



HAL
open science

Typing of Adaptation Connectors in MMSA Approach Case Study: Sending MMS

Makhlouf Derdour, Marc Dalmau, Philippe Roose, Nacira Ghoualmi-Zine

► **To cite this version:**

Makhlouf Derdour, Marc Dalmau, Philippe Roose, Nacira Ghoualmi-Zine. Typing of Adaptation Connectors in MMSA Approach Case Study: Sending MMS. *International Journal of Research and Reviews in Computer Science*, 2010, 1 (4), pp.39-49. hal-00541829

HAL Id: hal-00541829

<https://hal.science/hal-00541829v1>

Submitted on 1 Dec 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Typing of Adaptation Connectors in MMSA Approach

Case Study: Sending MMS

Makhlouf Derdour¹, Marc Dalmau², Philippe Roose², Nacéra Ghoulmi Zine¹

¹ University of Annaba

University of Badji Mokhtar - B.P.12, 23000-Annaba, Algeria

² LIUPPA – IUT of Bayonne

LIUPPA, I.U.T. of Bayonne / Pays Basque - 2 Allée du Parc de Montaury, 64600-Anglet, France
{m.derdour, ghoulmi}@yahoo.fr, {roose, dalmau}@iutbayonne.univ-pau.fr

Abstract: Pervasive systems are designed to make communication possible anytime, anyhow and anywhere. These systems must be used in different contexts depending on the environment of the user, his profile and his device. The mutualization of the means of communication and the tendency towards all multimedia caused a major problem of heterogeneity of the multimedia flows exchanged between the components of such a system. In this paper we propose a typing of connectors proposed in the MMSA approach (*Metamodel for Multimedia Software Architecture*), thus an implementation of the various concepts proposed by this approach, which aims is to facilitate communication between heterogeneous components. A case study of the multimedia messaging service (MMS) is presented at the end of this paper to highlight the proposal.

Keywords: component, software architecture, multimedia.

1. Introduction

The development of software components based multimedia application requires answering to two main questions: How to design the application so that it adapts itself to the behavior of components? How to design a system to ensure adaptation to dynamic context at runtime? The first question requires a higher reflection in the development process of the applications. The problem of heterogeneity caused by the use of multimedia (*text, image, sound, and video*) must be regulated initially at the internal level, between components of the application. In such a context, the applications obtained by the assembly of components must adapt itself automatically to this evolution. We need a correct specification of components (*complete, static and homogeneous*) which is rather difficult to realize.

The heterogeneity caused by the use of multimedia data must be regulated with external entities of the components of the application. This separation makes it possible to satisfy the principle of separation of the functional and nonfunctional concerns. The adaptation of data flows exchanged between components of an application is assigned to the connectors that are executed anywhere, according to the capacity of the components.

MMSA [1] approach (*Metamodel for Multimedia Software Architecture*) solved the problem of heterogeneity by

proposing connectors of adaptation to manage the problem of heterogeneity of data flow (*text, image, sound and video*). The objective of this paper is firstly to propose adaptation connector and a classification of the adaptation connectors by categories and by adaptation type. This allows easy selection of connectors, and ensures the replacement of the connector in case of unavailability. Next, we provide an implementation of the concepts defined in MMSA with the Java programming language to solve the problem of MMS (*Multimedia Messaging Service*).

In this paper we detail the concept of connector proposed in MMSA, and we propose a description of these adaptation connector allowing a classification of the adaptation services according to the type of media (*image, its, text, video*), and according to the format of each type. Then, we propose an implementation of the MMSA concepts, and we present an example of the MMS.

2. Related works

An ADL (Architecture Description Language) can analyze and verify very early in the development cycle a properties that the future system will have to be able to satisfy, particularly the properties of homogeneity and compatibility of components handling various media. Indeed, current applications such as embedded systems include the notion of media as an important feature of their behavior [15, 16].

Most existing ADL such as SPT-UML [17], MARTE [24], Fractal [18], SCA [21], Kmelia [19] and AADL [20] does not take into account the adaptation and the properties related to multimedia flow when designing software. Some deal with the problem of heterogeneity by changing the configuration settings (*adding, removing or replacing components*) [22] or by metamodel that checks the adequacy of the service to the context and research the adaptation strategy [23]. At a dynamic level, several recent research projects propose a multimedia adaptation architectures such as the one based on wrapper proposed by Metso [8], MAPS [9], M21 [10], BAIT [11], DCAF [12], NAC [13] and PAAM [14], based on an improved P2P model.

The ADL can be classified in three different categories:

ADL without connectors, ADL with a preset set of connectors, and ADL with explicit types of connectors. In the last case, the ADL provides connectors as first order elements of the language such as: Wright [2, 6], ACME C2 [7], xADL [5], AADL [4], etc. All these languages try to improve the reusability of the components and the connectors by separating the calculation and the coordination. In our approach, we choose the explicit category of connector. Thus, MMSA presents a generic and explicit type of connector that the system can specialize it according to the architecture and the components needs. The originality of MMSA connector comes from the function which it provides. It ensures the adaptation of the data flows according to the characteristics of the destination component. The architecture described by MMSA allows the detection of heterogeneities between the application components.

A simple component language [26] proposes a comparison of the principal characteristics of the components languages: component, interface, port, service and connector. The main objective of this work is to take into consideration the unforeseen connection of the developed components in an independent way. As a solution, it proposes the production of reusable and configurable connectors through the association of a particular service to the provided ports which will be used in the absence of the requested service at port level. A drawback of this work is the absence of the integration mechanisms of the new communication services which ensures the evolution of architecture towards new needs; it also does not provide efficient technics for checking the quality of architectures and the provided services.

C3 (*Component Connector Configuration*) presented by Amirat and Oussalah [25] is an approach based on software architectures. It makes it possible to describe a view of logical architecture in order to automatically generate physical architecture for all the application instances. The idea is based on the refinement and the traceability of the architectural elements. The software architecture is described according to the first three levels of modeling defined by the OMG (*Object Management Group*). Consequently, connectors proposed do not ensure the connection of heterogeneous components and do not take into account the semantics of configurations and the links between components.

3. MMSA approach

MMSA is software architecture metamodel for multimedia applications incorporating properties multimedia data flows. The adaptation of data flows is shifted onto connectors called adaptation connectors. The latter integrates the services of adaptation necessary as well as qualitative extensions of these services in order to offer a measurement of QoS reflecting the evolution of the data flow according to the adaptations.

MMSA proposes an extension of the component manifest adding information on provided and required flows. This information allows a better assembly of the component through a detection of incompatibilities of exchanged flows that affects the component interoperability when exchanging information multimedia.

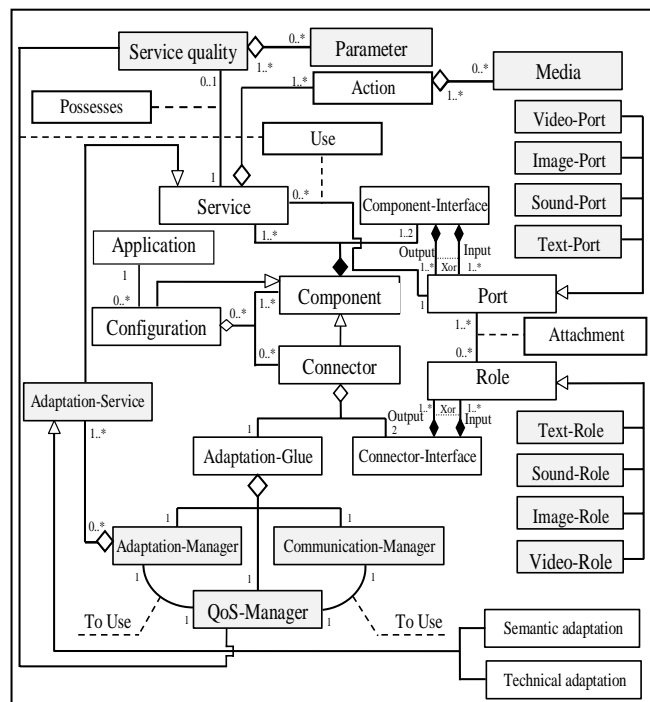


Figure 1. Class diagram of MMSA

The solution proposed by MMSA is based on the adaptation of exchanged flows between heterogeneous components, it is an adaptation provided by the connectors when communicating. This solution avoids having to find compatible components in terms of flows, and to avoid reconfiguration of applications and reassembly of components that can create problems of inconsistency. There are several techniques and adaptation services that depend on the type or the format of media. The various types of adaptation are: type adaptation (*transmoding*), format adaptation (*transcoding*), and management of parameters of adaptation services that represent the different quality produced by these services (*transformation*).

MMSA connector is defined by two interfaces "Input" and "Output" and a glue represented by three managers: communication, adaptation and quality of service (figure 1). They manage the data transfer between components and can make adaptation on the fly. An interface of a connector is composed of a set of roles. Each role is used as connection point between connector and component. Thus two components can be linked by a connector, so that two connectors can be linked together to create complex adaptations.

4. Typing of MMSA connector

Allowing a heterogeneous component to interact with each other is a significant task. The adaptation is considered as a nonfunctional task. It must be ensured by another element. The connector provides the nonfunctional aspects (*communication, adaptability, security, etc.*) that component needs. The role of adaptation connector is to receive data, to adapt them according to the directives of the QoS manager and to route it to the correct component/connector.

Compared with [2], the connectors that we propose can be

simple or composite and can even ensure services. These connectors do not only makes communications links but also the adaptation of data exchanged between components. In our approach, the connector constitutes the entity of communication and adaptation, i.e. it is able to transfer the multimedia data between the various components while ensuring the adaptation of the latter.

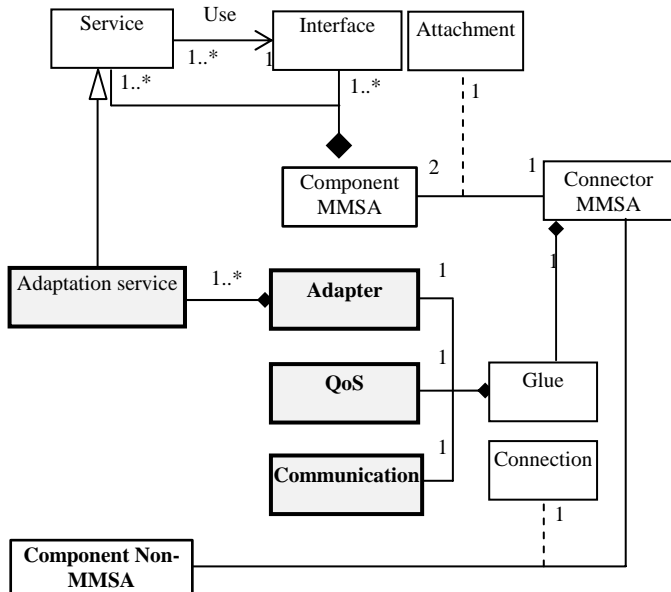


Figure 2. Component, service and connector relationship

The figure 2 shows that a service is offered to satisfy a need: it may be offered by a component (functional need), a connector (non functional need) or as an adaptation service (interaction need). An adaptation connector composed of a set of adaptation services and may be linked to other components not MMSA (e.g. web services). It is a special connector between two heterogeneous components.

A connector of adaptation is used to satisfy a non functional need for a component. This need is detected during the assembly of the components. It acts of an incompatibility of the data flows exchanged between the components of the same configuration, and this incompatibility is described as semantic heterogeneity.

There are several types of adaptation connectors (Figure 3) depending on the types and on the data formats. An adaptation connector consists in glue and two interfaces, one for inputs and the other for output. Each interface contains

roles; the type of the roles depends on the service of adaptation. For example, a connector of transformation of image format contains two roles of image type, for the input and for the output. The adaptation connectors have the same structure for the glue which is composed of three components (*communication, adaptation, QoS*). The only difference deals with the type of roles (*image, text, sound, video*).

We propose a graphical notation of the ports of multimedia interfaces allowing to visually identifying the type of connector.

Type	input	Output	Format
Text	□	┆	DOC, DOCX, ODT
Image	◀	▶	JPEG, BMP, PNG
Sound	▷	◁	WAVE, RM, MP3
Video	▷	◁	MPEG, AVI, MP4

TABLE I. PORT OF MULTIMEDIA INTERFACE

All connectors, including those assigned to communication, have the same internal structure. The only difference between connectors is in term of roles that allow the connection with components that have the same ports (no need for adaptation). This distinction offers two advantages:

1. During the execution one can ask the communication connector to ensure a task of adaptation to satisfy a new need related to a new execution context.

2. A quality management service ensures the adaptation according to the needs and the context.

For the first point, let us take the example of a user who travels in a train and who wants to watch a football match. For that it uses its cell phone connected directly to a component of acquisition and diffusion via a connector. The connector ensures connection and the communication between the two components (*see figure 4/1*). After half an hour the user has noticed a problem of energy and thinks that the battery will not hold until the end of the match. He decided to keep only the sound in order to listen to the match until the end. The ideal solution is to install a service adaptation at the connector to extract audio from video, and send the sound to the user (*see figure 4/2*).

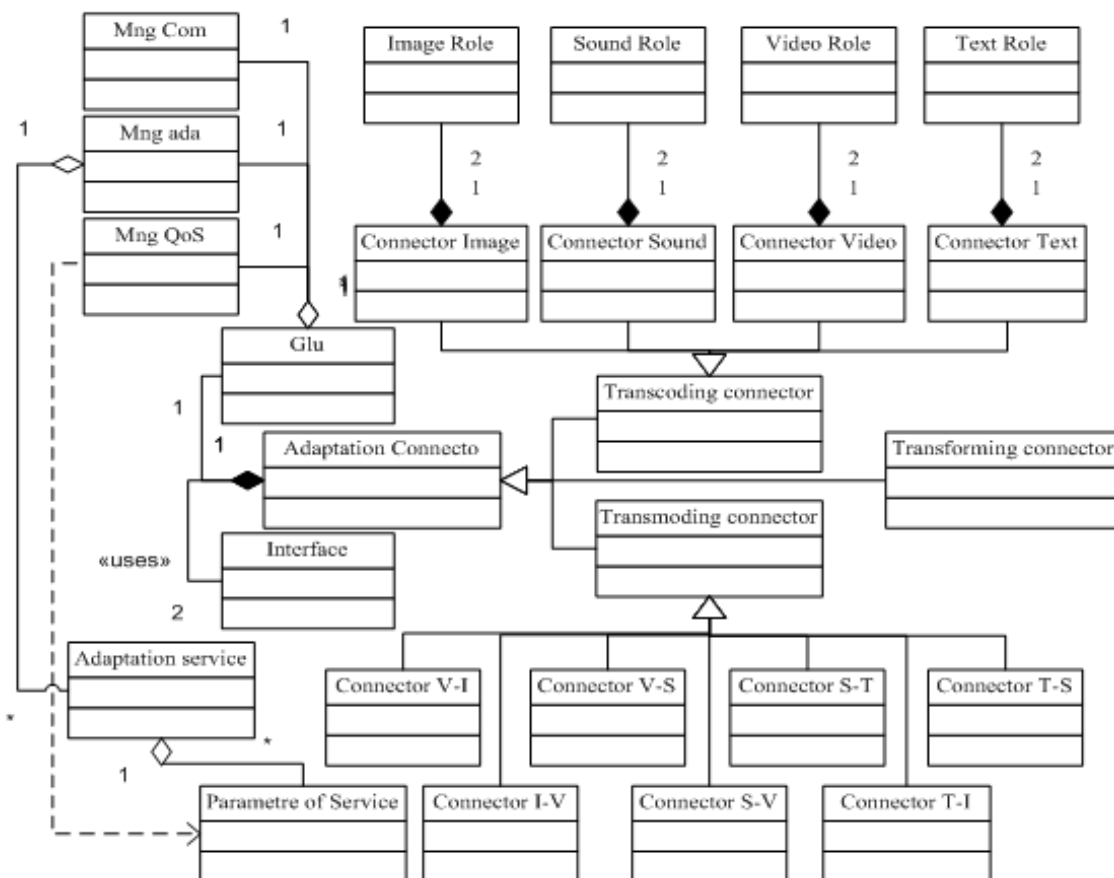


Figure 3. Presentation of the types of connectors

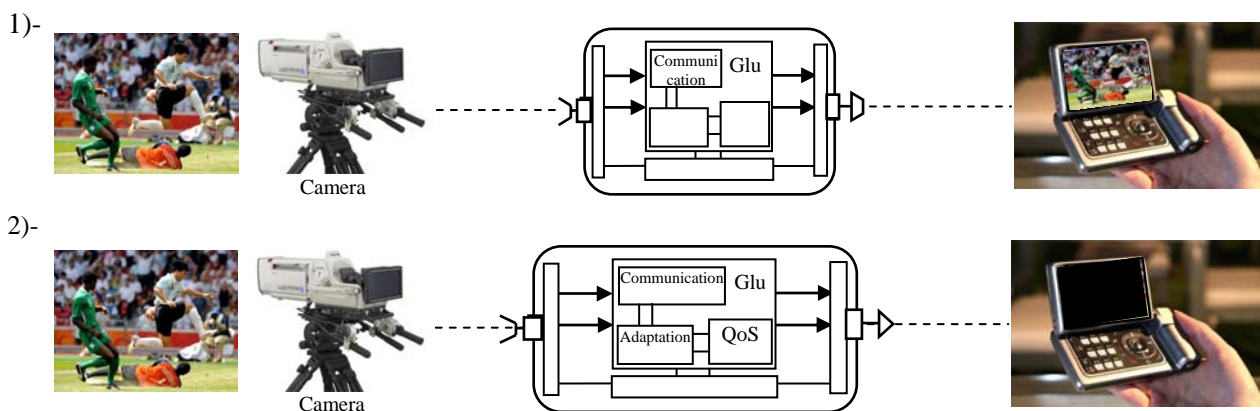


Figure 4. Changing the service of adaptation connector at runtime

Regarding the 2nd point, the following table shows the textual representation of a connector without adaptation service and a connector with adaptation service:

Communication connector
<pre> Class Adaptation-connector { Properties {flow =} Service {Connection} Glue { //simple case of a glue Communication {Synchronous} Adaptation service {} QoS {} Interface { Required-Roles {category A} Provide-Roles {category A}} } </pre>

TABLE II. COMMUNICATION CONNECTOR

daptation connector
<pre> Class Adaptation-connector { Properties {flow = proprieties of data} Service {Connection, adaptation} Glue { //adaptation glue Communication {Synchronous} Adaptation service {service of adaptation} QoS {parameters of adaptation}} Interface { Required-Roles{category A or category B} Provide-Roles{category A or category B} } </pre>

TABLE III. ADAPTATION CONNECTOR

4.1 Role (Provided and Required)

There exist two categories of adaptation connectors:

Connector of format: This category includes connectors that have the same type of role for the input and the output. The

connectors in this category are grouped in the following table:


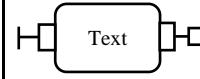
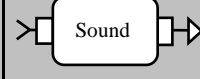
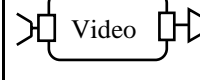
Format adaptation Connector	Chart	Class of connector
Image-connector: this connector is responsible for the adaptations of transcoding between media of image type.		<pre> Class Image-connector { Service {Connection, adaptation} Glue { Communication service {} Adaptation service {} QoS {} Interface { Required {Input-Role-Image} Provide { Output-Role-Image }} } </pre>
Text-Connector: this connector is responsible for the adaptations of transcoding between media of text type.		<pre> Class Text-connector { Service {Connection, adaptation} Glue { Communication service {} Adaptation service {} QoS {} Interface { Required {Input-Role-Text} Provide { Output-Role-Text}} } </pre>
Sound-connector: this connector is responsible for the adaptations of transcoding between media of sound type.		<pre> Class Sound-connector { Service {Connection, adaptation} Glue { Communication service {} Adaptation service {} QoS {} Interface { Required {Input-Role-Sound} Provide { Output-Role-Sound}} } </pre>
Video-connector: this connector is responsible for the adaptations of transcoding between media of video type.		<pre> Class Video-connector { Service {Connection, adaptation} Glue { Communication service {} Adaptation service {} QoS {} Interface { Required {Input-Role-Video} Provide { Output-Role-Video}} } </pre>

TABLE IV. FORMAT ADAPTATION CONNECTOR

Connector of type: This category includes connectors that do not have the same type of role for the output and input.

The following table shows some examples of connectors in this category (the list is not exhaustive connectors).

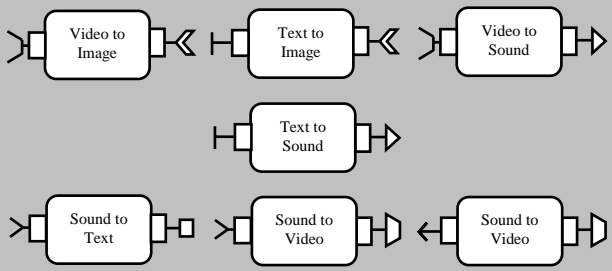
Adaptation Connector of type	Chart
Connectors to ensure semantic adaptation between different media types. This type of adaptation called transmoding.	

TABLE V. TYPE ADAPTATION CONNECTOR

4.2 Adaptation services

The connector ensures communication between two components (*including heterogeneous ones*) from the services provided by components or service providers, according to the quality required by the QoS manager.

The adaptation service participates in the realization of an adaptation of data exchanged between components (Figure 5). The representation of a component by a set of services makes it possible to use these services for adaptation tasks. The mechanism of this use is similar to using Web Services by considering components such as local service providers.

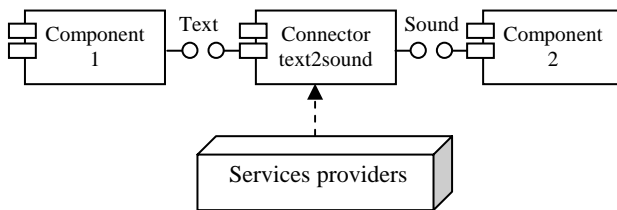


Figure 5. Connector-Provider Relationship

5. Implementation of MMSA concepts

After the specification of adaptation connectors and the selection of the adequate configuration, the purpose of this section is to give implementation directives of the connectors (*service, interface and role*). The execution platform instantiates the components assembled to produce the required application.

To implement the elements of our MMSA approach, we selected the Java language that provides all necessary mechanisms to represent the concepts described in the metamodel.

- An MMSA component is represented by a Java class;
- An MMSA connector is a class composed of three sub-classes;
- A service is a method in a class;
- An interface is represented by a class client or server, that uses a socket client or server to receive or send data (see example);
- For each media type is assigned a range of numbers (eg. [8000 to 8099] for image), and for each media format a port is identified by an integer. So, the concepts of Port and Role in MMSA are replaced by the concept of Port in Java, the types and colors are replaced using the port number;
- To send or receive data, we used the methods *Write ()* and *Read ()* of *OutputStreamWriter* and *InputStreamReader*.

Example:

```

Required interface class of the text
import java.io.*;
import java.net.*;
public class Serveur {

```

```

static final int port = 8080;
public static void main(String[] args) throws Exception {
    ServerSocket s = new ServerSocket(port);
    Socket soc = s.accept();
    // Un BufferedReader permet de lire par ligne.
    BufferedReader plec = new BufferedReader(
        new InputStreamReader(soc.getInputStream())
    );
    PrintWriter pred = new PrintWriter(
        new BufferedWriter(
            new OutputStreamWriter(soc.getOutputStream()),
            true);
    while (true) {
        String str = plec.readLine();
        if (str.equals("END")) break;
        System.out.println("ECHO = " + str);
        pred.println(str);
    }
    plec.close();
    pred.close();
    soc.close();
}
}

```

Provided interface class of the text

```

import java.io.*;
import java.net.*;
public class Client {
    static final int port = 8080;
    public static void main(String[] args) throws Exception {
        Socket socket = new Socket(args[0], port);
        System.out.println("SOCKET = " + socket);
        BufferedReader plec = new BufferedReader(
            new InputStreamReader(socket.getInputStream())
        );
        PrintWriter pred = new PrintWriter(
            new BufferedWriter(
                new OutputStreamWriter(socket.getOutputStream()),
                true);
        String str = "bonjour";
        for (int i = 0; i < 10; i++) {
            pred.println(str);
            str = plec.readLine();
        }
        System.out.println("END");
        pred.println("END");
        plec.close();
        pred.close();
        socket.close();
    }
}

```

The use of port numbers ensures typing of adaptation connectors depending on the type of data appropriate (sound, image, text and video).

6. Case Study: MMS (Multimedia Messaging Service)

Although mobile phone users can create and send their own MMS messages, their greatest use is made by companies that send messages to subscribers, customers or to answer questions. For example, a company could send its MMS route map to help visitors finding their office. Other possible applications include weather forecasting, news and sports news.

MMS service is responsible for the management of shipments of multimedia messages from one device to another. However, the message received by the receiver is not always compatible with the formats that it accepts. When this problem occurs, a message *"mismatch message"* is displayed, so that the receiver cannot read this message.

MMS is a standard in mobile messaging. Such as SMS (*Short Messaging Service*), MMS is a way to send a message from a mobile device to another. The difference is that MMS can include not only "text", but also: sound, images and video. It is also possible to send MMS messages from mobile phone to an e-mail.

To solve the incompatibility problem, we must adapt to the MMS service happens to make adaptations to the message sent according to the characteristics of the device and user. The best way is to have an independent service providing adaptation and making the appropriate message to the MMS that provides for sending messages to the receiver.

6.1 Heterogeneity problem and MMS

MMS is actually composed of multimedia files. These files can be downloaded or digital images taken directly from the camera included in the phone, sound recordings or video stream.

The objective is to ensure that MMS sent from a manufacturer specific mobile phone can be entirely displayed on another mobile phone from another manufacturer. The aim is avoiding the problem of unsupported file formats (*there is many multimedia file format such as JPEG, Bitmap, PostScript, MIDI, Real Media, MP3, QuickTime, etc.*). Another problem is the conversion of files, which is free. If a mobile phone has a large screen with a large color capability, the media clip produced by this phone and posted on another one with a small screen imply that some colors may not be correctly displayed. Moreover, some phones will not interpret the code and use a SMIL presentation preprogrammed pattern and timing that may not correspond at all.

To highlight the problem of heterogeneity between portable devices, the following table summarizes the characteristics of some devices.

Technical feature	Nokia 2610	Samsung SGH-X640	Sony Ericsson K320
Size	104 x 43 x 18 mm	87.4 x 47 x 23 mm	101 x 44 x 18 mm
Type	CSTN, 65K colors	UFB, 65K colors	UBC, 65K colors
Image quality	128 x 128 pixels	128 x 160 pixels	128 x 160 pixels, 1.8 inches
Memory Card	No	No	No
WLAN	No	No	No
Bluetooth	No	No	Yes
Infrared	No	No	Yes
USB port	No	Yes	Yes
Image format supported	GIF, JPEG, PNG, BMP	BMP, GIF, JPEG, PNG, X-NP-WPNG	GIF, JPEG, WBMP, BMP, PNG, VND.WAP, WBMP, CVG
Video camera	No	No	Available
Video Format supported	No	No	MPEG, mp4, 3gpp, mpeg4, mp4v-es
Audio format supported	Midi, mid, mp3, x-mid, amr, amr-wb, mpeg, x-amr	Melody, midi	Amr, rhz, midi, x-midi, sp-midi, midi melody, mpeg, mpeg3, mp3, wav, 3gpp, mp4,

TABLE VI. CHARACTERISTICS OF SOME DEVICES

6.2 Configuring MMS with MMSA

MMS extends the capabilities of SMS, which are limited to 160 characters, and notably to send image, audio and video. The MMS includes all the necessary mechanisms for receiving, sending and tracking of multimedia messages exchanged between mobile devices mobile. The best solution is to have independent services that provide accommodation and return the appropriate messages MMS for sending the message to the receiver.

MMSA adaptation connector is placed between the two components to transmit and receive multimedia messages. The representation of components and connectors are made according to the models proposed by MMSA.

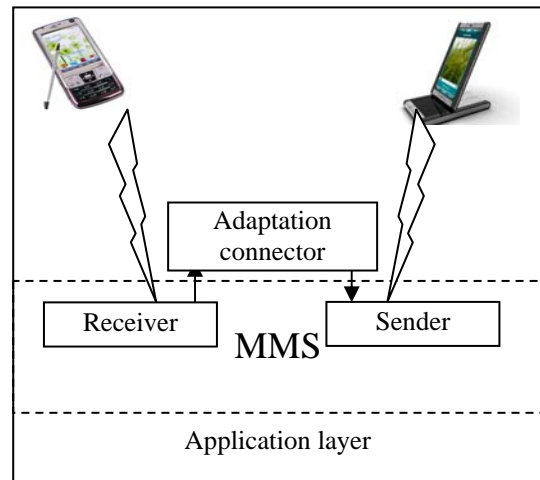


Figure 6. Architecture of MMS application

To make it suitable to the needs of adaptation, the MMS must be enriched with other components to improve the quality of its services and make possible the exchange of multimedia messages between heterogeneous devices.

Figure 6 shows a good application layer. The MMS is composed of several components, among them the emission component and the receiving component. The MMSA approach requires the presence of adaptation connector between two multimedia components, which is why we put an adaptation connector between the two components. This connector provides the adaptation of multimedia flow in case of need.

6.3 System modeling in UML

To describe the context in which the MMS will be used, we use UML diagrams. The architectural elements identified in our system are:

- **The sender and receiver:** it sends or receives the adapted messages to its characteristics.
- **Connector adaptation:** it provides communication between components by adapting messages. It is composed of an adaptation manager, QoS manager and communication manager.
- **Manager of context:** it is responsible for the necessary updates in the database of profiles (Adding new profiles, modifying profiles existing or deletion).

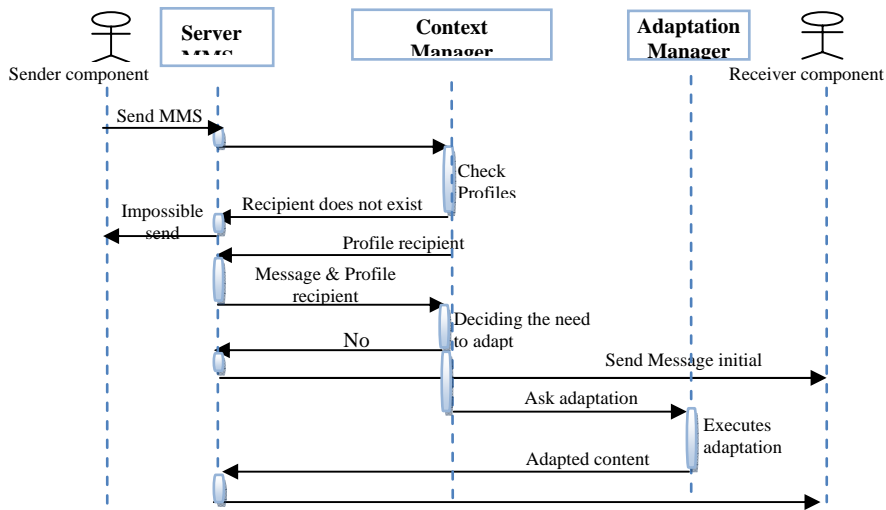


Figure 7. Sequence diagram

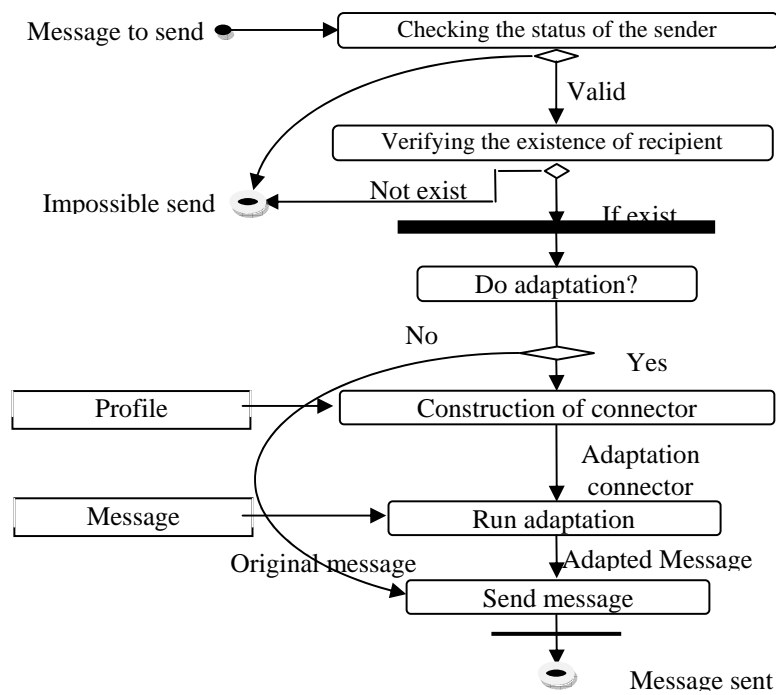


Figure 8. The activity diagram

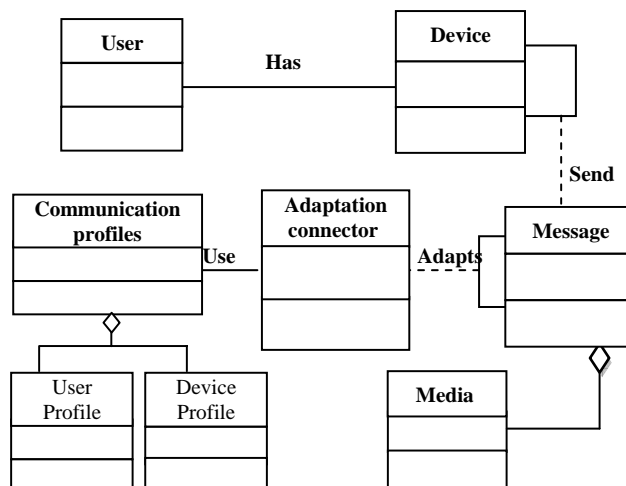


Figure 9. The class diagram

The UML sequence diagram models the exchanged flows in our system in a visual way, allowing us both to document and validate our logic. The sequence diagram focuses on the identification of system behavior. In addition to the device sender and receiver, the sequence diagram (Figure 7) focused on the MMS service, the manager of adaptation and the context manager to properly mount the cooperation between different services.

While sequence diagrams model the control flow between objects, the activity diagram (Figure 8) is used to model the control flow between activities. The activity diagram shows the sequence of events for the use case: Send MMS.

The class diagram (figure 9) gives a static view of the application. It described the objects and relations between them.

Modeling of MMS with UML allows showing need for this type of application to carry out adaptation policies to solve the problem of heterogeneity, and the satisfaction of the mechanisms proposed in previous sections to ensure interoperability of components of application.

6.4 Example of image adaptation

A user who wants to send a multimedia message from his mobile phone number of the phone selects the recipient. Then it sends the message to be received by the receiving component. The latter sends the destination mobile number to the context manager, which in the model apartment is MMSA, verify the need for adaptation. If there are any needs for adaptation, it sends a request to have the appropriate connector to the requested accommodation. The adaptation connector adapts and sends the message to the sending component to send to the recipient. The application provides the following services.

6.4.1 Resize

In the case where the screen dimensions of destination are different to that of the sender, we made a call to a sizing service. The adaptation connector polls context manager to get the screen dimensions of the recipient, and applies these dimensions on the image sent by the sender; the following figure illustrates the application of this service.

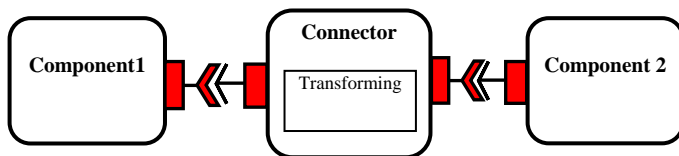


Figure 10. MMSA configuration for a resizing connector

```

Resize service
public void Resize (OutputStream, InputStream, int lar, int haut)
{BufferedImage initial;
 String format="jpg";
 try {
  initial= ImageIO.read( source8);
  Final= new BufferedImage(lar,haut, BufferedImage.TYPE_INT_RGB);
  Graphics2D dessin = Final.createGraphics();
  dessin.drawImage( initial,0,0,lar,haut,null);
  ImageIO.write(Final,format,destination8);
 }
 catch (IOException zz) {
  System.out.println("erreur");
 }
 }
    
```

Figure 11 shows the execution of application in the resizing case.

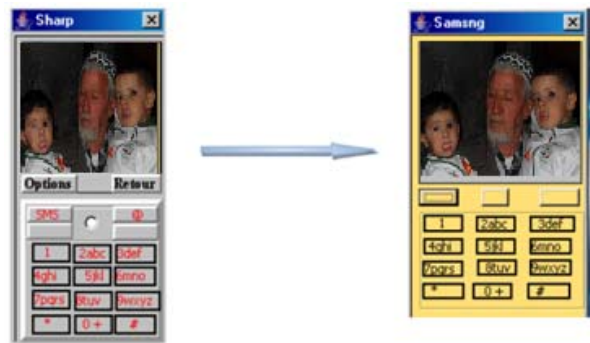


Figure 11. Execution of resizing Connector of image

6.4.2 Format change

In case the recipient does not support the image format sent by the sender, for example the image is .bmp and the recipient mobile media as .jpeg, it is necessary to have a format conversion service to compress the image in JPEG format.

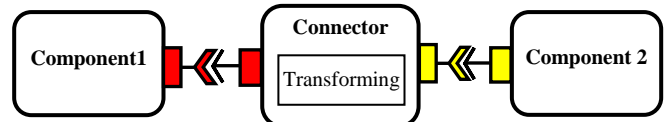


Figure 12. MMSA configuration of transcoding connector BMP2JPEG

```

Transcoding service BMP2JPEG
public void bmp_jpg(BufferedImage1 :InputStream, BufferedImage2:
 OutputStream,String compressionType) {
 try{
  initial=ImageIO.read(source5);

  Final=new                BufferedImage1(initial.getWidth()
 ,initial.getHeight(),BufferedImage1.TYPE_INT_RGB);
  Graphics2D g = (Graphics2D) Final.getGraphics();
  g.drawImage(initial,0, 0, initial.getWidth(), initial.getHeight(),null);
  ImageIO.write(Final,compressionType, destination5);
 }
 catch (IOException e)
 { e.printStackTrace();}
 }
    
```

Figure 13 shows the execution of application in the case of transcoding to JPEG BMP



Figure 13. Format adaptation Bmp2Jpeg (transcoding)

7. CONCLUSION

Complete adaptation architecture must therefore cover various aspects such as the consideration of all entities of the environment, the definition of how these entities can interact in adaptation profile and the adaptations techniques that allow transformation of content of its origin state to a new state more consistent to characteristics of the target component. The advantage of such representation for modeling, adaptation and transformation is obvious. It is based on modeling languages built on strong theoretical foundations, and powerful enough to describe and characterize the properties of the media.

The adaptation connector are the most interesting concept in this proposal, their use allows separation of nonfunctional concerns such as adaptation, communication and security of the functional concerns ensuring by components. Besides the communication, the connectors can ensure of other tasks, the adaptation is a nonfunctional concern, since it does not form part of the system, but it proposes solutions in the case of heterogeneity of the components. Typing connectors has a double interest. It allows classification of connectors according to data type and format, which facilitates the research and integration of connectors and, it allows to group connectors that have the same task in order to facilitate the replacement of a connector by another if needed.

The transition from modeling to the implementation is often ensured by transformation models, It represents a significant task in software engineering. In this paper we have presented the main directions of implementation of MMSA concepts in Java language, which ensures a passage of the models generated by MMSA to the realization of the components and the configurations.

As we saw, the multimedia environment is wide, and especially for image formats diversity.. So, we tried to implement the essential in the field of image processing, in order to better determine the problem of heterogeneity of the mobile devices.

In this article we presented a module of adaptation for the phone networks in order to allow to limited devices the exchange multimedia data. The architecture is based on the client/proxy/server model. The objective of the adaptation module is to improve the QoS of the exchanged the multimedia information on heterogeneous phones.

References

[1] Makhoulf Derdour, Philippe Roose, Marc Dalmau, Nacéra Ghoulmi Zine, Adel Alti - An adaptation

approach for component-based software architecture - 34th Annual IEEE Computer Software and Application Conference - COMPSAC 2010 - pp. 179-187, ISBN: 0730-3157/10 - DOI 10.1109/COMPSAC.2010.24, Seoul - 2010.

- [2] R. Allen, A Formal Approach to software Architecture, PhD thesis, School of Computer Science Carnegie Mellon University, Pittsburgh, Mai 1997.
- [3] Garlan D., Monroe R.T., Wile D., « Acme: Architectural Description of Component-Based Systems », Foundations of Component-Based Systems, Leavens G.T. and Sitaraman M. (eds), Cambridge University Press, 2000, pp. 47-68.
- [4] Allen, R., Vestal, S., Lewis, B., Cornhill, D., "Using an architecture description language for quantitative analysis of real-time systems", In Proceedings of the Third International Workshop on Software and Performance, ACM Press, Italy, pages 203–210, 2002.
- [5] Dashofy, E., Hoek, A.v.d., Taylor, R.N., "A comprehensive approach for the development of XML-based software architecture description languages", Transactions on Software Engineering Methodology (TOSEM), volume 14, issue 2, pages 199–245, 2005.
- [6] Medvidovic, N., Rosenblum, D. S., and Taylor, R. N. "A Language and Environment for Architecture-Based Software Development and Evolution", In 21st International Conference on Software Engineering (ICSE'99), Los Angeles, May 1999.
- [7] Garlan, D., Monroe, R.T., and Wile, D. "Acme: Architectural Description Component-Based Systems, Foundations of Component-Based Systems". Cambridge University Press, pages 47-68, 2000.
- [8] Metso M & Sauvola, J. The media wrapper in the adaptation of multimedia content for mobile environments. Proc. SPIE Vol. 4209, Multimedia Systems and Applications III, Boston, MA, 132 – 139. 2001
- [9] Lienhart R., Kozintsev I., Chen Y-K., Holliman M., Yeung M., Zaccarin A., Puri R., « Challenges in Distributed Video Management and Delivery ». chapitre 38 de « Handbook of Video Databases : Design and Applications ». CRC Press, Boca Raton, Floride , p. 961-990, 2003.
- [10] Vetro A., « MPEG-21 Digital Item Adaptation: Enabling Universal Multimedia access », IEEE Multimedia, janvier-mars 2004, vol. 11, n° 1, p. 84-87.
- [11] Lapayre J-C., Renard F., « Appat : a New Platform to Perform Global Adaptation », Actes de la 1st IEEE International Conférence on Distributed Frameworks for Multimedia Applications (DFMA'2005), Besançon, 6-9 février, 2005, p. 351-358
- [12] Berhe, G.Brunie, L.Pierson, Distributed content adaptation for pervasive systems, Information Technology: Coding and Computing, ITCC 2005. page 234- 241 Vol. 2
- [13] N. Layaïda, T. Lemlouma, V. Quint , «NAC : une architecture pour l'adaptation multimédia sur le Web», Technique et Science Informatiques, RSTI-TSI -- 24/2005, num. 7, pp. 789-813.
- [14] Zakia Kazi-Aoul, Isabelle Demeure et Jean Claude Moissinac. "Towards a Peer-to-peer Architecture for the provision of Adaptable Multimedia Composed

- Documents". DFMA (Distributed Frame for Multimedia Applications), Penang, Malaysia on 14th to 17th May, 2006.
- [15] Avizienis A., Laprie J.-C., Randell B., Landwehr C. « Basic Concepts and Taxonomy of Dependable and Secure Computing ». IEEE Transactions on Dependable and Secure Computing 1(1). January-March 2004. pp. 11-33.
- [16] Balsamo S., Bernado M., Simeoni M. «Performance Evaluation at the Architecture Level Formal Methods for Software Architectures». Lecture Notes in Computer Science 2804. Springer. Berlin, Germany. pp. 207-258. 2003.
- [17] Graf S. , Ober I. « How useful is the UML realtime profile SPT without semantics? » In SIVOES 2004, associated with RTAS 2004, Toronto Canada, April 2004.
- [18] Bruneton E, Coupaye T, Leclercq M, Quéma V, Stefani J-B. An open component model and its support in Java. In: Crnkovic I, Stafford JA, Schmidt HW, Wallnau KC, editors. CBSE. Lecture notes in computer science, vol. 3054. Berlin: Springer; 2004. p. 7–22.
- [19] Attiogbé, C. André, P. and Messabihi, M.: Correction d'assemblages de composants impliquant des interfaces paramétrées. In 3ième Conférence Francophone sur les Architectures Logicielles. Hermès, Lavoisier. (2009).
- [20] Society of Automotive Engineers (SAE). « Architecture Analysis & Design Language (AADL) ». SAE Standards no AS5506, November 2008.
- [21] Barros, T., A. Cansado, E. Madelaine, et M. Rivera. Model-checking distributed components: The vercors platform. Electron. Notes Theor. Comput. Sci. 182, 3–16. 2007.
- [22] Marcel Cremene , Michel Riveill , Christian Martel, Calin Loghin, Costin Miron . Adaptation dynamique de services. DECOR'04, 1ère Conférence Francophone sur le Déploiement et (Re) Configuration de Logiciels. Grenoble, France. October 2004.
- [23] Marcel Cremene, Michel Riveill, Christian Martel. "Autonomic Adaptation based on Service-Context Adequacy Determination" in Electronic Notes in Theoretical Computer Science (ENTCS), vol. 189, pages 35-50, Elsevier, jul 2007.
- [24] OMG, OMG Policies and Procedures, Version 2.7, pp/08-06-01, <http://www.omg.org/cgi-bin/doc?pp>
- [25] Amirat Abdelkrim and Oussalah Mourad: "First-Class Connectors to Support Systematic Construction of Hierarchical Software Architecture", in Journal of Object Technology, vol. 8, no. 7, November-December 2009, pp. 107-130
- [26] Luc Fabresse, Christophe Dony, and Marianne Huchard. Foundations of a Simple and Unified Component-Oriented Language. Journal of Computer Languages, Systems & Structures, editor Elsevier, Volume 34/2-3 (July-October 2008), p. 130-149.