Compositional and Architectural Evolution Issues: a comparison between JMF and MultiTEL*

M. Pinto, M. Amor, C. Canal, L. Fuentes
Depto. de Lenguajes y Ciencias de la Computación
Universidad de Málaga (Spain)
e-mail:{ pinto, pinilla, canal, lff,}@lcc.uma.es

Abstract

The growing complexity of Web-based services in general, and multimedia services in particular, makes it necessary to apply engineering methods to their development. In particular, Multimedia Engineering may be considered as the field in Software Engineering which deals with the development of open, reusable and quality multimedia and Web-based software. In this paper we apply two of the current trends in Software Engineering, such as component and framework technologies, to the development of multimedia services over the Web, presenting and comparing different solutions in use today.

Keywords

Component-Based Software Engineering, Framework Technology, Web-based Multimedia Services, Distributed Web-based Systems, Java Media Frameworks, MultiTEL.

1. Introduction

The spread of Internet and the World Wide Web contribute to the development of new and complex services, many of them involving the interchange of multimedia information. Although this Multimedia Services (MS) are not new, they are in constant evolution due to the advances in technology, which makes now possible the implementation of synchronous real-time services that exchange high-quality audio and video.

The rapid construction of reliable, portable and efficient MSs faces with the growing complexity of these systems, the heterogeneity of multimedia devices and the diversity of operating systems and communication platforms. This argues for the development of new systems that are reusable, extensible and open. In this context, Component-Based Software Engineering (CBSE) [1] and Framework Technology [2] are becoming essential in order to reduce the cost and improve the quality of software. Our work focuses on Multimedia Engineering as a discipline of Web engineering [3], to establish the development methods that guide the design and implementation of multimedia applications [4]. Consequently, we apply the aforementioned technologies to Multimedia Engineering.

Current platforms offer some kind of APIs to deal with the capture and management of multimedia devices (e.g. Silicon Graphics' Digital Media Library [5]), and many of them are based on component and framework technologies (e.g. Microsoft's Windows Media Technology [6], Sun's Java Media Framework (JMF) [7]). In a component-based platform, components act as replacement units that can be composed to build extensible, reusable and open systems. This technology allows the construction of an application by plugging commercial off-the-shelf components, developed at different times, by different people, and with different uses in mind [1]. However, the flexibility provided by components should be complemented by a framework that imposes some underlying architecture [8]. A framework is a reusable, "semi-complete" application that can be specialised to produce customised applications [2] and therefore, framework technology can be considered a better approach than object-oriented or component-oriented paradigms isolated.

* This paper is an extended abstract from "Towards Multimedia Engineering on the Web: a Case Study", which is now being considered for publication in the journal *Multimedia Tools and Applications* (Kluwer Academic Publishers).

The main goal of this paper is to present a comparative study between two development frameworks for the construction of MSs: JMF and MultiTEL [9], a non-commercial compositional framework which was developed by the authors of this paper[1]. The main criterion applied in this study are the extensibility and adaptability of each framework to the evolving requirements of MSs, like the use of different communication protocols, the negotiation of data formats, or the management of participants with different user profiles or that play different roles in the service. While making the comparison, we have found how some limitations and inconveniences of JMF (such as the difficulty to understand and extend the base components provided, and the lack of a full architecture for the construction of MSs) are easily faced using MultiTEL.

2. Technologies for Multimedia Services Development

The main difference between MSs and other distributed services is the diversity of hardware devices (e.g. cameras, microphones, speakers, etc.) managed, which increases the complexity of multimedia programming. These devices capture/present media data in different formats, which makes format negotiation between the participants necessary. Also, other requirements must be also taken into account, such as participant preferences, the capacity of the network, or the quality of service (QoS) required. Each participant in the service may have different platform-specific devices that produce real-time data in different formats. Users should be able to execute MSs in heterogeneous systems, and devices must be modelled by components that are managed locally by each participant, offering a common interface to the service.

Other important issues concerning multimedia programming are the media synchronisation and the use of different communication protocols in different versions of the same service, like UDP and RTP (Real-Time Transport Protocol) [10]. RTP is a transport protocol designed to be used by applications with real-time requirements. It provides information about the format and the time of the transferred data allowing the correct reconstruction of data in the target. Furthermore, it offers control information about the quality of current transmission and the participants engaged in the session by the RTCP (Real-Time Transport Control Protocol) protocol.

2.1. MultiTEL Overview

MultiTEL provides a distributed compositional platform for the management of network resources and multimedia devices in a MS [11]. This framework models multimedia devices as components that implement standard interfaces. Currently, several multimedia, video on demand and collaborative services developed using MultiTEL are being used in the area of educational services at the University of Málaga.

The MultiTEL platform is divided into two levels, as shown in Figure 1. The higher level provides a frame architecture to bind MSs by dynamically composing the service, multimedia, and network subsystems. In MultiTEL, a MSs is made up from a set of components which model entities of the real world, and a set of connectors that implement the coordination protocols between components. Component/connector communication is defined as that components throw events and receive input messages, while connectors catch events and send output messages. However, components and connectors are not wired together in a fixed manner, but are dynamically connected by using the services provided by the User Service Part (USP) object, located in the lower level of the architecture. When a component sends an event, the USP determines the connectors that must manage this event. Thus, a component does not know the address of the target connectors. From the user point of view, the USP is simply an applet that is downloaded through a web navigator.

Another important component is the *resource manager* (RM). In MultiTEL, the devices are captured through the RM that hides device and platform dependencies by providing a unique identifier and a common interface for each device type. The MS asks the resource manager for capturing a device, giving it the unique identifier of the device and the desired format. Hence, the MS does not need to know the actual implementation class; it only has to invoke the methods of the common interface implemented by the device. Only the local RM knows the implementation class. The components that model the devices join the service at runtime through a plug&play mechanism.

[1] The Web page of MultiTEL can be found at http://www.lcc.uma.es/~lff/MultiTEL

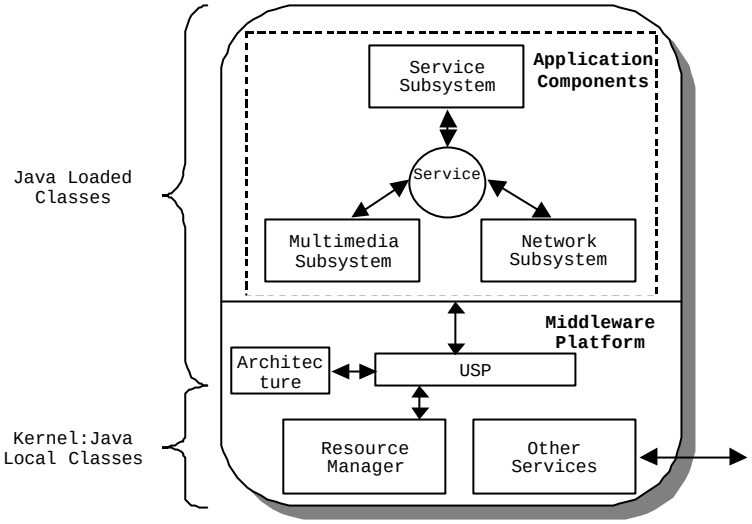

Figure 1. MultiTEL Platform

2.2. JMF Overview

The JMF 2.0 API, introduced in November of 1999 provides Java with support for capturing and storing media data. JMF defines two different ways of programming. The first one is the presentation of local or remote audio and/or video files, or real time data captured from a camera or a microphone. JMF offers plug-ins to do format conversions, multiplex or demultiplex data or synchronise several media. Secondly, to develop distributed MSs with real time streams JMF provides a set of components that implement the RTP protocol (through the RTP API).

Like MultiTEL, JMF provides a *resource manager* (RM), which deals with the registration of multimedia devices and provides device information to capture them. However it does not performs the actual capture. Moreover, JMF does not provide an atomic method to create a device. Instead of that, an application must invoke several methods that create a set of chained components, but only one of them really captures the device. In order to use them, the MS designer must have a deep knowledge in multimedia programming and JMF API classes.

3. MultiTEL versus JMF

We are going to compare MultiTEL and JMF based not only on the complexity of building multimedia services with them, but also on how extensible, reusable, maintainable, and interoperable they are. Another important issue is the extra functionality that both frameworks offer additionally to the capture and management of multimedia devices. The comparison is made on our own experience building basic videoconference and Video on Demand (VoD) services using these platforms.

Both JMF and MultiTEL are implemented in Java and define a set of classes for the management of multimedia devices independently of local platforms, and both provide a reference architecture used in the development of multimedia services.

However, JMF only provides a framework to capture multimedia devices and transmit the media data between the participants, while MultiTEL allows the definition of complete MSs, covering all the phases of the service, which includes not only device capture and data transmission, but also the connection, negotiation, and scheduling phases. In addition, MultiTEL offers a directory service with pre-developed components and applications that developers can use to build their own services.

Hence, JMF does not provide a reference architecture to guide the implementation of MSs. In the development of a videoconference service with JMF, we were only able to capture the participants multimedia devices (e.g. cameras, microphones, speakers and displays), and to manage the transmission of

multimedia data. The rest of tasks related to the development (for instance, the *participant connection phase*, where the system should check is a participant is registered for the conference, or in the *schedule phase*, that establishes the different roles played by the participants) had to be implemented from scratch.
On the contrary, MultiTEL offers several possibilities for each of these phases. For instance, it offers several *access protocols*, like the *manager access protocol* in which the manager decides whether a user can join the service, or the *database access protocol*, where there is an access control database containing the users subscribed to a service.

With respect to device capture, both frameworks use a resource manager to capture and manage multimedia devices, and both use a set of interfaces that must be implemented by the service components. Nevertheless, JMF allows the capture of devices without the use of the resource manager, by knowing the device identifier. This solution is platform-dependent because device identifiers change from one system to another. On the contrary, platform dependency is hidden in MultiTEL by the definition of a unique identifier for each device (i.e. "camera", microphone", …) and by always capturing the devices through the resource manager.

In addition, the main restriction imposed by JMF in the development of videoconference and VoD services is that you are compelled to use RTP as the transport protocol. We agree in that RTP is an appropriate protocol for multimedia data transmission, but it is not the only way to transfer multimedia data. Someone may like to use IP multicast, the services provided by an ATM network, or a reservation protocol like RSVP. Hence, the JMF architecture is not generic for multimedia services that require the use of an arbitrary communication protocol. Furthermore, JMF does not offer the common interface to model the connection between the participants in the service, so it is not easily reusable or extensible with new service requirements related to communication protocols. For instance, if a service needs to use proprietary protocols, or if a new protocol appears, improving the transmission of multimedia data, the service developers must wait until JMF provides the components implementing these protocols.

Now, we are going to explain the device capture and data transmission in MultiTEL. MultiTEL divides these tasks in separate, although very close, architectures: the *multimedia architecture*, and the *network architecture*. The main difference with JMF is that MultiTEL provides a common architecture to all types of videoconference and VoD services. Unlike JMF, where only the producer devices are captured, in MultiTEL both the producer and the receiver devices are captured prior to the transmission. Once the devices have been created, the service accesses them through a connector which encapsulates the device behaviour. This is more extensible because allows the RM to manage all the devices in a uniform way, and makes easy the addition of new receiver devices, for example a CD recorder. In MultiTEL the MS does not notice the difference between writing in a CD recorder or in a file, while in JMF it may have to be treated with a different component, which would impose a change in the developed service.

Once the devices are captured, the connectors in the network architecture connect them and they begin to control the transmission. The main advantage of our architecture with respect to JMF is the use in each participant of local components that encapsulate the communication protocol used in the service. Changing the communication protocol in a MultiTEL service is as easy as providing a new implementation for these components. Without changes in any other component or connector of the service, you could obtain a completely new service.

Another very important issue in multimedia programming is *data format negotiation*. The developer cannot expect that all the participants who join the service will transmit data in the same or compatible formats. On the one hand, each participant in the service will have different devices and these devices will generate or consume a different set of data formats. In many cases, these formats will be incompatible with other participants' device formats. On the other hand, there are multiple criteria to approach format negotiation. Firstly, we can take into account user preferences given in terms of the availability of plug-ins to manage the received data or the accessibility of codecs to change the data format and decodecs to understand the data received.

In MultiTEL, format negotiation is not a problem since the architecture includes connectors for implementing the negotiation protocol of a service. These connectors can encapsulate a negotiation protocol as complicated as needed, while the rest of the service remains unchanged. On the contrary, the JMF architecture does not contemplate how the different participants negotiate data formats. Every participant

knows which format is using for data transmission, and it can also change this format if it encounters problems in the transmission when analysing the control information that they receive through the RTCP control packets, but it does not know whether the other participants will be able to reproduce the data it is sending to them.

4. Conclusions

Our experience building Web-based multimedia services shows that it is possible to improve their design by applying CBSE methods and Framework Technology. Multimedia programming includes new issues, such as resource reservation, format negotiation, or real-time restrictions that make difficult the development of these services from scratch. The compositional framework technology offers a way to abstract all these aspects allowing the construction of open, reusable and extensible services.

The central aim of this work has been the comparison between JMF and MultiTEL. We have observed important differences between the current compositional frameworks of the multimedia domain. We have focused in how ease-of-use, extensible and adaptable both approaches are.

We have shown that: 1) Using MultiTEL, the development of a multimedia service is easier that using JMF, because JMF only provides the reference architecture for the capture, transmission and presentation of multimedia data, and developers must implement the rest of the service from scratch. On the contrary, MultiTEL offers a full service architecture; 2) The learning curve of JMF is higher than that of MultiTEL, due to the complex relationships between the components imposed by the JMF architecture; 3) Videoconference and video on demand services in JMF impose the use of the RTP communication protocol; 4) It is more difficult to extend a service in JMF that in MultiTEL; and 5) It is more difficult to add new behaviour (e.g. incorporate a new multimedia device) in JMF than in MultiTEL.

Sun is continuously providing new versions of JMF. Recently, it has announced a new version, JMF 2.1.1. However, we think that the complexity in the development of multimedia services with JMF will remain the same, because the JMF API will not change in this new release.

However, we are conscious that another significant criteria to compare both frameworks is efficiency. JMF offers a more efficient implementation because it access directly to device drivers, while MultiTEL implements the multimedia device components through Java native methods.

As a result of this comparison, JMF extensions which solve some of its limitations can be defined, in order to obtain a framework more standard than MultiTEL. The resulting framework would provide a reference architecture to develop MSs over the Web using JMF.

5. References

[1] W. A. Brown and C.W. Kurt, “The Current State of CBSE”, *IEEE Software*, September/October 1998.

[2] M.E. Fayad, D.C. Schmidt, "Object-Oriented Application Frameworks", *Communications of the ACM*, October 1997, Vol. 40, No. 10.

[3] A. Ginige, S. Murugesan, Special Issue on Web Engineering, IEEE Multimedia, 2001, Vol 8, No. 1 & 2.

[4] S. Murugesan, et al. , "Web Engineering: A New Discipline for Web-Based System Development", *First ICSE Workshop on Web Engineering*, Los Angeles (USA), May 1999.

[5] Silicon Graphics, “Digital Media Programming Guide”, IRIS Insight Library.

[6] Microsoft Corporation, “An introduction to Windows Media Technology”, http://msdn.microsoft.com/workshop/imedia/windowsmedia/IntroToWMT.asp

[7] Sun Microsystems, Inc, “Java Media Framework Guide”, http://java.sun.com/products/java-media/jmf/2.0/jmf20-fcs-guide/JMFTOC.html

[8] D. Krieger, “The Emergence of Distributed Component Platforms”, *IEEE Computer*, March 1998.

[9] L. Fuentes, J.M. Troya, “A Java Framework for Web-Based Multimedia and Collaborative Applications”, *IEEE Internet Computing*, March/April 1999, Vol. 3, No. 2.

[10] H. Schulzrinne. “RTP. A Transport Protocol por Real-Time Applications”, RFC 1889, Enero 1996

[11] L. Fuentes, J.M. Troya, “A Component-Oriented Architecture to design Multimedia Services on a distributed platform”. *Proc. of WorldWide Computing and its Applications. WWCA’97*. Tsukuba (Japan), Springer LNCS No. 1274, 1997.

Authors' background

The authors of this paper are members of the Software Engineering Group (GISUM[1]) at the University of Málaga (Spain). We include a short biography of the author(s) who most likely would attend the workshop.

Carlos Canal

Carlos Canal is an assistant professor at the Department of Computer Science of the University of Málaga (Spain), where he received an MSc. Degree in Computer Science in 1993. In 2000 he completed his PhD. Thesis on Software Architecture and Architectural Description Languages. Apart from these architectural issues, his research interests address Component Based Software Engineering, program composition and compatibility, and interface description using process algebras.

Lidia Fuentes

Lidia Fuentes is an associate professor at the Department of Computer Science of the University of Málaga, Spain. She received an MSc degree in Computer Science from this university in 1992, completing afterwards her PhD thesis on multimedia middleware, component frameworks and telecommunications services in 1998. Her research interests address practical applications of composition and frameworks to collaborative work inside intranets, and she also works on multimedia programming in distributed systems.

Expectations towards the workshop

Most of our research work deals with architectural, component and framework issues and its applications to the development of complex systems, in particular to multimedia services, where we have applied these technologies to the development of MultiTEL, a platform for building multimedia services such as videoconferences and VoD systems. For this reason we are very interested in participate in this workshop, where we could discuss and interchange our points of view with other researches and groups working on similar issues. We hope that the discussions and activities of the workshop will enrich both our work and that of the rest of the participants.

[1] The Web page of GISUM can be found at http://www.lcc.uma.es/~gisum