Extended sensations on interactive telecommunication

Juan Antonio Álvarez[1], Juan Antonio Ortega[1], Alejandro Fernández[1], Manuel David Cruz[1], Pablo Castilla[2]

[1] Dept. Lenguajes y Sistemas Informáticos, Univ. Sevilla.
Av. Reina Mercedes s/n. 41012 Sevilla, Spain
{alvarez,ortega}@lsi.us.es, {alejandro.fdez, manuel3086}@gmail.com
[2] Dept. Administrativo, INDRA Sistemas.
Avda. de San Francisco Javier, 22
41018 Sevilla, Spain.
pcastilla@indra.es

Abstract. Although computer science and telecommunications have evolved from the 90's to nowadays, the way people communicate each other through electronic devices has stuck since the appearance of the videoconference. The audiovisual interaction seems enough, but we think the sensation of presence gets higher when we allow the person who we are talking to, taking part in our environment using and enriching it. In that way, the interaction is improved with new sensations which come from outside of our device. In this article, we propose a communication service which provides not only videoconference, but manages the smart environment where the interlocutor of the communication is, allowing him to access to any device connected through domotic and/or wireless network. The application is developed under the OSGi open specification and it almost doesn't imply any cost increase over the bandwidth used during communication. Once explaining the technical foundations, its diverse applications in ubiquitous meetings and on-line games will be commented.

1. Introduction

The evolution of communication systems has allowed the improvement of interaction methods. Nowadays, we can start a high quality videoconference from our mobile phone. However a face-to-face conversation is always pleasanter and more effective, due to the sensation of presence. The relation between this perception and the level of attention has caused the success of a wide range of communication applications.

Application\Features	Sensation of presence	Level of necessary attention	Communication speed
E-mail	Null	Low	Low
Instant Messaging	Reduced	High	Medium
Conference (audio)	Medium	Medium	High
Video-conference	High	High	High
Face-to-face conversation	Maximum	Maximum	Maximum

Table 1: Interactive communication applications

A row for the face-to-face conversation was added to compare the next features:

Sensation of presence: When we are reading an e-mail, this sensation does not exist, but seeing our interlocutor this perception goes up.

Level of necessary attention: The need to pay attention to a speaker is the highest if we are in the presence of the speaker. We must attend if we want to observe all their gestures and obtain all the information. However, this attention is reduced if we do not sense his presence.

Communication speed: A basic feature, the rate of exchange of information.

As we can see, sensation of presence is in direct proportion to the level of necessary attention. The exception is in the instant messaging. Attention must be paid reading the messages and writing the response, so the speed is reduced too. As we have said, our behaviour changes when we see or listen to our interlocutor. What about when we feel the sound of his shoes or the light of his room? Sensation of presence is increased when we let our interlocutor acts over our environment, (e.g. switching on lights, turning on TV, etc.). Without almost any cost on necessary attention the feeling of presence could improve our interaction.

In this paper, this idea will be developed through two applications. The first one is an instant messaging client that includes a management interface for OSGi [5] bundles implemented by OSCAR [7]. Second one uses an extension of the famous communication client skype 2.0 [9] (that offer beta videoconference). This extension allows the connection with Knopflerfish [10] through web services offered by a bundle from Apache Axis [2].

Firstly we show a reduced study of the management of OSGi components on its Framework. Next we will describe both implemented applications and we will propose some scenarios for illustrate their possibilities. Finally we will explain our future work.

2. Development

The OSGi specification [6], oriented to supply services from WANs to LANs or device networks, is perfect to increase the users perceptions of a communication through electronic devices. OSGi allows to use the Internet power and to combine it

with the potential of a home management through domestic wiring networks or wireless communications. Although the fourth OSGi specification appeared in 2005 October, two open implementations called OSCAR and Knopflerfish exist that implement the third specification and these are very complete.

Next we describe how the bundles installation is managed [4] and how these are loaded in the framework.

When install a bundle is decided to, its manifest file, that describes the dependencies that it has with other bundles, is verified, and if it would be necessary, it installs those components before installing the chosen one. After this check is done, an identifier (called bundle Identifier) is assigned to the bundle. This identifier allows other user bundles to access it to start, stop, uninstall or even update it.

2.1. Instant messaging client + remote management interface:

The first implemented approach was to modify the code of an instant messaging bundle implemented by Richard Hall to OSCAR bundle repository. In its interface was introduced some HttpAdmin component's parts to manage the life cycle of the components through theirs bundle Identifiers using HTTP requests.

The application was tested with a series of components as mp3 player bundles and X10 devices management. The following problems were found:

- The messaging client was very simple and its strength and functionality had to be improved.
- In order to be able to use it, it was necessary to install OSCAR in all devices with which it was desired to allow a conversation.
- Difficulties to install OSCAR in mobile devices.
- Problems derived from instant messaging: too much attention for a slow communication and a reduced presence sensation.

In spite of mentioned disadvantages, it was checked that the access through the interface to initiate music services and activate and stop devices using X10, was very intuitive and functional. For that reason, it was decided to look for a second option that allowed to improve the communication and to extend the functionality with conferences and videoconferences.

2.2. Skypebot for Skype 2.0

In this section we are going to explain an example application that connects one of the most popular videoconference programs, Skype 2.0, with a web service which connects to OSGi.

The second implementation that was used was Knopflerfish. Its component's warehouse offers some bundles with different functionality that OSCAR offers, emphasizing one called osgi-axis, a bundle which permits to explore the services offered by the OSGi Framework to a web services server. In this way, our connector offers its services through the framework and through SOAP messages, without

needing the installation of an OSGi's implementation in all the machines that we pretend to communicate, as occurred in the previous system.

As we noticed that the instant messaging client in chapter four was poor, we look for VoIP clients with these requisites: they must be extensible and popular. After analyzing a few of them we decide to use Skype. This client has connected 6 millions people simultaneously and offers an API that could be used by others applications.

The way for extending it was though a skypebot programmed under Visual C# using Visual Studio 2005. After added a component to the application that encapsulate Skype's API, skypebot were programmed as a message parser. When the user sends a command, they are shown at the beginning of the conversation, which the parser recognizes skypebot invokes through a web service offered by axis the starting or stopping of an OSGi bundle. To access a concrete bundle is used the getBundles method which shows the bundles Identifiers which are installed in the framework and choose the one which name correspond with the desired service.

In this way, users can connect to conferences, or videoconferences, and enrich the environment through the instant messaging client that offers Skype. In addition, as Skype has clients for mobile devices, using it is not limited to PC's or laptops.

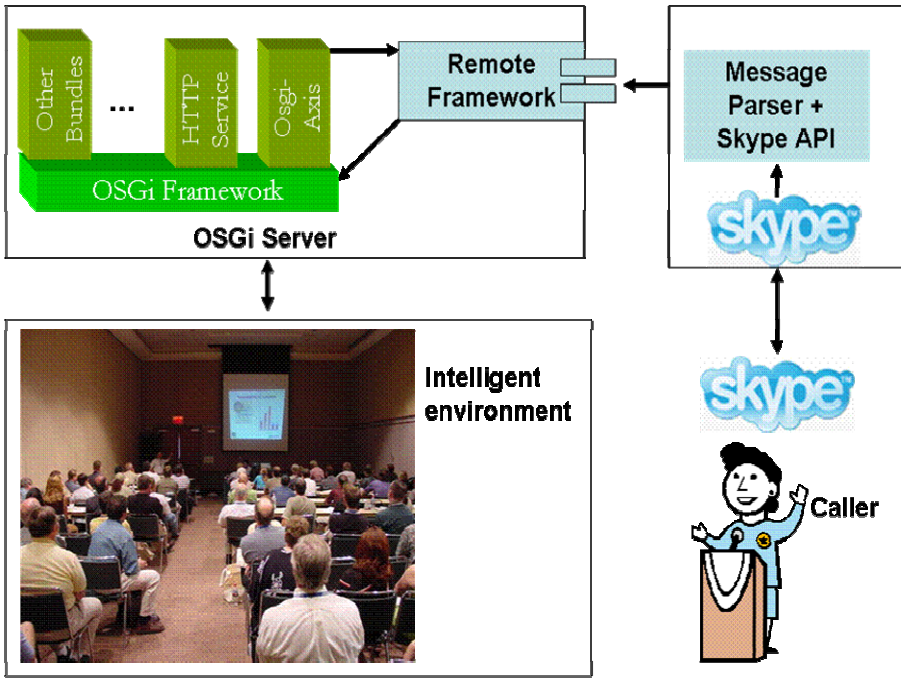

Image 1: Architecture of second application

Although the testing was done by determined bundles (mp3 player and X10 manager), we developed another web service that provided two methods:

ServiceList(): returns an object which contains a list with the names of the services that the building in which the other caller is provide. This could be: lower video projector, raise video projector, turn on lights, play a video…

InvokeService(serviceName): invokes an OSGi Framework's service.

As in all Internet communications we need security, for achieving this we could use security features that are provided by WSE 3.0, or if this is not admitted by the server we could develop another method, validate(name, password), which should return a ticket. We could use a VPN too.

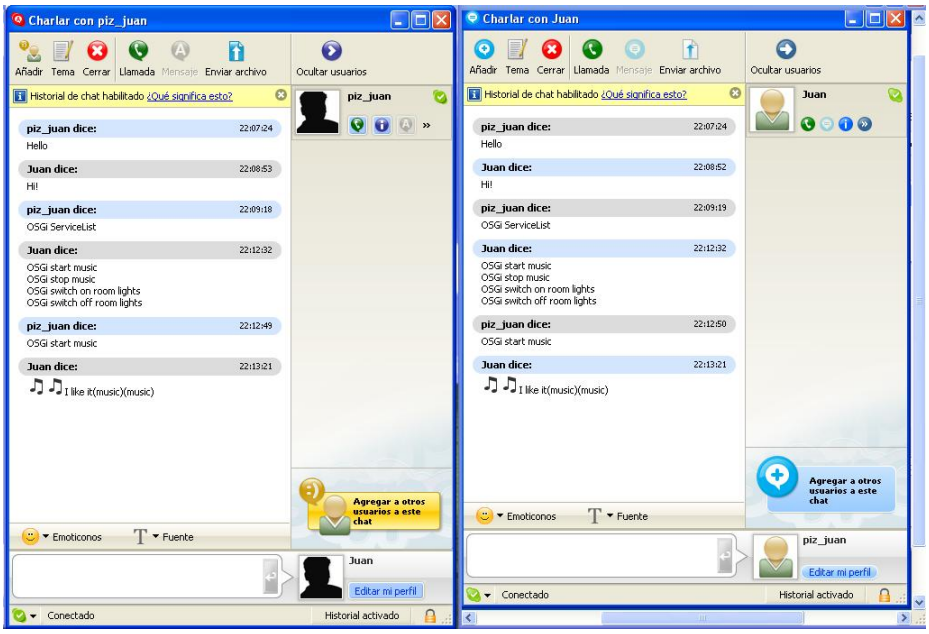

Image 2: SkypeBot example

3. Scenarios

- **Ubiquitous meeting**: The director of an international company needs a meeting with some executives in USA and Germany. He does not want to disturb them with flights so he organizes an enhanced videoconference. He connects through his PDA to Skype and begins an audio conference. Done the salutations, he obtains the services offered by both meeting rooms and starts a web presentation for German people and a video conference with Americans, switching off the lights of both meetings and setting up the screen where the speech will be shown. Then, he begins his locution. after the ending of both presentations, the director switches on the lights and the services of videoconferences of both rooms and he talks simultaneously to American and German audience.

- **Hiding online games**: With several webcams spread through the callers' homes, with two axis rotation capabilities, the game starts. Webcams allow looking for the user hidden into some home's rooms. The skill of the user who manages the cameras should be fast to win user hides and the other controls the cameras of the hidden user building. The one who find the opponent in less time wins.

4. Conclusions and Future work

Although the functionality of existent personal communication systems like skype chat have increased, the level of necessary attention was the same due to the need of writing messages. According to this, we are orienting our work to voice interfaces for using devices without the needing of using instant messaging clients. The technology that we intend to use is WinFX [8], the new speech recognition and synthesis API in Windows Vista.

We work also on the remote management of wearable devices, so embedded OSGi Framework could offer through WiFi connection web services to act over these devices. The application on enhanced wheelchair will be helpful if someone could help remotely a handicapped person. Tele-care environments are also another scope of our research.

5. References

1. Andreas Frei, Gustavo Alonso. A dynamic lightweight Architecture. Published in the Proceedings of the 3rd International Conference on Pervasive Computing and Communications (PerCom 2005), March 2005.
2. Axis. Apache Web Services Project. http://ws.apache.org/axis/
3. Chris Loeser, Wolfgang Mueller, Frank Berger, Heinz- Josef Eikerling. Peer-to-Peer Networks for Virtual Home Environments. hicss, vol. 09, no. 9, p. 282c, September 2003.
4. Humberto Cervantes and Richard S. Hall. Automating Service Dependency Management in a Service-Oriented Component Model. In Proceedings of the 6th Workshop on Component-Based Software Engineering (CBSE), May 2003.
5. Open Source Gateway Initiative. OSGi. 1999.
6. OSGi Alliance. OSGi Service Platform Release 3. 2003
7. Richard S. Hall. OSCAR, An OSGi framework implementation, http://oscar.objectweb.org
8. Robert Brown. Exploring New Speech Recognition And Synthesis APIs In WindowsVista. MSDN Magazine, January 2006.
9. Skype 2.0, http://www.skype.com
10. The Knopflerfish project, http://knopflerfish.org/index.html