LEICA: Loosely-coupled Environment for Integrating Collaborative Applications

Roberta Lima Gomes [+]
LAAS – CNRS
rgomes@laas.fr

Guillermo de Jesús Hoyos Rivera[§]
LAAS – CNRS
ghoyos@laas.fr

Jean Pierre Courtiat
LAAS – CNRS
courtiat@laas.fr

Abstract

Group activities generally involve several people presenting different collaborative tasks and requirements. Accordingly, collaboration is rarely supported by a single CSCW technology. Different types of collaborative applications are then applied in order to support group work. But in spite of being used with a common purpose, applications are executed independently. The integration of such applications would allow them to dynamically interoperate, combining their different functionalities in a controlled way. To achieve integration, we propose LEICA[*], a loosely-coupled integration environment which allows collaborative applications to interact without loosing their autonomy.

1. Introduction

As nowadays organizations become widely distributed, they present a growing demand for collaboration technologies. Accordingly, different collaborative applications, or CSCW (Computer Supported Cooperative Work) systems, have been developed.

Since collaborative activities can present different requirements, they are usually supported by multiple technologies. Thus some CSCW systems try to merge different functionalities into a single environment. The weakness of this approach is the challenge behind the anticipation of all the requirements of cooperative situations involving several people, with different group tasks and needs.

In practice, current collaborative environments consist of a range of applications, working side by side but independently, without getting advantage of each other. The integration of these applications could bring significant benefits to users: different functionalities of existing applications could be dynamically combined and controlled (enhancing flexibility/tailoring possibilities).

In order to achieve integration of existing CSCW systems avoiding to deal with their low-level features, we propose LEICA, a "Loosely-coupled Environment for Integrating Collaborative Applications". Relying on Web services technology [1] and an event notification system, different collaborative applications can interoperate (or interact) by exchanging information within the context of a global *SuperSession*. Interactions are controlled by a set of collaboration policies defining how the collaborative activity supported by one application is affected by information received from other applications.

The paper is structured as follows. Section 2 presents related work regarding the integration of CSCW systems. Section 3 overviews the integration framework proposed by LEICA. Section 4 describes how a *SuperSession* is configured. Section 5 explains LEICA's architecture and some implementation issues. Section 6 draws some conclusions and perspectives.

2. Related Work

The work presented in [2] addresses basic issues in interoperating heterogeneous CSCW systems that concurrently manipulate the same artifacts (like text documents). However, it does not regard the interoperation of CSCW systems that, despite being involved in the same collaborative tasks, do not deal with the same artifacts.

In [3] authors propose an integrative framework based on a three-level model: ontological, coordination and user interface. An internal knowledge of the collaborative application is needed so that its functionalities can be mapped into the three-level model in order to achieve integration. Accordingly, the integration of third party applications becomes a complex (even impossible) task.

[+] Supported by a scholarship from CNPq, Brazil.
[§] Supported by scholarship 70360 from CONACyT /PROMEP, Mexico.
[*] Research partially supported by the IST project *Lab@Future*.

In order to avoid dealing with internal implementation details of applications (allowing then the integration of existing applications), some environments have employed a loosely-coupled integration approach. Loosely-coupling overcomes two problems related to integration environments: (i) it does not require a true semantic integration of applications, (ii) once integrated to the environment, applications can keep their autonomy.

AREA [4] and NESSIE [5] have proposed a loosely-coupled integration for supporting cross-application awareness. Like LEICA, these systems are based on the exchange of activity relevant events. However, these environments only aim to provide users with a common awareness of the whole collaborative activity.

Another integration solution also based on a loosely-coupled approach is presented in [6]. The authors define a framework where Web services are used to wrap collaborative applications in order to integrate them. Since they leverage open Internet standards, Web services overcome the interoperability issues associated with general integration solutions like CORBA and EJB. Web services based integration is also simpler to develop and quite flexible, as it is built on loose coupling.

A drawback presented by the Web services wrapping approach (in particular related to the use of SOAP [1]) is that it represents an additional tier causing some overhead in processing messages [7]. Besides, depending on the architecture of collaborative applications, the complete wrapping of these applications as Web services may imply great development efforts and even applications redesign.

Therefore, following the recommendations of [7], we decided to use Web services for coarse-grained operations only. Thus, LEICA applies Web services as an initial mechanism for (i) registering newly integrated applications, (ii) creating and starting up integrated collaborative sessions (*SuperSessions*). Then, a different infrastructure is used to implement the *Event Notification System* in charge of interconnecting applications during the *SuperSession* execution.

3. LEICA Overview

LEICA is an integration environment where collaborative applications can interact through the exchange of event notifications. According to a set of predefined collaboration policies, these notifications may lead to performing specific actions in these applications.

The interaction degree among integrated applications depends obviously on the nature of the events they are able to exchange, and actions they are able to perform. Three main cases may be considered when integrating applications: a) open source applications, b) API-based applications, and c) applications without any API. Integration of open source applications can achieve the tightest interaction degree, since any internal event/action can be exported/performed; it might however imply great development efforts. API-based applications integration is straightforward, and interaction is limited to the provided API. Applications without API are constrained to interact only through application start and stop actions. LEICA's integration approach has been mainly driven by case (b). We believe that developers are certainly interested in creating specific and performable collaboration tools that can be used either stand-alone or integrated with other applications (through a flexible API), thus being able to get a great share of the market. This is for instance the case of Skype™ [8], a successful example of communication tool that has recently released its API.

In order to integrate collaborative applications to LEICA, a *Wrapper* must be attached to this application. Each *Wrapper* comprises a Web services interface allowing the collaborative application to register itself with LEICA as an integrated application. Through its Web services ports, the integrated application can interact with the *Session Configuration Service*.

The *Session Configuration Service* is a Web service used to create and start up new *SuperSessions*. A *SuperSession* is an integrated collaborative session holding the whole collaborative activity. Within the context of a *SuperSession*, different *specificSessions* can exist. A *specificSession* is a conventional collaborative session defined within one collaborative application (*e.g.* a videoconference or a shared whiteboard session).

During the *SuperSession* configuration, the *Session Configuration Service* dynamically contacts each integrated application in order to request them specific information about its communication API. This information will be used to create *specificSessions* and to specify the collaboration policies of a *SuperSession*.

Collaboration policies are a set of rules following a condition/action model. These rules define how collaborative applications might react when events coming from other collaborative applications are notified. Thus the collaboration polices specify the desired coupling between the integrated applications and hence the dynamic behavior of a *SuperSession*.

Once a *SuperSession* is created and configured, the *Session Configuration Service* can finally start it. Then the integrated applications (that will be used in this *SuperSession*) are interconnected through an event notification service. As previously explained, from this point Web services are not used anymore.

As collaborative activities progress, *Wrappers* exchange event notifications in a P2P fashion. Meanwhile, they are in charge of managing the collaboration policies: as events are notified, policy rules are enabled and action requests are sent to the respective application.

Note that LEICA is not intended to support low-level physical events (*e.g.* mouse events) or high frequency synchronization events (*e.g.* position of moving objects). It aims to support meaningful events that carry some semantics. For example, consider the integration of a shared 3D world with a collaborative web browsing tool. Suppose that we want to represent in the 3D world what is happening inside the web browsing session. In this case, each time the group browses a new web page, an event of type "NEW_URL" is notified. Then a policy rule associates this event notification to an action requesting the insertion of a 3D text object inside the 3D world.

4. *SuperSession* Definition

The *SuperSession* configuration process is accomplished in two steps: (i) Session Management and (ii) Collaboration Policies configuration.

4.1 Session Management Configuration

In the first configuration step, two groups of information are provided:

- *General Session Management information* (*GSMinfo*). It carries general management information such as scheduling, membership and user roles.
- *Integrated Applications information* (*IAinfo*). It defines the list of integrated applications to be used during this *SuperSession*. For each application, a list of *specificSessions* is defined, where specific data required by this application for creating sessions is provided (*e.g.* a videoconference application is provided with an IP multicast address).

4.2 Collaboration Policies Configuration

The second step of the configuration process deals with the specification of the collaboration policies. These policies are responsible for linking the collaborative activities supported by different *specificSessions* in the context of the global *SuperSession*. This is carried out through a set of rules associating *n* event notifications to the execution of *m* actions (under certain conditions).

In [9] we describe how policy rules are graphically specified through the composition of *policy widgets*. Based on the graphic specification, a XML specification is generated which is appended to the respective *SuperSession* configuration file. The rules' semantics associated with the XML syntax has been defined using the RT-Lotos formal description technique [10]. This semantics is implemented by a *Wrapper*'s sub-module.

5. LEICA Development

We describe here the LEICA's architecture and different implementation aspects of a prototype currently under development.

5.1 LEICA's Architecture

To precisely describe the main architecture elements, let us consider the five necessary steps to achieve the execution of a *SuperSession*.

a) Integrating a Collaborative Application

Three different distribution architectures are considered when integrating collaborative applications: (i) client/server, (ii) multi-server, and (iii) P2P applications.

When integrating client/server or multi-server collaborative applications, a *Server Wrapper* must be added to the server(s). In the case of a P2P application, a *P2P Wrapper* is used. The difference between these two *Wrappers* (figure 1) regards the *Web services* (WS) interface, not present in the second case. Since P2P applications are usually executed when users get connected, they cannot be permanently available as Web services. To overcome this problem, a *P2P Proxy* is used.

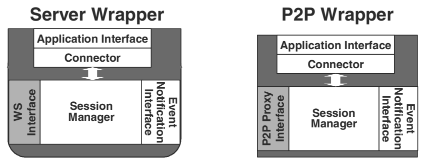

Figure 1. The LEICA *Wrappers*

To decrease development efforts when integrating applications, a special module called *API Factory* is defined. This module works as a *Wrapper* factory: based on XML specifications of application's API, it generates a *Wrapper* adapted to it. Adaptation concerns the creation of an *Application Interface* sub-module defining all the necessary methods (corresponding to the events/actions API) for communicating with the application. The *Application Interface* is the component of the *Wrapper* which to be tied to the collaborative application.

The *Session Manager* implements the core functionalities of the *Wrapper*. It is in charge of (i) receiving and handling *specificSession* configuration data; (ii) managing the collaboration policies; and (iii) sending event notifications to other applications.

b) Registering a Newly Integrated Application

To register with LEICA, the *Wrapper* publishes its services in a *Private UDDI Registry* (figure 2). In multi-server applications, a *Master Server* is designated to register the application. In P2P applications, registering is made through the *P2P Proxy*.

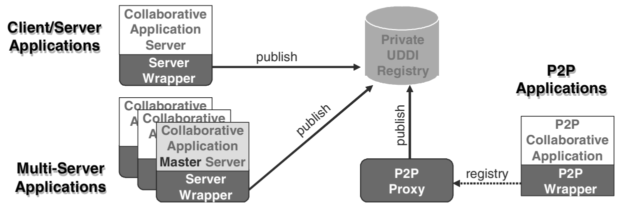
Client/Server Applications
Collaborative Application Server
Server Wrapper
publish
Private UDDI Registry
P2P Applications
Multi-Server Applications
Collaborative Application Master Server
publish
publish
P2P Proxy
registry
P2P Collaborative Application
P2P Wrapper

Figure 2. Registering collaborative applications

c) Creating SuperSessions

Figure 3 schematizes the necessary steps for creating a new *SuperSession*.

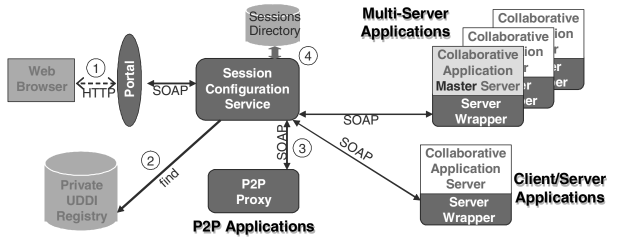
Web Browser
1
HTTP
Portal
SOAP
Session Configuration Service
Sessions Directory
4
Multi-Server Applications
Collaborative Application Master Server
Server Wrapper
SOAP
SOAP
3
SOAP
2
find
Private UDDI Registry
P2P Proxy
P2P Applications
Collaborative Application Server
Server Wrapper
Client/Server Applications

Figure 3. Creation of new *SuperSessions*

1. A Web portal is used to access the *Session Configuration Service* and start the creation process.
2. The *Session Configuration Service* accesses the *Private UDDI Registry* to find out which are the integrated collaborative applications.
3. Each integrated application is contacted and asked about which specific data it needs to create *specificSessions*, and which kind of events it can notify and action requests it can treat.
4. At the end of the configuration process, a *SuperSession* configuration file is created and stored.

d) Starting SuperSessions

Figure 4 illustrates how a *SuperSession* is started.

1. A Web portal is used to start a *SuperSession*.
2. The *SuperSession* configuration file is retrieved and parsed. The collaborative applications to be used in this *SuperSession* are identified.
3. The *Session Configuration Service* contacts the concerned *Server Wrappers* to set up s*pecificSessions* and to send the collaboration policies of this *SuperSession*.
4. *Server Wrapper*s are interconnected through the *Event Notification System* (Web services are not used anymore).

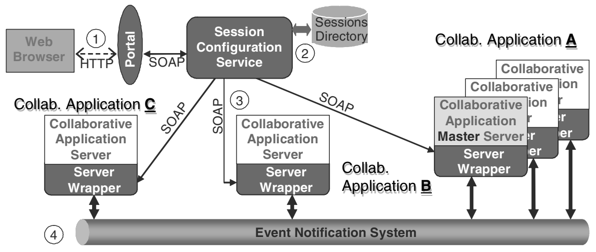

Figure 4. Starting up a *SuperSession*

The *Event Notification System* used by LEICA is based on the publish/subscribe paradigm [11]. Publish/subscribe interaction is really well adapted to loosely-coupled environments. Subscribers register their interest in patterns of events and then asynchronously receive events matching these patterns, regardless of the events' publishers.

After connecting to the *Event Notification System*, each *Wrapper* analyses the collaboration policies in order to discover: which type of events it needs to publish, and which type of events it needs to subscribe to. A *Wrapper* just needs to publish *Event*s that could enable policy rules, and subscribe to *Event*s that could enable a policy rule defining an *Action* to its associated application.

e) Connecting to a SuperSession

In order to connect to a *SuperSession*, users must run the *LClient* application. Figure 5 shows how the connection of a new user is treated.

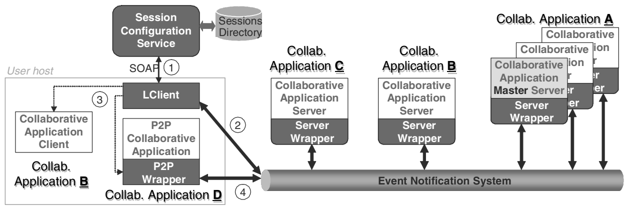
Session Configuration Service
Sessions Directory
User host
SOAP
1
LClient
3
Collaborative Application Client
Collab. Application B
P2P Collaborative Application
P2P Wrapper
Collab. Application D
2
4
Collab. Application C
Collab. Application B
Collab. Application A
Collaborative Application Server
Server Wrapper
Collaborative Application Master Server
Event Notification System

Figure 5. User connection to a *SuperSession*

1. The *LClient* contacts the *Session Configuration Service* and it receives the *GSMinfo*, *IAinfo*, and collaboration policies defined to the chosen *SuperSession*.
2. The *LClient* plays the role of a local launch point for P2P and client applications. As it needs to execute the collaboration policies in order to know when P2P/client applications must be launched, it joins the *Event Notification System* to also receive event notifications.
3. Suppose that, initially, this user must be connected just to two *specificSessions*, concerning the collaborative applications "B" and "D". Then, the *LClient* runs the client application "B" and the P2P application "D".
4. The *Wrapper* of the P2P application "D" connects to the *Event Notification System* and executes the same publishing/subscribing process described for *Server Wrappers* in step *d*.

5.2 Design and Implementation Issues

In order to get a concise and correct implementation of LEICA, we have specified a design model of the environment. To design LEICA's architecture entities, the UML-based modeling tool TAU G2 [12] has been used.

During the design process, a top-down development approach was adopted. Architecture elements have been decomposed and dynamic behaviors have been defined for each sub-component. Simulations using the UML case tool have been carried out in order to validate the model.

Based on the UML model, LEICA's first prototype has currently been developed. Java™ has been chosen as underlying technology[1]. Special Java APIs have been used in order to implement the Web services interactions:

- Apache jUDDI for implementing the *UDDI Registry*;
- the *Wrapper*'s *WS Interface* and the *P2P Proxy* use (i) UDDI4J (from IBM) to interact with the *UDDI Registry* and (ii) Apache Jakarta Tomcat 5.0 and Apache SOAP 2.3.1 to interact with the *Session Configuration Service*;
- *LClient* also uses Apache SOAP 2.3.1 to contact the *Session Configuration System*.

To implement the *Event Notification System* based on the publish/subscribe approach, the Scribe [13] infrastructure has been used. Scribe implements a large-scale, peer-to-peer, topic-based publish/subscribe approach with an efficient application level multicast.

6. Conclusions and Future Work

This paper has presented LEICA, a loosely-coupled environment for integrating collaborative applications. Existing collaborative applications can be integrated using Web services as integration technology. In the context of a *SuperSession*, a global collaborative activity is supported where different integrated applications are used in a parallel and coordinated way. Based on the specification of collaboration policies, LEICA defines applications' behavior in response to event notifications.

In order to test the LEICA's prototype, two open source applications are currently being integrated: a collaborative web browsing tool (CoLab[14]) and a multi-rooms chat tool (Babylon [15]). Further, API based applications will be integrated (*e.g.* Skype). The current prototype implementation confirms the fact that open source collaborative applications achieve richer interaction levels since we have all the needed flexibility for binding *Wrappers* to applications. However, as new software applications are increasingly coming out of the box with API specifications (sometimes based on Web services), their integration with LEICA tends to be straightforward.

Regarding Web services, an important effort has been deployed to propose solutions for optimizing the transmission and/or wire format of SOAP messages. In this perspective, the current *Event Notification System* of the LEICA could also become a Web services-based system without the performance problem actually inherent to SOAP.

[1] Java Native Interface is used for integrating non Java applications.

7. References

[1] Web Services Activity. http://www.w3.org/2002/ws/

[2] Dewan, P. "An experiment in interoperating heterogeneous collaborative systems". *6th European Conference on Computer Supported Cooperative Work*, Copenhagen, 1999.

[3] Iqbal, R., James, A., Gatward, R. "A practical solution to the integration of collaborative applications in academic environment". *5th International Workshop on Collaborative Editing Systems*, Helsinki, 2003.

[4] Fuchs, L. "AREA: a cross-application notification service for groupware". *6th European Conference on Computer Supported Cooperative Work*, Copenhagen, 1999.

[5] Prinz, W. "NESSIE: an awareness environment for cooperative settings". *6th European Conference on Computer Supported Cooperative Work*, Copenhagen,. Kluwer Academic Publishers, 1999.

[6] Fox, G. et al. "A Web services framework for collaboration and videoconferencing". *Workshop on Advanced Collaborative Environments*, Seattle, Washington, 2003.

[7] Alonso, G., Casati, F., Kuno, H., Machiraju, V. "Web Services" (chapter 5). In Web Services – Concepts, Architectures and Applications, *Springer Verlag*, 2004.

[8] Skype website. http://www.skype.com/

[9] Gomes, R.L., Hoyos-Rivera, G.J., Courtiat, J.P. "Integrating Collaborative Applications with LEICA". *3rd IEEE International Conference on Information Technology: Research and Education* (ITRE 2005), Taiwan, Juin, 2005

[10] Courtiat, J.P., Santos, C.A.S, Lohr, C., Benaceur, O. "Experience with RT-LOTOS,a temporal extension of the LOTOS formal description technique". *Computer Communications*, v.23, n.12, pp.1104-1123, July 2000.

[11] Eugster, P., Felber, P., Guerraoui, R., Kermarrec, A.-M. "The many faces of publish/subscribe". *ACM Computing Surveys*, v.35, i.2, pp. 114-131, 2003.

[12] Telelogic's TAU Generation2: http://www.taug2.com/

[13] Castro, M., Druschel, P., Kermarrec, A.-M., Rowstron, A. "Scribe: A large-scale and decentralized application-level multicast infrastructure. *IEEE Journal on Selected Areas in Communications* (JSAC), 20(8), 2002.

[14] Hoyos-Rivera, G.J., Gomes, R.L., Courtiat, J.P. "CoLab: a flexible collaborative web browsing tool". *19th IEEE International Conference on Advanced Information Networking and Applications*, Taiwan, March 2005.

[15] Babylon: http://visopsys.org/andy/babylon/index.html