Service-Oriented Architecture for Grid-Enabled Computer Simulation Software

Anatolii Petrenko[1], Volodymyr Ladogubets[1], Oleksii Finogenov[1], Bogdan Bulakh[1]

1. CAD Department, Educational Scientific Complex "Institute of Applied System Analysis", National Technical University of Ukraine "Kyiv Polytechnic Institute", UKRAINE, Kyiv, P. Myrnyi street 19

Abstract - **In this paper architectural decisions for development of grid-aware simulation software are presented. The solution relying on the composition of web and grid services to provide the ability to customize scenarios of analysis is proposed. The concept of analysis workflows is described briefly and web service workflows execution procedure is covered.**

Keywords - **Simulation, Grid computing, SOA, NetALLTED.**

I. Introduction

There are a number of challenges being faced by the developers of modern CAE/CAD tools. Interdisciplinary research tools capable of modelling objects of different physical nature are in high demand affecting the internal models and logic. The problem of the interoperability with third-party software that can be involved in the complex analysis scenarios is also urgent. At the same time the complexity of such objects of analysis forces the use of high performance computing resources as it may be impossible to reach the desired accuracy in the reasonable time with only desktop PC's computing power. Modern tools must provide some collaboration opportunities, which lead to the virtual lab environments capable to support collective work on projects by the groups of engineers or researchers. It's worth noticing that both computing resources and their users can be (and often are) geographically distributed across multiple countries.

Such wide range of today's demands to simulation software design can be summed up in the following requirements. Functional capabilities: multiple physical domains support, rich analysis and visualization toolkit, compatibility with popular existing tools. Performance: harnessing the power of computing clusters, grids, clouds. User environment: highly customizable models and computing scenarios, collective work support, user-friendly visual user interface. Deployment and maintenance: cross-platform, distributed solutions, easy to extend and to upgrade and so on.

II. Service-Oriented Approach

In order to cope with requirements mentioned above the architecture of modern simulation tools must be flexible and extensible, as opposed to traditional monolithic standalone applications. The latter variant exposes the following shortcomings: it must be installed and upgraded on every machine where it may be used and usually utilizes computing power of only single machine. To mitigate these restrictions a client-server model can be implemented, where client side provides a lightweight front-end to server-side functionality that can be installed on remote high performance computing resource.

Such general approach that separates user interface from logic is well-known and widely accepted in the architecture of many software packages including engineering calculation and simulation tools. But there still are issues like compatibility, extensibility, harnessing grid computing issues. The solution proposed leverages the advantages of the service-oriented architectural pattern for the server-side implementation to cope with previously listed demands.

The architecture proposed is a further refinement of the client-server model where server-side functionality is decomposed to "ecosystem" of loosely-coupled services. Service-oriented architecture (SOA [1]) has become popular solution for distributed enterprise-scale software systems (like ERP), as it has the following specifics:

- *abstraction and modularity*: loosely-coupled services as the basic functional units having no hard-coded references to other services, hiding their logic behind the public interfaces and communicating with clients and each other in a message passing style can simplify maintenance and introduction of new functionality;
- *discoverability*: services are provided with metadata that helps them to be categorized and discovered;
- *composability and reuse*: with their public interfaces described in a standard way, services are able to be composed (orchestrated) statically or even dynamically, manually or automatically to form higher-level services with desired functionality.

Although SOA can be built upon many existing technologies like RPC/RMI, CORBA, Jini, DCOM etc. the most successful implementation of the SOA methodology is web services (WS). WS are compliant to adopted W3C standards, rely on well-known protocols and technologies like HTTP and XML, are platform and language independent thus effectively helping to overcome interoperability problems.

With these advantages web services can be successful not only for business applications but for engineering software as well. Besides previously mentioned general reasons of using web services by computer simulation tools there are the following specific ones:

1. WS-standards are also adopted by the grid community (e.g. OGSA, WSRF specifications declaring the concept of grid services) that allows easier utilization of grid computing resources for compute-intense analysis.
2. Service orchestration mechanism can be leveraged for organization of customizable scenarios of computations in the form of web service workflows.

III. Analysis Workflows

The basic idea of the proposed solution is a representation of the analysis scenario as the workflow: a set of activities orchestrated to be executed in the specific order (the most common visual representation of the workflow is a graph, usually DAG). Fig.1 shows the example of the analysis workflow from electronic circuit design involving several types of analysis and having both sequential and parallel branches.

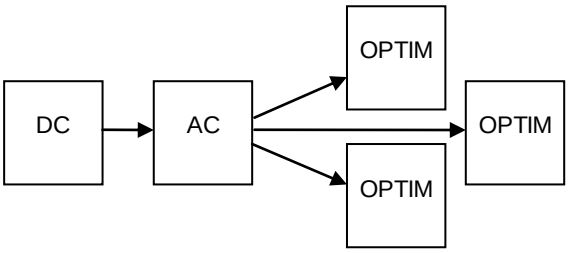

Fig.1 Electronic circuit analysis workflow: DC – direct current analysis, AC – alternate current analysis, OPTIM – optimization procedures.

This idea is also shared by so-called Scientific Workflow systems [2] (e.g. Kepler, web services-oriented Taverna [3], grid-aware Askalon), but they are general frameworks and are not specially tailored for engineering tasks like the CAE tools are.

The main difference between web service orchestration tools (orchestrators) like Taverna and the proposed approach is that the latter has an extra abstraction layer (abstract workflow or task). This means that user does not compose the workflow from web services directly but connects abstract activities being later mapped to concrete web service invocation scenario automatically by the system. So, the web service registry should be also supplemented with activities library and mapper logic.

This additional overhead from the architectural point of view allows less overhead from the user's point of view. First, user does not need to know about and to deal with all web service metadata and concrete orchestrator's specifics like invocation details or XML message handling. Second, non-trivial mapping becomes possible when single activity can be mapped to a several communicating web services as well as when several activities can be mapped to a single web service invocation. In other words, such intermediate abstract workflow concept allows separating services and orchestrator details from user scenarios increasing overall flexibility of the system.

IV. General Architecture

The proposed architecture of the grid-enabled service-oriented simulation platform consists of the following layers (Fig.2).

User interface is organized as web interface in form of web/grid portal. As soon as the current web technologies and tools like JavaScript, AJAX or RIA (Rich Internet Application) frameworks allow the creation of user friendly but sophisticated and powerful enough graphical user interface this approach can reveal many benefits. It needs no specific client software preinstalled but web browser (thus enabling access from wide variety of networked devices besides desktop PCs like netbooks, smart phones, tablet PCs etc.). It also simplifies the provision of the virtual collaborative lab environment.

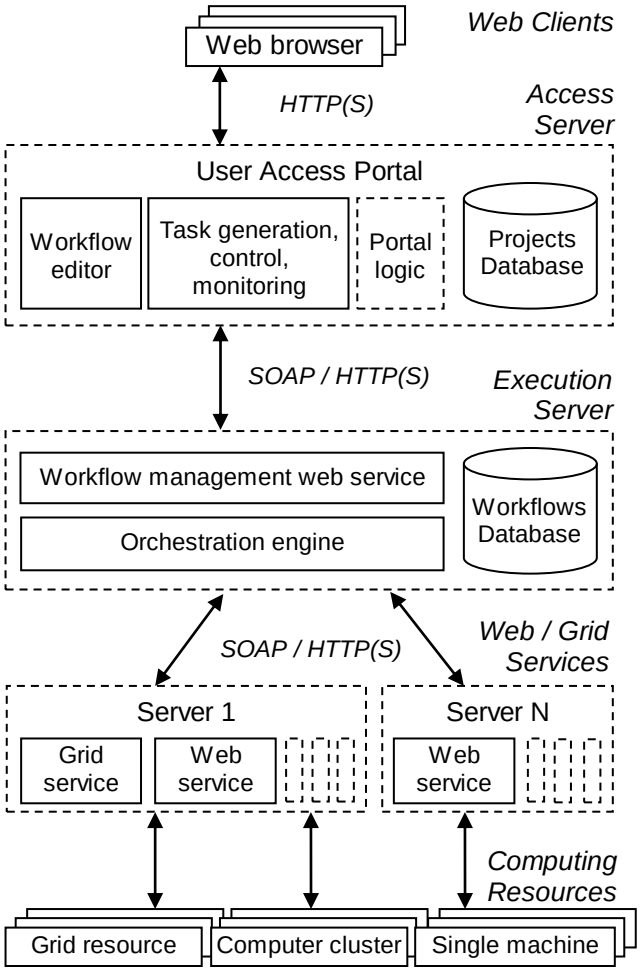

Fig.2 Main elements of service-oriented architecture of grid-enabled computer simulation system.

User interface provides the following functionality: authorization, graphical workflow editor, project artefacts browsing (input and output files management, simulation results visualizers etc.), task execution monitoring and others.

The server-side part of the architecture has several layers to reflect the abstract workflow concept described above. First tier is the *portal* which organizes user environment: holds user data and preferences, controls user access, provides information support, organizes user interface. Its modules are also responsible for: abstract workflow description generation according to user inputs, passing this task description to lower architecture layers for execution, retrieving finished task results and storing all the project artefacts in the database.

The next tier is the *workflow manager* running on the execution server. It is responsible for mapping (with the help of service registry) the abstract workflow description to the concrete web services orchestration scenario expressed in the orchestrator-specific input language (like WS-BPEL for

BPEL engines or t2flow for Taverna). It also initiates the execution of the concrete workflow with the external orchestrator, monitors its state and fetches the results.

Concrete workflow operates with functional SOAP *web services* representing the basic building blocks of system's functionality: data preparation and adaptation, simulation, optimization, results processing etc. Compute-intensive steps are implemented as grid services interacting with *grid resources* to run computations as grid jobs. Introduction of the new functionality to the system is accomplished through the registration of the new web or grid services.

The overall sequence of user scenario execution is as follows. User passes login procedure on the portal and accesses workflow editor. He may choose and setup the activities available in repository to compose the scenario workflow he wants to execute. It must be noted that some activity sequences may not be allowed due to logical incompatibility (e.g. transient analysis as a predecessor or successor of frequency domain analysis) or data formats incompatibility (e.g. between activities from different providers). These rules must be checked at design time by editor.

Then the execution phase is initiated by user. User task description is passed to workflow management service on execution server, where this abstract workflow is translated to the concrete one. Workflow manager parses the description and checks for errors, requests metadata from service registry and performs mapping from activity sequence to web service invocations sequence, described in one of the standard orchestration languages. Mapper unit of the workflow manager should arrange web services in correct invocation order according to abstract workflow, organize XML messages and variables initializations and assignments between calls, and provide the ways for run-time control (workflow monitoring, cancelling, intermediate results retrieving etc.). Then this concrete scenario is executed by orchestrator.

When orchestrator invokes grid service the latter initiates the submission of grid job to grid resource: it prepares job description and communicates with grid middleware to schedule and execute grid job.

User is informed about the progress of the workflow execution by monitoring unit communicating with workflow manager. When execution is finished user can retrieve the results, browse and analyse them and repeat this sequence if needed.

V. Implementation

These presented architectural principles were implemented during the work on the project of development of the "Interdisciplinary complex of optimal mathematical modelling in grid environment with the automatic composition and solving of equations of corresponding mathematical models" executed by ESC "IASA" of NTUU "KPI". Server-side concrete workflow management is based on standard web service orchestration description language WS-BPEL 2.0 (Business Process Execution Language) [4]. Simulation grid services rely on the functionality of the ALLTED [5] simulation software and compatible with computing resources accessible through Nordugrid ARC [6] grid middleware. The test prototype of this system was deployed at the resources of the NTUU "KPI" HPC Centre.

References

[1] T. Erl. *Service-Oriented Architecture: Concepts, Technology & Design*. New York: Prentice Hall/PearsonPTR, 2005, 792 p.

[2] V. Curcin, M. Ghanem. Scientific workflow systems - can one size fit all? // *Proc. of Biomedical Engineering Conference CIBEC2008*. Cairo International, 2008, P.1-9.

[3] Taverna Workflow Management System: http://www.taverna.org.uk/

[4] Web Services Business Process Execution Language: http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf

[5] A. Petrenko, V. Ladogubets, V. Tchkalov, Z. Pudlowski, *ALLTED – a computer-aided engineering system for electronic circuit design*. Melbourne: UICEE, 1997,205 p.

[6] Nordugrid Advanced Resource Connector: http://www.nordugrid.org/arc/

VI. Conclusion

The multi-layered architecture of the grid-enabled computer simulation software was presented. This architecture is characterized by the following: it is web-accessible, its functionality is distributed across the ecosystem of both web services and grid services (enabling utilization of grid computing resources); it is compatible with adopted standards and protocols; it supports custom user analysis scenario development and execution; it hides the complexity of web-service interaction from user with abstract workflow concept and graphical workflow editor. The prototype developed according to this architecture has proven the working ability of the proposed solution. The further development of this approach can involve semantic technologies for automatic workflow synthesis and analysis.

VII. Acknowledgements

Results presented in the paper have been carried out within the works supported by the Ukrainian State goal-oriented scientific-technical programme for implementation and application of Grid technologies and a Marie Curie International Research Staff Exchange Scheme Fellowship within the 7th European Community Framework Programme – EduMEMS - Developing Multidomain MEMS Models for Educational Purposes, no. 269295.