszyn, K. and Prusiewicz, K., "Educational services recommendation using social network approach," *LNAI 6591*, Springer, pp. 327–336, 2011.

34) Karakoc, E. and Senkul, P., "Composing semantic Web services under constraints," *Expert Systems with Applications, 36, 8*, pp. 11021–11029, 2009.

35) Klusch, M., Fries, B. and Sycara, K., "OWLS-MX: A hybrid Semantic Web service matchmaker for OWL-S services," *Journal of Web Semantics: Science, Services and Agents on the World Wide Web, 7, 2*, pp. 121–133, 2009.

36) Kobsa, A., "Generic User Modeling Systems. User Modeling and User-Adapted Interaction," *The Journal of Personalization Research, 11, 1-2*, pp. 49–63, 2001.

37) Kobsa, A., Jurgen, K. and Pohl, W., "Personalised hypermedia presentation techniques for improving online customer relationships," *The Knowledge Engineering Review, 16, 2*, Cambridge University Press, pp. 111–155, 2001.

38) Kołaczek, G., "Multiagent security evaluation framework for service oriented architecture systems," *LNCS. LNAI, 5711*, pp. 30–37, 2009.

39) Kołaczek, G. and Juszczyszyn K., "Smart Security Assessment of Composed Web Services," *Cybernetics and Systems, 41, 1*, pp. 46–61, 2010.

40) McIlraith, S. and Son T., "Adapting Golog for composition of semantic web services," in *Proc. of the 8th International Conference of Knowledge Representation and Reasoning 2002*, pp. 482–493, 2002.

41) Milanovic, N. and Malek, M., "Current Solutions for Web Service Composition," *IEEE Internet Computing, 8, 6*, pp. 51–59, 2004.

42) Montaner, M., Lopez, B. and De La Rosa, J., "A Taxonomy of Recommender Agents on the Internet," *Artificial Intelligence Review, 19, 4*, pp. 285–330, 2003.

43) Murray, D.M., "Modelling for adaptivity," in *Proc. of 8th Interdisciplinary Workshop, Informatics and Psychology, 1989*, Scharding, Austria, 1989.

44) Newcomer, E. and Lomow, G., *Understanding SOA with Web Services*, Addison Wesley Professional, 2004.

45) Newell, A., *Unified Theories of Cognition*, Harvard University Press, 1990.

46) Ponnekanti, S.R. and Fox, A., "SWORD: A developer toolkit for Web service composition," in *Proc. of the 11th World Wide Web Conference 2002*, pp. 83–107, 2002.

47) Prusiewicz, A., "Managing Web services in SOKU systems," *LNCS. LNAI 5991, part II*, Springer, pp. 57–64, 2010.

48) Prusiewicz, A. and Zięba, M., "On some method for limited services selection," *International Journal of Intelligent Information and Database Systems, 5, 5*, pp. 493–509, 2011.

49) Prusiewicz, A. and Zięba, M., "The proposal of Service Oriented Data Mining System for solving real-life classification and regression problems," *IFIP Advances in Information and Communication Technology 349* (Springer), pp. 83–90, 2011.

50) Resnick, P. and Varian, H., "Recommnder Systems," *Communications of ACM, 40, 3*, pp. 56–58, 1990.
51) Rudek, R., "The single processor total weighted completion time scheduling problem with the sum-of-processing-time based learning model," *Information Sciences*, 2012, doi: 10.1016/j.ins.2012.02.043 (in press).
52) Rudek, R., "Scheduling problems with position dependent job processing times: computational complexity results," *Annals of Operations Research*, 2012, doi: 10.1007/s10479-012-1098-1 (in press).
53) Rutkowski, L., *Methods and Techniques in Artificial Intelligence*, Polish Scientific Publishers PWN (in Polish), 2005.
54) Rygielski, P. and Świątek, P., "Graph-fold: an efficient method for complex service execution plan optimization," *Systems Science, 36, 3*, pp. 25–32, 2010.
55) Rygielski, P., Tomczak, J. M., "Context change detection for resource allocation in service-oriented systems," *LNCS. LNAI, 6882*, pp. 591–600, 2011.
56) "SOAP – Communication protocol," *W3C documentation*, http://www.w3.org/TR/soap12-part1/.
57) Stelmach, P., Grzech, A., Juszczyszyn, K., "A Model for Automated Service Composition System in SOA Environment," *IFIP Advances in Information and Communication Technology, 349*, Springer-Verlag, Heidelberg, pp. 75–82, 2011.
58) Świątek, P., Grzech, A., Rygielski, P., "Adaptive packet scheduling for requests delay guaranties in packet-switched computer communication network," *Systems Science, 36, 1*, pp. 7–12, 2010.
59) Tintarev, N. and Masthoff, J., "A Survey of Explonations in Recommender Systems," in *Data Engineering Workshop*, pp. 801–810, 2007.
60) Tomczak, J. M., "On-line change detection for resource allocation in service-oriented systems," *IFIP Advances in Information and Communication Technology, 372*, Springer-Verlag, Heidelberg, pp. 51–58, 2012.
61) Tomczak, J. M. and Świątek, J., "Personalisation in Service-oriented Systems Using Markov Chain Model and Bayesian Inference," *IFIP Advances in Information and Communication Technology, 349*, Springer-Verlag, Heidelberg, pp. 91–98, 2011.
62) "Web Services Overview," *http://publib.boulder.ibm.com/infocenter/rtnlhelp/v6r0m0/ index.jsp?topic=/com.ibm.etools.webservice.doc/concepts/cws.html*.
63) Zeng, L. and Kalagnanam, J., "Quality Driven Web Services Composition," in *Proc. of the 12th International Conference on the World Wide Web 2003*, pp. 411–421, 2003.

Paweł Świątek, Ph.D.: He received his M.Sc. and Ph.D. degrees in computer science from Wrocław University of Technology, Poland, in 2005 and 2009, respectively. From 2009, he is with Institute of Computer Science, Wrocław University of Technology, where from 2010 he works as an assistant professor. His main scientific interests are focused on services optimization and personalization, optimization of service-based systems, resources allocation, QoS delivery in heterogeneous networks and mobility management in wireless networks.

Paweł Stelmach M.Sc.: He is a Ph.D. student at the Institute of Computer Science, Wrocław University of Technology. He focused his Ph.D. thesis on the problem of automated service composition fulfilling both the functional and non-functional requirements of the user. He participates in a large European project focused on Web Services and SOA paradigm.

Agnieszka Prusiewicz, Ph.D.: She received the M.S. and Ph.D. degrees in computer science from the Faculty of Computer Science and Management, Wrocław University of Technology, Poland in 2000 and 2004, respectively. She is now an assistant professor at the Institute of Informatics, Wrocław University of Technology, Poland. Her research interests include artificial intelligence: agent and multi-agent systems, knowledge management and information retrieval: recommendation systems, machine learning approach to personalization, systems based on service-oriented architectures and Web services.

Krzysztof Juszczyszyn, Ph.D.: He works as an Assistant Professor at the Institute of Computer Science, Wrocław University of Technology, Poland. He also received his M.Sc. and Ph.D. degrees in Computer Science from this University in 1997 and 2001, respectively. He participated in a regional programme for visiting researchers at Murcia University, Spain, in 2006 and acted as an expert and reviewer in EU Framework Programmes during 2006-2008. In 2008, he served as an expert and panel secretary in the National Foresight Programme "Poland 2020." Currently he is a task leader in two large European projects concentrated on SOA architectures and Future Internet technologies. His research concentrates on dynamic network models applied to social networks based on communication technologies, local topology analysis and the semantic Web systems.