Model-Driven Design of VoIP Services for E-Learning

Nicola Aste[§], Aldo Bongio[+], Stefano Ceri*, Marco Fais[§], Maristella Matera*, Alessandro Raffio*

[§]AbbeyNet S.p.A.
ex S.S. 131 Km 8,200, 09028 - Sestu, Cagliari - Italy
e-mail: [nicola.aste, marco.fais]@abbeynet.it

[+]WebModels s.r.l.
P. zza Cadorna, 10 – 20123 - Milano – Italy
e-mail: aldo.bongio@webratio.com

* DEI – Politecnico di Milano
P.zza L. da Vinci, 32 – 20133 - Milano - Italy
e-mail: [ceri, matera, raffio]@elet.polimi.it

Abstract. Project-centered learning increasingly demands for communication tools to enhance learners' virtual interaction. In the context of the COOPER EU project a model-based approach has been conceived to support and facilitate the introduction of a set of VoIP (Voice over IP) services within collaborative environments. This paper discusses the integration of two up-to-date technologies enabling Web application design and VoIP services, which have been adopted for the development of the COOPER collaborative environment.

Keywords: VoIP Services, Web Application Design, Conceptual Modelling, Collaborative Environments.

1. Introduction

Project-centered learning is increasingly becoming popular. Learners, who very often are geographically dispersed, are required to perform concrete tasks to master specific projects; in doing so, they need facilities for organizing their collaboration, and also communication tools to enhance their virtual interaction.

In the context of the COOPER EU project [3], a collaborative platform has been produced, by means of a framework that integrates the up-to-date technologies for the model-driven Web application development [2][5][11], with the provision of communication tools based on Voice-Over-IP (VoIP) technologies [1]. The produced platform is totally Web-based, and integrates a synchronous environment (principally supporting the definition and execution of collaborative processes [6]), with synchronous VoIP communication tools, such as audio/video conference, co-browsing, and chat.

The COOPER development framework is strongly supported by a model-driven approach, based on the adoption of the WebML conceptual model for Web application

design [5] and of its accompanying CASE tool WebRatio [2][11]. This paper illustrates the extensions of the WebML model-driven development with primitives for the invocation of VoIP services, specifically conceived for the COOPER project. In particular, Section 2 describes the COOPER platform. Section 3-5 illustrate the main ingredients characterizing the adopted technologies for the model-driven development of Web application and for VoIP communication, and also discuss their integration. Section 6 then presents an example showing the extended WebML model at work for the design of VoIP services. Section 7 finally draws our conclusions.

2. The COOPER Platform

The COOPER platform aims to support team-based, project-centered learning processes, where learners belonging to distributed teams are asked to work on projects and to cooperate to produce some results. The platform is specifically intended to support team members' collaboration and coordination. It therefore integrates technologies for collaborative and flexible processes [6], knowledge sharing and recommendation [4], also supporting pedagogical scenarios and assessment [9].

The distribution of team members using the platform also requires support for the effective communication. A prominent feature of the COOPER platform is therefore the adoption of a set of interactive tools for synchronous communication, which enable the on-the-fly creation of environments for doing synchronous online activities, such as:

- *Moderated One-to-Many Audio/Video Conferencing*, which allows the users to talk and "see" either the conference moderator or other participants. The moderator owns a "token", i.e. it decides either who talks or transmits his own image.
- *Chat*, which allows conference participants to exchange messages in a chat.
- *Application sharing*, which allows conference participants to share applications, also the entire desktop or a remote server.
- *Co-browsing*: the moderator can bring the participants into a synchronous Web guided tour: the Web pages chosen by the moderator are shown on the PCs of the participants.
- Synchronous interaction is also complemented by additional features, such as *tokens* (token passing is equivalent to "raising hands" in presence environments), *voting mechanisms* and *polls*.

In order to cope with the previous requirements, the development of the COOPER's platform is based on the use of advanced methods and technologies provided by the two SMEs involved in the COOPER project:

- *VoIP technologies*, provided by Abbeynet (www.abbeynet.com), are integrated within the platform to invoke synchronous communication services.
- *Model-driven technologies*, provided by Web Models (www.webratio.com) and based on the WebML modelling language [5] and the WebRatio CASE tool [2][11],

support the ease of design and development of the Web-based collaborative environment, also allowing the integration of the Abbeynet VoIP services.

The following sections will focus on the integration of such two technologies.

3. Overview of the Abbeynet VoIP Technologies

The background technology provided by Abbeynet is a customizable platform, *X-VOW (active X Voice Over Web)* [1], which provides communication services that are pervasively integrated into a Web-based environment. As represented in Figure 1, such services allow a complete communication, binding three communication channels easily and efficiently: PC connected to Internet, traditional telephony (PSTN) and Web sites.

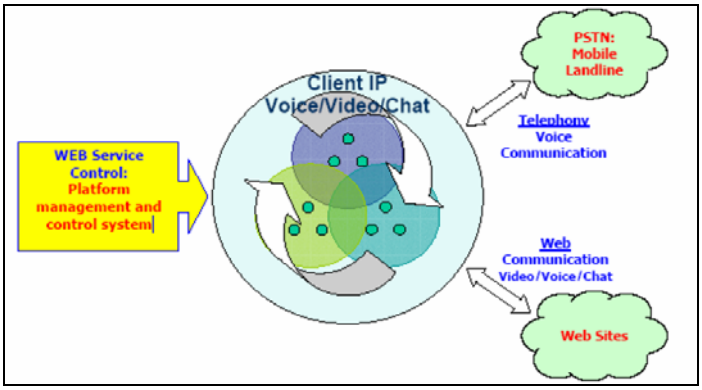


Figure 1. VoIP Communication Platform

The main characteristics of X-VOW, which are most relevant to the development of the COOPER platform, are:

- Supporting all kinds of phone calls (PC to PC, PC to Phone, Phone to PC);
- Supporting chats and all forms of interpersonal communications;
- Supporting video conferencing;
- Supporting co-browsing of documents.

Additional features concern the support of classical communication platforms (call logging, address translation, user identification through aliases, outgoing call barring, detailed record of all calls, directory services and Web Personal Address Books).

X-VOW is based on a hosted model, with server farms offering any communication facility: therefore, third party applications willing to integrate the X-VOW services are not required to deal with maintenance, updates, release issues and the other common concerns that normally arise on a managed-mode provisioning.

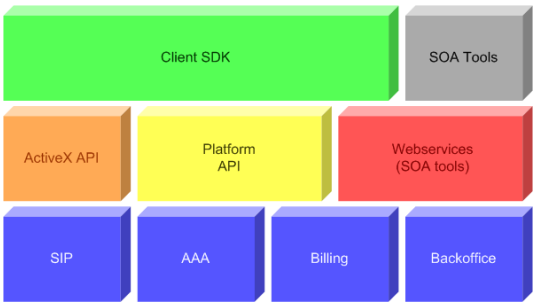


Figure 2. The component-block schema of X-VOW.

Figure 2 illustrates the schema of X-VOW, composed of three different layers:

- The lower layer is made of basic platform components, not directly accessible to developers: they identify the main components that are needed to deploy an enterprise VoIP application.
- The middle layer provides full control of all the platform components, via two set of APIs and Web Services. Interactions with this layer require dealing with SIP and VOI related technical issues. On the other hand, it is at this level that the most creative applications can be composed.
- The highest layer finally allows developers, with low skills on VOI, to deploy a fully functioning Web application, by means of a Web-based software development kit.

Such a layered, component-based model has the advantage of allowing the developer to move gradually to the deeper (and more customizable) layer, by activating only a few of the underlying APIs functionalities. As illustrated in Figure 3, according to such a model, if the need of an external Web application is to invoke VoIP services, without requiring any kind of service personalization, then the designer just needs to enrich some Web pages with some JavaScript code. Such a code instantiates an ActiveX SIP client that enables the connection to the X-VOW server, thus establishing the audio/video calls through the SIP protocol [1].

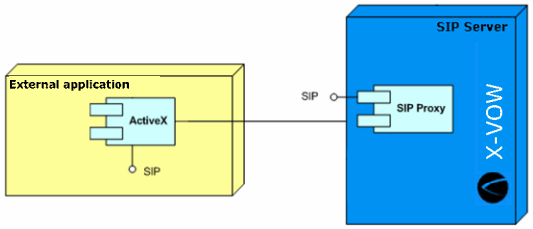


Figure 3. Interfacing an external Web application with the X-VOW services.

4. Overview of WebML and WebRatio

WebML [5] is a conceptual model for Web application design, providing graphical, yet formal, specifications, embodied in a complete development process assisted by tools for visual design and automatic code generation. The main objectives of the WebML development process are:

- Expressing the structure of a Web application with a high-level description, which can be used for querying, evolution, and maintenance;
- Separating the information content (*data layer*) from its composition into pages, navigation, and presentation (*hypertext layer*), which can be defined and evolved independently;
- Storing the meta-information collected during the design process within an XML repository, which can be used during the lifetime of the application for dynamically generating Web pages;
- Enabling the specification of data manipulation operations for updating the site content or interacting with arbitrary external services.

The key aspect of WebML is the capability of defining hypertext diagrams consisting of *pages*, *content units*, and *operation units*, linked to form a specification of the required content publishing or management functions.

The data and hypertext design have been implemented in WebRatio [2][11], a commercial CASE tool for designing data-centric applications using WebML. As showed in Figure 4, the architecture of WebRatio consists of two layers:

- A *design layer*, providing functions for the visual editing of specifications;
- A *runtime layer*, implementing the basic services for executing WebML units on top of a standard Web application framework.

A third module (called *EasyStyler Presentation Designer*) then offers functionality for defining the presentation style of the application, allowing the designer to create XSL style sheets from XHTML mockups, associate XSL styles with WebML pages, and organize page layout, by arranging the relative position of content units in each page.

The design layer is connected to the runtime layer by the WebRatio *code generator*, which exploits XSL transformations to translate the XML specifications visually edited in the design layer into application code executable within the runtime layer, built on top of the J2EE and Struts platforms. In particular, a set of XSL translators produce a set of *dynamic page templates* and *unit descriptors*, which enable the execution of the application in the runtime layer. A dynamic page template (e.g., a JSP file) expresses the content and mark-up of a page in the mark-up language of choice (e.g., in HTML, WML, etc.). A unit descriptor is an XML file that expresses the dependencies of a WebML unit from the data layer (e.g., the name of the database and the code of the SQL query computing the population of the unit).

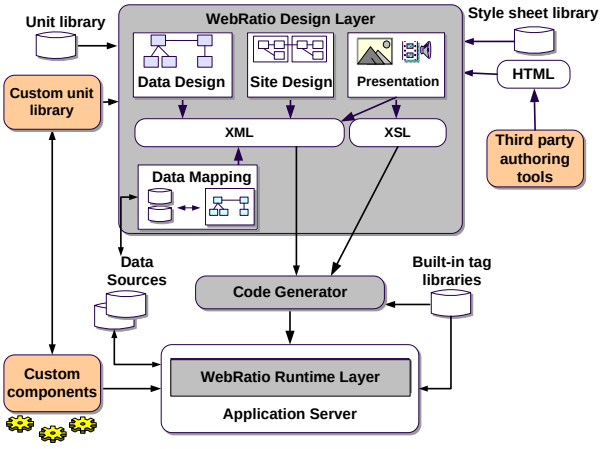


Figure 4. Overview of the WebRatio Architecture.

WebML and WebRatio are built on very few, highly composable concepts, which can be used to assemble complex hypertext applications for publishing or updating content. However, the core units provided in WebML may not be sufficient for covering the entire spectrum of application requirements. To this purpose, WebRatio include the notion of *custom units* (also called *plugin units*).

A custom unit is a content unit or an operation unit defined by the developer, and not pre-defined by the WebML language. It is both similar and different with respect to the standard WebML units:

- Like standard units, it is possible to use a custom unit in a hypertext diagram, link it to other units, with input or output links transporting some parameters, and define the unit's properties.
- Unlike a standard unit, a custom unit has a behaviour that is completely defined by the unit designer.

5. Technology Integration

The integration of the X-VOW services in WebML-based applications required the introduction of new WebML custom units, to represent the invocation of such services within Web pages. It also required reorganising the Abbeynet platform by means of components for synchronous communication that can be independently integrated with the COOPER environment. As better explained in Section 6, such components (represented in a WebML schema by the new custom units) are in fact able to produce parameters (as any other WebML unit), so as to be easily exploited within a WebML conceptual schema.

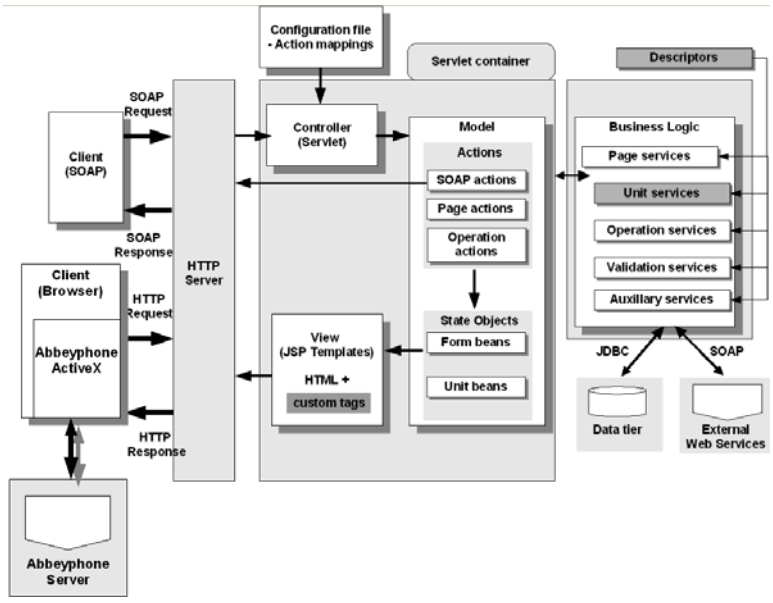


Figure 5. Run-time architecture of a VoIP enabled application generated by WebRatio.

The integration also required the extension of the WebRatio code generation techniques, to produce pages embedding the JavaScript code needed to download at client-side the X-VOW Active X controlling the VoIP service and managing all the occurring events. The COOPER version of WebRatio therefore includes new XSL transformations[1], which allow enriching the Web pages where the audio-video service must be invoked with such JavaScript code.

Figure 5 depicts the run-time architecture of a typical application generated with WebRatio, enhanced with the VoIP services of the X-VOW platform. The architecture leverages the MVC2 (Model-View-Controller) model, where each JSP template is a *View* component, which generates the rendition of a page in HTML, on the basis of the content computed through *page actions* and *page services*. The former are instances of Java classes, able to extract the input from the HTTP request of a page and call the corresponding page service, passing to it the required parameters. The page service is a business function, which computes the page and updates the objects that represent the state of the application in the *Model*. The binding among user's HTTP requests, page

[1] The WebRatio techniques for automatic code generation make extensively use of XSLT technologies to transform the XML-based representation of the conceptual schemas into JSP page templates.

actions and page views is obtained through an action mapping, declared in a configuration file implementing the *Controller*.

Figure 5 clearly shows that the VoIP component directly interfaces the client browser with the X-VOW server, thus providing a loosely coupled integration scheme. This is achieved by means of the X-VOW ActiveX that at execution time is downloaded every time a VoIP service is invoked by a Web page.

6. VoIP Service Specification

In order to integrate VoIP services, two new WebML custom units have been defined:

- The *Push&Speak Unit*, specifying the X-VOW service for audio/video calls.
- The *Conference Unit*, specifying an audio-video conference that also integrates application sharing capability, a chat and a vote system.

The two units are characterized by a set of parameters that they can accept in input (see for example Table 1), which allow managing user identification. They may also produce output parameters. For example, Table 2 reports the parameters produced by the Conference unit, which can be used to log conference events.

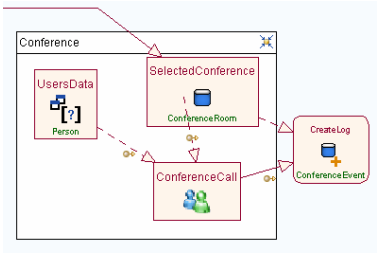
Table 1. Input parameters accepted by the Conference Unit

Current Alias	The X-VOW alias of the user entering the conference. Usually, it is the user logged in the site-view where the unit is placed.
First Names	An array of strings storing the first name of the allowed participants.
Last Names	An array of strings storing the last name of the allowed participants.
X-VOW Aliases	An array of strings storing the Abbeyphone aliases of the allowed participants.

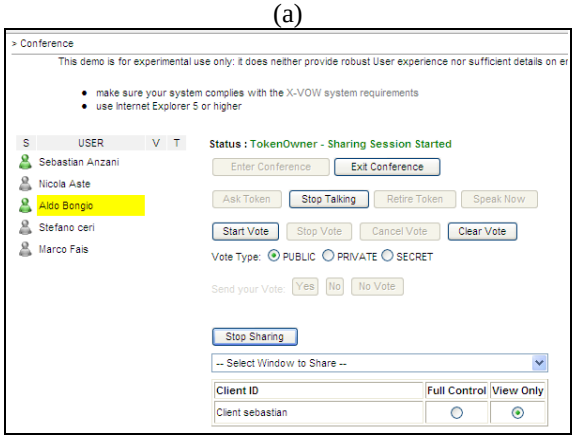
Table 2. Output parameters produced by the Conference Unit

User Alias	The X-VOW alias of the user generating the event.
Event Type	The type of generated event. Possible values are: START, END, STARTSPEAK, ENDSPEAK, VOTE and CHAT.
Event Details	A brief description of the occurred event.
Timestamp	The timestamp of the occurred event (yyyy-MM-dd hh:mm:ss.SSS).

Figure 6(a) depicts the WebML-based specification of a Web page supporting the use of the conferencing service. The *Conference* page is accessed after selecting a conference room in a previous page, as represented by the link entering the page and reaching the *SelectedConference* unit. This unit represents the data of the conference room, which are then sent in input to the *ConferenceCall* unit, which is in charge of managing the service invocation. This unit has another input link, which transports some parameters for users' identification, represented and computed through the *UsersData* unit. The link exiting the *ConferenceCall* unit then exposes parameters representing the conference events. As represented by the operation unit *CreateLog*, such events are logged by storing them within the application data source.



(a)



(b)

Figure 6. The WebML specification of a page invoking the Conference service (a) and its HTML rendition (b).

Figure 6(b) then shows the HTML page rendition[2]: the users' data, specified by the unit *UserData*, are visualized by the index placed at the left side of the page. The central panel then shows the set of commands to control the conferencing services, as provided by the X-VOW ActiveX control.

7. Conclusions

Modeling is an essential tool in the development of complex software. In recent times models have gained importance becoming de-facto development artifacts that replace traditional code writing. In this paper, we have discussed the use of models in the development of collaborative environments that integrate VoIP services. In particular, we have shown the tight integration of a conceptual model for the design of synchronous Web applications with synchronous tools enabled by state-of-the-art VoIP technologies.

[2] The page design recalls some other popular conferencing services, such as those provided by flashmeeting (www.flashmeeting.com).

Several research and commercial collaborative platforms (see for example CURE [7], XCHIPS [10], and also WebCT, BlackBoard, IBM LearningSpace) support synchronous and asynchronous communication. Such systems are however "rigid": the customization of the provided collaborative environments and the addition of new tools is difficult (or even impossible), since it requires modifications at the code level. The main merits of the illustrated framework with respect to such systems can be summarized as follows:

- *Ease of deployment:* the model-driven approach facilitates the application design and development, thanks to the adoption of high-level intuitive models, and the availability of code generators that automatically produce the application code.
- *Ease of customization* (by means of wizards): the usability of the development framework has been enhanced through a set of wizards, able to guide the user in the process of adding VoIP services to an application.

Some preliminary experiments conducted on the COOPER platform with real users [8] revealed that the COOPER VoIP services are considered useful to support collaboration. Our current work is however devoted to extensively testing the X-VOW services to improve their quality.

8. References

[1] Abbeynet S.p.A. X-VOW – Platform Overview. http://www.abbeyphone.com/VOW/sdk/platform_overview.php. 2007.

[2] R. Acerbis et al. WebRatio, an Innovative Technology for Web Application Development. Proc. of ICWE 2004, pp. 613-614, Springer Verlag, 2004

[3] A. Bongio et al. COOPER: Towards a Collaborative Open Environment of Project-Centred Learning. Proc. of EC-TEL 2006, Springer Verlag, pp. 561-566, 2006.

[4] A. Bozzon, T. Iofciu, W. Nejdl, S. Tönnies. Integrating Databases, Search Engines and Web Applications: a Model-driven Approach. Pro. of ICWE 2007, Springer Verlag. In print.

[5] S. Ceri, P. Fraternali, A. Bongio, M. Brambilla, S. Comai, M. Matera. *Designing Data Intensive Web Applications.* Morgan Kaufmann, 2002.

[6] S. Ceri, M. Matera, A. Raffio, H. Spoelstra. Flexible Processes in Project-Centred Learning. Proc. of EC-TEL'07. Springer Verlag. In print.

[7] J. M Haake, T. Schmmer, A. Haake, M. Bourimi, B. Landgraf. Supporting flexible collaborative distance learning in the CURE platform. Proc. of IEEE HICSS'04, IEEE Press, 2004.

[8] V. Posea, S. Trausan-Matu, V. Cristea. "Online Evaluation of Students' Opinions about the Collaborative Learning System they Use", submitted for publication to ICCP 2007.

[9] H. Spoelstra, M. Matera, E. Rusman, J. van Bruggen, R. Koper. Bridging the Gap Between Instructional Design and Double Loop Learning. In: *Current Developments in Technology-Assisted Education.* ISBN 84–690–2472–8, 2006.

[10] W. Wang, J.M. Haake, M. Wessner. Supporting Cooperative Learning in Distributed Project Teams. Proc. of CoopIS 2002, Springer, 2002.

[11] WebModels s.r.l. WebRatio CASE tool. http://www.webratio.com. 2007.