Extensible Digital Library User Portals for e-Infrastructures

Massimiliano Assante, Luca Frosini

Institute of Information Science and Technologies (ISTI),
Italian National Research Council (CNR), Via G. Moruzzi, 1,
56124 Pisa, Italy
{massimiliano.assante, luca.frosini}@isti.cnr.it

Abstract. Large scale Digital Library e-Infrastructures are emerging as a new approach for facilitating the development of Digital Libraries (DLs) serving diverse user communities with specific and continuously evolving requirements. Such an approach promotes the building of large-scale DLs via the dynamic, automated management of highly-distributed DL resources – e.g. collections of information objects, services, computing and storage resources – and enforcing controlled sharing of such resources across different DLs. This paper discusses an extensible and new approach for User Interfaces development, facilitating the *sharing* and *reuse* of UI resources, i.e. user interface constituents. In particular a design model and a concrete implementation, carried out in the context of DILIGENT and D4Science projects is presented in detail.

Keywords: Large scale Digital Library E-Infrastructures, Digital Library Management, User Interface development

1 Introduction

Research work today is often a collaborative effort carried out by communities of individuals possibly affiliated to different Institutions sharing common goals and funding opportunities. These communities aggregate themselves into *virtual organizations* [1] (VOs) [1] with the aim of sharing their resources, e.g. knowledge, experimental results, instruments, for the whole period of their collaboration, so as to create rich and powerful research environments.

Such communities manifested novel and peculiar Digital Library (DL) requirements: they may come from different research branches, integrating methods and knowledge from the relevant subject areas, notwithstanding they feature common *dynamic requirements* of large-scale production, sharing, aggregation, diffusion, and manipulation of research outcomes in digital form. Moreover, VO members are intrinsically world-wide, multidisciplinary, and transient with respect to traditional communities. For instance,

[1] Groups of individuals, institutions and resources, possibly remotely distributed, which are grouped together to achieve a common goal.

communities may include an arbitrary, unforeseeable number of users, information objects coming from heterogeneous and potentially world-wide dispersed data sources, and functions.

DLs should provide support for the collaboration of these dynamic and highly distributed communities; systems capable of providing working environments that guarantee access to quality content and functionality resources tailored to their time-evolving needs and large-scale requirements.

In the last few years, e-Infrastructure technologies have been adopted as a new approach for facilitating the development of Digital Libraries (DLs), ranging from institutional to very-large scale, serving diverse user communities with specific and continuously evolving requirements. e-Infrastructures implement environments facilitating the *sharing* and the *reuse* of services, data and research results by supporting a level of agreed standards and participation policies. In this context, a *Virtual Research Environment* (VRE) is defined as a set of resources, e.g. services, cooperating together to deliver the functionality expected by the end users of the VRE's VO.

VREs have to be accessible by users which may be occasional and spread worldwide. Consequently the best way for VO members to interact with VREs would be through web portals providing user interface access to all resources belonging to the VRE at hand.

However, since VREs are dynamic sets of resources that may vary in time, traditional static web portals are not sustainable solutions; i.e. coping with the potential modifications would imply high development costs. A solution for this problem of finding ways for building DL User Interfaces in a general and efficient way is proposed in this paper, trying to reduce the developer effort as much as possible and lower costs by using dynamic web portals. In a VRE context a Portal can be conceived as the high-level interface providing VRE users with an integrated and organized view of the VRE constituents, i.e. data and services. The portal acts as a *hosting environment* for all VRE UI components.

In Section 2 we present Service Oriented Infrastructure (SOI) as a possible VRE implementation. In Section 3 and 4 we describe a design to adapt web portals to VRE followed by our solution and its current implementation as realized in the DILIGENT and its continuation D4Science EU Project [3]. Finally, Section 5 concludes the paper by reporting on the potentialities of the proposed approach and on the future evolution of the proposed framework.

2. e-Infrastructures and VREs

In Service Oriented Infrastructures (SOI) [4] [5] applications are defined as dynamic sets of services which can be shared or reused across different applications. e-Infrastructures can be implemented as specific forms of SOI, where VREs can be seen as dynamic sets of possibly shared *resources*, interacting with each other in respect of given security policies.

Specifically, in the context of DILIGENT and D4Sscience infrastructures VREs definition and management functionalities are implemented by the following *enabling* services:

- Security Service: The Services addressing security, by ensuring that resources are to be consumed only by authenticated and therefore authorized resources.
- Information Service (IS): The Services implementing the functionality for supporting resource registration, discovery and notification in the e-Infrastructure. Its main task is keeping an up-to-date map of the VRE; resources partaking the system must be registered to the IS by providing a profile and keeping it updated to evidence their existence.
- Manager Service (MS): The Services managing VREs by (i) orchestrating the resources of a specific VRE in order to deliver the expected functionality, and (ii) monitoring resources of a specific VRE in order to optimize efficiency. The MS is the "brain" of the e-Infrastructures. This service is conceived in order to manage VRE resources in terms of their creation, modification, and deletion with the goal of maximizing the quality of service in the VREs running on top of the e-Infrastructures.

Most importantly, the Service Oriented Infrastructure approach enables the construction of sustainable VREs by reducing their maintenance costs and coping with scalability issues. VREs are directly configured by the organizations defining them through management and orchestration rules automatically administered by the infrastructure enabling services. Moreover, the criteria according to which e-Infrastructures are defined and managed by Information and Manager Services are general enough to not constrain participating organizations to specific design and implementation choices.

3. Web Portals for VREs

The resources, i.e. the services, which partake in a VRE, can be very-large in number and change in time. Consequently, Web portals for VREs should face the same variability and dynamicity by adapting to the user interface functionalities relative to the services available at a specific time. This paper proposes a solution to this problem: individual services are delegated the implementation of *web portal subcomponents* that embed the user interface logic of their functionality; Web portals are then conceived as dynamically configurable services, capable of accessing and combining only those subcomponents required by the scientific community at hand. It is worth noticing that the same service may expose its web portal subcomponent to different VREs and according to different requirements of the VREs. In that sense, subcomponents own different states, to be associated to the VREs they are joining.

Above all, web portals are customizable with respect to:

- what user functionality they should support, i.e. what services subcomponents it should address;
- how these functionalities should be combined, logically and graphically, in order to generate an usable interface;
- how the status of each subcomponent should be configured to reflect the requirements of the given VRE.

Next section describes how such a solution was implemented in the context of the DILIGENT and D4Science e-Infrastructure, together with the issues faced during the development work, and how they have been solved.

4. Implementation in D4Science

D4Science is one of the main European e-Infrastructure project, involving 11 participants co-funded by the European Commission's Seventh Framework Programme for Research and Technological Development. The project started in January 2008 and has a life of 2 years. D4Science aims to continue the path that the GÉANT [8], EGEE [2], and DILIGENT [9] projects have initiated towards establishing networking, grid-based, and data-centric e-Infrastructures that accelerate multidisciplinary research by eventually overcoming several crucial barriers that stand in the way, primarily those related to heterogeneity, sustainability and scalability.

D4Science consists of a number of services that implement mediation functions between the *resources providers*, i.e. organizations that decide to share their resources under certain policies, and *resource consumers*, i.e. virtual research organizations that need DLs for supporting their activities.

By exploiting a D4Science infrastructure a *virtual research organization* can create a DL that satisfies its needs by sending a request to the system. This request must specify a set of requirements on the information space and on the expected functionalities. The infrastructure responds to this request by transparently instantiating, and then making available through a portal, a reliable and secure DL that meets the specified criteria. This DL is activated on a subset of the resources that are accessible to the research organization. Many DLs, serving different virtual research organizations, thus co-exists on the same set shared resources.

DILIGENT and D4Science infrastructures implement SOI approach by Web-Services, in particular it adopts *Web Service Resource Framework* [14] (WSRF) as enabling technology. In such scenario web-services communicate each other through *Simple Object Access Protocol* [15] (*SOAP)* messages.

Web portals in D4Science that match the requirements specified in Section 3 are built on the following features:

- Web based interfaces: accessible world-wide;
- Component-oriented approach: reflect from User Interface prospective the SOI paradigm;
- Web-Service based environment integration: the portal is perfectly suited in the D4Science e-Infrastructure.

In particular *Java portlet* (JSR 168) has been adopted as the enabling technology. In the following we describe this technology and its customization to support Extendible Web Portals in the context of DILIGENT and D4Science e-Infrastructure.

4.1 Portlet Technology

The Java Portlet Specification [6] [13] defines a contract between the portlet container and portlets providing a convenient programming model for portlet developers. The Specification V1.0 was developed under the Java Community Process as Java Specification Request (JSR) 168 to introduce the *basic portlet programming model*, mainly characterized by:

1. two phases of action processing and rendering in order to support the Model-View-Controller pattern;
2. portlet modes, enabling the portal to advise the portlet what task it should perform and what content it should be generated;
3. window states, indicating the amount of portal page space that will be assigned to the content generated by the portlet;
4. portlet data model, allowing the portlet to store view information in the render parameters, session related information in the portlet session and per user persistent data in the portlet preferences;
5. a packaging format in order to group different portlets and other J2EE artefacts needed by these portlets into one portlet application which can be deployed on the portal server.

Such web based technology, allows each single portlet to represent not the whole interface but just a portion of it, with the possibility of being plugged together. In particular, portlet-based Web portals interact with a *portal engine service* whose purpose is that of combining the *portlet* exposed by a number of services, i.e. *servers*, to shape a uniform interface. Besides, the same engine can generate/support an arbitrary number of *layouts*, combining the server portlets available in the local container according to different user community needs.

Due to such peculiarities portlets have appeared to be the natural solution to implement web portals for dynamic, possibly very-large VREs. In particular, the web portal requirements of different VREs can be modelled as different layouts in the context of one portlet engine.

Among the portal engines technologies available in the market, the choice has fallen on Gridsphere [10] because of its open-source nature and the major functionalities offered for JSR 168 management and compliancy.

Portlet technology also offers extensions allowing for remote portlet deployment integration in a web-service based environment. Such an extension is called Web Service for Remote Portlet (WSRP) [7]. The WSRP specification defines a web service interface for interacting with presentation-oriented web services. Java's portlet specification, JSR 168, and WSRP are not competing technologies: JSR 168 is used to define a portlet, while WSRP may be used to enable deployment of portlets into remote containers so as to support transparent interaction between local engines and remote portlets.

4.2 Adapting Portlet in D4Science

Java Portlet technology addresses most of the issues required for dynamic Web portal construction. However, due to the highly-distributed architecture of the underlying e-Infrastructures, this technology required proper customization in order to guarantee the following requirements in the presence of multiple services interaction:

- responsiveness;
- dynamic configurability
- inter-portlet communication and events signaling.

Responsiveness

The JSR 168 standard forces a web page reload each time a user fires a client side event; e.g. opening a tree-menu. This is generally acceptable for portlet-based Web Applications, where portlet servers are typically running local to the portal engine and performing local business logic actions. The DILIGENT and D4Science infrastructures, in contrast, portlet servers can run at different sites, vary in number, remotely from the engine, possibly interacting with others in order to accomplish their tasks. This feature would imply unnecessary server side interactions thereby causing “frustrating” user interface rendering delays. For instance, consider a user action executed in a page composed by three portlets. According to the JSR 168 standard, this would cause three server-to-engine communications, two of which would have been useless since anything could have changed in their rendering context.

For this purpose we looked for a technology which could make us overtake this shortage, moving part of the application logic from the server to the engine. Analysing the state of the art we found in the set of technologies grouped under the acronym AJAX [12] a solution that best suited our needs.

Ajax (Asynchronous JavaScript and XML), or AJAX, is a group of interrelated web development techniques used for creating interactive web applications or rich Internet applications. With Ajax, web applications can retrieve data from the server asynchronously in the background without interfering with the display and behaviour of the existing page.

Furthermore, to simplify the development work, we decided to adopt a framework which would abstract from the different implementation details and overhaul multi-browser incompatibility. Among the available frameworks our choice has fallen on the use of Google Web Toolkit (GWT) [13] since apart being Java based, it has a higher level of abstraction both server and client side.

The nature of AJAX technology allows user actions to modify the interface without server interaction, this raises another issue: server can not maintain the client interface status anymore with respect to JSR 168 specification.

For this reason, each user action needing status restore invokes an asynchronous server call registering the current status information on the server using the session. On users page reloads or portal page switch, a portlet can restore its last status, before it is displayed again by the web browser, asking it to the server.

Dynamic configurability

JSR 168 portlet servers expose portlet subcomponent whose configuration can be provided through a special configuration file. Besides, the same portlet will run according to the same configuration file in the context of all layouts involved.

On the other hand, in DILIGENT and D4Science, web portals of different VREs share portlets but expecting different rendering a/or behaviour, i.e. the same portlet may belong to different VRE layouts according to different configurations. In this environment, portlet dynamicity has been accomplished, relying on the Information System (IS): (*i*) the IS stores the configuration files relative to each portlet with respect to the VRE layouts it belongs to (*ii*) when invoked in the context of a VRE, the portlet looks up the IS to fetch the right configuration. The IS in fact is the ideal candidate due to its context related nature.

Inter-portlet communication and events signaling

Standard inter-portlet communication occurs when an action made on a portlet also modifies the status on another one in the page. In the JSR 168, this is obtained by means of continuous server interactions, i.e. having portlets sharing some data through the server and firing the refresh of all portlets involved when one changes such data. The refresh forces each portlet to regenerate its status with the following drawbacks: portlets not affected by the status change are refreshed anyway, "frustrating" end user wait, and unnecessary overhead in terms of bandwidth and CPU consumption.

We have studied and implemented a solution to enable inter-portlet communication and avoid such refresh and the relative drawbacks. Portlets can synchronize on some shared semaphore events and thus limit data exchange to the strict necessary.

The solution is an implementation of a subscription/notification mechanism based on HTTP Requests and semaphores mechanisms. It works by sharing server-side (in the session) a semaphore among those portlets needing to communicate: a portlet which needs to be notified on a given event invokes an asynchronous server call request with the effect of blocking itself on a shared semaphore through a *wait*(). Since asynchronous server calls requests use the HTTP request method, in which requests typically last 30 seconds, it is necessary to always renew them when they time out. As a result, the portlet executing a *wait*() on a given semaphore can continue to work on the client, since it has invoked an asynchronous call request. In the meantime that request is blocked on the server and returns to the client only if another portlet generating an event executes a signal() on that semaphore. Hence portlet content changes depending on the event which is raised.

5. **Conclusions**

Web Portal sustainability, i.e. well-balanced trade-off between portal developing effort (and costs) and community benefits derived from the use of VREs, is motivated by the technological evolution from traditional DL technology to e-Infrastructures.

This work presented Dynamic Web Portals as one promising solution to the construction and maintenance of large-scale sustainable DLs that satisfy the new

challenging requirements brought in by Scientific Communities. Specifically, it also presented a particular implementation based on portlet technology provided within the DILIGENT and D4Science infrastructures. In both cases the approach proves to be sustainable and effective.

6. Acknowledgments

This work would have not been possible without the work of one other DILIGENT and D4Science technical partner: University of Athens.
Special gratitude goes to Leonardo Candela and Andrea Manzi from ISTI – CNR.

References

[1] Foster, I., Kesselman, C., Tuecke, S.: The anatomy of the Grid: enabling scalable virtual organization. Int. J. High Perform. Comput.Appl. 15(3), 200–222 (2001)
[2] EGEE: Enabling Grids for E-science. http://public.eu-egee.org/. INFSO 508833
[3] D4science: DIstributed colLaboratories Infrastructure on Grid ENabled Technology 4 Science, www.d4science.org
[4] Lublinsky, B.: Defining SOA as an architectural style. http://www28.ibm.com/developerworks/webservices/library/ar-soastyle/index.html (2007). IBM developerWorks
[5] Newcomer, E., Lomow, G.: Understanding SOA with Web Services. Addison-Wesley (2005)
[6] JSR 168: http://www.jcp.org/en/jsr/detail?id=168.
[7] WSRP: http://java.sun.com/javaee/javaserverfaces/.
[8] GEANT: http://www.geant.net/
[9] DILIGENT: A DIgital Library Infrastructure on Grid ENabled Technology. http://www.diligentproject.org/. IST No. 004260
[10] GRIDSPHERE: http://www.gridsphere.org/
[11] GWT: Google Web Toolkit, http://code.google.com/webtoolkit/
[12] AJAX: Asynchronous Javascript and XML, www.ajax.org
[13] Introducing Java Portlet Specification**:** JSR 168 and JSR 286, http://developers.sun.com/portalserver/reference/techart/jsr168/
[14]Banks, T.: Web Services Resource Framework (WSRF) - Primer. Committee draft 01, OASIS (2005). http://docs.oasisopen.org/wsrf/wsrf-primer-1.2-primer-cd-01.pdf
[15] Candela, L.; Akal, F.; Avancini, H.; Castelli, D.; Fusco, L.; Guidetti, V.; Langguth, C.; Manzi, A.; Pagano, P.; Schuldt, H.; Simi, M.; Springmann, M. & Voicu, L. DILIGENT: integrating Digital Library and Grid Technologies for a new Earth Observation Research Infrastructure. International Journal on Digital Libraries, 2007, 7, 59-80
[16] Candela, L.; Castelli, D.; Manghi, P. & Pagano, P. M. Agosti, F. E. & Thanos, C. (ed.) Enabling Services in Knowledge Infrastructures: The DRIVER Experience. Post-proceedings of the Third Italian Research Conference on Digital Library Systems (IRCDL 2007), DELOS: a Network of Excellence on Digital Libraries, 2007, 71-77
[17] Candela, L.; Castelli, D. & Pagano, P. gCube: A Service-Oriented Application Framework on the Grid. ERCIM News, 2008, 48-49