Computing and Informatics, Vol. 26, 2007, 447–464

SEMANTIC SERVICES GRID IN FLOOD-FORECASTING SIMULATIONS

Marian Babík, Ondrej Habala
Ladislav Hluchý, Michal Laclavík

Department of Parallel and Distributed Computing
Institute of Informatics
Slovak Academy of Sciences
Dúbravská cesta 9
845 07 Bratislava, Slovakia
e-mail: `Marian.Babik@saske.sk`

Revised manuscript received 15 March 2007

Abstract. Flooding in the major river basins of Central Europe is a recurrent event affecting many countries. Almost every year, it takes away lives and causes damage to infrastructure, agricultural and industrial production, and severely affects socio-economic development. Recurring floods of the magnitude and frequency observed in this region is a significant impediment, which requires rapid development of more flexible and effective flood-forecasting systems. In this paper we present design and development of the flood-forecasting system based on the Semantic Grid services [3]. We will highlight the corresponding architecture, discovery and composition of services into workflows and semantic tools supporting the users in evaluating the results of the flood simulations [15]. We will describe in detail the challenges of the flood-forecasting application and corresponding design and development of the service-oriented model, which is based on the well known Web Service Resource Framework (WSRF). Semantic descriptions of the WSRF services will be presented as well as the architecture, which exploits semantics in the discovery and composition of services. Further, we will demonstrate how experience management solutions can help in the process of service discovery and user support. The system provides a unique bottom-up approach in the Semantic Grids by combining the advances of semantic web services and grid architectures.

Keywords: Web services, grid services, workflow, discovery, composition, annotation

1 INTRODUCTION

Establishing a viable flood-forecasting system for communities at risk requires a combination of data, forecast methods, tools, as well as trained forecasters. A flood-forecasting system must provide sufficient lead time for communities to respond. Increasing lead time increases the potential to lower the level of damages and loss of life. Further, forecasts must be sufficiently accurate to promote confidence so that communities will respond when warned. If forecast become inaccurate, the credibility of the programme can be questioned and no response actions will occur. Flood-forecasting systems must be reliable and designed to operate during severe floods. The implementation of a flood-forecasting, warning and response system consists of many components. In this paper we will focus on design and development of the grid-based flood-forecasting system, which can perform very accurate predictions while providing a complex support for the user interactions.

Our flood-forecasting system is based on the *Web Service* (WS) technologies, which are nowadays gaining importance in the implementation of distributed systems, especially grids. One such example is the *Web Service Resource Framework* (WSRF) [13], which extends the current WS technologies by modeling the grid services. Design and development of the service-oriented distributed system is quite common and there are several emerging WS initiatives, which try to automate the process of discovery, composition and invocation of services. The *Semantic Web services* are a typical example, showing the potential of how ontological modeling can improve the shortcomings of service oriented computing [11].

In this paper we will present the process of design and development of the flood-forecasting simulation system. The novelty of our approach is the use of grid services and semantic technologies in the domain of flood-forecasting. Specifically, we provide a novel grid architecture, which can automatically compose and execute workflows based on the user query. Furthermore, we will present how it can support the user in his/her decisions during the course of composition and execution of the workflows based on the previous experience gained by other users. The system was fully implemented and tested during the project K-Wf Grid [15].

The structure of the paper is as follows: In the next sections, we present a brief introduction to the flood-forecasting domain and describe the underlying technologies used in the architecture, i.e. WSRF and OWL-S (Sections 3 and 4). In Section 5 we present the workflow model based on the High-Level Petri Nets. Section 6 presents our approach in mapping the workflow model to the semantic web services and the discussion of the addressed issues. In Section 7 we provide an overview of the overall architecture of the system with description of each component. We conclude with a flood-forecasting application scenario demonstrating the benefits of the system and description of related work.

2 FLOOD-FORECASTING SIMULATIONS

Flood-forecasting application aims at providing current and accurate predictions of the water level and possible flood scenarios for the given river basin or its part. The grid-based model of the forecasts for the basins of the major Slovak rivers was successfully developed and deployed during the project Cross-GRID [21]. This model was based on the simulation cascade with five major groups of methods, which have reflected the work of meteorologists and hydrologists of the Slovak Hydrometeorological Institute (SHMI). The groups mentioned cover the areas of meteorology, watershed integration, hydrology, hydraulics and visualization. Meteorological methods aim at computing the possible weather forecasting scenarios, which are then used to compute the actual watershed in the given area (e.g. MM5 and Aladin being the most common ones). The watershed is then fed into several hydrological methods (e.g. HSPF, NLC), producing the actual water level for the rivers. In case the water level is above some critical point a flood occurs and it is necessary to calculate how the actual water level influences the surrounding country. This is done by employing different hydraulics methods (e.g. DaveF, Mike). The results of each stage can be visualized to give an expert a better overview of the situation and possibility to change the existing parameters or method based on his/her experience. A sample workflow showing all the stages of the flood-forecasting is shown in Figure 1.

Although the model has provided many innovative ideas, it has also introduced new problems and obstacles:

- The model was based on the series of grid jobs, which were hard to maintain and with growing number of methods the complexity of inter-connections due to the heterogeneity of the methods has become an issue.
- The user had to compose and run the workflow manually without the possibility to store the workflows or re-use the workflows run by other users. Such feature is quite important since many flood-forecasting workflows have the tendency to be repeated many times to get many different possible scenarios, which can then be evaluated and integrated.
- Although the model introduced a portal-based user interface, it was very complex and users were many times confused with the large number of different options.
- There was no notification mechanism, which would allow the users to interact with long running workflows and take decisions during the runtime of the workflow. There was also no possibility to discuss the results of the workflows and suggest new ideas for the future runs.

Evaluation of the mentioned shortcomings has lead to the following set of requirements:

- Automated discovery, composition and execution of the flood-forecasting methods.
- Possibility to store and re-use existing workflows and their results.
- Composition of dynamic workflows within a user-friendly interface offering complex functionality.

- Support for complex user interactions, e.g. creating workflows based on queries, collaboration with other users, user as part of a workflow (user proxy).

3 WEB SERVICE RESOURCE FRAMEWORK (WSRF)

WS-Resource Framework (WSRF) is a set of specifications, which are based on the concept of modeling state as stateful resource and codify the relationship between Web Services (WS) and stateful resource in terms of a set of conventions on current (i.e. stateless) WS technologies [13]. A stateful resource is defined as having specific state data expressible as an XML document and a well defined life-cycle. It can be acted upon by one or more stateful services, e.g. files in a file system or rows in a relational database are considered a stateful resource. Figure 2 shows a sample implementation scenario for the stateful service in the globus toolkit environment (GT4) [14].

In the example a sample meteorological service called MM5 is shown. A sample stateful resource is actually a grid job of MM5 running on a PC cluster. It is defined by various resource properties, e.g. jobState, workflowID, MM5-DomainInput, MM5-TerrainData, MM5-PredictionOutput, etc. Submitting a particular MM5 job is performed by contacting a service factory, which creates an instance of the resource (binding of service and particular resource) and returns an identifier, i.e. so called EPR (end-point reference). The client can interact with the given resource by calling the methods exposed by the service instance interface, i.e. configureFromProperties, start, abort, etc. The pairing of the service instance and the resource is called a *WS-Resource*. The normal usage of the MM5 WS-Resource is to setup the resource properties, call the configureFromProperties method and then call the start method, which submits the job to a PC farm. The actual resource instances are homed by the Resource home, which can be implemented as either persistent or transient store. Detailed description of the WSRF is beyond the scope of this paper; for additional details see [13].

During the project we have designed and developed a network of loosely coupled grid services representing the flood-forecasting methods. The services were deployed into an experimental testbed used for evaluation and testing of the possible flood-forecasting scenarios. A more detailed overview of the design and development can be found in [1].

4 WEB ONTOLOGY OF SERVICES (OWL-S)

OWL-S is an ontology-based approach to the semantic web services [11]. The structure of the ontology consists of a *service profile* for advertising and discovering services, a *process model* which supports composition of services, and a *service grounding*, which associates profile and process concepts with underlying service interfaces (see Figure 3). Service profile (*OWL-S profile*) has functional and non-functional properties. The functional properties describe the inputs, outputs, pre-

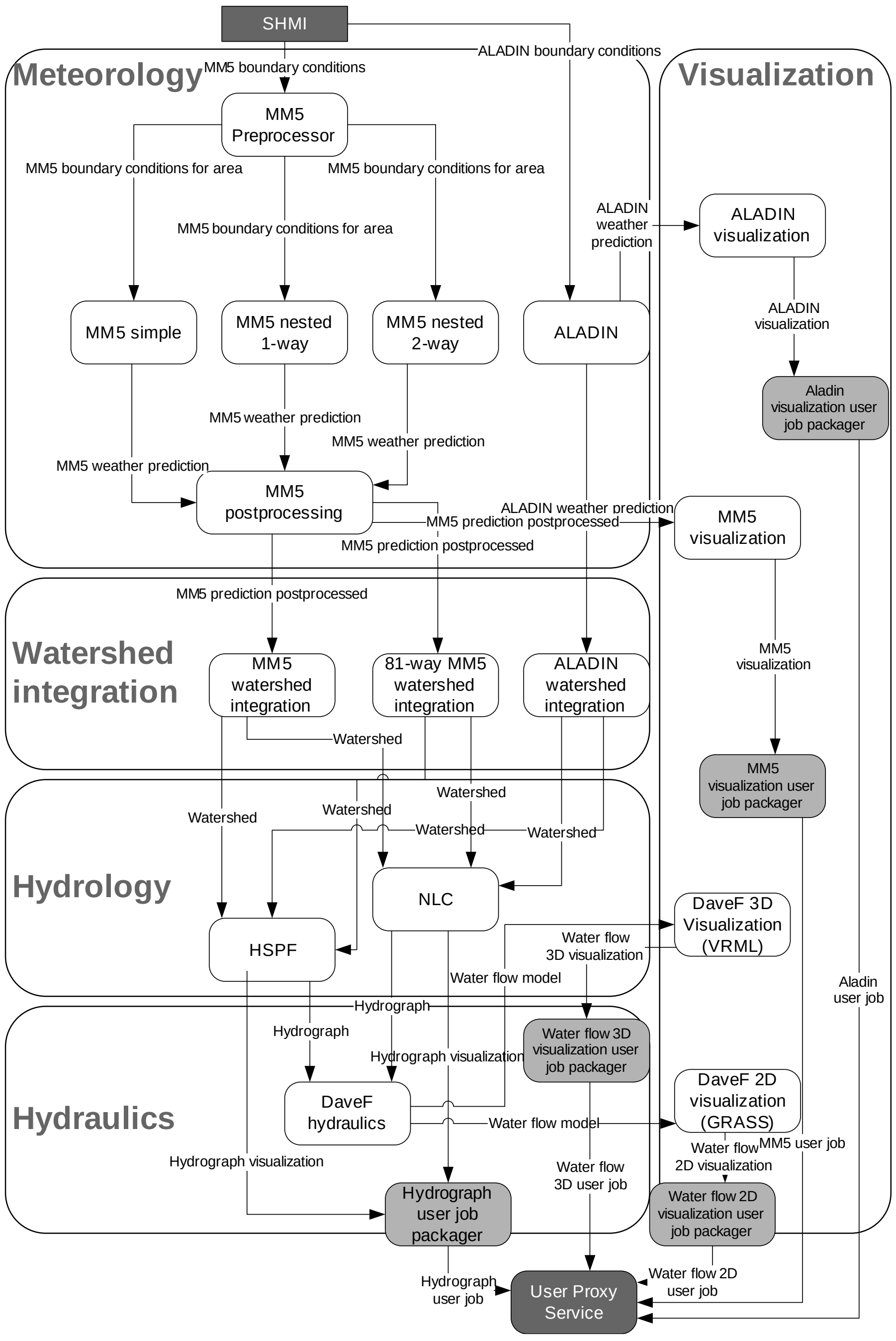

SHMI
ALADIN boundary conditions
Meteorology
MM5 boundary conditions
Visualization
MM5 Preprocessor
MM5 boundary conditions for area
MM5 boundary conditions for area
MM5 boundary conditions for area
ALADIN weather prediction
ALADIN visualization
MM5 simple
MM5 nested 1-way
MM5 nested 2-way
ALADIN
ALADIN visualization
Aladin visualization user job packager
MM5 weather prediction
MM5 weather prediction
MM5 weather prediction
MM5 postprocessing
ALADIN weather prediction
MM5 prediction postprocessed
MM5 visualization
MM5 prediction postprocessed
MM5 prediction postprocessed
Watershed integration
MM5 watershed integration
81-way MM5 watershed integration
ALADIN watershed integration
MM5 visualization
MM5 visualization user job packager
Watershed
Watershed
Watershed
Watershed
Watershed
Watershed
Hydrology
NLC
HSPF
DaveF 3D Visualization (VRML)
Water flow 3D visualization
Water flow model
Aladin user job
Hydrograph
Hydrograph
Hydrograph visualization
Water flow 3D visualization user job packager
Hydraulics
DaveF hydraulics
Water flow model
DaveF 2D visualization (GRASS)
Water flow 2D visualization
MM5 user job
Hydrograph visualization
Water flow 3D user job
Hydrograph user job packager
Water flow 2D visualization user job packager
Hydrograph user job
User Proxy Service
Water flow 2D user job

Fig. 1. A complete data flow diagram of the K-WfGrid's flood application

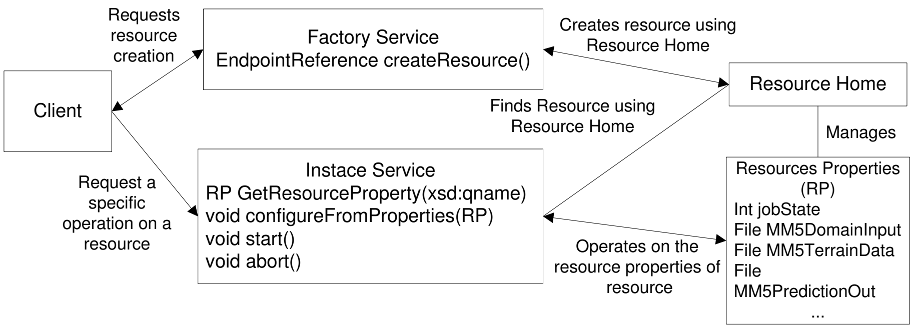

Client
Requests resource creation
Factory Service
EndpointReference createResource()
Creates resource using Resource Home
Resource Home
Finds Resource using Resource Home
Manages
Request a specific operation on a resource
Instace Service
RP GetResourceProperty(xsd:qname)
void configureFromProperties(RP)
void start()
void abort()
Operates on the resource properties of resource
Resources Properties (RP)
Int jobState
File MM5DomainInput
File MM5TerrainData
File MM5PredictionOut
...

Fig. 2. Sample implementation scenario for the WS-Resource

conditions and effects (IOPE) of the service. The non-functional properties describe the semi-structured information intended for human users, e.g. service name, service description, and service parameter. Service parameter incorporates further requirements on the service capabilities, e.g. security, quality-of-service, geographical scope, etc. Service grounding (*OWL-S grounding*) enables the execution of the concrete Web service by binding the abstract concepts of the OWL-S profile and process to concrete messages. Although different message specifications can be supported by OWL-S, the widely accepted *Web Service Description Language* (WSDL) is preferred [12]. In the following we will denote the WSDL operations as O_i and input, output messages as M_i^in, M_o^iut.

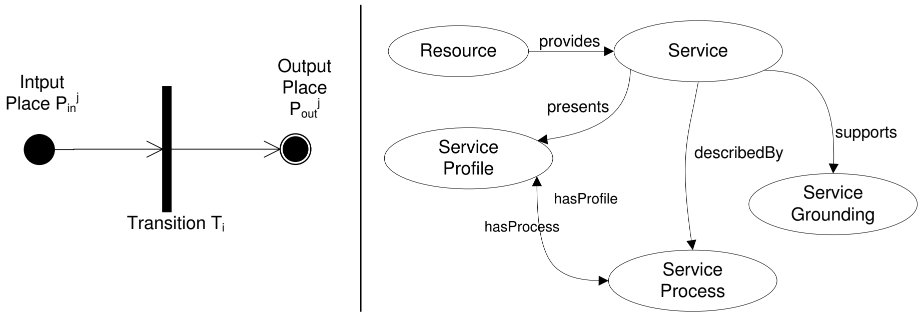


Fig. 3. Schema of the workflow model (left). Basic concepts of the OWL-S ontology (right)

5 WORKFLOW MODEL

The services and the middleware for discovery, composition and execution is based on the Web Service Resource Framework (WSRF), a recent standard for grid services.

Since we do not rely on the single service, but on the set of services, it is necessary to describe the application in terms of the *workflow*, which represents a control and data flow of the corresponding services. The representation of the workflow should be easy to use and intuitive, however it should also be able to cope with the complexity of the Grid, i.e. large number of services, dynamic scheduling, automatic handling of resource properties, etc. Although OWL-S provides OWL-S Process, we have decided to replace this part of the OWL-S as it provides a limited expressiveness in terms of loops and implicit parallelism, which makes it difficult to use and also lacks a clear and understandable graphical presentation. However, it would be possible to extend OWL-S Process to cover these areas based on our approach.

We have relied on the workflow model based on the High-level Petri Nets, which allows to compute the output tokens of a transition from the input tokens [20]. Service operations are represented as transitions as shown in Figure 3. Each service operation is represented by the *transition* T_i, denoted by a thick vertical line. The variables of the formal parameters are represented as input, output *places*, P^i_{in}, P^i_{out}, shown as empty circles. Each place can hold a number of *tokens*, which represent the data items of the parameters. The input and output parameters of the operation are shown as input and output *edge expressions*, respectively. Additionally each transition can have a set of Boolean *condition* functions. Firing transition and thus calling the service operation is only allowed if all of its conditions evaluate to true.

Given the individual service representation, we can describe the complex workflows by connecting the output places of one transition with input places of another transition. Additionally, it is possible to introduce more complex conditions to realize standard control flow structures such as conditions and loops. All of the mentioned concepts and relations are defined in the form of an XML-Schema as Grid Workflow Description Language (GWorkflowDL). Detailed description of the GWorkflowDL and High-level Petri Nets is beyond the scope of this document and can be found in [20].

6 SEMANTICS

6.1 Overview

In order to support the complex user interactions, automated discovery, composition and execution of service operations, it is necessary to provide a mechanism for semantically describing the set of services, users, application domains and grid environment. Semantic descriptions together with related Semantic Web technologies can then allow to automatically discover the services based on the user query, compose the services into the workflow, optimize the execution of the service operations and present the results in clear and understandable way. Further, it can support collaboration among users, possibility to store and re-use the important results and workflows gathered by previous runs.

The knowledge captured by the semantic descriptions in the form of ontologies has two major areas: semantic description of services allowing to discover, compose and execute the workflows and semantic description of user context and experience. In the area of semantic service descriptions we have used the upper ontology of services OWL-S [11]. In the area of the user context and experience we have relied on the ontological descriptions and mechanism defined by the knowledge and experience management [22]. In the next section we will describe the service descriptions and its mappings to the workflow model as well as the issues we faced during the extension of the OWL-S standard to the grid services. In Section 6.3 we will present our approach in describing the system context and experience.

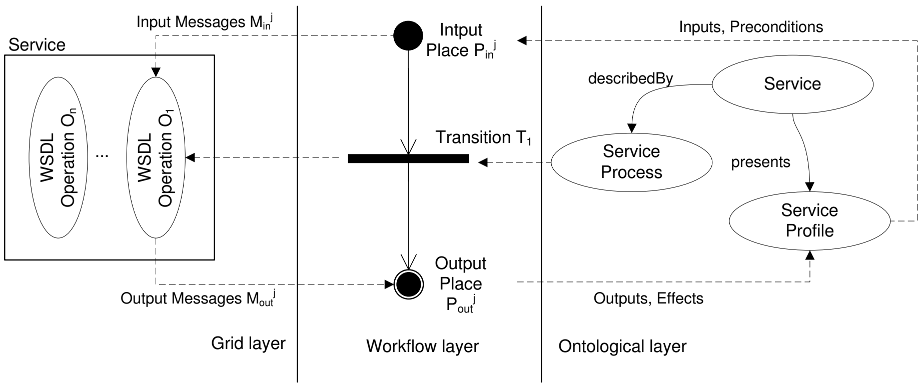


Fig. 4. Layers of the service description with corresponding mappings

6.2 Semantic Description of Grid Services

Our semantic description of the services has three layers: *grid layer*, represented by the concrete WSDL operations and messages, *workflow layer*, represented by the GWorkflowDL elements and *ontological layer*, represented by the OWL-S and OWL ontologies. The corresponding mapping between grid and workflow layer is quite straightforward. Each workflow transition T_i represents one WSDL operation O_i, whose input/output messages $M^i_{in/out}$ are mapped to input/output tokens $P^i_{in/out}$. Ontological layer provides semantic description of services in terms of service *profile* and *grounding*. Service *process* is atomic thus leaving the workflow description to the *workflow layer*. Service *profile* functional properties, i.e. inputs, outputs, preconditions and effects, are described by the corresponding flood-forecasting domain concepts, e.g. river, basin, location, watershed. Non-functional properties are described by service name, service providers, quality of service metrics and service classes (hierarchy of services). Service *grounding* maps the functional properties to the actual WSDL messages. Since we already have such information in the workflow layer, we provide service *grounding* descriptions only for compatibility purposes. Mapping the WSDL operations is not necessary, since we provide semantic descriptions per service operation. This means that based on the service *profiles* we can infer whether it is possible to connect inputs and outputs of two services and thus create a workflow. Since OWL-S is based on the OWL we can use any OWL-S capable planner for this task. Another benefit of such approach is that we have a direct mapping between workflow transitions and service *profile*. Figure 4 shows the three layers and the corresponding mappings between them.

Grid services, i.e. WS-Resources, are composed of a service and a stateful resource [13]. WSRF specification defines a stateful resource as a single XML Global Element Declaration (GED) in a given namespace. Such GED defines the type of the stateful resource and is motivated by the modeling of complex objects for stateless services. For modeling semantics of the stateful resources this means there are no major differences between grid and web services. There are, however, few important issues, that should be noted.

Resource properties (RP), as defined in the WSRF specification, can be dynamic. This means that it is possible to create and destroying properties on the fly, i.e. if stateful service is providing access to the filesystem and resource properties are listing the file attributes, it is possible to add or remove a file attribute at any given time. In order to model such dynamic behavior by corresponding semantics, it is necessary to consider more advanced techniques for ontological mapping and concept definition. Since such procedures can become quite complicated, we have concentrated our work on the area of static resource properties, i.e. it is possible to change the values of the properties, but the set of properties for given resource remains constant over time.

In the process of designing the stateful services it is possible to use inheritance of the resource properties. Although explicit hierarchy of the resource properties can help in the generation of the semantics, there is no standard, which would describe in details how RP inheritance should be implemented. This can cause major difficulties in parsing of such services and it is necessary to introduce special parsers to extract the RP hierarchies. Furthermore, resource properties can often be used to model the actual inputs and outputs of the service, i.e. a service submitting specific jobs to the cluster can represent the inputs and outputs as properties of the job, thus hiding the inputs/outputs in the resource properties of the service. This has to be considered then in the composition of the services.

Apart from the issues in the process of service discovery, there are also slight differences in the process of service invocation. The WS-Resource is composed of a service and a stateful resource, i.e. it is identified by the so called EPR (end point reference), which describes not only service address but also the identification of a resource. The service identification in the OWL-S Grounding has to be then extended to a more complex structure. For the grid services such extension can also consider the possibility of having multiple instances of the same service hosted by different servers. This can, however, be solved simply by introducing multiple OWL-S Groundings.

6.3 Semantic Description of User Context and Experience

Capability to store and re-use knowledge is presented in several system requirements such as possibility to store and re-use previous workflows and results, guide the users during the workflow composition according to previous experience or creating workflows based on user queries. These requirements reflect the fact that in the system with many services and users, everyone can benefit from the previous cases experienced by others. An example of such functionality is creation of a workflow from a simple query such as, flood-forecasting for tomorrow for the basin of Danube river. The answer to such query is a list of services together with the list of notes describing each service an its suitability for the requested area and time. Upon selection of the concrete service the system should be able to either compose the workflow or show all fitting workflows computed previously by other users. It should also suggest the location of available results for the given query. In order to provide such functionality it is necessary to find a suitable technology, which can semantically describe the user context and experience.

Semantic description of user context and experience allows the system to determine similarity and relations among different contexts experienced by other users. This is the domain of the knowledge and experience management [22, 28], where such functionality is defined as a capability to collect lessons from the past. Experience of the user is modeled as a Case-Lesson pair (C, L), where C denotes the *Case Space* and L denotes the *Lesson Space*. The Case Space C is usually represented as a set of vectors c, whose elements determine the user context. The Lesson Space L is a set of notes (i.e. free text), workflows and results, which have the form of some global identifier, e.g. URI. The elements of the user context vector c are also just instances of the ontological concepts, i.e. services or instances of the domain concepts such as concrete time, city, basin, etc.

Given the Case-Lesson pair database and the current user context vector c_u, the system can compare its similarity to the previous cases and list the most relevant Lessons L to the user. The similarity of the two vectors c, c_u is determined by using an ontological similarity metric based on similarity graphs [26, 27].

The actual build-up of the current user context vector c_u is based on user queries and actions in the user interface, e.g. user query: "Flood-forecasting for tomorrow in Bratislava" would result in the context having the following instances $(DaveF2DVisualization, Bratislava, Danube, 241206)$; the vector contains instances of the concepts Service, City, River, Date, which were determined from the user query. As previously described, evaluating the current user context vector c_u results in the list of lessons from previous cases. Based on the user actions in this list, the user context can be further extended/updated, e.g. by clicking on the note describing the service *DaveF2DVisualization*; the system can present other similar services and update the context based on next user decisions. If the user is interested in the concrete workflow or result, its URI is then used to enact the workflow engine and either run the workflow or present its results. A special case is the possibility to compose the workflow based on the identified service (as was described in Section 4).

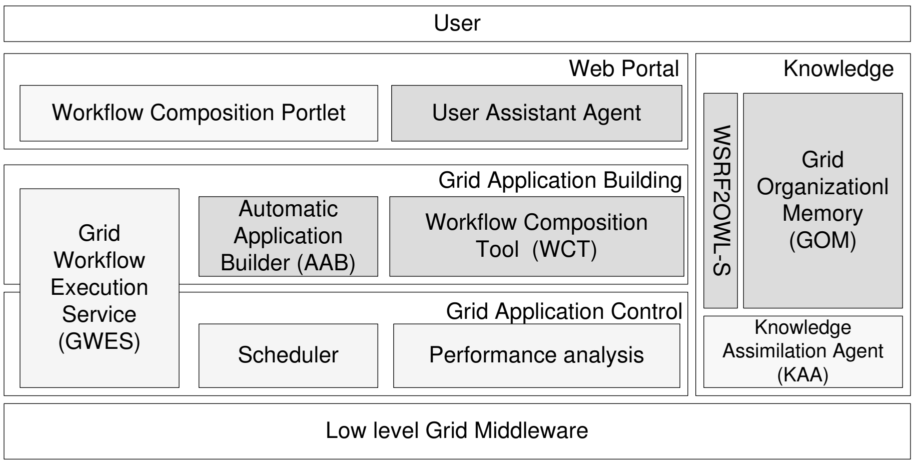

User
Web Portal
Workflow Composition Portlet
User Assistant Agent
Knowledge
WSRF2OWL-S
Grid Organizationl Memory (GOM)
Grid Application Building
Grid Workflow Execution Service (GWES)
Automatic Application Builder (AAB)
Workflow Composition Tool (WCT)
Grid Application Control
Scheduler
Performance analysis
Knowledge Assimilation Agent (KAA)
Low level Grid Middleware

Fig. 5. A simplified schema of the K-Wf Grid architecture

7 K-WF GRID ARCHITECTURE

Figure 5 presents the overall architecture of the system components of the workflow orchestration and execution environment described in the previous sections. The main user interface for developing semantic-based Grid applications is the *User Assistant Agent* (UAA)[26], which contacts the *Grid Workflow Execution Service* (GWES) [20], that manages the process of composing and executing the services. The automated semantic service composition is partly delegated to the *Automatic Application Builder* (AAB), the *Workflow Composition Tool* (WCT), and the user (by means of the UAA)[1]. The *Automatic Application Builder* and the *Workflow Composition Tool* are knowledge-based semi-automatic modeling services, which in cooperation with the *User Assistant Agent* can propose known solutions to problems solved in the past [25, 19]. The semi-automatic composition of the services is enabled by the semantic description of the grid services, which is created and maintained by the *Grid Organizational Memory* (GOM) and supporting tools [24][2]. When parts of the workflow are ready to be executed on the Computing Grid, GWES asks the Scheduler for the optimal resource, due to some user-defined metrics. Then, the corresponding Web Service operation is invoked remotely on the Grid middleware using WSRF protocols. The events triggered by the workflow orchestration and execution will be published by means of an event system. The Knowledge Assimilation Agents (KAA) [23] consume these events and generate knowledge that

[1] Grid Workflow Execution Service and Workflow Composition Tool are available at: `http://www.gridworkflow.org`

[2] Grid Organizational Memory and WSRF2OWLs tool are available at: `http://gom.kwfgrid.net/web/` and `http://www.tuke.sk/fei-cit/babik/wsrf2owls/`

is stored in the Grid Organizational Memory (GOM). This knowledge will later be reused by the components of the workflow orchestration and execution environment.

Workflow Composition Tool (WCT) provides the functionality of composing abstract workflows of Grid applications from simple user requirements [19, 20]. It employs semantic reasoning techniques over OWL-S descriptions (i.e. subsumption, classification) and it tries to propose a solution to the user's problem by using provided descriptions of available resources. The basis for the composition is the model and corresponding mapping described in Section 6.2. The composed workflow is executed by the GWES, which uses the workflow model described in Section 3. During the composition and refinement of the workflow the User Assistant Agent is used to guide the user according to the experience it gained in the past compositions. UAA and KAA are both based on the semantic description of user context and experience, cf. Section 6.3[3].

8 APPLICATION SCENARIO

In this section we present a scenario which showcases the functionality described in previous sections[4]. The scenario is hosted by the K-WfGrid user portal, which was developed on top of the well known GridSphere portal [2]. As shown in Figure 6, the portal has only three main components: *control panel, user assistant and workflow panel.* The *user assistant* guides the user throughout the composition and execution of the workflow, which is shown in the *workflow panel.* The workflow can be started, stopped or suspended at any time by using the *control panel*, which also provides a list of tasks that the user needs to accomplish in order to make progress (e.g. provide input data).

The system allows the user to type a free text query, e.g. flood forecasting in Bratislava. The *user assistant* parses the query and determines the user context, e.g. in this case the list of services appropriate for the region and concepts reflecting the location and basin. This context is presented as a list of clickable concepts, which the user can select. Based on the user selection the *user assistant* displays the list of relevant lessons, e.g. past workflows, results and notes submitted by previous users. The next steps vary based on the user actions. The user can browse through the existing notes, workflows and results to see if someone has already computed any similar results or workflows. Based on the user selection the system will display the results and notes describing each service. By clicking on the workflow the system will load it into the *workflow panel* and the user can restart it with his/her data by using the *control panel.* Generally, the process of user guidance is based on the semantic descriptions of user context and experience as described in Section 6.3.

[3] User Assistant Agent and Knowledge Assimilation Agent are available at: `http://ikt.ui.sav.sk/?page=software.php`

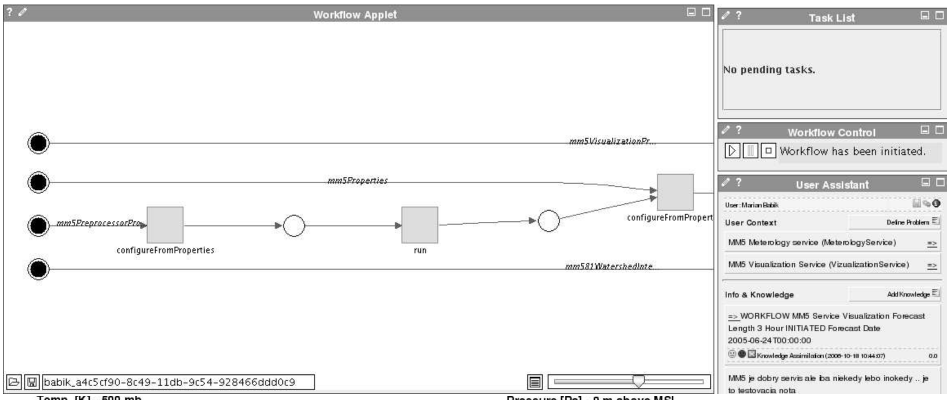
[4] A captured demo session can be found at `http://www.gridworkflow.org/kwfgrid/distributions/movie-09-ffsc-red.avi`

The benefits of this approach are mainly the simple and understandable interface, possibility to compose workflow based on user query as well as user support and guidance throughout the process.

Another alternative path is followed when the user selects a concept, which describes a flood-forecasting method, e.g. DaveF2DVisualization. Such concept is related with the semantic description of the service hosting the method. Based on the semantic description of the service the system can automatically compose an abstract workflow and present it to the user. This process is based on the semantic description of services as described in Section 6.2. This abstract workflow is then analyzed, relevant services capable of creating the solution are found, and a more concrete workflow is created, where the abstract transition is replaced with several classes of services. The user is presented with a list of tasks that he/she needs to fulfill in order to continue execution of the workflow, e.g. specifying time and date of the forecast. The user can then start the workflow by using the *control panel.* Consequently, the system looks for real service instances and executes them. The process of instance lookup and execution is visualized in real-time in the *workflow panel.* After the workflow finishes the user can get the results by clicking on the corresponding output place in the *workflow panel*, e.g. in this case it would display an animation of the possible flooding in the region. The workflows and results can be annotated by the user during the whole process. The overall process of composition and execution is captured by the semantic description of context and experience and stored by the system for later retrieval. The benefits of this approach are mainly the user-friendly interface with a possibility to compose workflows based on the user query, as well as the user support and guidance throughout the process of composition and execution.

9 RELATED WORK

One of the challenges of the loosely coupled distributed systems is the ability to dynamically discover and integrate services needed by the applications. Interoperability among services is especially important in the distributed environments hosting a large number of services, i.e. grids. Semantic descriptions facilitate the process by expressing the characteristics of the service, which is one of the goals of the Semantic Grid initiative [3]. There are many projects, which are trying to develop an architecture for the Semantic Grid applications such as [4, 6, 5, 7]. S-OGSA is trying to extend OGSA-based architecture and provide a reference architecture with explicit handling of semantics as well as defining the associated knowledge services. Guided by the set of design principles it defines a model, the capabilities and mechanisms for the Semantic Grid [4]. InteliGrid aims at developing a grid architecture based on three layers, i.e. conceptual, software and basic resource [5]. Unlike our approach the mentioned projects are trying to address the Grid semantics by a top-down approach, creating reference architectures, which should cover a broad range of applications and requirements. On the contrary, our approach can be seen



No pending tasks.

Workflow has been initiated.

MM5 Meterology service (MeterologyService) =>

MM5 Visualization Service (VizualizationService) =>

=> WORKFLOW MM5 Service Visualization Forecast Length 3 Hour INITIATED Forecast Date 2005-06-24T00:00:00

Knowledge Assimilation (2006-10-18 10:44:07) 0.0

MM5 je dobry servis ale iba niekedy lebo inokedy .. je to testovacia nota

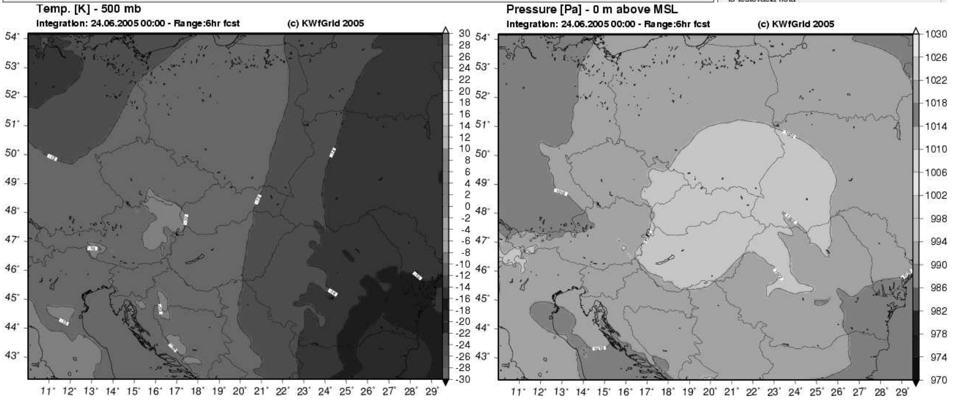

Temp. [K] - 500 mb
Integration: 24.06.2005 00:00 - Range:6hr fcst
(c) KWfGrid 2005
Pressure [Pa] - 0 m above MSL
Integration: 24.06.2005 00:00 - Range:6hr fcst
(c) KWfGrid 2005

Fig. 6. K-Wf Grid portal (top) with *control panel* (top right), *user assistant* (bottom right) and *workflow panel* (left). A sample workflow for weather prediction is shown; grey boxes denote transitions/service operations; circles denote places/service parameters. A sample results of the MM5 prediction method for the area of the Central Europe is shown at the bottom.

as a bottom-up approach, which is trying to leverage as much as possible from the existing Semantic Web Service technologies. A similar approach can be seen in the ^{my}Grid, which is a pioneering Semantic Grid project, providing a set of tools and services to enable workflow composition in biological domain [6]. It is, however, more focused on the support for the OGSA and OGSA-DAI, while we aim at supporting WSRF and OWL-S, which have shown to be more suited for the domain of the flood-forecasting.

There are many existing workflow-based graphical problem solving environments such as [8, 10, 9]. Our approach differs mainly in the exploitation of the user context and experience, which enables storing and re-using past workflows and results, composition of the workflows based on the queries and complex support for the user interactions during the process.

In the context of the *Semantic Web Services*, WSMO, WSMX and IRS-III provide an industry scale framework for discovery, composition and execution of web services [16, 18]. A similar approach to ours is taken by the WSDL-S and METEOR-S, which attempts to add semantics to the basic web service descriptions [17]. Generally, none of the existing semantic web service frameworks provide support for composition and execution of the grid services or any possibility for user assistance based on experience or knowledge re-use.

10 CONCLUSIONS

We have described a working flood-forecasting system based on the semantic grid services. We have shown how it can compose and execute workflows while supporting complex user interactions and assisting technologies. The system is currently tested and evaluated by the Slovak Hydrometeorological Institute on the datasets of the major Slovak river basins. We have tried to address the most important topics related to the flood-forecasting workflow execution environment, however there are other interesting areas such as performance monitoring, scheduling, etc., which are beyond the scope of this paper; refer to the homepage of the project for additional details [15].

Acknowledgments

The research reported in this paper has been partially financed by the EU within the project IST-2004-511385 K-WfGrid and by Slovak national projects NAZOU SPVV 1025/2004, RAPORT APVT-51-024604, VEGA 2/6103/6, VEGA 2/7098/27.

REFERENCES

[1] Habala, O.—Mališka, M.—Hluchý, L.: Workflow-Based Flood Forecasting in K-Wf Grid. In: Proc. of International Workshop on Environmental Applications and Distributed Computing – EADC, 2006, VEDA, pp. 82–89, ISBN 80-969202-4-3. October 2006, Bratislava, Slovakia.

[2] Novotny, J.—Russell, M.—Wehrens, O.: GridSphere: An Advanced Portal Framework. EUROMICRO '04: Proceedings of the 30th EUROMICRO Conference (EUROMICRO '04), 0-7695-2199-1, pp. 412–419, IEEE Computer Society, Washington, DC, USA.

[3] Goble, C.—de Roure, D.: The Semantic Grid: Myth Busting and Bridge Building. In Proceedings of the 16th European Conference on Artificial Intelligence (ECAI-2004), Valencia, Spain, 2004.

[4] Alper, P.—Corcho, O.—Kotsiopoulos, I.—Missier, P.—Bechhofer, S.—Goble, C.: S-OGSA as a Reference Architecture for OntoGrid and for the Semantic Grid. GGF16 Semantic Grid Workshop. Athens, Greece. February 2006.

[5] Turk, Z.—Stankovski, V.—Gehre, A.—Katranuschkov, P.—Kurowski, K.—Balaton, E.—Hyvarinen, J.—Dolenc, M.—Klinc, R.—Kostanjsek, J.—Velkavrh, J.: Semantic Grid Architecture, Deliverable 13.1 Interoperability of Virtual Organizations on a Complex Semantic Grid (InteliGrid), IST-004664, `www.inteligrid.com/data/works/att/d13_1.content.00913.pdf`, 2005.

[6] Wroe, C.—Goble, C. A.—Greenwood, M.—Lord, P.—Miles, S.—Papay, J.—Payne, T.—Moreau, L.: Automating Experiments Using Semantic Data on a Bioinformatics Grid. IEEE Intelligent Systems, Vol. 19, 2004, pp. 48–55.

[7] Zhuge, H.: China's E-Science Knowledge Grid Environment. IEEE Intelligent Systems, Vol. 19, 2004, No. 1, pp. 13–17.

[8] Taylor, I.—Shields, M.—Wang, I.—Harrison, A.: Visual Grid Workflow in Triana. Journal of Grid Computing, Vol. 3, 2005, pp. 153–169.

[9] Deelman, E. et al.: Pegasus: Mapping Scientific Workflows onto the Grid. Grid Computing, 2004, Springer, LNCS.

[10] Hategan, M.—von Laszewski, G.—Amin, K.: Karajan: A Grid Orchestration Framework. Supercomputing 2004, Pittsburgh, November 6–12, 2004.

[11] Ankolekar, A. et al.: OWL-S: Semantic Markup for Web Service. 2003, `http://www.daml.org/services/owl-s/1.1`.

[12] Christensen, E.—Cubera, F.—Meredith, G.—Weerawarana, S.: Web Services Description Language (WSDL) 1.1. Technical report, WWW Consortium, 2001, `http://www.w3c.org/TR/wsdl`.

[13] Web Service Resource Framework. `http://www.globus.org/wsrf/`.

[14] Globus Toolkit. `http://www-unix.globus.org/toolkit/`.

[15] The Knowledge-based Workflow System for Grid Applications FP6 IST project. `http://www.kwfgrid.net`.

[16] Fensel, D.—Bussler, C.: The Web Service Modeling Framework WSMF. Eletronic Commerce: Research and Applications, Vol. 1, 2002.

[17] Rajasekaran, P.—Miller, J.—Verma, K.—Sheth, A.: Enhancing Web Services Description and Discovery to Facilitate Composition. International Workshop on Semantic Web Services and Web Process Composition, 2004.

[18] Motta, E.—Domingue, J.—Cabral, L.—Gaspari, M.: IRS-II: A Framework and Infrastructure for Semantic Web Services. 2nd International Semantic Web Conference (ISWC 2003), Sundial Resort, Sanibel Island, Florida, USA, 2003.

[19] Gubala, T.—Bubak, M.—Malawski, M.—Rycerz, K.: Semantic-Based Grid Workflow Composition. In: Proc. of 6th Intl. Conf. on Parallel Processing and Applied Mathematics PPAM 2005, R. Wyrzykowski et al. eds., 2005, Springer-Verlag, Poznan, Poland.

[20] Neubauer, F.—Hoheisel, A.—Feiler, J.: Workflow-Based Grid Application. Future Generation Computer Systems, Vol. 22, 2006, pp. 6–15.

[21] Hluchý, L.—Tran, V. D.—Habala, O.—Šimo, B.—Gatial, E.—Astaloš, J.—Dobrucký, M.: Flood Forecasting in CrossGrid project. In: Grid Computing, 2nd European Across Grids Conference, Nicosia, Cyprus, Janu-

ary 28–30, 2004, LNCS 3165, Springer-Verlag, 2004, pp. 51–60, ISSN 0302-9743, ISBN 3-540-22888-8.

[22] Bergmann, R.: Experience Management: Foundations, Development Methodology, and Internet-Based Applications. Lecture Notes in Artificial Intelligence, ISBN 3540441913, 2002.

[23] Balogh, Z.—Gatial, E.—Laclavík, M.—Mališka, M.—Hluchý, L.: Knowledge-Based Runtime Prediction of Stateful Web Services for Optimal Workflow Construction. Proc. of 6th Intl. Conf. on Parallel Processing and Applied Mathematics PPAM 2005, R. Wyrzykowski et al. eds., 2006, LNCS 3911, Springer-Verlag, pp. 599–607, ISSN 0302-9743, ISBN 3-540-34141-2, Poznan, Poland.

[24] Kryza, B.—Slota, R.—Majewska, M.—Pieczykolan, J.—Kitowski, J.: Grid Organizational Memory-Provision of a High-Level Grid Abstraction Layer Supported by Ontology Alignment. Future Generation Computer Systems, Vol. 23, 2007, No. 3, pp. 348–358.

[25] Dutka, L.—Kryza, B.—Krawczyk, K.—Majewska, M.—Slota, R.—Hluchý, L.—Kitowski, J.: Component-Expert Architecture for Supporting Grid Workflow Construction Based on Knowledge. In Cunningham, P., Cunningham, M. (Eds.): Innovation and the Knowledge Economy, Issues, Applications, Case Studies, Vol. 2, IOS Press 2005, pp. 239–246.

[26] Laclavík, M.—Babík, M.—Balogh, Z.—Hluchý, L.: AgentOWL: Semantic Knowledge Model and Agent Architecture. In: Computing and Informatics, Vol. 25, 2006, No. 5, p. 419–437, ISSN 1335-9150.

[27] Armengol, E.—Plaza, E.: Symbolic Explanation of Similarities in Case-based Reasoning. In: Computing and Informatics, Vol. 25, 2006, No. 2–3, ISSN 1335-9150.

[28] Carminatti, N.—Borges, Marcos R. S.—Gomes, J. O.: Analyzing Approaches to Collective Knowledge Recall. In: Computing and Informatics, Vol. 25, 2006, No. 6, ISSN 1335-9150.

Marián Babík is a researcher at the Institute of Informatics of the Slovak Academy of Sciences at Center for Intelligent Technologies (CIT) and a member of the computing group at the Institute of Experimental Physics, SAS. In 2003 he received his M. Sc. degree in artificial intelligence from the Technical University, Košice. He was a member of the CDF collaboration at the Fermi National Laboratory and worked as a technical student at the European Organization for Nuclear Research (CERN). He is the author and co-author of several scientific papers and participates in the EU IST projects K-Wf Grid, EGEE-2, Degree, as well as in several national projects. He is experienced in the Semantic Web technologies and languages; development of large scale systems; development of knowledge based systems and semantic web services.

Ladislav Hluchý is the director of the Institute of Informatics of the Slovak Academy of Sciences and also the head of the Department of Parallel and Distributed Computing at the institute. He received M. Sc. and Ph. D. degrees, both in computer science. He is R & D Project Manager, Work-package Leader in a number of 4FP, 5FP and 6FP projects, as well as in Slovak R & D projects (VEGA, APVT, SPVV). He is a member of IEEE, ERCIM, SRCIM, and EuroMicro consortiums, the editor-in-chief of the journal Computing and Informatics. He is also (co-)author of scientific books and numerous scientific papers, contributions and invited lectures at international scientific conferences and workshops. He also gives lectures at Slovak University of Technology and is a supervisor and consultant for Ph. D., master and bachelor studies.

Michal Laclavík is a researcher at the Institute of Informatics of the Slovak Academy of Sciences. In 1999 he received his M. Sc. degree in computer science and physics. He received his Ph. D. degree in applied informatics with focus on knowledge oriented technologies in 2006. He is the author and co-author of several scientific papers and he participates in the Pellucid and K-Wf Grid IST projects as well as in several national projects. His research interests include artificial intelligence, agent technologies, knowledge management and semantic web.

Ondrej Habala is a researcher and Ph. D. student at the Department of Parallel and Distributed Computing of the Institute of Informatics, Slovak Academy of Sciences. His primary research interests are data integration, semantical metadata repositories, and applications of peer-to-peer networks in grid computing.