Computing and Informatics, Vol. 30, 2011, 137–164

COMBINING WEB 2.0 AND WEB SERVICES IN COLLABORATIVE WORKING ENVIRONMENTS

M. Antonia Martínez-Carreras, Antonio F. Gómez-Skarmeta

Department of Information and Communication Engineering
Faculty of Computer Science
30071 Campus de Espinardo
Murcia, Spain
e-mail: {amart, skarmeta}@um.es

Manuscript received 25 September 2008; revised 18 May 2009
Communicated by Michal Laclavík

Abstract. Collaborative applications offers significant benefits in business sector. Usually, team members need to use several systems to carry out their tasks. What these users need is an environment which permits them to carry out these tasks automatically, considering the flow of information between the different systems and offering interoperability and composition features. Nowadays, Web Services have gained their prominence in providing these both features. On the other hand, the use of Web 2.0 allows to create web applications in which the user constitutes a key element. What we propose in this paper is the combination of both approaches for creating a Collaborative Working Environment (CWE).

Keywords: Web services, web 2.0, CSCW, interoperability, distributed systems and information systems

1 INTRODUCTION

The web has influenced our social patterns of communication deeply. Moreover, it has facilitated new ways of communication among people geographically dispersed, providing virtual places where people can share opinions, information or ideas. Regarding the communication and cooperation between people, CSCW (Computer Supported Cooperative Work) field emerged with the objective of studying the creation and the social impact of collaborative applications, also well-known as group-

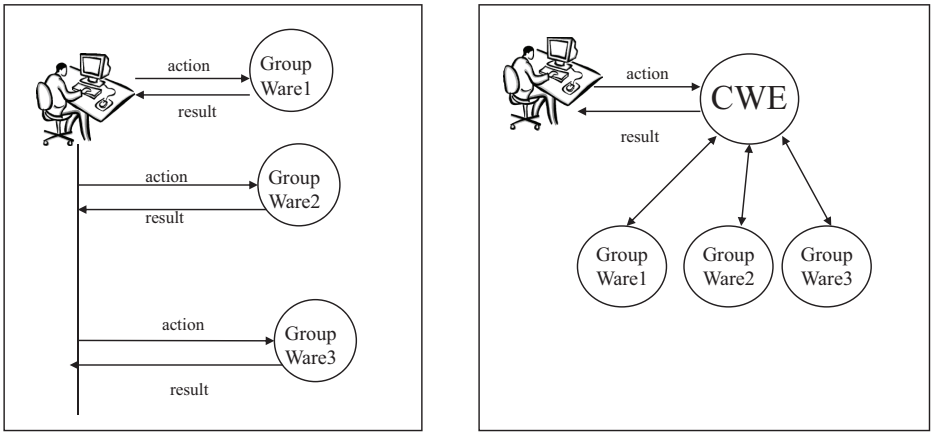


Fig. 1. The left image depicts the interaction between the user and the groupware in which the user is responsible for exchanging information between different systems. In the right part, the ideal solution is shown in which the user only interacts with only one application which is responsible for transferring information between different systems.

ware, which favor the coordination and the collaborative work of teams through the use of computers. Namely, Wilson [1] defined CSCW as a "generic term which combines the understanding of how people work in groups with the enabling technologies of computer networking, and associated hardware, software, services and techniques".

The importance of CSCW results in affecting several areas such as education, health and business. In the latter, several collaborative systems have been developed, such as the following a) Shared Work Spaces, which allow to share objects, such as documents, between the collaborative workers (co-workers), b) Forums, which allow a group to discuss a certain topic and c) E-mails, which allow general communication between co-workers. Namely, the set of collaborative applications that a company needs for maintaining its collaborative work constitute its Collaborative Working Environment (CWE) [2, 3].

With the aim of fulfilling the collaborative tasks inside an organization, co-workers may need to use different groupware. Currently, in order to complete a collaborative task using several of these systems, the co-workers have to login in different systems and pass the information from one system to another, due to the differences in implementation of them.

Moreover, they are responsible for following all the required steps for completing the work. Regarding this problem, Henri ter Hofte [4] stated that this imposes on users the full burden of switching between groupware applications, including:

- logging on to a variety of groupware applications to initiate a particular collaborative task;

- copying data between applications when users shift from asynchronous to synchronous collaboration, from distributed to non-distributed collaboration, or from a single-user program to a collaborative program;
- moving the result from a groupware application used for one collaborative activity into the groupware application that is used for another collaborative activity.

The left part of Figure 1 illustrates the above-mentioned problems, in which the coordinator between the different systems is the user. Apart from the above problems, this kind of work implies also human errors such as forgetting one of the steps or delaying between tasks.

The ideal solution for solving these problems is to offer a CWE that performs these tasks automatically, thus reducing human errors. For doing that, the CWE has to be responsible for the flow of information between the different systems, following the required structure for the completion of the collaborative task (see right part of Figure 1).

With the aim of obtaining CWEs of this kind, the European project EcoSpace "e-Professional Collaboration Space" [5] has emerged. In this project, several companies provide their systems, mostly groupware, with the aim of integrating them in a CWE, allowing the creation of more enriched services that facilitate the work to co-workers. The desirable features in the creation of a CWE are the following [3, 6]:

- Easy use of applications. The CWE should offer user-friendly and intuitive applications.
- Interoperability. Services offered by different collaborative applications should interoperate in order to facilitate aspects such as composition, flexibility and reusability.
- Low cost of entry. The boundaries between different CWEs should be reduced in order to improve interoperability, fostering the development of new CWEs based on existing services and applications.
- Goal-oriented. The services offered in a CWE should be oriented to solve different problems based on decomposing activities in goals.

As to create such architecture it is interesting to follow a Service Oriented Architecture (SOA), where different services can be defined with the aim of being reusable in other tasks or processes. Bearing in mind the communication between different systems, it is needed to deal with interoperability. In this sense, aspects such as the use of a common communication protocol and a common data format have to be solved. In this line, Web Services specifications provide key elements for carrying out the previous aspects [7]. In addition, these specifications give support for the composition of several services, obtaining enriched services which may allow the automation of tasks.

The use of web applications offers several advantages to co-workers such as

1. working always with the same version of the application,
2. providing the same graphical interface and
3. favoring the independence of the operating system.

Nowadays Web 2.0 has gained its prominence for the building of web applications, offering friendly tools for the user participation and collaboration. In this sense, Web 2.0 joins with the purposes of a CWE offering the ease of use feature. From our point of view the combination of these trends, Web 2.0 and Web Services, may offer valuable features and advantages in the creation of a new CWE. Therefore, in this paper we offer the design of a CWE based on the use of these trends. On one side, this design allows the automation of tasks that have to be carried out using different systems and on the other side it provides Web 2.0 applications favoring the creation of user-friendly tools. As a proof of concepts we describe the creation of two applications based on this architecture.

This paper is structured as follows. In Section 2 we deal with the main questions regarding interoperability and we revise some previous standards showing the benefits of using Web Services as well. In Section 3, we analyze the main concepts of the Web 2.0, offering more detailed information about Ajax and the use of Widgets. In Section 4, we propose a layered architecture for the creation of any CWE, based on Web Services and Web 2.0. As a proof of concepts, we show the development of two applications following this layered architecture. After that, Section 5 describes the necessary effort for building some applications of this CWE. In Section 6, we deal with the usability issues, measuring the efficiency comparing the functionalities of our CWE with other environments and the satisfaction and effectiveness with some tests with six users. Finally, we offer the conclusions and future work in this research.

2 INTEROPERABILITY AND WEB SERVICES

Interoperability is one of the hot topics in the business and research area. In order to provide interoperability between different systems it is necessary to deal with the following questions:

- What technology will be used to support the communication between different systems. This aspect is really important because it determines how the different systems pass information between them.
- What kind of data format will be used to understand the shared information between different systems. Due to the fact we try to communicate different systems, we have to indicate how each system represents the information, in order to provide the understanding of the data.

With the aim of solving the questions mentioned above, a Service Oriented Architecture (SOA) seems to be the most appropriate paradigm for designing an in-

teroperable CWE. SOA is not a new paradigm, although nowadays it plays a vital role in the Web Services research area. A service-oriented architecture is essentially a collection of services. These services communicate with each other, and the communication can involve either simple data passing or it could involve two or more services coordinating some activities. One of the important aspects is that "SOA is not tied to any specific technology, indeed it can be implemented using a wide range of protocols", that is, SOA is a technology-independent paradigm [7].

For many people in the past, the first service-oriented architecture was performed with the use DCOM or Object Request Brokers (ORBs) based on the CORBA specification. Although this first approach helped in the creation of reusable code, it had the problem of being technology-dependent. More concretely, in this first approach the implemented operations in each service are quite engaged to the transport protocol. Thus, whenever the transport protocol changes, it is required to re-code part of the components again.

Another problem is the data format of shared objects. The use of DCOM or ORB requires an agreement with the objects to share. Hence, new integrations imply changes in the existing code. After them, XML-RPC appeared as the first solution based on XML to exchange data and to carry out the communication based on HTTP.

Nowadays, the design of interoperable systems is driven to the use of Web Services specifications, providing a common way for developing services and communicating them [7, 8]. One of the most attractive features of Web Services is the use of XML, which provides protocol and language independence. Using them, both the data exchange and the transport protocol is based on XML, solving the two main questions about interoperability. In this way, changes in the transport mechanism do not imply recoding.

Among the most important specifications recommended by the WS-I (Web Service Interoperability Organization) [9], we can find the following:

SOAP. The XML-based transport protocol for exchanging information among computers. It works over several transport mechanisms such as HTTP, SMTP, JMS, etc.

WSDL (Web Services Description Language). The specification defines how to describe Web Services based on a specific grammar. Based on this specification developers can define the location, operations and parameters of each service.

UDDI (Universal Discovery Directory Interface). The specification of how to publish and find services.

Regarding the specifications recommended by WS-I, SOAP and WSDL are widely accepted and used in the Web community. With the aim of enabling the service composition, the Business Process Execution Language (BPEL) [7] emerged. The use of BPEL allows the design and definition of business processes, indicating how the information flows from one service to another. Using this specification, we

can compose new services by means of others. Moreover, a BPEL process is a service and can be used for the creation of other BPEL processes.

Another important language to allow services composition is the Web Service Choreography Description Language (WS-CDL). More concretely, "WSCDL attempts to organize information exchange between multiple organizations" [7].

According to Motahari et al. [9] "BPEL presents protocols from a service point of view, whereas WS-CDL describes the entire choreography of message among multiple partners". More precisely, BPEL standardizes an orchestration, in which a central element "controls almost every facet of a complex activity" [7].

On the other hand, WS-CDL reflects a choreography in which the execution of tasks is carried out by collaboration between different participants. In this approach, there is not a process that orchestrates the information between different services. From our point of view, BPEL allows the execution of several services following a specific structure. In addition, BPEL is a mature industry standard for orchestration. As a consequence, with the use of BPEL processes we can create services that automate the work of the team, avoiding users changing between applications; that is to say, we can create composite applications. In the same line, Debevoise [10] indicates that "composite applications are orchestration of different programs or applications that act as one application. You use BPEL to create the composite application".

Some business processes need to follow an orchestration. For instance, imagine the case of a co-worker who needs to perform some fixed tasks after uploading a document, such as creating a forum for discussing some topics and notifying the rest of co-workers about these changes. The execution of these services should be performed following a specific flow of information, a specific order in the execution. Thus, the use of BPEL is most appropriate for carrying out processes of this kind.

Concerning the definition of business processes, the Business Process Management Initiative (BPMI) [11] defines the necessary elements for designing business process at a high level, obtaining as a result the Business Process Management Notation (BPMN). In this sense, there are some modeling tools which offer this notation for the graphical design of business processes and their translation to executable BPEL, such as Intalio [12].

3 WEB 2.0: MAKING THE USE OF THE WEB EASY

A new trend that has brought a significant shift in the development of web applications is represented by Web 2.0. Tim O'Reilly is the first person coining the Web 2.0 concept [13]. According to him, this new trend in creating web applications considers the users, their opinions and the collaboration between them. For that reason, this kind of web is also well-known as "social web" and offers important concepts for creating collaborative systems.

In more details, Web 2.0 is an attitude or philosophy for creating web tools based on open standards, offering user friendly tools. According to Kwei-Jay [14],

Web 2.0 is presented as a new way of conceiving the web "representing a paradigm shift in how people use the Web". Therefore, in Web 2.0 the key element is the user, and its main focus is on providing simpler User Interface (UI) allowing users to customize their applications. Hence, technologies and standards under the umbrella of Web 2.0 are intended to make the Web a more intuitive and smoother place [14]. Among these technologies and standards we underscore the following: Ajax, Ruby on Rails, JavaScript, RSS, REST [15] or XML-RPC.

From our point of view, several of these technologies and standards, such as Ajax, Ruby on Rails, RSS and JavaScript are more related to improve the UI. Following a Web 2.0 attitude, these technologies should be used in such a way that they provide users with friendly and intuitive tools for performing their work. On the other hand, the use of REST and XML-RPC is devoted to establish the communication between different services. These protocols are related to interoperability issues and may constitute the transport protocol for the communication between services.

More concretely, defenders of Web 2.0 see REST as a simpler and easier communication protocol than SOAP. The relevance of REST is such that the last specification of WSDL, version 2.0, includes it as a transport protocol. Therefore, developer can choose between REST or SOAP.

However, SOAP provides the support for several other specifications such as Web Services Security, providing security in the access to the services as well as in the information sent and received. Furthermore, the lack of protocol for composing services in Web 2.0 implies the use of BPEL, whose last version works with WSDL 1.1; moreover, it does not include the use of REST. Considering these aspects, from our point of view SOAP is the best choice for supporting the communication between services.

Regarding the use of Web 2.0 in the UI, Ajax (Asynchronous JavaScript and XML) has become an important element for building web applications. More concretely, Tim OReilly defined it as a set of technologies which allows the creation of enriched web pages. Namely, he stated that "Ajax incorporates: standards-based presentation using XHTML and CSS; dynamic display and interaction using the Document Object Model; data interchange and manipulation using XML and XSLT; asynchronous data retrieval using XMLHttpRequest; and JavaScript binding everything together" [13].

Smith defines Ajax as "a standards-based programming technique designed to make Web-based applications more responsive, interactive, and customizable" [16]. Indeed, this author indicates that the major benefit of using Ajax for building web applications is that it provides quick responses as well as it offers users more control in the applications.

More precisely, Ajax allows local processing of the graphical information without requesting for each step to the server. In this sense, the entire UI can be loaded locally and thus each change in the UI is managed locally by the Ajax motor. Once the needed information is filled in, the application can invoke the required service for processing it. Following this style, the user benefits from quicker responses to

his/her interactions in the UI. For that reason, Ajax constitutes a good underpinning of Web 2.0.

With the aim of illustrating the benefits of using Ajax, we are going to consider our previous example of uploading documents, creating forum and notifying co-workers about these events. Basing the web application on Ajax style, we can offer an application in which all the forms are managed locally by the Ajax motor. In this way the user may benefit of a quick transition from one form to another, avoiding multiple requests to the server for the visualization of each form. Once the information is provided, the required service can be executed.

Other important fact of the use of Ajax is that it facilitates the creation of widgets. Lawton [17] stated that "widgets are portable, lightweight Web applications than can be embedded in HTML-based pages". Thus, users can configure a personal web space where they can place those widgets they usually need in their daily tasks.

4 COMBINING WEB 2.0 AND WEB SERVICES IN COLLABORATIVE WORKING ENVIRONMENTS

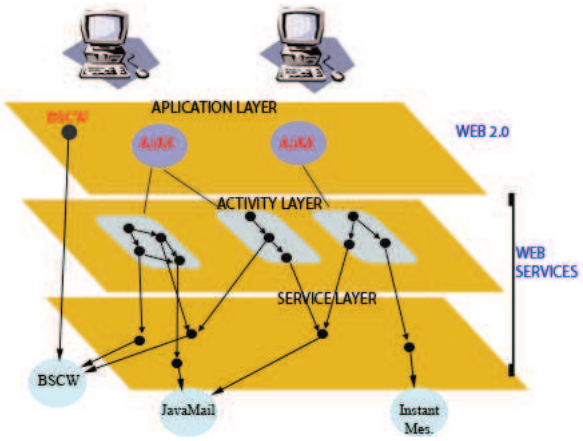


Fig. 2. Layered design of a Collaborative Working Environment using Web 2.0 and Web Services

Some authors see Web 2.0 as substitute of Web Services [18]. From our point of view and considering the comments of Schroth [19] and Cetin et al. [8], both trends may be combined for obtaining the best performance in the creation of web applications. According to previous sections, we may benefit of the use of Web Services specifications and Web 2.0 technologies for creating enriched web applications, which need interoperability issues. In fact, the use of Web Services specifications allows the creation of enriched composite services which enable the interoperation

between different systems. Indeed, these composite services are intended for automating necessary tasks. In this sense, we fulfill interoperability and goal oriented objectives of a CWE.

On the other hand, by following Web 2.0 recommendations we can design simpler and more efficient web applications which communicate with the aforementioned enriched services, obtaining the ease of use desirable feature in a CWE. Additionally, the use of open standards such as Web 2.0 and Web Services provides the low cost of entry feature in a CWE.

With the objective of creating a CWE which offers all the previous aspects we have combined Web 2.0 and Web Services as shown in Figure 2. This CWE does not replace the use of existing legacy applications, indeed it provides new applications that requires automation by using Web Services and Web 2.0 which are based on the existing legacy applications. This CWE architecture is based on the following three layers:

- The Application Layer. It represents the applications and we have based the creation of new ones on the use of Web 2.0. More concretely, the implementation of the new applications can be done using Ajax and can be integrated as widgets for allowing workers to customize their webs with their preferred applications. In fact, the applications in this layer are the vehicles for allowing the users interactions in the system as well as collecting the necessary information which is passed to the corresponding process in the following layer. Thus, this layer constitutes the link between the Web 2.0 and Web Services.
- The Activity or Composite Services Layer. It is based on the use of business processes or composite services, such as BPEL. The main purpose of this layer is to collect those new services, or activities, which require the combination of several other services, groupware in our case, which are offered in the lower layer or included in this layer.
- The Services Layer. It offers several services regarding groupware and core services for the management of any business, such as directories or database access. This layer is based on the use of WSDL for allowing the interoperation with any kind of service provided in the CWE.

The design of this architecture provides reusability. In fact, the services and the composite services can be reused for creating new applications or for being integrated in existing applications offering advanced features. Moreover, these services can be integrated in other companies business processes providing interoperability regarding intra and inter-organizational boundaries.

Following this structure in the creation of new applications, the user indicates the actions to perform by means of web applications offered in the upper layer. The execution of the required actions is carried out by the right composite service in the Activity Layer. Therefore, the web application serves as the link between the Web 2.0 and Web Services.

In subsequent subsections we are going to describe how we have designed and implemented two applications following our CWE: one application is for uploading document, creating forum and notifying by e-mail about this event and the other is for uploading a document and notifying by Instant Messaging or E-mail. These applications deal with all the layers involved in the CWE architecture depicted in Figure 2.

4.1 Building the Service Layer

Previous to the introduction of services in the CWE, we have to analyze the necessary services for each activity or composite service that has to be integrated for automating the work. Moreover, these services have to be defined in a generic way with the aim of being reusable in other developments, following in this sense the SOA principles.

Considering our previous example about uploading a document, commenting ideas about it in a forum and sending e-mail notification to co-workers, we are going to use the following services: shared work spaces, forum and e-mail services. Additionally, information about the organization should be integrated in this CWE, such as the role of each person, his/her personal data and e-mail. Thus, a directory service is needed as well.

In the development of the aforementioned application, we have used the following existing collaborative applications:

- BSCW (Basic Support for Cooperative Work) [21] is a shared workspace system which supports document uploading, group management, forums, polls, and much more. More concretely, the collaborative work in this system is based on the use of folders in which team members can collaborate. It constitutes a good basis for exchanging information among co-workers.
- JavaMail is an API which provides a platform and protocol independent framework to build mail and messaging applications [22].
- OpenLDAP [23] is an open source implementation of Lightweight Directory Access Protocol (LDAP), which is a protocol for querying and modifying directory services running over TCP/IP. This implementation helps us store personal information about users or workers inside an organization.

In order to integrate some of these systems in the architecture presented in Figure 2 we have created wrappers or adapters [20]. The purpose of these adapters is to translate the information from our CWE platform, based on Web Services specifications, to the systems they model, based on the API of this system, and vice-versa. More concretely, these adapters are integrated in the Services Layer of our CWE and thus they provide WSDL interfaces and can be accessed using SOAP as a transport protocol. The use of a common data format is ensured due to the fact that the services follow WSDL language.

Considering the software mentioned previously, we have defined the following services: Shared Work Spaces, Forum, E-mail and Directory Services. The two first ones are built on top of BSCW, E-mail is built on top of JavaMail, and the Directory Service is built on top of OpenLDAP. All of them have been included by using wrappers. More concretely, in the case of BSCW the wrappers of Shared Work Spaces and Forum communicate with this system by using its XML-RPC API [24].

Table 1 describes the operations of each one of these services. These operations constitute the underpinning for building the composition of services, as we describe in the following section.

As a proof of concept, we have also created other application following similar structure. This application is intended for uploading a document and notifying by using instant messaging depending upon whether the user is connected in the Instant Messaging system, otherwise the notification is done by e-mail. This new application is built re-using some of the previous applications and services. Particularly, for the development of this application we use the Presence and Instant Messaging infrastructure which has been developed in ECOSPACE project [25]. For using this application we have integrated the following two services in the system:

Presence and Availability Service. This service provides the information about the online users and their status in the Instant Messaging tool. As commented previously, this functionality is a wrapper for the Presence and Instant Messaging Infrastructure of ECOSPACE project.

Instant Messaging Service. This service makes available the functionalities for sending messages through the synchronous infrastructure developed in the project. This is a new service which sends messages to the Instant Messaging Infrastructure.

The available operations in each of these last services are listed in Table 2.

4.2 Building the Activity Layer

The Activity Layer represents processes or activities which are built as the composition of several of the services provided in the lower layer or even using processes of this layer. The aim of this composition is to create enriched services which automate some tasks, permitting workers to perform theirs tasks without changing between systems. In this sense, the worker perceives the activity as a whole, and he/she only includes the information required for the whole activity using only one application. In fact, an activity can be seen as a service which requires some parameters as entries and returns a result.

As commented before, in the current implementation of our architecture we rely on the use of BPEL for creating these activities or composite services. Considering the goal of our applications, we have created some BPEL processes which reflect how the information flows between the services described in Table 1 and Table 2. Figure 3

Shared Work Spaces	Forum	Directory	E-mail
addDocument	createForum	getUserEmail	sendEmail
addFolder	deleteForum	getUserRole	
addNote	editForum	login	
copyObject	replyToForumEntry		
deleteObject	createForumEntry		
deleteNote	deleteForumEntry		
editNote			
getDocument			
getObjectEvents			
getObjectNotes			
getUserHome			
getMembersObject			
renameDocument			
renameObject			
replaceObject			

Table 1. Functionalities for SharedWorkSpaces, Forum, E-Mail and Directory services

Presence and Availavility	Instant Messaging
getUserStatus	sendMessage
getOnlineUsers	

Table 2. Functionalities for Presence and Availability and Instant Messaging services

offers the design of the Upload document, creates a forum and sends a notification about these events by e-mail; Figure 4 depicts the structure of the Upload document and notification of users by using e-mail or instant messaging depending on their status in the Instant Messaging tool. More concretely, this process is composed of the Notify process which is responsible for doing the notifications. Thus, other processes can reuse this way of notification.

More precisely, several of the steps in the composite service in Figures 3 and 4 imply the invocation of operations in the related services, except the Notification

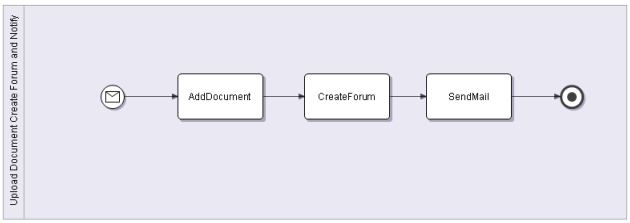


Fig. 3. Business Process Model for the activity of uploading a document, creating Forum and notifying to co-workers

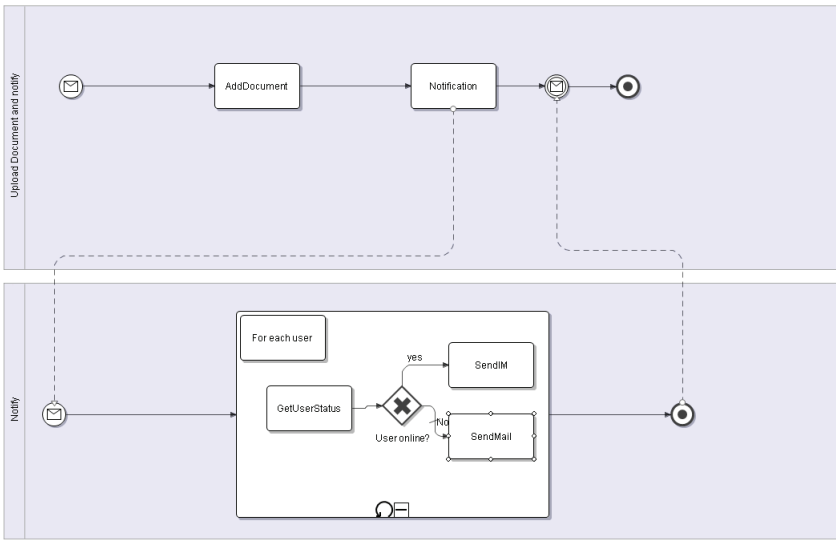


Fig. 4. Business Process Model for the activity of uploading a document and notifying by using Instant Messaging or E-mail

task in Figure 4 that invokes another process. For example, as we can appreciate in the example, the AddDocument task implies the invocation of the addDocument operation of the SharedWorkSpaces service and the sendMail task implies the invocation of sendEmail from the Email service (see Table 1). In order to simplify the diagrams depicted in Figures 3 and 4, we have not included the management of errors that can occur in each step.

As can be seen in Figures 3 and 4, the fact that our CWE is based on Web Services specifications helps reuse services for a different process. Actually, from Figure 4 we can appreciate the reuse of composite services in the creation of new ones, due to the use of BPEL specification.

4.3 Building the Application Layer

The application layer is one of the most important elements in our architecture due to the fact it constitutes the link for combining both approaches, the Web 2.0 and the Web Services, and additionally, it constitutes the vehicle for the communication between co-workers and the CWE.

As we have commented previously, we are going to base this layer on the use of Web 2.0, with the purpose of obtaining a simpler and user-centered interface. Thus, we have designed a web application based on Ajax; we have used ZK Ajax [26]. Namely, we have followed Ajax style in our developments, in which several forms are offered in only a web page, avoiding multiple requests to the server (see Figure 5). In this way, the Ajax motor performs locally the transition from one form to another.

Fig. 5. AJAX application for uploading a document in the shared workspace, creating a forum and notifying users about this event

Fig. 6. Widget of the uploading document, creating forum and notifying application integrated in iGoogle

Therefore, the information is not send to the composite service or activity until all the forms are filled in and checked out.

Once the necessary information is provided, the Ajax application invokes the required composite service in the Activity Layer passing the information provided by the user.

In addition, the use of Ajax facilitates the creation of widgets as well. Thus, the applications can be integrated as widgets in platforms such as iGoogle, FreeWebs or Netvibes. In this way, users can design their own web pages, obtaining more

personalized web platforms which adapt to their needs and preferences. As a proof of this integration, Figure 6 depicts the widget of our previously commented Ajax application integrated in the iGoogle platform.

5 EFFORT IN THE DEVELOPMENT OF CWE APPLICATIONS

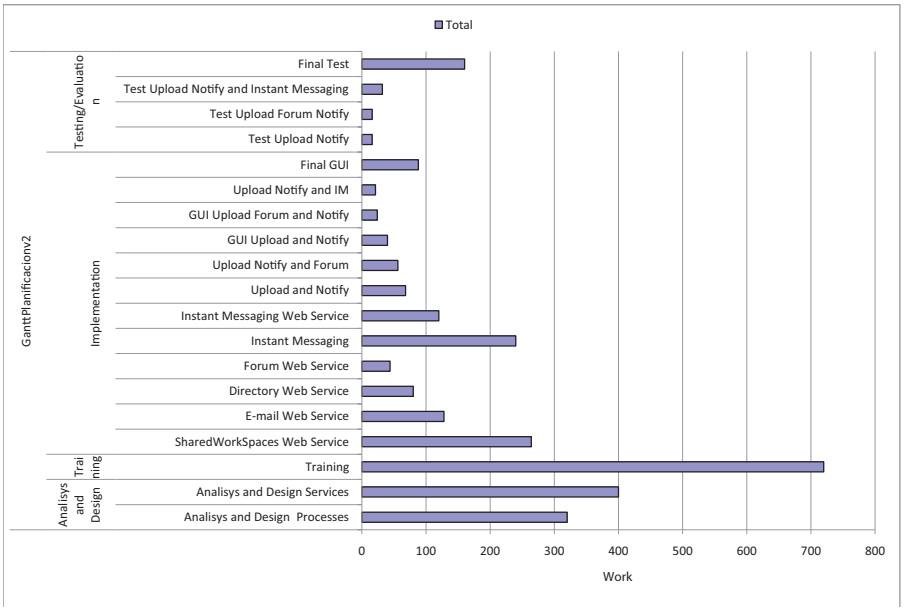


Fig. 7. Efforts building the CWE

Basically, the major part of the CWE architecture presented in this paper is based on the use of Web Services. As Hutchison et al. [20] indicate, the creation of architectures of this kind implies a significant shift in how organizations implement or deliver new business functionalities. From our point of view and following the same line that these authors propose, we believe that a progressive-evolution strategy is the best way for creating a CWE of this kind. For that reason, and in order to mitigate the effort we have based our developments in the use of wrappers [20].

Previous to the creation or adaption of applications in a CWE, it should be analyzed and designed which processes are going to be implemented and what elements are needed for them. The choice of these processes and their structure should be guided by the business needs. In this phase paradigms and designs are indicated. This part is quite important and it requires a considerable period of time as it is depicted in Figure 7 as "Analysis and Design Processes". Namely, three analysts of different partners were involved in this task. At the end of this phase the technologies for building these processes were proposed as well.

In order to convert the functionalities of a system in services, the system should be rigorously analyzed for offering the generic functionalities. In this phase analysts should consider all the possible applications that may need the service, and what are the functionalities they need. For those systems which offer an API the conversion to web services is easier, because the methods in the API are converted to services in the Wrapper WSDL file. More concretely, in the Ecospace project 12 systems were analyzed (Blog, E-Mail, Forum, Shared Work Space, Whiteboard, Instant Messaging, Video Conference, Wiki, Voting, Polls, Calendar, Directory) and the description of 22 services was obtained. The effort for performing this task is depicted as "Analysis and Design Services" in Figure 7.

Let us explain the tasks illustrated in Figure 7 which are grouped in four major tasks: "Analysis and Design" gathers the two aforementioned tasks, "Training" represents the learning period of three developers which were in charge of building the applications indicated in this schema, "Implementation" means the tasks for performing the implementation of the services, the business processes and the GUI for accessing to them, finally, the task "Test Upload and Notify" refers to testing and evaluation of the system from the developer side. It should be noticed that in the Implementation task most of services built are wrappers for existing systems.

As with other emerging technologies, the learning period of web services is quite long, however once the developers are in touch with these technologies, the development of subsequents services or composite services (BPEL) requires less time.

It can be seen in Figure 7 that the creation of the first web services consumes more time; however, as soon as developers get used to the development of web services, the implementation is carried out faster (depending as well on the number of operations involved in the system). For example, the developer in charge of the creation of E-Mail Web service is also responsible of creating the Directory Web Service. Although the latter contains more functionalities, it needs less effort. Furthermore, building of the BPEL process presents analogous issues. While the process of "Upload Notify and IM" is more complex, the fact that part of this process has been previously built in "Upload and Notification" reduces the effort for building it.

The implementation of web services may be carried out by means of different IDE such as Eclipse, NetBeans or Intalio. More concretely, in our development we have made use of NetBeans and the Sun Application Server (in last versions changed to GlassFish). The use of the IDE makes easy the design and the deployment of the web services and BPEL procesess by means of graphical elements.

6 USABILITY EVALUATION

Previously to the creation of any process or application on this architecture, some issues should be considered. First of all, the new process has to facilitate some business goals, and what is more the application should be structured efficiently

eliminating the user-perceived latency. For the former question, this information is gathered in the "Analysis and Design Processes" offered in Figure 7. For the latter, the use of AJAX technology by means of asynchronous interactions reduces this factor [29]. In this sense, the user interface should be intuitive, mitigating the learning of the application interface.

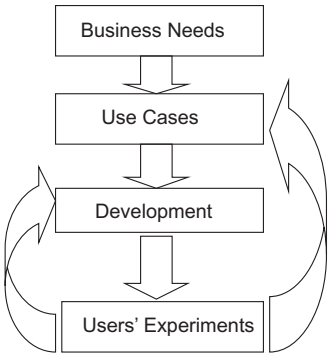


Fig. 8. Steps followed in the experimentation

As commented in [27, 32], users should be introduced during the development of systems so as to improve the usability. Applications are generated by real business needs, and while producing these applications several tests have to be performed. More precisely, the sequence followed is depicted in Figure 8.

Concerning the usability of any product, it allows to achieve specified objectives with efficiency, effectiveness and satisfaction. Namely, efficiency is related to the accuracy for achieving the goals and the resources expended for achieving them. In this sense the performance of the application is quite significant [27, 28]. Effectiveness is related to the fact that the application permits users to achieve certain goals; and finally satisfaction defines the user comfort and positive attitude to use the application.

In the following subsections we are going to deal with efficiency of the new application as well as with the users' experiences, which reveal information regarding the effectiveness and satisfactions parameters.

6.1 Efficiency of CoCos Applications in the CWE

Until now, there exist some integrated environments which incorporate several collaborative features and systems which allow the collaboration between team members. However, these legacy systems are not enough for the different activities each organization should include, and the user is responsible for performing these tasks. Regarding these integrated environments there exists several such as BSCW, Alfresco [37] and EMC Documentum [38]. Despite the inclusion of several collaborative

systems these environments lack interoperability and reusability features. Moreover, excepting EMC Documentum, BSCW and Alfresco do not allow the integration of business processes for automating processes. As a consequence, the comparison we are going to establish respecting our CWE and these integrated systems is how a certain composite task is performed in each environment.

In the followings paragraphs we are going to explore each of these environments by illustrating how the process of uploading and notification is performed. Regarding the notification via instant messaging, none of these systems integrates instant messaging, thus we are going to compare only with the notification via e-mail service.

In previous sections we have introduced BSCW indicating some of the tasks that can be carried out in this system. In BSCW the collaborative tasks are carried out around the concept of "folder", and document, forums, calendars, and other artifacts can be managed inside a folder. Considering the task we are analyzing, BSCW allows the uploading of a document, and once the user has uploaded it, he/she can send a link of this document to others by clicking an action button. The composition of tasks from one task to another is a user's responsibility.

Alfresco is Shared Work Space which provides a commercial and a lab (free) versions. Furthermore, it incorporates functionalities of a Document Management System (DMS) and a Content Management Systems (CMS). Similarly to BSCW, this system allows creation of folders, uploading documents, versioning control, creation of discussion and calendar. However, the structure and the way this environment manages the information is slightly different, and the shared information is managed by "spaces" concept. Once the user is in a "space" he/she can manage the uploading of documents, forums, or other content. Concerning our analyzed activity, once the user has uploaded the document there is no action available in the system for notifying this fact. Therefore, the user is forced to use an e-mail system for communicating this event to some of their collaborators.

EMC Documentum is a commercial Shared Work Space which incorporates functionalities of CMS and DMS, and, moreover, it allows the creation of workflow and business processes. Unlike the rest of systems, the shared data is managed in the concept of "container". Concerning the activity of uploading and notification, similar steps as in BSCW have to be performed. In contrast with the other environments, EMC Documentum allows the creation of business process. Nevertheless, this process is not based on open standard which makes the interoperability with other processes difficult.

Additionally, the CWE we present in this paper can be built upon any of these integrated environments, allowing the creation of interoperable activities, giving at the same time added-value operations in the CWE.

In order to measure the efficiency of the Upload and Notify applications of our CWE, we are going to compare it with the tasks users should do for uploading documents and sending notifications by e-mail in BSCW, which is similar to EMC Documentum. More concretely, we are going to consider the user response time of each application. Note that the tool upon which we have based the performance

is the one that users have tested during the experimentation which is described in the next subsection, except sending messages that can also be done through instant messaging.

As for measuring the response time to users for achieving this activity we have to consider each step user and the applications should perform in both environments. Regarding the performance of the applications of our CWE, it should be noticed that the use of web services specifications is going to reduce it. Particularly, it is widely accepted that the use of web services specifications implies performance penalties [36] which result from additional processing time of XML structures. Moreover, the performance of their execution depends on the kind of servers they are deploying. However, the major advantage of these technologies is the reduced cost of integration with other systems and the enhancement of connectivity between systems across logical and physical boundaries. Furthermore, the automation of tasks avoids human errors such as forgetting doing some tasks.

Let T_{CoCos} be the user response time for executing the upload and notifying in our CWE and T_{BSCW} the user response time for executing the same task in BSCW. Bearing in mind the above mentioned considerations we obtain the subsequent equations:

$$\begin{aligned} T_{\text{CoCos}} &= T_{\text{logCocos}} + T_{\text{showFolders\_CoCos}} + T_{\text{uploadDocuments\_CoCos}} \\ &\quad + T_{\text{notify\_CoCos}} + T_{\text{click}} + T_{\text{BPEL}} \end{aligned} \tag{1}$$

$$T_{\text{BSCW}} = T_{\text{logBscw}} + T_{\text{showFolders\_Bscw}} + T_{\text{uploadDocuments\_Bscw}} + T_{\text{notify\_Bscw}} \tag{2}$$

where T_{logBscw} and T_{logCoCos} represent the time needed for logging in BSCW and the CoCos, respectively, $T_{\text{showFolders\_CoCos}}$ and $T_{\text{showFolders\_BSCW}}$ indicate the time needed for representing the user's folders in each application, $T_{\text{uploadDocuments\_Bscw}}$ and $T_{\text{uploadDocuments\_CoCos}}$ measure the time for uploading documents in BSCW and CoCos, respectively, $T_{\text{notify\_CoCos}}$ and $T_{\text{notify\_Bscw}}$ represent the time for notifying in JavaMail and BSCW, T_{BPEL} indicates the time needed for processing the XML of the BPEL schema and finally T_{click} measures the time needed for making a mouse click.

The use of a BPEL process forces that the necessary information for its execution is collected in the GUI and once the information is gathered, the execution of the process is performed. For that reason Equation (1) needs a T_{click} for starting the invocation of the BPEL process. In order to compare both systems, we are going to include the invocations of web services in each of the steps in which we have decomposed Equation (1). It should be noticed that each of these parts which includes a web service invocation adds $2T_{\text{webservice}}$ for processing the XML involved in the request and the response of a web service. Although this time will depend on the kind of parameters involved in the request or the response, for simplicity we consider the request and the response time as equivalent.

Now, we are going to decompose each of these equations according to the steps performed by users and the resources needed by the execution of the activity of

uploading documents and notifying a group of users. First, the user should login in the application. Consider the following parameters:

- T_{fillform} indicating the time needed by the user for entering information in a form
- $T_{\text{call\_loginBSCW}}$ representing the time needed in the invocation of BSCW for logging the user
- $T_{\text{call\_loginLDAP}}$ measuring the time needed in the invocation of LDAP for logging the user.

According to the steps and the invocations involved in the login of the user, we can model T_{logBscw} and T_{logCoCos} as follows.

$$T_{\text{logBscw}} = 2T_{\text{fillform}} + 3T_{\text{click}} + T_{\text{call\_loginBSCW}} \quad (3)$$

$$T_{\text{logCoCos}} = 2T_{\text{fillform}} + 3T_{\text{click}} + T_{\text{login\_userLDAP}} + 2T_{\text{webservice}}. \quad (4)$$

Notice that both equations are quite similar, offering differences in the invocations to the services. Namely, in Equation (3) it is performed in BSCW and in Equation (4) it is performed in LDAP. Furthermore, the CoCos consumes additional time $2T_{\text{webservice}}$ due to the processing of the XML files of the request and response for invoking the Directory Web Service.

Once the user is logged into the system, both systems automatically show the information on the user's home. Let $T_{\text{get\_folders\_Bscw}}$ be the time needed for obtaining the information on a user's home in BSCW system and $T_{\text{repres\_folders\_Ajax}}$ and $T_{\text{repres\_folders\_Bscw}}$ be the time needed for representing the folder in Ajax and in BSCW, respectively. Then:

$$T_{\text{showFolders\_BSCW}} = T_{\text{get\_folders\_Bscw}} + T_{\text{repres\_folders\_Bscw}} \quad (5)$$

$$T_{\text{showFolders\_CoCos}} = T_{\text{get\_folders\_Bscw}} + 2T_{\text{webservices}} + T_{\text{repres\_folders\_Ajax}}. \quad (6)$$

Due to the fact that our CoCos application is based on the use of a service based on BSCW, we obtain Equation (6). Moreover, in the same equation $2T_{\text{webservices}}$ is added for representing the processing of the request and response of the SharedWorkSpace Web service.

After that, the user has to include the documents for uploading in the BSCW. Regarding the steps involved in the GUI of these applications, we obtain the following equations for $T_{\text{uploadDocuments\_CoCos}}$ and $T_{\text{uploadDocuments\_Bscw}}$:

$$T_{\text{uploadDocuments\_CoCos}} = 2T_{\text{click}} + numdoc \times 3T_{\text{click}} + T_{\text{call\_uploadDocuments\_Bscw}} + 2T_{\text{webservices}} \quad (7)$$

$$T_{\text{uploadDocuments\_Bscw}} = 2T_{\text{click}} + numdoc \times 3T_{\text{click}} + T_{\text{call\_uploadDocuments\_Bscw}} \quad (8)$$

Again, in Equation (7) a service in the SharedWorkSpace is invoked and thus it consumes $2T_{\text{webservices}}$.

In order to finalize the process of uploading documents and notifying, the user should notify some co-workers about the new documents. In this part, both systems present significant differences. While in the CoCos the notification is automatically done in the same process and the links of the documents are included in the e-mail, in BSCW the user is responsible for sending the notification for each document.

Let *numdoc* be the number of documents to upload in each application, *numUsers* be the number of users to notify, $T_{\text{intr\_usu}}$ be the time needed for introducing the name of the user, T_{subject} be the time needed for introducing the subject of the message, T_{body} be the time needed for introducing the body of the message, $T_{\text{sendJavaMail}}$ be the time for sending the mail in JavaMail and $T_{\text{sendMailBSCW}}$ be the time for sending the mail in BSCW. Therefore:

$$\begin{aligned} T_{\text{notify\_CoCos}} &= 2T_{\text{click}} + T_{\text{subject}} + T_{\text{body}} + numUsers \times T_{\text{click}} \\ &\quad + T_{\text{sendJavaMail}} + 2T_{\text{webservices}} \end{aligned} \tag{9}$$

$$\begin{aligned} T_{\text{notify\_Bscw}} &= numdoc \times (6T_{\text{click}} + T_{\text{subject}} + T_{\text{body}} \\ &\quad + numUsers \times T_{\text{intr\_usu}}) + T_{\text{sendMailBscw}}. \end{aligned} \tag{10}$$

Applying (3), (5), (8) and (11) to (2) we obtain the following equation:

$$\begin{aligned} T_{\text{BSCW}} &= 2T_{\text{fillform}} + 5T_{\text{click}} + T_{\text{call\_loginBSCW}} + T_{\text{get\_folders\_Bscw}} \\ &\quad + T_{\text{repres\_folders\_Bscw}} + T_{\text{multiple\_select}} + T_{\text{call\_uploadDocuments\_Bscw}} \\ &\quad + numdoc \times (9T_{\text{click}} + T_{\text{subject}} + T_{\text{body}} + numUsers \times T_{\text{intr\_usu}}) \\ &\quad + T_{\text{sendMailBscw}}. \end{aligned} \tag{11}$$

Considering Equations (4), (6), (7) and (9) to (1) we obtain the following equation:

$$\begin{aligned} T_{\text{CoCos}} &= 2T_{\text{fillform}} + 8T_{\text{click}} + T_{\text{login\_userLDAP}} + 8T_{\text{webservice}} \\ &\quad + T_{\text{get\_folders\_Bscw}} + T_{\text{repres\_folders\_Ajax}} + \\ &\quad + numdoc \times 3T_{\text{click}} + T_{\text{call\_uploadDocuments\_Bscw}} + T_{\text{subject}} + T_{\text{body}} + \\ &\quad + numUsers \times T_{\text{click}} + T_{\text{sendJavaMail}} + T_{\text{BPEL}}. \end{aligned} \tag{12}$$

Analyzing Equations (11) and (12) we can notice that several elements are similar. Simplifying the equal terms and supposing that $T_{\text{sendJavaMail}}$ and $T_{\text{sendMailBscw}}$ consume the same time as $T_{\text{repres\_folders\_Bscw}}$ and $T_{\text{repres\_folders\_Ajax}}$, we can obtain the following equations, where $T_{\text{diffCoCosBscw}}$ and $T_{\text{diffBscwCoCos}}$ are the differences in time between the CoCos and the BSCW, and vice-versa:

$$\begin{aligned} T_{\text{diffBscwCoCos}} &= numdoc \times 6T_{\text{click}} + (numdoc - 1) \times T_{\text{subject}} \\ &\quad + (numdoc - 1) \times T_{\text{body}} + numdoc \times numUsers \times T_{\text{intr\_usu}} \end{aligned} \tag{13}$$

$$T_{\text{diffCoCosBscw}} = 3T_{\text{click}} + 8T_{\text{webservice}} + numUsers \times T_{\text{click}} + T_{\text{BPEL}}. \tag{14}$$

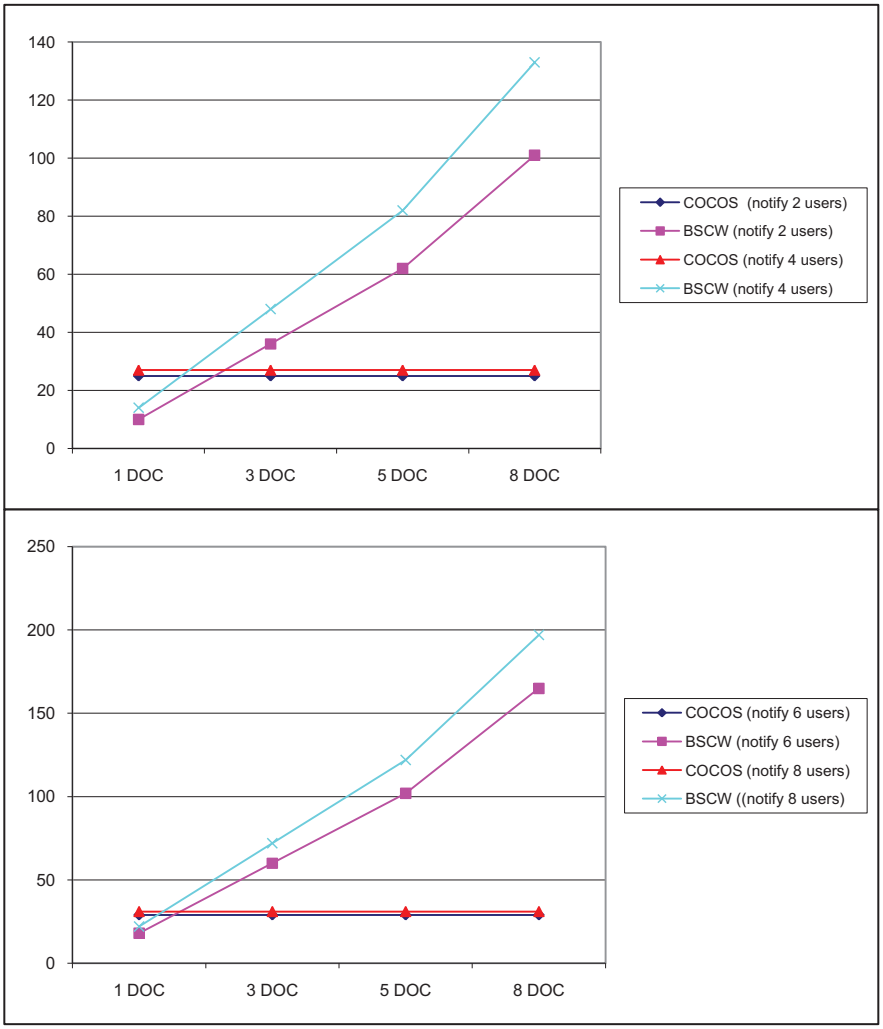


Fig. 9. Results of the comparative of BSCW and CoCos respect the number of documents to upload and the number of users to notify

As we can see, the performance of BSCW will decrease with the number of documents to upload and users to notify. On the other hand, the performance of web services is critical for the performance of the application based on the CoCos. As Woodall, Brereton and Budgen stated [30], the distributed nature of web services makes it difficult to obtain their performance. Moreover, depending on the kind of SOAP implementation [36] or the BPEL engine used for running the process [31] the time can vary. In order to measure the processing time of the XML of a web

service we are going to consider the work of Woodall, Brereton and Budgen [30]. These authors make a study on how to measure it and this study reflects that when the number of attributes is less than ten, the estimated time for the request and response of a web service is less than 4 seconds. The web services that compose the CoCos application work with a mean of 4 attributes. Therefore we can consider $T_{\text{webservice}} = 2\,\text{s}$.

T_{BPEL} depends on the structure of the defined process [31]. In the case we are comparing in this section we work with a sequence structure that does not imply nested activities, in this sense [31] the time needed for processing of the BPEL depends on the services invoked. Due to the fact we have considered them in $T_{\text{webservice}}$ in the equations, we can consider $T_{\text{BPEL}} = 4\,\text{s}$. Considering $T_{\text{click}} = 1\,\text{s}$, $T_{\text{subject}} = 1\,\text{s}$, $T_{\text{body}} = 2\,\text{s}$ and $T_{\text{intr\_usu}} = 2\,\text{s}$ we obtain the results depicted in Figure 9. While the CoCos application is slightly affected by the number of users to be notified and by the number of documents to upload, the performance of BSCW is quite affected when these two parameters are increasing. Although the use of web services imposes a burden in the performance due to the XML processing, in this case the extra actions that users have to perform in BSCW for the notifications and the fact that notifications have to be sent for each document affect the performance of this system when these two parameters increase.

Additionally, due to the automation of our application, users always have to follow the uploading and the notification for performing the task. In BSCW users can forget the notification which leads to connecting again in the system for performing the final tasks and therefore consuming more time. Furthermore, using web services technologies different functionalities and interoperability with other systems can be introduced, such as notification by instant messaging.

6.2 Users' Experiences

Previous to the users' experiences, training documents about how to use the application were prepared and distributed. Currently two tests have been performed with six people belonging to different institutions. It should be noticed that the experimentation is still underway and more information will be gathered at the end of the project. All of them were using the upload documents and the Notify application with the aim of preparing documentation and presentations for the creation of a common project.

The first testing period produced several suggestions from the users indicating new functionalities such as: uploading several documents, including an option for notifying all users, showing alphabetical users' list to notify, integrating this tool in BSCW, enabling notification to groups and enabling notifications between different shared work spaces systems.

All these issues were analyzed by the developers and almost all of them were included in a new version. Namely, the two last ones imply changes in the business needs specification and building them implies different business processes. Another feature that was not included for the second test was the integration of this tool in

Questions	Mean	SD
Does the tool allow you to perform the uploading of documents and notifying users?	4.66	0.52
Do you think this tool improve the quality of the BSCW?	3.83	0.75
Are you going to use this tool for uploading documents in BSCW?	3	1.26
Do you think that the GUI is easy to understand?	4.83	0.40

Table 3. Results of the tests in the usage of Upload and Notify tools

BSCW. After the implementation of these new features, a new test was performed and the users filled in the test presented in Table 3.

Analyzing the results, the issue that provides different opinions is the third one ($SD > 1$). Some of the users indicated that they prefer to have this tool integrated in BSCW and maybe that is the reason why some users are not willing to use this application. This problem may be solved once the BSCW shows its main functionalities through the use of widgets allowing sharing of information between them as well. Although some widgets regarding BSCW have been developed [34], in the period when the testing was performed the widgets were not available, and currently, further developments should be done for obtaining information from them.

7 CONCLUSIONS AND FUTURE WORK

The use of the Web has increased considerably in the last decade. This increase has produced the emergence of several trends for developing web systems such as Web Services and Web 2.0. While Web Services specifications are intended to solve interoperability issues between heterogeneous systems as well as to offer matured specifications for the composition of services, Web 2.0 offers a significant shift for designing efficient and simpler web graphical user interface.

The influence of the Web is also obvious in areas such as Computer Supported Cooperative Work (CSCW), in which the use of collaborative web applications offers considerable advantages to collaborators. In fact, applying Web Services and Web 2.0 trends to the design of a Collaborative Working Environments (CWE) helps fulfill desirable features such as ease of use, interoperability and goal oriented.

In this paper we have described how we have designed a layered architecture for creating a CWE by combining Web 2.0 and Web Services. Thus, this architecture benefits from enriched services created by the composition of different collaborative systems by means of Web Service specifications, and from an attractive and efficient user interface following a Web 2.0 attitude.

With this kind of CWE, on one hand co-workers benefit from enriched services which automate the flow of information from different services, and on the other hand, co-workers benefit from more efficient interfaces which provide faster interactions and can be customized according to users preferences.

As a proof of concept, we have explained the implementation of two applications following the architecture of the CWE. Thus, we have shown that the combination

of Web 2.0 and Web Services brings significant advantages for the collaborative community, despite the fact that several authors consider that the trends are opposed.

Regarding the evaluation results of one of the tool built following this architecture, we have noticed that users demand integration on it in BSCW. This fact can be solved once this application shows the information in widgets and these widgets can exchange information.

Acknowledgment

This work has been funded by ECOSPACE "Integrated Project on e-professional Collaboration Space" (FP6 IST-035208) and partially supporting for the project 04552/GERM/06. The authors of this paper thank to all the staff involved in the Ecospace project. The authors would like to thanks to Antonio Ruiz for his contribution in the Ecospace and to the team developers for their support: Alejandro Piqueras, Vicente David Guardiola, Manuel Bernal Llinares, Daniel Vicente Fernandez and Jesus Martínez.

REFERENCES

[1] Wilson, P.: Computer Supported CooperativeWork. Kluwer Academic Publishers, Great Britain, 1991.

[2] Fontaine, M. A.—Parise, S.—Miller, D.: Collaborative Environments: An Effective Tool for Transforming Business Processes. Business Journal-Improving the Practices of Management, May/June 2004, pp. 1–10.

[3] Laso Ballesteros, I.—Prinz, W: New Collaborative Working Environments 2020. Proceedings of the 15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2006.

[4] ter Hofte, G.: Working Apart together: Foundation for Component GroupWare. Telematica Instituut, Enschede 1998.

[5] Ecospace Link. Available on: `http://www.ip-ecospace.org/`.

[6] Laso Ballesteros, I.: Research Perspectives on Collaborative Infrastructures for CollaborativeWork Environments. Report on industry-led FP7 consultations and 3rd Report of the Experts Group on Collaboration@Work, European Commission 2006.

[7] Erl, T: Service-Oriented Architecture. Concepts, Technology, and Design. Prentice Hall, Crawfordsville 2005.

[8] Cetin, S.—Altintas, N. I.—Iguztuzun, H.—Dogru, A. H.—Tufekci, O.—Suloglu, S.: Legacy Migration to Service-Oriented Computing with Mashups. Proceedings of the International Conference on Software Engineering Advances 2007.

[9] Motahari Nezhad, H. R.—Benatallah, B.—Casati, F.—Toumani, F.: Web Services Interoperability Specifications. IEEE Computer, Vol. 39, 2006, No. 5, pp. 24–32.

[10] Debevoise, T.: Business Process Management with a Business Rules Approach. Implementing the Service Oriented Architecture. BookSurge Publishing, Roanoke 2007.

[11] Business Process Management Initiative link. Available on: http://www.bpmi.org/.

[12] Intalio link. Available on: http://www.intalio.com/.

[13] O'Really, T.: What is the Web 2.0. Design Patterns and Business Models for the Next Generation of Software. Available on http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html, 2005.

[14] Kwei-Jay, L.: Building the Web 2.0. IEEE Computer, Vol. 40, 2007, No. 5, pp. 101–102.

[15] Fielding, R. T.—Taylor, R. N. :Principled Design of the ModernWeb Architecture. ACM Transactions on Internet Technology, Vol. 2, 2001, No. 2, pp. 115–150.

[16] Smith, K.: Simplifying Ajax-Style Web Development. IEEE Computer, Vol. 39, 2006, No. 5, pp. 98–101.

[17] Lawton, G.: These Are Not Your Father's Widgets. IEEE Computer, Vol. 40, 2007, No. 7, pp. 10–13.

[18] Fox, G. C.—Pierce, M. E.—Mustacoglu, A. F.—Topcu, A. E.: Web 2.0 for EScience Environments. Proceedings of the 3rd International Conference on Semantics, Knowledge and Grid 2007.

[19] Schroth, C.: Web 2.0 versus SOA: Converging Concepts Enabling Seamless Cross-Organizational Collaboration. Proceedings of the 9th IEEE International Conference on E-Commerce Technology and the 4th IEEE International Conference on Enterprise Computing, E-Commerce and E-Services, 2007.

[20] Hutchinson, J.—Kotonya, G.—Walkerdine, J.—Sawyer, P.—Dobson, G.—Onditi, V.: Migrating to SOAs by Way of Hybrid Systems. IEEE IT Professional, Vol. 10, 2008, No. 1, pp. 34–42.

[21] BSCW link. Available on: http://bscw.fit.fraunhofer.de/.

[22] Java Mail link. Available on: http://java.sun.com/products/javamail/.

[23] OpenLdap link. Available on: http://www.openldap.org/.

[24] X-BSCW: XML-RPC Application Programming Interface to BSCW. Fraunhofer FIT. Available on: http://www.bscw.de/english/documentation.html, 2004.

[25] Martínez-Carreras, M. A.—Piqueras, A.—Hernandez, M.—Gómez-Skarmeta, A. F.: Designing of an Interoperable Infrastructure for Synchronous Collaboration. Proceedings of the 3th International Conference on Collaborative Computing: Networking, Applications and Worksharing, 2007.

[26] Ajax ZK link. Available on: http://www.zkoss.org/.

[27] Teo, H. H.—Oh, L. B.—Liu, C.—Wei, K. K.: An Empirical Study of the Effects of Interactivity on Web User Attitude. International Journal of Human Computer Interactions Study, Vol. 58, 2003, No. 3, pp. 281–305.

[28] Folmer, E.—van Gurp, J.—Bosch, J.: Software Architecture Analysis of Usability. In Proceedings of the IFIP Working Conference on Engineering for Human-Computer Interaction, 2004.

[29] Mesbah, A.—van Deursen, A.: An Architectural Style for Ajax. In Proceedings of the Working IEEE/IFIP Conference on Software Architecture, 2007.

[30] Woodall, P.—Brereton, P.—Budgen, D.: Investigating Service-Oriented System Performance: A Systematic Study. Software – Practice and Experience, Vol. 37, 2007, pp. 177–191.

[31] Rud, D.—Schmietendorf, A.—Dumke, R.: Performance Modeling of WS-BPEL Based Web Service Composition. In Proceedings of the IEEE Services Computing Workshops, 2006.

[32] US Department of Health and Human Services. The Research Based-Web Design and Usability Guidelines. U.S. Government Printing Office Official Editions, 2006.

[33] Open Living Labs, available at `http://www.openlivinglabs.eu/`.

[34] Link of available widgets in BSCW `http://bscw.coolfusion.de/`.

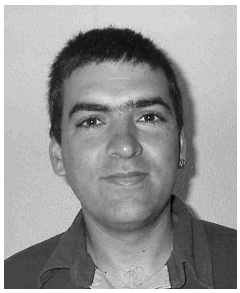
[35] `https://jax-rpc.dev.java.net/`.

[36] Chen, S.—Yan, B.—Zic, J.—Liu, R.—Ng, A.: Evaluation and Modeling Web Services Performance. In Proceedings of the IEEE International Conference on Web Services, 2006.

[37] Alfresco system link `http://www.alfresco.com/`.

[38] EMC Corporation. "Documentum Web Top User Guide".

M. Antonia Martínez-Carreras has a Ph. D. in Computer Science from the University of Murcia (Spain). She is an Assistant Professor for Collaborative Environments at the University of Murcia. She also participates as a teacher in the Master Information and Communication Technologies and Telematics, teaching in the course New Paradigms for Building Information Systems. Her research area is devoted to cooperative systems, CSCW, groupware, CSCL and interoperability issues. She has been working in the Department of Information and Communication Engineering since 1999 until now, where she was involved in several research projects funded by the European IST such as ITCOLE, COLAB and ECOSPACE, a contract to act as a consultant in a Leonardo Project Replika and some funded by the Spanish Foundation Seneca and the Spanish Education and Science Ministry. She has published several papers in national and international conference proceedings and journals.



Antonio Gómez-Skarmeta received the Ph. D. degree in Computer Science from the University of Murcia (Spain). Since 1993 he is Professor at the same department and University. He has worked on different research projects in the national environment, either in the distributed artificial intelligence field (M2D2 project), as in the tele-learning and computer support for collaborative work. He has also been involved in several research and development projects funded by the Spanish Research and Development programs (SABA-Ext, SABA-2, ISAIAS and CROWN Project). He was also coordinator of a Socrates CDA (European Master on Soft Computing) and a Leonardo project for Distance and Open Learning. He has worked in several European IST projects such as COLAB, ITCOLE, Euro6IX and 6Power. He is currently working in the European IST ECOSPACE and POPEYE projects. He has published over 50 international papers.