Service Oriented Architecture for Biomedical Collaborative Research

José Antonio Heredia[1], Antonio Estruch[1], Oscar Coltell[2], David Pérez del Rey[3], Guillermo de la Calle[3], Juan Pedro Sánchez[4], and Ferran Sanz[5]

[1] Departamento de Tecnología, Universitat Jaume I, Castellón
[2] Systems Integration and Re-Engineering Research Group, Universitat Jaume I
[3] Biomedical Informatics Group, Department of Artificial Intelligence, Universidad Politécnica de Madrid
[4] Área de Bioinformática y Salud Pública, Instituto de Salud Carlos III, Ministerio de Sanidad y Consumo, Madrid
[5] GRIB, IMIM/UPF Barcelona, Spain

Abstract. Following a systems engineering approach we have identified the information system requirements for biomedical collaborative research. We have designed a Service Oriented Architecture following a dynamic and adaptable to change approach, using technology and specifications that are being developed in an open way, utilizing industry partnerships and broad consortia such as W3C and the Organization for the Advancement of Structured Information Standards (OASIS), and based on standards and technology that are the foundation of the Internet. The design has been translated in a pilot implementation infrastructure (INBIOMED) that is now been populated with web services for data and images analysis and collaborative management.

1 Towards the Collaborative Management of Biomedical Research

Publishing and sharing the research results is something inherent to the scientific method and has been performed since the dawn of science by means of written publications. Thanks to the possibilities offered by electronic formats it has become possible to reduce the time that elapses before they see the light, and use of the network makes them far more easily accessible. At the same time, in many disciplines the academic journals require researchers to make the data they used to reach their conclusions publicly available and some communities have even set up large pools of experimental data. These data, as a manifestation of the communities that generated them, are unconnected and the fact that they were produced by different technologies makes it difficult to integrate them. This basic level collaboration can be understood as a **first level of collaboration** in which the model of information exchange is limited to making both the data and the findings publicly available (table 1).

Table 1. Levels of collaboration and models of information exchange

Level of collaboration	Model of information exchange
Work in isolation	Publication of data and findings
Coordinated projects	Sharing applications
Collaboration processes	Sharing information system

J.L. Oliveira et al. (Eds.): ISBMDA 2005, LNBI 3745, pp. 252–261, 2005.

Yet, it is becoming increasingly clearer that isolated biomedical research does not lead to any significant solutions in clinical practice. Thus, the current trend is for public institutions to back research projects involving increasingly large numbers of research groups in order to be able to make more efficient use of the scarce resources available. At the same time, these research groups participate in networks where they try to coordinate with other groups by sharing knowledge as well as tangible resources. This approach to research is very recent and we are at present in a phase in which cultural changes are taking place, and structures and mindsets are therefore still being adjusted to the new environment. This means that, although in general terms research groups appear to be assuming this new situation and are willing to adapt to it, for the time being many shortcomings can be observed in the incipient processes of collaboration. Indeed, collaboration is often essentially limited to sharing funding and findings. In this phase, which is the one that currently predominates, we would be in a **second level of collaboration** where the models of information are characterised by sharing applications, as well as data and findings.

The challenges biomedical research faces in the medium and long term require large-scale coordinated projects involving collaborative research processes that really break down the walls between different departments. These projects will be fostered and will require the use of suitable infrastructures for computing and communication over the Internet – *cyberinfrastructures* [1] – that will allow both hardware and software resources to be shared in a network.

The fact is that reaching an understanding of the molecular mechanisms underlying diseases, such as the different types of cancer, is now seen as an increasingly more distant objective. Just as different research groups deal with their work in depth, the gap between what they currently know and what they actually need to know appears to be getting bigger. We think that required experimental technology appears to be already available, although it will be continually improved and new techniques will be proposed. However, the real challenge is to manage the complexity and scale of the problem being tackled: the organisational complexity that entails cultural changes related to managing research, and the huge amount of data generated by technologies used in molecular scale experiments. Handling such information requires sophisticated database management systems, in addition to complex data mining algorithms for its analysis. For certain applications, it is believed that it will be necessary to use Grid-based computing services [2].

The processes involved in disease research include research projects that are interrelated on different scales and which focus on a variety of aspects (molecular, cellular, organs, individuals, family, community, population). These kinds of research are usually carried out by researchers from different disciplines, with different languages and cultures, and this will also make it more difficult to integrate the research processes.

Hence, as happens in other scientific and technological disciplines, it appears that we are entering a new age in which it will be necessary to work in a large-scale collaborative manner with many other groups. We will therefore become part of a management process with common final goals and which shares large amounts of resources that enable extensive, costly and well designed experiments to be conducted in order to reach conclusive outcomes. This is what many call *Big Science*. At first, perhaps large emblematic projects will be implemented (by means of platforms), but

alongside such work individual research will undoubtedly continue to be carried out. These studies will also have to benefit from collaborative work with other groups that are dealing with the same issues.

A paradigmatic example of the collaborative management of a large project was the Cooperative Human Linkage Center (CHLC) consortium set up as part of the genome project with the intention of creating gene maps [3]. The CHLC acted as a geographically distributed virtual centre that connected small highly specialised laboratories by means of an Internet-based computer system. Each group worked within the context it was familiar with and the different parts were put together to make maps using predefined workflows. To do so, when necessary, they also used distributed processing over a network of computers. The data, intermediate analyses and maps were distributed over the Internet by web servers.

This example is a proof-of-concept that the key objectives of collaborative management can be reached even with previous-generation computer technology [1]. The design and management of the research processes are even more important than specific computer solutions. The computer systems, although insufficient, like any technological tool, are necessary.

Collaboration during the research process requires the sharing of information systems, that is to say, working on the same information system or on systems that are truly interoperable so that the information flows from one group to another in the most effective and efficient manner.

2 The Service Oriented Architecture

The information technology industry is basing on web services technology the construction of the latest generation of distributed computer systems. This technology allows the internal resources to be encapsulated within the service, thus providing a logical application layer between those resources and their consumers. The owners of the services can modify the resources as time goes by without the need to carry out changes in the messages used to communicate with other consumer services, which means that they can continue to interoperate as they did before the modifications were introduced. This independence among the parts that go to make up a system makes it possible to build very robust systems that are open and in continual development, without the need for anyone to be in control of the system.

The specific way that certain information architects design the basic system or infrastructure that is going to be built with web services is called, as is only natural, Service Oriented Architecture (SOA). The services in this architecture are loosely coupled [4], since one application does not necessarily have to know the technical details of another – in fact, it does not even need to know which platform or language it has been programmed in. In contrast, however, if they are to be able to communicate with one another they must have well and clearly defined communication interfaces. Although Web services technology is not the only approach to realizing an SOA, it is one that the IT industry as a whole has enthusiastically embraced. With Web services, the industry is addressing yet again the fundamental challenge that distributed computing has provided for some considerable time: to provide a uniform way of describing components or services within a network, locating them, and accessing them. The difference between the Web services approach and traditional ap-

proaches (for example, distributed object technologies such as the Object Management Group – Common Object Request Broker Architecture (OMG CORBA), or Microsoft Distributed Component Object Model (DCOM)) [5] lies in the loose coupling aspects of the architecture. Instead of building applications that result in tightly integrated collections of objects or components, which are well known and understood at development time, the whole approach is much more dynamic and adaptable to change. Another key difference is that through Web services, the IT industry is tackling the problems using technology and specifications that are being developed in an open way, utilizing industry partnerships and broad consortia such as W3C and the Organization for the Advancement of Structured Information Standards (OASIS), and based on standards and technology that are the foundation of the Internet.

Many developers, architects, managers and academics still see web services as the next episode in the continued saga of distributed object technologies such as CORBA, DCOM and RMI. Web services and distributed objects systems are both *distributed systems* technologies, but that is where the common ground ends. They have no real relation to each other, except maybe for the fact that web services are now sometimes deployed in areas where in the past the application of distributed objects has failed. If we look for relationships within the distributed technology world it is probably more appropriate to associate web services with messaging technologies, as they share a common architectural view, but address different types of applications.

Web services are based on XML documents and document exchange, and as such one could call the technological underpinning of web services *document-oriented computing*. Exchanging documents is a very different concept from requesting the instantiation of an object, requesting the invocation of a method on the specific object instance, receiving the result of that invocation back in a response, and after a number of these exchanges, releasing the object instance [6].

2.1 Service Oriented Architecture Technologies: XML and WEB Services

The foremost technology for the description and interaction of services is based on XML and HTTP, although extended with a set of specifications. From which the most widely used are SOAP and WSDL. SOAP (Simple Object Access Protocol) is a "binding protocol" utilised to coordinate the query/response interactions among Web services. WSDL (Web Services Description Language) specifies how to describe a service using a document with a special XML format. The big advantage of using XML standards for Web services is that all the computer platforms (such as Microsoft, IBM, Oracle, BEA, Apache) support them.

The World Wide Web Consortium (W3C), which has managed the evolution of the SOAP and WSDL specifications, defines Web services as follows: "*A software system designed to support interoperable machine-to- machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with XML serialization in conjunction with other Web-related standards"[7]*

Along with the standards for describing and the transfer of Web services, it is also necessary to use methods that enable communication to be performed with high levels of security by encrypting the messages. The standard that is currently becoming the

most popular is the VPN (Virtual Private Network). The VPN (Virtual Private Network) concept was to produce the virtual "dedicated circuit", pump it over the internet, and use cryptography to make it secure providing protections against eavesdropping and active attacks. Different alternatives are available to achieve secure networking in VPNs. IPSec was the first major effort to develop a standard. Traditional IPSec implementations required a great deal of kernel code, complicating cross-platform porting efforts. By contrast, other standards like SSL (Secured Sockets Layer) matured quickly, due to heavy usage on the web. SSL runs in user space, simplifying implementation and administration.

2.2 Layers of the Service Oriented Architecture

The basic functionalities of a SOA on a conceptual level are usually classified in layers. We propose the following three layers.

On the lowest level there are the functions and components that allow the infrastructure to connect with other services and sources of data, which is why this is usually called the *connectivity layer*. On top of the connectivity layer, the *orchestration layer* allows us to organise the different services and data sources that are accessed previously to enable the composition of the workflows. These workflows are the ones that allow us to reach the intended goal by implementing the information system. These may include the different workflows needed for collaborative management, knowledge management, project management, querying distributed databases, grid computing [8], or transactional processes.

The services made available through the infrastructure constitute a catalogue that must be administered so that service suppliers and customers can locate and communicate with each other. In addition to the services, the other resources that are managed by the infrastructure, such as data sources and users, also need to be administered. The architecture must therefore include capabilities that allow users to subscribe and unsubscribe, as well as the management of roles and the access privileges to the different resources. This is the basic function of the *administration layer*.

The distinct layers can be made functional by means of specialised services and/or through a low level language of the infrastructure itself that glues the different layers together. This SQL-type language makes the system very strong and flexible.

The infrastructure can be considered to be middleware, since it can be seen how it is situated between the interfaces with the people and the services and sources of data that supply the logic and the data (Figure 1).

The middleware infrastructure provides a common set of tools and services that enable both people and other applications to behave as if the computation, the data pools and other distributed resources were all part of one large virtual system. This common service infrastructure simplifies the development of applications, enhances robustness and interoperability, and increases efficiency in the development and maintenance of information systems.

3 The INBIOMED Infrastructure

Based on this Service Oriented Architecture approach, we have developed an infrastructure suited to biomedical research that is currently being enhanced in a pilot

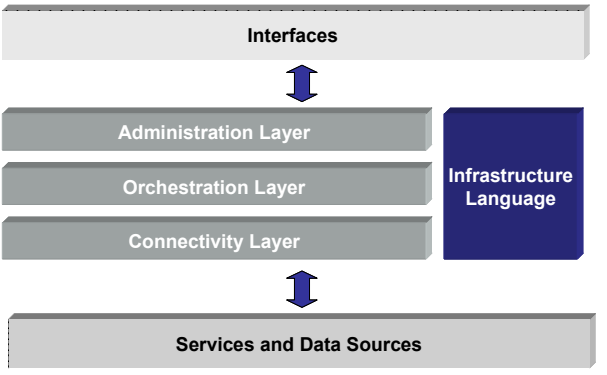

Fig. 1. Diagram of the Service Oriented Architecture

experience by the incorporation of data and imaging analysis services, collaborative management services [9], distributed databases (CaCore [10], SRS[11]), interfaces, administration applications, ontology managers, workflow tools, and so forth.

The software development is guided by the objectives and requirements identified by the INBIOMED research network [12] and the Web Services Architecture Working Group [13]. The INBIOMED research network is made up of several bioinformatics research groups that, in turn, collaborate with prestigious biomedical research groups both in Spain and around the world, from whom requirements have been gathered through personal interviews, dedicated workshops and surveys.

The system has as nucleus a Site Web Service that serves as a point of entry to the client applications. It receives the query and processing requests in a language oriented to the batch execution called IQL (Inbiomed Query Language). This language is the key to communicate with the platform. A more detailed description of how the infrastructure was implemented and the language definition can be found in [14].

3.1 The INBIOMED Query Language

Every request to the Site Web Service is made by calling a unique method (see figure 2), where must be specified, at least, the sentences that have to be executed, in a string format. In that way, it is consolidated a basic communication interface that will not be modified by the addition of new functionality to the IQL language.

```
String Login(String a_sLoginName, String a_sPassword)
   Starts a new session for the specified Login.
String ExecuteQuery(String a_sGuidSession, String a_sQuery,
String[] a_dsArguments, Boolean a_bWait)
   Executes an IQL script with the specified arguments,
   instructing the Executing Engine to wait the end of the
   script execution to return his result, or return the Guid of
   the associated thread to query his result later.
String Logout(String a_sGuidSession)
   Close the specified session from the client.
```

Fig. 2. Site Web Service calling method description

Every request received is executed in a separated thread, so that can be controlled its execution separately. The sentences are executed in a sequential way, allowing in one request ask for data from several data sources and call a data processing service that processes the data obtained, delegating all the processing load in the server. Input data included as part of the request are sent to the server with the query sentences that reference these input data as parameters.

All the data manipulation in IQL is done using an ADO.NET data type called DataSet [15]. This data type can contain traditional relational information including tables and relations in memory, or scalar information implemented like one table with one column with one row of data. A lot of operators had been implemented in IQL to work with this data type, including queries with joins and filters, mathematical and string operators.

To make easy the data manipulation, during the execution of the IQL sentences, variables can be defined to store temporally partial results and be reused during the execution of a sequence of instructions. Once the sentences execution is finished, the variables are removed. In case that you want to store the results to use them later, there is a Persistent Data Repository, which has an analogue structure to a conventional file system with folders and subfolders and with archives that store the result of the execution of IQL sentences. This sentences can be saved and load whenever, whether the user has the proper rights.

As the execution of a sequence of IQL sentences can take a long time, the system can leave running the query in the server and return an ID to the client, that can be used by the client to interrogate to the system about the state of the query, and when the query is finished to recover the result. When the client waits until the end of the execution the result is returned to the client that made the request.

For complex tasks, IQL Procedures can be declared containing other IQL sentences to access to Data Sources located in several nodes, do basic data manipulation, process data calling Web Services located in other nodes, do flow control if necessary and return only the final result. These Procedures are stored and can be executed lately when is needed or be called from other stored Procedures.

The language IQL contains basic sentences to query data from local data sources with ADO.NET or from remote data sources managed by other Site Web Services. ADO.NET grants the accessibility to data through .NET managed providers, OLE DB Providers or widely used ODBC drivers. To call Data Processing Web Services, SOAP is used to invoke the standard method passing all data in strings with XML serialized data to ensure maximum interoperability from different development platforms.

The client applications only have to implement the User Interface with their input forms and the presentation results, delegating in the infrastructure the responsibility to query, store and process the information as much as possible.

The communication between the client applications and the Site Web Service must be made using the SOAP protocol, so the applications have to be implemented using a programming language that is able to consume Web Services. The common Data Model will establish the data structures used in the data flows between the several layers.

As part of the platform development, among others, an administrative tool has been developed named INBIOMED Manager (figure 4), that facilitates the connection with

the platform, throwing the execution of IQL sentences and has the necessary options to manage all the elements in the platform such as users, roles, data source connections, or the Data Process Services Catalogue, among others.

```
SET @var = expr;
IF (boolexpr)
{
truesentences;
}
ELSE
{
falsesentences;
}
WHILE boolexpr DO
{
loopsentences;
BREAK;
CONTINUE;
}
CREATE PROCEDURE myproc (@arg1, …)
{
bodysentences;
RETURN expr;
}
SET @var = CALL myproc (expr, …)
```

Fig. 3. Complex Procedure structure example with IQL

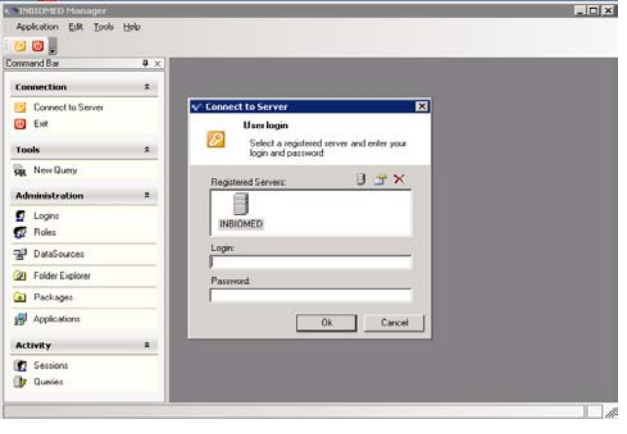

Fig. 4. INBIOMED Manager typical screen

4 Conclusions: Challenges for Collaborative Research in Biomedicine

After developing the computer infrastructure and beginning to use it in the first pilot trials, one of the fundamental conclusions we reached was that adopting collaboration processes in biomedical research has to overcome two basic obstacles, one of a technological nature and the other involving organisational aspects.

Infrastructures developed following a SOA facilitate interoperability among computer systems but do not completely resolve the problem. If a service is to operate as part of an infrastructure it will have to comply with the standards languages that have been specified for that particular system. Therefore the bioinformatics community will have to define its own standards for collaborative management processes. Such standards will have to go beyond one-off developments such as MAGE-ML [16] (for representing microarray data). These specifications will represent yet another step towards the development of software that is designed to be interoperable from its earliest stages, but until such applications become available the services and databases that are developed for other infrastructures will have to be adapted manually.

Moreover, research process models must also be elaborated to guide the development of specifications. Unlike the field of business management, where work has been carried out for decades on the development of a process perspective (which was initially internal and more recently has become focused on inter firm collaboration), very little has been done to define the processes involved in biomedical research. The term *processes* is understood to mean all the activities that are needed to accomplish a particular objective, their interrelations and the resources that are used at each stage [17]. In order to manage processes we need an information system and an organisational structure. To date, the need to carry out research into suitable models of organisational structures and research processes and their representation in formalised languages has not been covered. The progressive definition of biomedical research process models will provide us with a clearer idea of what scientific workflows [18] are more relevant in biomedicine, as well as the priorities in the development of specifications. Moreover, the reference models, which should include best practices, can be used by the scientific community to gain a deeper and easier understanding of the advantages of collaboration, different ways of approaching it and the organisational structures that are best suited to each project.

The collaborative research in Science needs interoperable information systems to share data, applications, computational resources, and workflows, but technology is not enough. We have develop an infrastructure following a Service Oriented Architecture that can be used to integrated research processes as a proof of concept that technology does not represent a integration barrier. The Web Services technology facilitates interoperability and foster systems integration with flexibility and efficiency. Although there are technical challenges to be solved with this new technology (mainly interoperability issues between different platforms and performance), what is most needed is to develop research process models and organizational approaches to guide the cultural change, from an isolated to a collaborative way of doing science.

References

1. Kenneth H. Buetow. Cyberinfrastructure: Empowering a “third way” in Biomedical Research. Science. Vol 308. pag. 821-824. 6 May 2005.
2. Oliveira IC, Oliveira JL, Sanchez JP, Lopez-Alonso V, Martin-Sanchez F, Maojo V, Sousa Pereira A. Grid requirements for the integration of biomedical information resources for health applications. Methods Inf Med. 2005;44(2):161-7
3. J.C. Murray et al. A comprehensive human linkage map with centimorgan density. Science 265, 2049-2054. 1994.4. D. Booth et al., Web Services Architecture (WC, Working draft, 2003; /www.w3.org/TR/ws-arch/)

4. Sanjiva Weerawarana, Francisco Curbera, Frank Leymann, Tony Storey, Donald Ferguson. Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging, and More. Prentice Hall PTR.2005
5. Werner Vogels. Web services are not distributed objects. IEEE Internet Computing. 59-66.Nov-Dec 2003.
6. Hugo Haas, W3C Web services Working Group. Web services glossary. http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/
7. I. Foster. The grid: computing without bounds. Sci Am. 2003 Apr; 288(4):78-85
8. Manuel Pastor, Paolo Benedetti, Angelo Carotti, Antonio Carrieri, Carlos Diaz, Cristina Herraiz, Hans-Dieter Höltje, M. Isabel Loza, Tudor Oprea, Fernando Padin, Francesc Pubill, Ferran Sanz, Friederike Stoll & the LINK3D Consortium. Distant collaboration in drug discovery: The LINK3D project. Journal of Computer-Aided Molecular Design, 16: 809–818, 2002.
9. Cacore. http://ncicb.nci.nih.gov/NCICB/core
10. SRS. www.lionbioscience.com
11. Jose A. Heredia. Plataforma Inbiomed: Objetivos y Arquitectura. 21-31. Informática Biomédica. Madrid. 2004. Editorial: Inbiomed.
12. Daniel Austin, Abbie Barbir, Christopher Ferris, Sharad Garg, Web Services Architecture Requeriments. http://www.w3.org/TR/wsa-reqs
13. Estruch, A., Heredia J.A., "Technological platform to aid the exchange of information and applications using web services" Lecture Notes in Computer Science, 3337, 458-468, Nov. 2004.
14. NET Framework Class Library. DataSet Class. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpref/html/frlrfsystemdatadatasetclasstopic.asp
15. Mage-ML. www.mged.org
16. José A. Heredia. Modelo de información de los procesos de análisis en la investigación de patologías.223-247. Informática Biomédica. Madrid. 2004. Editorial: Inbiomed.
17. William W. Stead, Randolph A. Miller, Mark A. Musen, and William R. Hersh Integration and Beyond: Linking Information from Disparate Sources and into Workflow. J Am Med Inform Assoc. 2000 Mar–Apr; 7(2)**:** 135–145.