

UML-profile for multimedia software architectures

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

▶ **To cite this version:**

Makhlouf Derdour, Nacira Ghoualmi-Zine, Philippe Roose, Marc Dalmau, Adel Alti. UML-profile for multimedia software architectures. International Journal Multimedia Intelligence and Security, 2010, 1 (3), pp.209-231. hal-00544154

HAL Id: hal-00544154
https://hal.science/hal-00544154

Submitted on 8 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.

UML-Profile for Multimedia Software Architectures

M. Derdour*, N. Ghoualmi Zine

University of Annaba
Computing Department
Annaba – Algeria
E-mail: {m.derdour, goualmi}@yahoo.fr
*Corresponding author

P. Roose, M. Dalmau

LIUPPA – IUT of Bayonne, Computing Department, France
E-mail: {roose, dalmau} @ iutbayonne.univ-pau.fr

A. Alti

Université of Setif, Computing Department, Algeria
E-mail: altiadel2002@yahoo.fr

Abstract: Multimedia technology is increasingly being used to create reliable and effective communication environments. However, the design of multimedia applications is currently driven more by intuition than by empirically or theoretically derived design guidelines. In a multimedia application, the software architecture is defined as a set of components manipulating various multimedia data types with specific constraints that we must take into consideration at the architectural design. For instance, the problem of heterogeneity related to the exchanged of multimedia data flows. In the absence of prescriptive architectural design principles, MMSA (Meta-model Multimedia Software Architecture) enables the description of software architectures expressing a multimedia software system as a collection of components which handle various types and formats of multimedia data, and interacts between them via adaptation connectors. This paper proposes a modeling of architectural elements such as: multimedia, application components, communication, etc. and an UML profile for verification and validation of MMSA architectures and detection of heterogeneities between components communicating with multimedia flows.

Keywords: Adaptation; Component; Connector; UML; heterogeneity; Architecture; Multimedia.

Biographical notes:

Makhlouf Derdour is a student in the department of Computer Science at the University of Annaba, Algeria. His research interests include software architecture, multimedia applications, adaptation and self-adaptation of applications, design & modeling by UML.
Philippe Roose is an Assistant Professor in the department of Computer Science at the University of Pau, France. He is responsible of the TCAP project - Video flow transportation on sensor networks for on demand supervision. His research interests include wireless sensors, software architectures for distributed multimedia applications, software components, quality of service, dynamic reconfiguration, COTS, distributed software platform, information system for multimedia applications.

Copyright © 200x Inderscience Enterprises Ltd.

Marc Dalmau is an IEEE member and Assistant Professor in the department of Computer Science at the University of Pau, France. He is a member of the TCAP project. His research interests include wireless sensors, software architectures for distributed multimedia applications, software components, quality of service, dynamic reconfiguration, distributed software platform, information system for multimedia applications.
Nacira Ghoualmi-Zine has state doctorate in computer sciences mention network and she is a lecturer in the department of Computer Science at Badji Mokhtar University-Annaba-Algeria since 1985. She was a head of department in 2000-2002. She is a head of Master and Doctoral Option entitled (Network and computer security). She is a head of project entitled "mobility in network and multimedia adaptation". Her research includes wireless networks, distributed multimedia applications, quality of service, security in protocol, optimization in networks…
Adel Alti is a Ph.D. student at UFAS, university of Sétif , Algeria . The subject of his thesis is mapping architectural description into UML and integrating architectural concepts in MDA. During his work he has published number of articles for international journals and conferences concerning these subjects.

1. Introduction

With recent progress in software and material technologies, the systems multimedia became increasingly sophisticated and complex. Today, the companies require multimedia applications that combine a variety of sources, such as audio, video, text and image, and of the multiparty interactive communications. The multimedia communication needs the services able to face heterogeneity on several levels: the context, the access devices, the communication network, the user, etc. It is necessary to integrate capacities to deal the heterogeneity problem, and to answer the changes of the context caused by the user, the application, the network or the access device. The future multimedia ubiquitous systems must have capacities of adaptation, and thus beings able to modify the system configuration and/or the multimedia contents any time. This requires taking into account the contents presentation and the components interaction of application in the early development phases of application.

Among the software architecture for pervasive applications, there exist component-based architectures that allow the reasoning about complex software systems at an abstract level, i.e. ignoring the details of design and of implantation. Architecture is an abstract and modular description of a system. At this level, the architecture is perceived as a collection of components (in the sense of software entities), a collection of connectors (to describe the interactions between components) and of configurations (assemblies components and connectors). The concerns functional and/or non-functional can concern the components assembled in architectures as well as the assemblies themselves. They cover the structural and the dynamic aspects of applications. The adaptation is one of the concerns that we consider non-functional and serves to ensure the interoperability of heterogeneous components.

The software architectures are commonly categorized in: "Component-Based Software Architecture", described with ADL "Architecture Languages Description" (Clements et al, 2002) (Medvidovic et al, 2000) and "Object-Based Software Architecture" (Khammaci et al, 2005) described using UML (Unified Modeling Language) (Booch et al, 1998) (Jacobson, 1992) (OMG, 2001).

After you define a meta-model MMSA in (Derdour et al, 2010) for multimedia applications, that offering an architectural description of components, and that is capable of detecting the heterogeneity (non-interoperability) between component of architecture

and propose adaptation connectors ensuring such interoperability. We need a language that allows a formal specification of architectural concepts and a tool for verification and validation of software architectures.

Recently, UML has become a standard of specification, visualization, construction and documentation of the software systems. The concepts of UML in its version 2.0 (OMG, 2004) are sufficiently generic to be used in the description of software architectures by providing a rich and a complete documentation and allowing the expression of a non-ambiguous semantics of their concepts. UML 2.0 offers means that are more explicit than version UML 1.4 to represent certain architectural concepts, such as, the components, the interfaces and the ports. Other architectural concepts such as the connectors of adaptation, the multimedia components, the interfaces of multimedia flows, etc. presented in MMSA cannot be directly expressed in UML 2.0. It is therefore necessary to define a profile, which is the subject of this paper.

The objective of this UML profile is to provide a rich and complete documentation and to produce a non-ambiguous semantics of the multimedia concepts allowing developers to express the elements of MMSA (Derdour et al, 2010) taking into consideration the heterogeneity generated by manipulating of several media types (image, sound, video, text). We define here a set of stereotypes to describe a complete and formal specification of multimedia software architectures.

This paper is organized as follows: we present some works related to this one in order to position our contribution compared to state of the art. Then we present the MMSA meta-model. We introduce thereafter the UML profile for MMSA with an illustrative example the application of rules and constraints proposed into profile. Finally, this article ends with a conclusion and prospects.

2. Related works

Modern applications which have software preponderance are more and more developed by ADL-based development processes (Avgeriou and Zdun, 2005). The ADLs allow analysis and verification of properties early in the development cycle that the future system will have to satisfy, in particular the homogeneity and compatibility properties of components manipulating various media. Indeed, the current applications (multimedia, embedded systems, communication systems, etc.) consider the media notion as an important characteristic of their behavior (Avizienis et al, 2004) (Balsamo et al, 2003). Most of existing ADLs such as SPT-UML (Graf and Ober, 2004), MARTE (OMG, 2006), and AADL (SAE, 2008) do not take into account the adaptation and the properties related to multimedia flow during the software construction phase. Some of them, treat the heterogeneity problem by modification of the configuration parameters (addition, withdrawal or replacement of components) (Marcel et al, 2004) or by a meta-model which verifies the adequacy of service regarding its context and research of the adaptation strategy (Marcel et al 2007). In particular, their use does not allow the detection of the incompatibilities caused by the diversity of media during exchanged flows.

The Model-Driven Engineering (MDE) is an approach of software development that puts the model concept (rather than code) in the center of the development cycle. This approach is mainly based on UML and the initiative MDA (Model-Driven Architecture), led by the OMG (Object Management Group). The current of meta-modeling, very vigorous in the 1990, has also produced MOF standard (Meta-Object Facility) and mechanisms defining the specific languages of modeling for specialized fields (DSL). The

duality UML/Profile - DSL/MOF exists at the heart of the standard, since it was expressed by the OMG, in sometimes conflicting positions.

Regarding the advantages and the disadvantages of each approach DSL/Profile, the sterile debates on the technical preferences between these approaches are secondary (Desferay, 2009). It remains a permanent ambiguity, on the definition of a DSL: the concept of DSL does not stop at Profile/Meta-model border, because these two techniques allow defining languages dedicated to certain fields. Considering that UML integrates in its definition the ability to be expanded to target a particular field, UML integrates the concept of DSL which is a necessity. We chose the UML Profile that allows us to profit from UML standard, in terms of learning model, exchange between different workshops, and an equipped support very responded. The profiles can be applied to existing models, and combined between them. Their limit is that it imposes that the new concepts introduced is a natural extension of the UML semantics.

Many works have been realized on the projection of ADL concepts in UML. In (Midvidovic et al, 2002), Medvidovic propose two approaches to express the architectural elements with notations of UML 1.4 language. The first approach uses UML language "such as it is", whereas the second proposes to use extensions such as the stereotypes, the marked values and the constraints in certain of architecture description languages, like C2, Wright and Rapide. In (OMG, 2007) (Belloir, 2008), OMG has presented tow UML profiles. The first one is an UML profile for Marte (OMG, 2007) intended to real-time and embedded systems. The second called SysML (Belloir, 2008), it is an UML 2.0 profile providing the elements of modeling systems that they lacked in UML. In particular, their use does not allow the detection of the incompatibilities caused by the diversity of media during the exchange of flows.

In (Garlan, 2000), the authors have selected UML 1.4 notations to represent architectural elements, holding in counts the advantage and limitations of each notation. They found that aspects of components-based software architecture are not easily representable in UML 1.4. This leads us to say that early versions of UML were not adequate to represent the architectural concepts such as components, connectors, configurations, interfaces (ports and roles) or the architectural styles.

Therefore, UML 2.0 (OMG, 2004) proposed new architectural concepts such as connectors, ports, and structural classifiers and redefined the concept of components which becomes a subclass of UML meta-model class. Moreover, a component has more expressive characters as classes (it can have interfaces and contain other components or classes). In (Ivers et al, 2004), Ivers studied the suitability of new UML 2.0 notation for the projection of the view components and connectors (C & C) of software architecture, especially the architecture description language ACME. Thus, they studied the projection of each concept of ACME language linked to the view C&C (component, connector, ports, and roles) towards UML 2.0. Goulao and Al. (Goulao and Abreu, 2003) consider each concept of the language ACME as a stereotype, and so it does not benefit from the new notations of UML 2.0 language. More recently, Oquendo (Oquendo, 2006) proposed a profile UML 2.0 for the formal ADL ArchWare.

Mauro (Caporuscio and Issarny, 2006) proposed a UML profile to define and analyze software architectures that exploit explicitly the domain properties of B3G (Beyond Third-Generation). This work tries to integrate the various networks available in the B3G application domain and QoS properties defined by (UMTS, 2005) in the profile dually (Inverardi et al, 2005). While exploiting the representation of the connectors by a stereotyped UML component, symbolizing both the functional and nonfunctional properties of connectors.

3. MMSA: Meta-model for Multimedia Software Architecture

Currently, the multimedia data flows must be executed on many platforms (Cell phones, PDA, PC or portables, etc). This diversification of the uses and the supports requires the adaptation of flows to their execution context, sometimes unforeseeable at the stage of the preparation and the design of data. The flow is a main constituent of the functional components, it is often specified like constraint to associate with a functionality of communication involving several components. The constraints of data flows such as the type, the format and the parameters of media must be specified at the architectural level. For that, we consider a new type of component intended to ensure a non-functional concerns that of the adaptation, which one calls the adaptation connector related to the component which provides and/or requires the data multimedia.

We propose a graphical notation of the ports of multimedia interfaces allowing to identify visually the heterogeneity points per media type and to highlight the need for the search of adaptation connectors.

Table 1. PORT OF MULTIMEDIA INTERFACE

Type	Output Input	Format	
Text	DOC	DOCX	ODT
Image	JPEG	BMP	PNG
Sound	WAVE	RM	MP3
Video	MPEG	AVI	MP4
Color	Black	Red	yellow	Blue

The detection of heterogeneity is done automatically by checking of the constraints of forms and colors. For example the port of type "Text" must be linked with only one port of type "Text" having the same multimedia format and so on for the other types.

MMSA is used to describe software architecture as a collection of components (homogeneous and heterogeneous) that interact by intermediate of connector. Components and connectors have the same abstract level and are defined explicitly by the separation of their interfaces and their implementations. MMSA integrates most adaptation mechanisms of multimedia flow by introducing the adaptation connector concept (Derdour et al, 2009). Figure 1 presents the class diagram of MMSA meta-model showing the basic architectural elements that are the components, the connectors and the configurations. These architectural elements are types which can be instantiated to build several architectures. An architectural element can have its own properties (functional and non-functional), its constraints and several implementations and can be composed of several interfaces. Finally, it can be composed of several architectural sub-elements.

In order to respond the insufficiencies of ADL (lack of expression, absence of semantics, etc.) we proposed MMSA (Meta-model for Multimedia Software Architecture) to describe software architecture based on the multimedia components, it is based on the definition of four types of interfaces according to data flow (Image, Sound, Text, Video)

and adaptation strategy of multimedia flow (type, format, property) to three levels to solve the problem of components heterogeneous.

The basic concepts of MMSA software architecture are the same ones as in most of the software architecture, namely: configuration, component and connector. The software architecture model of MMSA is a hybrid model based on the concepts of component-oriented architecture (CBSE) and service-oriented architecture (SOA).

A component is defined by a set of services that interact to fill a role of component and communicate with environment through its required/provided interface. Generally, the connectors define abstractions which encapsulate the mechanisms of: communication, coordination and conversion (type, number, frequency and order of interactions) between the components. A connector is represented by an interface and glue (Goulao and Abreu, 2003) (Maillard et al, 2007). This description considers the connector as a mediator between components, which limits its role in communication. The specification of glue describes the functionality which is expected from a connector. It represents the hidden part of a connector. The glue can be a simple protocol of communication linking the ports, or a complex protocol that uses various operations especially that of: links, conversion of data format, transfer, adaptation, etc. Generally, the glue of connector is the connection type of this connector.

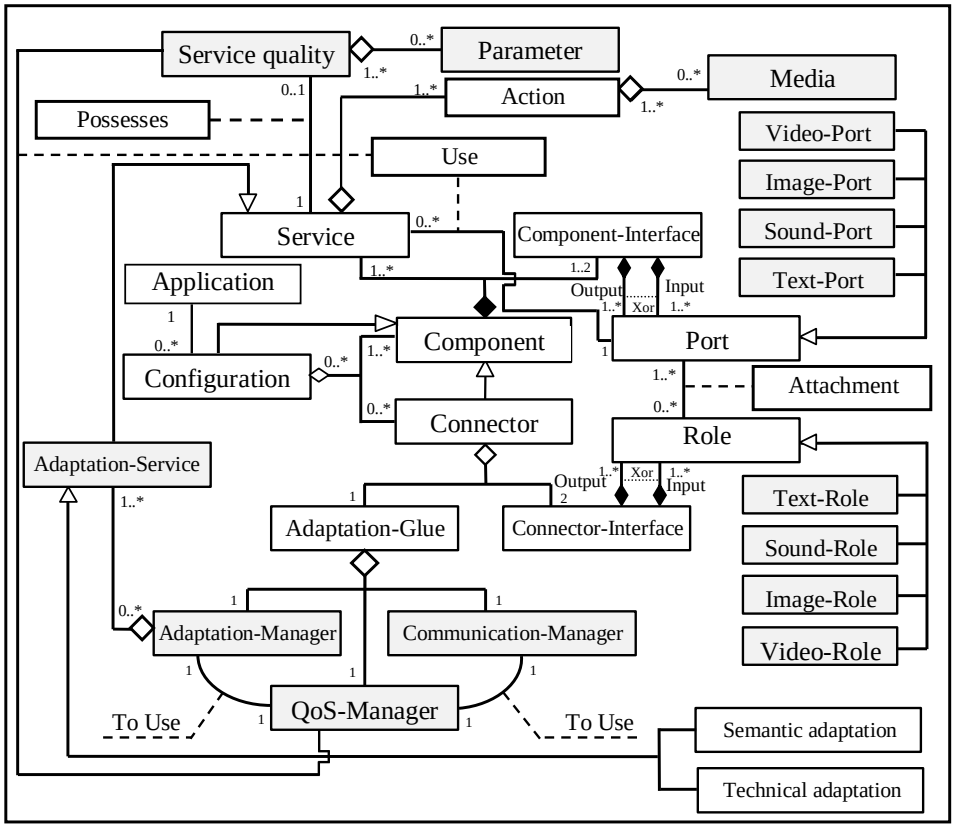


Figure 1. Class diagram of software architecture MMSA

The ADL can be classified in three different categories Amirat (Amirat and oussalah (A), 2009): 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 (Allen and Garlan, 1997) (Medvidovic et al, 1999), ACME C2 (Garlan et al, 2000), xADL (Dashofy et al, 2005), AADL (Allen et al, 2002), etc. All these languages seek 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 present 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.

In MMSA, a connector is a set of services (communication, adaptation, Quality, etc.) ensuring connection between the components, it can ensure the non-functional concerns of components (such as security, data transformation, communication, etc). This allows a possible change of the adaptation services during the execution of the application (dynamic and real time adaptation), and preserves the abstract specification of the component.

4. Definition of UML Profile

The main interest of the MMSA meta-model (Derdour et al, 2010) is to express the architectural multimedia concepts that are not explicitly defined in UML 2.0. In other words, the use of stereotypes, constraints and marked values allowing better specifying and better capturing the concepts of MMSA meta-model (multimedia interface, multimedia component, multimedia connector, adaptation glue, etc.).

In this objective, we will exploit the profile approach that constitutes a key aspect for the validation of the non-functional assembly of heterogeneous components and allow deriving the automatically analyzable models by tools like Eclipse and Rational Software modeler (http://www.ibm.com/developerworks/). Thus, we benefit from MDA approach and UML profiles to separate the architecture and the implementation contexts from a multimedia application. This provides an architecture model that better responds the various constraints of the multimedia component, and offers a support for the systematic adaptation of models, thus of the automatic adaptations according to various contexts for heterogeneous components of conceptual model of application.

The definition of UML 2.0 profile requires the respect of the structural and semantic characteristics of MMSA meta-model (c.f. section 3) in order to ensure the quality of the multimedia components assembly with formal techniques of specification (of multimedia flow and constraints of the components multi-media) and automatic checking of assembly.

To define a UML 2.0 profile for MMSA, we propose three hierarchical levels of abstraction where the basic concepts of MMSA are represented in distinct levels of abstraction. The level of pre-configuration presents a basic model of the application from the component and attachment concepts of MMSA meta-model. The level typing and formatting of the interfaces specify the types and formats of data exchanged between components. It is defined by the concepts of multimedia ports (Text, Image, Video, and Sound). At this level, we detected heterogeneity between components in terms of type and format of data encoding. Finally, the integration level places the adaptation connectors between components according to the heterogeneity type and uses concepts (Connector

and Glue). This hierarchy provides to software architects the opportunity to verify the architecture model with each change in order to ensure its semantic coherence and to detect heterogeneity in terms of data exchanged.

4.1 Pre-configuration level

Context of MMSAComponent. The multimedia component is a basic concept of our meta-model. This concept has no explicit correspondence in UML. Thus, UML profile must include a stereotype to represent it. We call this stereotype «MMSAComponent». It corresponds to the Component meta-class of UML meta-model. The latter is more expressive than UML 2.0 class and provides services from the ports associated to the "provided" or "Required" interfaces. Any component stereotyped "MMSAComponent" must have at least one port stereotyped "MMSAInterface". This constraint can be defined in OCL (Object Constraint Language) (OMG, 2005) as follows:

```
context UML::InfrastructureLibrary::Core::Constructs::Component
inv : self.isStereotyped("MMSAComponent")
implies
  ( self.ownedPort->size()>=2) and
  ( self.ownedOperation->IsEmpty()) and
  (self.ownedPort->forAll(p|p.isStereotyped("ComponentInterface"))) and
 (self.clientDependency.target->forAll(t|t.oclIsKindOf(Interface)))->isEmpty())
```

We propose the following graphical notation:

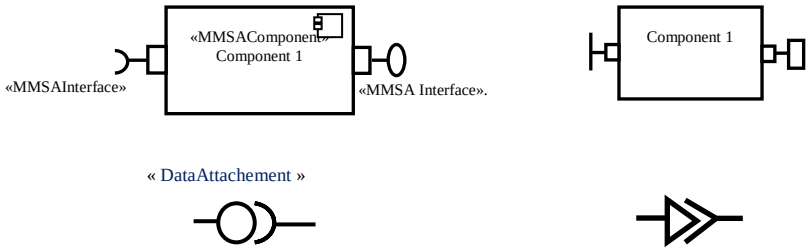


Figure 2. Graphical notation of MMSA component

4.2 Level of typing and formatting of interfaces

In our profile, an UML port has multiple interfaces (provided and required), matches MMSA interfaces. A UML port has multiple interfaces (provided and required), matches MMSA roles or ports.

Context of ComponentInterface. In MMSA, the interface of component is composed of several input/output ports of multimedia flow according to their format. The UML port corresponds to the interface concept that has several ports of provided/required type and support unidirectional and bidirectional communication. The MMSA port "MMSAPort" supports only the unidirectional communication since a port MMSA is directed and can ensure only one required or provided service. Thus, a connector ensures the adaptation of data and, generally, the adaptation service does not function in both directions (ex: transformation text to sound). The interface has only «InputPort» or «OutputPort» stereotyped ports.

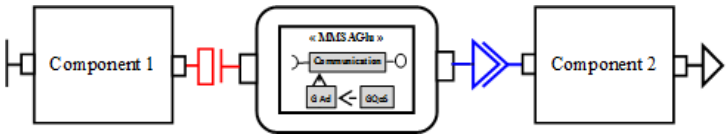


Figure 3. Transformation connector: Text to Sound

Contrary to connectors defined in UML, the adaptation connector is unidirectional. Indeed, the adaptation service of media towards another is neither automatic, neither symmetrical, nor even sometimes feasible (the transformation of Text toward sound is feasible, the reverse is not realistic), it's another service. This constraint is expressed in OCL as follows:

```
context UML::InfrastructureLibrary::Core:: Constructs::Port
inv : self.isStereotyped("ComponentInterface")
implies
  (self.owner.isStereotyped("MMSAComponent")) and
  ( self.ownedOperation->IsEmpty()) and
  (self.required->size() <= 1 or self.provided->size() <= 1) and
  (self.required->forAll(p|p.isStereotyped("InputVideoPort") or
                         p.isStereotyped("InputAudioPort") or
                         p.isStereotyped("InputTextPort") or
                         p.isStereotyped("InputImagePort")) ) and
  (self.provided->forAll(p|p.isStereotyped("OutputVideoPort")
                         p.isStereotyped("OutputAudioPort") or
                         p.isStereotyped("OutputTextPort") or
                         p.isStereotyped("OutputImagePort")) )
```

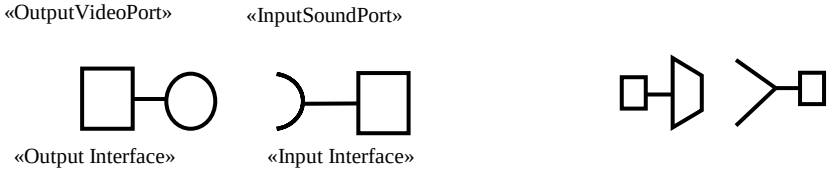


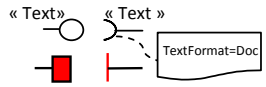
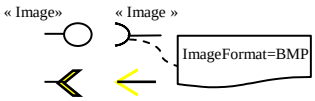
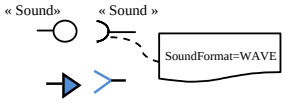
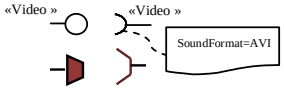
Figure 4. Graphical notation of MMSA Port/Role

Context of MMSAPort. We distinguish in the MMSA meta-model four categories of multimedia ports: text, image, sound and video. A port can be of type «Input» or «Output». It provides a set of services suitable for the media type. For example, a port of video type has the attributes: speed, sampling, etc. In UML, the concept of interface is perfectly identical to the «MMSAPort» concept, but it remains to define its semantics with following constraint OCL:

```
context UML::InfrastructureLibrary::Core:: Constructs::Interface
inv : self.isStereotyped("MMSAPort")
implies
   (self..isStereotyped("ComponentInterface")) and
   (self.ownedOperation->forAll(p|p.isStereotyped("MMSAMedia")))
```

The MMSAPorts are stereotyped as texts «InputText»/ « OutputText », videos «InputVideo» / « OutputVideo», images «InputImage» / « OutputImage», or sounds «InputSound» / « OutputSound». Each stereotype has labelled values (see table 1 and Table 2). Each interface has, for each type of media, a value labelled according to the format.

Table 2. VALUES LABELED OF PORTS OF MULTIMEDIA INTERFACE

	A UML interface "Text" has a value labeled Format: *TTextFormat*.
	A UML interface "image" has a value labeled Format: *TImageFormat*.
	A UML interface "sound" has a value labeled Format: *TSoundFormat*.
	A UML interface "video" has a value labeled Format: *TVideoFormat*.

A stereotyped UML interface “OutputSoundPort” or “InputSoundPort” export or import only data of type “Sound” in only one format, this constraint is expressed in OCL as follows:

```
context UML::InfrastructureLibrary::Core:: Constructs::Port
inv : self.isStereotyped(“OutputVideoPort”)
implies
  self.ownedOperation->forAll(op|op.formalParameter->forAll(fp|fp.direction= #output
  implies  fp.Type.oclASType(Sound).Format  = #WAVE  or
           fp.Type.oclASType(Sound).Format  = #MIDI     or
             ……..
           fp.Type.oclASType(Sound).Format  = #MP3      or
           fp.Type.oclASType(Sound).Format  = #PCM ))
```

4.3 Level of integration of adaptation connectors

Context of MMSAConnector. We include in the UML profile two stereotypes: a stereotype to represent the concept of component “MMSAComponent” corresponding to the component metaclass of metamodel UML and a stereotype to represent the concept of connector “MMSAConnector” corresponding to the connector meta-class of the meta-model UML. The components and the connectors remain distinct with their associated stereotypes (Port and Role). A connector is represented in UML by the class “MMSAConnector”. The class “MMSAConnector” must have at least two ports UML “ConnectorRole” and only one “AdaptationGlue”. This constraint is expressed in OCL:

```
context UML::InfrastructureLibrary::Core::Constructs::Class
inv :self.isStereotyped("MMSAConnector")
implies
(self.ownedPort->size ()>=1)and
(self.ownedOperation->isEmpty())
(self.ownedPort->select(p|p.isStereotyped("ConnectorRole"))-> size()=2) and
(self.nestedClassifier->select(m| m.oclIsTypeOf(Class))->
forAll(g|g.isStereotyped("AdaptationGlue")))-> size()=1) and
(self.clientDependency.target->forAll-> (t|t.oclIsKindOf(Interface))) -> isEmpty())
```

We propose the following graphical notation:

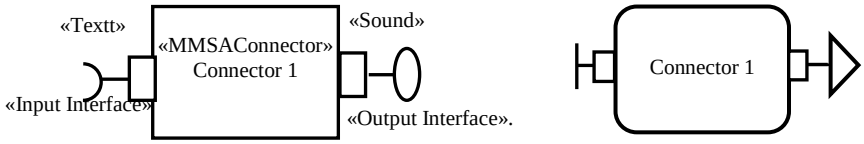


Figure 5. Graphical notation of MMSA connector

An adaptation connector is a mediator between two heterogeneous components or a component and a connector who do not have same MMSA interface.

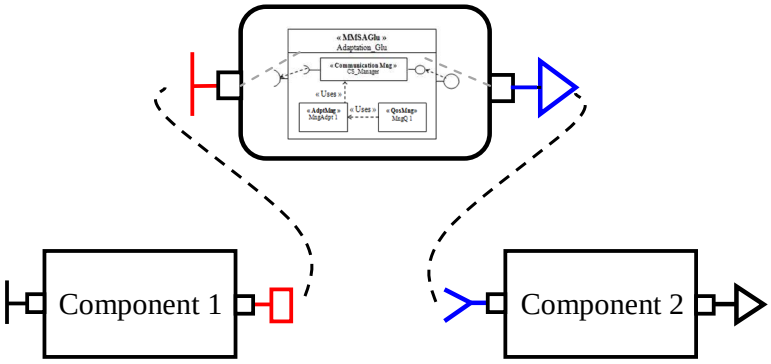


Figure 6. Tronsmoding connector of Text to Sound

Context of ConnectorInterface. An interface of connector contains a set of roles, a role of connector must be related to a wearing of component or a role of another connector. A role is of required type "InputRole" or provided type "OutputRole". A role MMSA "MMSARole" supports only the unidirectional communication since a role MMSA is directed and cannot be used that in only one direction required or provides. A port UML "MMSARole" has only one interface UML "TextRole", "ImageRole", "SoundRole" or "VideoRole". This constraint is expressed in OCL as follows:

```
context UML::InfrastructureLibrary::Core:: Constructs::Interface
inv : self.isStereotyped("ConnectorInterface")
implies
   (self.owner.isStereotyped("MMSAConnector")) and
   (self.required->size() = 1 or self.provided->size() = 1) and
   (self.required->forAll(p|p.isStereotyped("InputTextRole") or
```

```
                    p.isStereotyped("InputImageRole") or
                    p.isStereotyped("InputVideoRole") or
                    p.isStereotyped("InputSoundRole")))and
(self.provided->forAll(p|p.isStereotyped("OutputTextRole") or
                    p.isStereotyped("OutputImageRole") or
                    p.isStereotyped("OutputVideoRole") or
                  p.isStereotyped("OutputSoundRole")))
```

Context of AdaptationGlu. The role of the adaptation connector is to receive the data, to adapt them according to the directives of QoS manager and delivering them to a connector or component recipient. The glue consists of one: "CommunicationMng" "AdapatationMng" and "QualityMng", it describes the work made by each manager in order to ensure the interaction between the components. The concept of MMSAGlu is identical to the concept of association class in UML in the direction where it ensures the adaptation and ensures the communication between the components. Its semantics is defined with following constraint OCL:

```
context UML::InfrastructureLibrary::Core:: Constructs::Connector
inv : self.isStereotyped("AdapattionGlue")
implies
  (self.owner.isStereotyped("MMSAConnector")) and
  (self. nestedClassifier ->select(m| m.oclIsKindOf(Class))->select(cg|
        cg.isStereotyped("CommunicationMng "))-> size()=1)) and
  (self. nestedClassifier ->select(m| m.oclIsKindOf(Class))->select(am|
        am.isStereotyped("AdapatationMng "))-> size()=1)) and
  (self. nestedClassifier ->select(m| m.oclIsKindOf(Class))->select(qm|
      qm.isStereotyped("QualityMng "))-> size()=1))
```

We propose the following graphical notation:

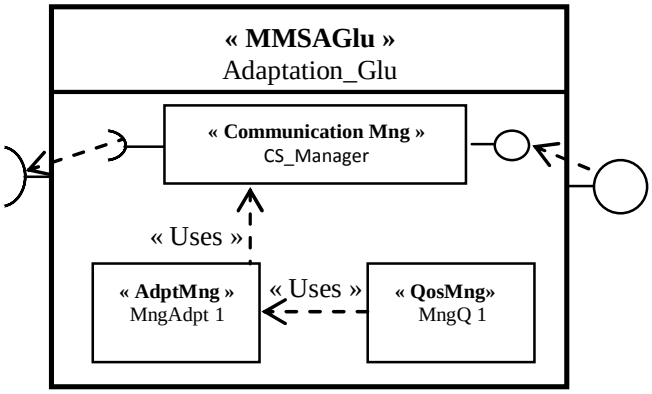


Figure 7. Internal structure of glue

Context of attachment. The attachment is a communication link between two roles or a port and a role (a role of "Output" must be only linked with a role / Port of "Input", and reciprocally). A connector assembly in UML is the concept of attachment that defines a link between an interface "Provided" and an interface "Required". This constraint is expressed in OCL as follows:

```
context UML::InfrastructureLibrary::Core:: Constructs::Connector
inv : self.isStereotyped("MMSAAttachment")
implies
    (self.kind=#assembly) and
    (self.memberEnd.type->forAll(m|m.oclIsKindOf(Interface))) and
    ( self.end->(exists(cp1,cp2|cp1.name <> cp2.name and
 cp1. oclASType(Media).Format=  cp1. oclASType(Media).Format))) and ((self.end-
>select(cp1|cp1 .isStereotyped("InputVideoRole"))->size =1) and
self.end->select(cp2|cp2 .isStereotyped("OutputVideoRole"))->size =1)) or
(self.end->select(cp1|cp1 .isStereotyped("InputImageRole"))->size =1)  and
self.end->select(cp2|cp2 .isStereotyped("OutputImageRole"))->size =1)) or
(self.end->select(cp1|cp1 .isStereotyped("InputSoundRole"))->size =1)  and
self.end->select(cp2|cp2 .isStereotyped("OutputSoundRole"))->size =1)) or
(self.end->select(cp1|cp1 .isStereotyped("InputTextRole"))->size =1)  and
 self.end->select(cp2|cp2 .isStereotyped("OutputTextRole"))->size =1))
```

… And we use the same idea for a port of component with a role of connector and reciprocally.

Context of MMSAConfiguration. An important aspect of MMSA architecture is that of configuration, it is represented by graph of components and connectors. As a UML component can contain sub-components and sub-classes, MMSA configurations are projected towards a graph of UML components with the OCL constraint follows:

```
context UML::InfrastructureLibrary::Core::Constructs::Component
inv :self.isStereotyped("MMSAConfiguration")
implies
    (self.ownedPort->size()=1) and
    (self.ownedOperation->isEmpty()) and
    (self.ownedPort->forAll(p|p.isStereotyped("ComponentInterface"))) and
    (self.member->select(m|m.oclIsKindOf(Component))->forAll
      ->(c|c.isStereotyped("MMSAComponent")))->size()>= 1) and
    (self.member->select(m|m.oclIsTypeOf(Class))->forAll
      ->(c|c.isStereotyped("MMSAConnector")))->size()>= 0)
```

5. An illustrative example: the monitoring system

To illustrate our strategy of projection, we consider an automatic surveillance system; that includes surveillance cameras, an information system and alert equipments. We have the following software components:

- A video capture component (provides video in MPEG format)
- An image improvement component (requires / provides PNG)
- A face detection component (requires / provides PNG images)
- A face recognition component (requires BMP/provides Text)
- A component of querying multimedia DB (provides the image in BMP format)
- An alarm management component (provides sound in wave format)

The Figure 8 describes the surveillance system with MMSA and Figures 9 to 12 shows the representation of that system after the implementation of the proposed UML profile.

```
Class Configuration  Monitoring {
Class Component Acquisition {
    Properties { data-type = video; data-format =MPEG;}
    Constraints {max-persons=1;}
    Interface { Connection-Mode : synchronous ;
            Ports provide{provide_MPEG;}
            Services provide {acquisition-video;} }
}
Class Component Preparation {
    Properties { data-type = image; data-format =PNG;}
    Interface {Connection-Mode : synchronous ;
            Ports provide{ProvImage _PNG;}
            Ports request{ReqImage_PNG;}
            Services provide { treatment -image;} }
}
Class Component Treatment {
    Properties { data-type = image; data-format =PNG;}
    Interface {Connection-Mode : synchronous ;
            Ports provide{ProvImage _PNG;}
            Ports request{ReqImage_PNG;}
            Services provide { treatment -image;} }
}
Class Component Alarm {…….. }
Class Component Recognition {…….. }
Class Component SGBDImage {…….. }
Class Component DataBase {…… }
Class Text-Connector connector1{
    Properties {flow = data}
    Glue {   //simple case of a glue
    Communication {Com_Service}
    Adaptation service {}
    QoS {}}
    Interface {Connection-Mode : synchronous
            Roles_Required {ProvText.OL}
            Roles_Provide { ReqText.OL}}
            Service {Connection} }
Class V-I-Connector connector2{
    Properties {flow = data}
    Glue {   //simple case of a glue
    Communication {Com_Service}
    Adaptation service {MpegToJpeg}
    QoS {resolution}}
    Interface { Connection-Mode : synchronous
            Roles_Required {ProvVideo.Mpeg}
            Roles_Provide { ReqImage.Jpeg }}
            Service {connection, adaptation}  } }
Class Image-Connector connector3 {…….}
Class Image-Connector connector4 {…….}
Class Image-Connector connector5 {…….}
Class Image-Connector connector6{…….}
Instance Monitoring {
 Instances {   CP1 : Acquisition; CP2 : Preparation;
            CP3 : Treatment;    CP4 : Alarm;
            CP5 : Recognition; CP6 : SGBDImage;
            CP7 : DataBase; CN1 : Connector1;
            CA1: Connector2; CA2: Connector3; …}
Attachement {
            CP1toCA1; CA1toCA2; CA2toCP2;
            …….. } }
```

Figure 8. The monitoring system in MMSA

5.1 The functional diagram of application

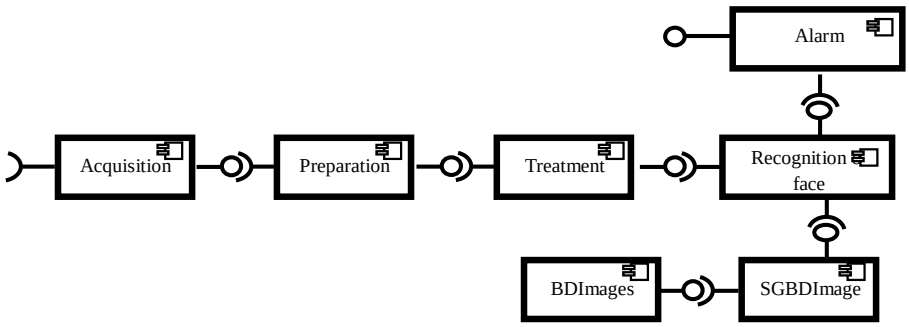


Figure 9. Description of monitoring system in UML 2.0

The figure 9 shows the component diagram of our example, as described in UML 2.0. This modeling with UML ADL does not allow the detection of heterogeneity between the various components

5.2 The use of UML profile

Level 1: Pre-configuration

In this step we represent the monitoring system with the specification of input/output data of each component of architecture. We use here the concepts defined by MMSA

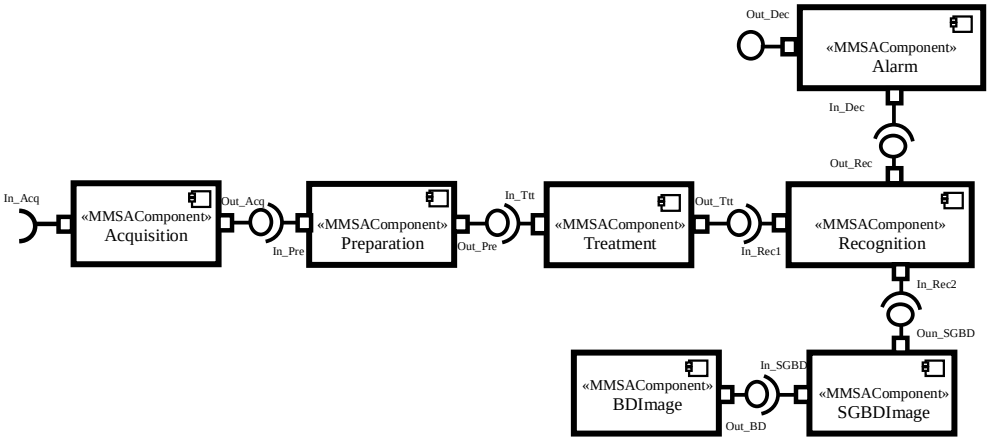


Figure 10. Pre-configuration of system

Level 2: Typing, formatting interfaces and detection of heterogeneity

In this step, we use the notations of the profile that we defined in Section 4. This notation allows us to locate and identify heterogeneity points (data type and data format) of the architecture components.

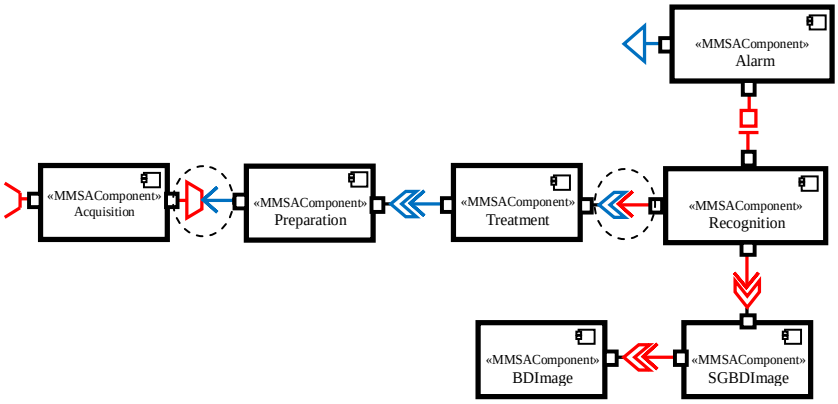


Figure 11. Use of profile components

Level 3: Integration of connectors

In this step, we integrate the connectors between components with the inclusion of heterogeneities, if they exist.

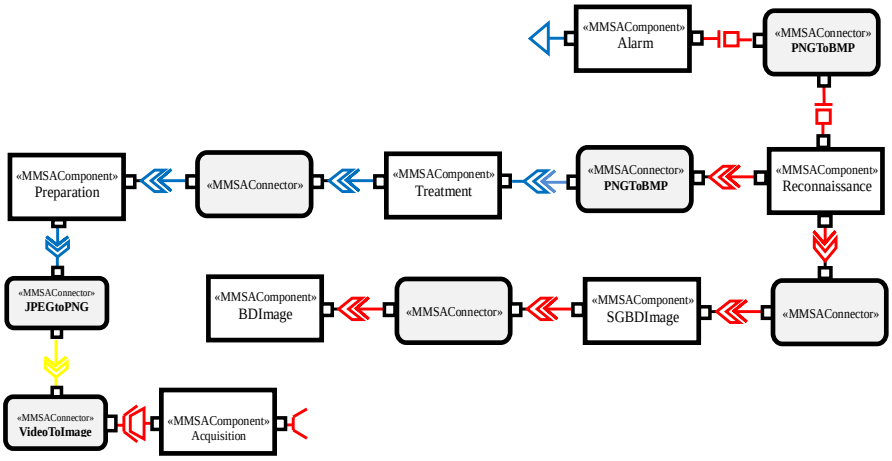


Figure 12. Integration of MMSAprofil connectors

In order to examine the projection of MMSA in UML 2.0, the OCL constraints will be dynamically evaluated on the model of monitoring system. We propose to the software architects the possibility of checking the architecture model to each modification in order to ensure its structural and semantic coherence. The various tests and validations made on the architecture models guarantee perfectly our projection.

6. MMSAplug-in: A Software Architecture Profil Tool

This section presents the development of the MMSA Profile in Rational Software Modeler (RSM) for Eclipse. For this, we choose to use the mechanisms of creating profiles of RSM. Next we focus on what tooling is needed to detect heterogeneity (data type and format) of a given system and to validate its semantics with MMSA approach.

After that we present an example from the tool and we end up with a comparison of the tool with other existing tools.

6.1 Implementing the profil tool

Once we have the MMSA Meta-model mapped into an UML model, we can take advantage of the tools developed around Rational Software Modeler. The UML 2.0 profile for MMSA is implemented in IBM Rational Software Modeler for Eclipse 3.1. This visual modeling tool supports creating and managing UML 2.0 models for software applications, independent of their programming language, and provides a common language for describing formal semantics with OCL language and have been used successfully to define profiles and to valid models of complex systems.

The Plug-In is developed with two levels of abstraction. In the high level, the meta-model of MMSA with all tagged values and its OCL 2.0 constraints is defined by the UML 2.0 profile. This diagram plays an important role in the second level when it is used by the model of software architecture. Once we ensure that the given model complies with the semantic constraints defined by the profile, a set of instances for the types are defined and evaluated in this level.

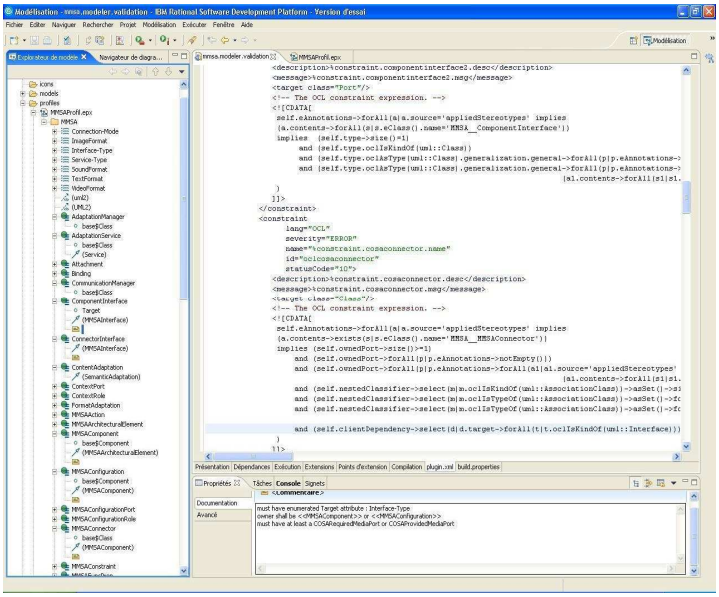


Figure 13. The MMSA-Profile in RSM for Eclipse 3.1.

The main objective of this plug-In is to show the ability to apply the profile for complex applications. The plug-In offers to the architects the possibility to verify the structural coherence of a given system and to validate its semantics with MMSA approach. First we create a diagram of components in UML 2.0 for the described system and then we add the needed OCL constraints. After that, the model is evaluated by the profile. MMSA is defined in UML 2.0 by using the mechanisms of creating profiles of RSM. Figure 13 shows the profile with its stereotypes, all tagged values and OCL 2.0 constraints expressed in the meta-model UML 2.0 – EMF (Eclipse Modeling Framework).

6.2 Final Results

For the surveillance system, we elaborated the system by a components diagram and OCL constraints. Once, MMSA profile is applied from the select profile dialog, shown in Figure 14, all its stereotypes will be available, applied and contributed by the tagged-values. The model then checks to remove any constraints violation.

The model is tested and validated with the semantic constraints defined by the profile. One of the strengths of the MMSA profile tool is their ability to link a model space (i.e. UML) to an architectural space (i.e. MMSA) using model extensions mechanisms (i.e. Stereotype concepts in UML meta-model), and therefore automatic detection of heterogeneity by type/format of media.



Figure 14. Selecting the MMSA profile for the serveillance system.

6.3 Comparison and lessons learned

Our tool MMSAPlug-In can be compared with similar architecture profile tools, such as UML 2.0 Profile for π-ADL (Amirat and Oussalah (B), 2009) and UML 2.0 Profile for C3 (Oquendo, 2006). Indeed, these two applications allow graphical representation of architectures and automatic constraints verification of models using OCL standard.

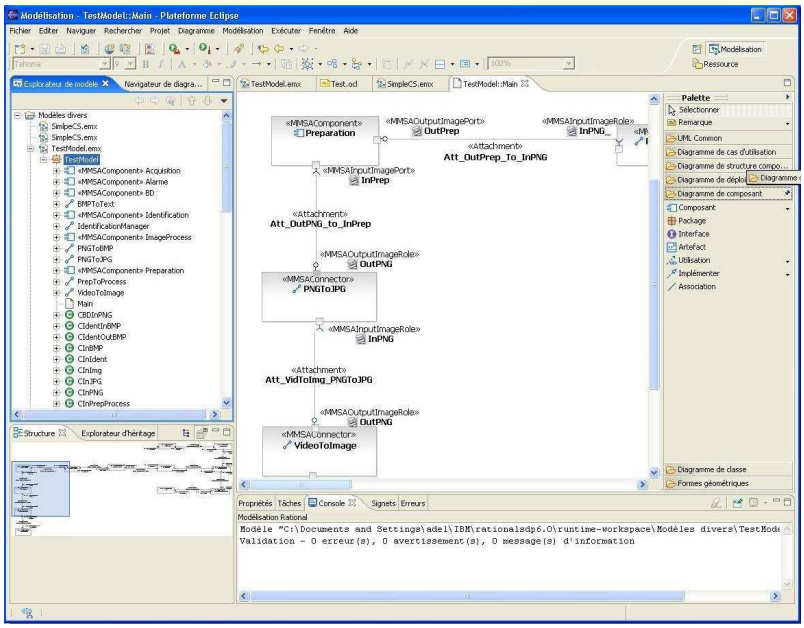


Figure 15. Validating of monitoring system in UML 2.0 with RSM

The use of MMSAPlug-in offers number of advantages compared to these tools, including:

- Providing an easy way to describe complex software architectures in one easy-to-use visual editor and diagramming facilities.
- Implementing most architectural multimedia concepts (Medias ports such as video, audio, text and image, user defined connectors, structures such as configurations of complex components and complex connectors).
- The detection of heterogeneity is done automatically by checking of the constraints of formats and data type.
- Providing a more suitable representation of adaptation connectors which are defined at the meta-level (Class concept of UML 2.0) rather than using a simple attributes for this purpose.

7. Conclusion

To make available to the UML users the concepts and subjacent mechanisms from the ADL, we proposed a specific profile for multimedia applications. Thus we proposed rules allowing translating an UML 2.0 architecture into MMSA architecture. This opens perspectives related to the formal verification of MMSA architectures. The MMSA approach describes in an abstract way the software architectures based multimedia components.

In this paper, we have developed an UML profile for MMSA approach. This profile enabled us to project the concepts (multimedia component, adaptation connector) of MMSA towards the concepts of UML 2.0. An illustrative example was presented at the end of the article. Our profile «MMSAProfile» contains a set of stereotypes where all the values are marked and all OCL constraints are expressed in the UML 2.0 meta-model. We have also developed a plug-In in Rational Software Modeler for Eclipse 3.1 for the profile.

Our future works will be the automatic transformation of models defined by MMSA to .NET using this profile and the integration of this profile in the approach MDA (Model Driven Approach) to ensure the automatism of the transformation process. Actually this profile is limited to MMSA architectural concepts, but we intended to include other multimedia concepts in the profile. Therefore we could have a complete profile for all multimedia concepts and next this profile can be integrated in the approach MDA as a transformation model for all architectural concepts.

8. References

Allen R, Garlan D. A Formal Basis for Architectural Connection. In ACM Transactions on Software Engineering and Methodology, vol. 6, no 3, 1997, p. 213–249.

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, Rome, Italy, 2002, p. 203–210.

Amirat A and Oussalah M. First-Class Connectors to Support Systematic Construction of Hierarchical Software Architecture. In Journal of Object Technology, vol.8, no.7, 2009, p. 107-130.(A).

Amirat A, and Oussalah M. «Towards an UML Profile for the Description of Software Architecture». Proceedings of International Conference on Applied Informatics (ICAI'09), 2009, pp. 226 – 232. (B).

Avgeriou P. Uwe Zdun. “Modeling Architecture Patterns using Architecture Primitives”. OOPSLA’ 05, ACM. 2005.

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. pp. 11-33. 2004.

Balsamo S, Bernado M, Simeoni M. «Performance Evaluation at the Architecture Level Formal Methods for Software Architectures». LNCS 2804. Springer. Berlin, Germany. p. 207-258. 2003.

Belloir N., Bruel J.-M., Hoang N., and Pham C., «Use of SysML for modeling networks of wireless sensors». Conference LMO'08, Montreal, Canada, 2-7 March 2008. P.171-186. RNTI.

Booch G., Rumbaugh J. and Jacobson I., «The unified modeling language user guide», Addison-Wesley Professional, Reading, Massachusetts, 1998.

Clements P., Bachmann F., Bass L., Garlan D., Ivers J., Little R., Nord R. and Stafford J., « Documenting software architectures: views and beyond ». Boston, MA: Addison-Wesley, 2002.

Dashofy, E., Hoek, A.v.d., Taylor, R.N., “A comprehensive approach for the development of XML-based software architecture description languages”, TOSEM, 2005,volume 14, issue 2, p. 199–245,

Derdour M., Ghoualmi-Zine N., Roose P. and Dalmau M., «Toward a dynamic system for the adaptation multimedia fluxes in the P2P architectures», the Fifth International Symposium (FINA) helded in the IEEE 23rd International Conference AINA-09, 2009.

Derdour .M, Roose .P, Dalmau .M, Ghoualmi-Zine .N, Alti .A. «An adaptation approach for component-based software architecture». 34th Annual IEEE Computer Software and Application Conference - COMPSAC 2010 - Seoul - South Corea - July 2010.

Garlan D., « Software Architecture and Object-Oriented Systems », the IPSJ Object-Oriented Symposium, Tokyo, Japan, August 2000.

Generation Partnership Project Technical Specification Group and Universal Mobile Telecommunications System (UMTS). « Quality of Service (QoS) concept and architecture », June 2005.

Goulao. M, Abreu F.B., «Bridging the gap between ACME and UML 2.0 for CBS». Workshop of Specification and Verification of Component-Based Systems, Helsinki, Finland, 2003.

Graf S. , Ober I. “How useful is the UML realtime profile SPT without semantics?”. In SIVOES and RTAS, Toronto Canada. 2004.

Inverardi. P, Muccini. H, and Pelliccione. P. «Dually: Putting in synergy UML 2.0 and ADLs», in: WICSA, Pittsburgh, 2005.

Ivers J., Clements P., Garlan D., Nord R., Schmerl B. and Silva J.R., « Documenting component and connector views with UML 2.0 », Technical Report CMU/SEI-2004-TR-008, 2004.

Jacobson I., « Object-oriented software engineering: a use case driven approach », Addison Wesley Professional, 1992.

Khammaci T., Smeda A. and Oussalah M., «Coexistence of object-oriented modeling and architectural description», Handbook of Software Engineering and Knowledge Engineering, Vol. 3: Recent Advances, World Scientific, 2005.

Maillard S, Smeda A, Oussalah M: «COSA: An Architectural Description Meta-Model». ICSOFT (SE) 2007, p: 445-448.

Marcel C, Michel R, Christian M, Calin L, Costin M. "Dynamic adaptation of services". DECOR'04, Grenoble, France. 2004.

Marcel C, Michel R, Christian M. "Autonomic Adaptation based on Service-Context Adequacy Determination". In ENTCS, vol. 189, p. 35-50, Elsevier. 2007.

Mauro Caporuscio, Valerie Issarny. «A UML 2.0 Profile for Architecting B3G Applications». 3rd International Workshop RISE, pp.18-34. 2006.

Medvidovic N, Rosenblum D-S, and Taylor R-N. A Language and Environment for Architecture-Based Software Development and Evolution, ICSE'99, Los Angeles, May 1999.

Medvidovic N., Taylor R. N. A Classification and Comparison Framework for Software Architecture Description Languages - IEEE Transactions on Software Engineering, vol. 26, no 1, p. 70–93. 2000.

Medvidovic N., Rosenblum D., Redmiles D. and Robins J., « Modeling Software Architecture in the Unified Modeling Languages », ACM TOSEM, Vol. 11, pp. 2-57. January 2002.

Philippe Desferay. « MDE, DSL et UML : where is it? ». Magazine softwares and systems engineering, model driven engineering. september 2009. N°90.

Oquendo F., «Formally Modelling Software Architectures with the UML 2.0 Profile for -ADL», ACM SIGSOFT Software Engineering Notes, 2006, Vol. 31, no. 1.

Object Management Group. « Unified modeling language specification V.1.4 », September 2001.

Object Management Group. « UML 2.0 superstructure specification: revised final adopted specification », Octobre 2004. http://www.omg.org/docs/ptc /04-10-02.pdf

Object Management Group. " UML OCL 2.0 specification: revised final adopted specification", June 2005, http://www.omg.org/docs/ptc /05-06-06.pdf. 2006.

Object Management Group. A UML Profile for MARTE. OMG document ptc/07-08-04, Object Management Group. 2007.

Society of Automotive Engineers "Architecture Analysis & Design Language (AADL)". SAE Standards no AS5506. 2008.