CONTEXT-AWARE MODEL DRIVEN DEVELOPMENT: APPLICATIONS TO WEB SERVICES PLATFORM

Slimane Hammoudi[1] , Samyr Vale[1,2]
[1]*ESEO – Angers, FRANCE*

Stéphane Loiseau[2]
[2]*LERIA, Université d'Angers, FRANCE*

Keywords: Model Driven Development, Context-aware Application, Model, CPIM, CPSM, Merging.

Abstract: Context-aware applications have been developed since the last decade but until today there has not yet been a common definition about what context means. One thing everybody agrees on is that applications must be more adaptable for each user. Context-aware applications aggregate different types of context to be more personalized and to respond more effectively the user needs. Nevertheless inserting personal context information into application code inhibits software development productivity and reuse. We propose in this work the separation of business logic from context properties in different models, as context depends essentially on a particular environment and user situation. We suggest the use of the Model Driven Development (MDD) approach to design context-aware applications. Context information as models can improve reusability and interoperability in contextual applications. We use a context metamodel and suggest a merging model technique to build CPIM (Contextual Platform Independent Model). By traditional transformation techniques a CPSM (Contextual Platform Specific Model) can be built from a CPIM. Web Service is the target platform used on account of its distributed and interoperable characteristics.

1 INTRODUCTION

Context-aware computing is a research theme of Ubiquitous Computing. The term "ubiquitous" in Computer Science was first applied by Weiser (Weiser, 1991). The goal is to adapt systems to respond to the human needs in a transparent way. The most relevant researches in ubiquitous computing focus on user interfaces, context-aware applications and automatic access in human activities. Context-aware computing studies the user-system interaction and how managing context information can adapt services to offer particular solutions to each user need. Some works use client name, address, current user location, IP address or profiles as context information. A simple example is a cookie registered by a web site on the user PC with information about identification, preferences, etc.

In (Dey, 2001), context is defined as: "any information that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant".

From our point of view context is not only information about the environment of the user application but also all information linked to the users' characteristics and needs.

Their needs are revealed by their behaviour in the application (what services they invoked, what information they requested, etc.). We can also extend the notion of environment to the activities that the application achieves by client invocations and what functionalities are being requested to answer the users needs.

These needs can change many times, so to just compose the services by the same domain does not guarantee an effective solution. For Coutaz (Coutaz et al., 2005) the two critical processes are to recognize users' goals and activities, and to map these goals and activities adaptively onto the population of available services and resources.

It is necessary therefore that the context management creates an environment with the possible services or functions that could serve a user in same situation.

The key points are that context could be specific for each user and for each situation, so it is not reasonable to put context properties into application models (business logic). However, context could be represented as a model which contains context information.

The OMG has stimulated the use of Model Driven Architecture-MDA (OMG-MDA, 2001) to define an approach to software development based on modelling and techniques of mapping and transformation to achieve implementation on specific platforms.

In this paper a PIM with context or CPIM (Contextual Platform Independent Model) improves models with context and the CPSM (Contextual Platform Specific Model) achieves contextualized execution. We suggest a merging technique as the most appropriate operation to integrate context models with application models. Separating these concerns in different models will provide model and context reusability and interoperability to application development.

From another point of view, representing context by models leads to a design of adaptive systems through MDD.

This paper is organized as follow. Section 2 briefly recalls and discusses the notion of context and the domain of context-aware application. Section 3 introduces our general approach of context-aware model driven development by emphasizing a merging operation as a main concept for the contextualization of MDD. In section 4 we suggest web services as a target platform to build context-aware applications. Finally, section 5 concludes this paper and discusses future works.

2 CONTEXT AND CONTEXT-AWARE APPLICATIONS

Firstly, we will introduce a non-computational view about context. For a conversation to make sense, two or more people have to be talking about the same issue. This issue is the context of the conversation. If another person arrives, he or she must have to know the context of the conversation to be able to talk about the same issue. If the participants of the conversation talk about different issues then this conversation will be only noise without any sense. The context is the element that provides relevant information to a conversation.

The idea of context-aware applications is to provide an effective system response using user behaviour information and adapt the system to provide the solution desired.

Building context-aware applications is a real challenge. Managing context is difficult because there is no pattern about how to capture context, abstract its information, model or use it.

So in the current context-aware systems, the context is usually represented inside the application model (business logic). The designer takes the user context provided by some context provider and integrates it into the application code. One problem of this approach is that it covers only one kind of context, while generally context is not something static and it could change in different situations: different behaviours, users, devices, environments, etc.

3 APPLYING CONTEXT IN MODEL DRIVEN DEVELOPMENT

The Model Driven Development uses a set of patterns to create, implement, deploy and reuse systems by models. This paradigm uses different layers with different abstraction levels to represent the application by model. The OMG`s MDA (Model Driven Architecture) (OMG-MDA, 2001) defines three principal layers.

- Computational Independent Model (CIM), without system structure details.
- Platform Independent Model (PIM), without platform details.
- Platform Specific Model (PSM), specifies execution platform requirements.

These layers target different levels of abstraction. The essence of MDA is the transformation capability of different models (Lopes et al., 2005). The MDA models are based on a metamodel that defines the model language and the metamodels are based on a meta-metamodel. Model transformations are defined by transformation rules using transformation languages (Bézivin et al., 2004).

As said before, usually context-aware applications combine context characteristics with the application logic. It augments the level of complexity of the application, because business logic is mixed with user context. Another problem with this approach is changing the context characteristics will change all the design of the application.

In MDA through the use of models we aim to "disconnect" business logic from a specific domain, and, by this way, to obtain applications which will be adaptable to many contexts.

The objective of this work is to improve models of context independently of the business logic based on MDA and afterwards provide the integration of these context models to different applications.

With this approach we identify some benefits. In one hand the application model could be integrated in different user contexts and the user context could be integrated to different application models. In this case changing context does not affect the business logic.

3.1 The Context Model Development and the Merging Process

Mappings and transformations are the heart of MDA (Bézivin et al. 2004). Building contextual applications implies that a context model has to be integrated with an application model.

Mapping and transformations are not fitted to this task because the objective here is to aggregate context characteristics with the business logic. To do this task we propose merging techniques that are used when different models contain specifications of different requirements.

In (Bottoni et al. 2006) the authors consider that the merging operation is more suitable when dealing with multiple input models to create one output model.The same work says that model merging is a multiple model transformation and the semantics of the source models are preserved in the merge model.

In Figure.1 we can see our proposition of merging techniques to aggregate context information with the application model followed by traditional transformations rules which can be applied to build platform specific models.

We can also use the mapping techniques specified in (Lopes et al. 2005) to map Contextual PIM (CPIM) in others CPIMs and Contextual PSM (CPSM) in others CPSMs. Mapping rules can also be used to construct CPSM from CPIM as carried out for the PIM to PSM transformation.

The context model is completely independent of the application model. In Figure.1 the Context Metamodel is built separately from the Source Application Metamodel.

The Context Metamodel only has to contain the elements of the context. Both conform to the same Meta-metamodel, i.e. to be merged they must be represented in the same formalism. If they are represented in different formalisms a mapping operation has to be made to put all input models in the same formalism.

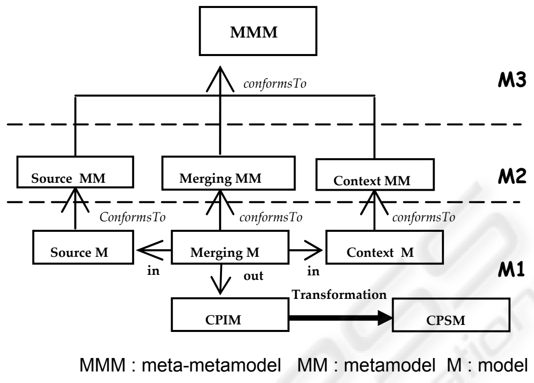

Figure 1: MDA-based model merging for Context-aware development.

As presented in (Lopes et al., 2005), mapping techniques can be used to generate automatically transformation rules between models, which will be applied from CPIM to CPSM.

The other metamodels: merging metamodel, transformation metamodel, and target ones must obey the same rule.

The Context Model and the Application Model are the input models to be merged. The merging model is based on the merging metamodel (which defines the Semantic of Merging Rules).

After the Merging operation we have as an output model the Contextual Application Model. Note that the contextualization could be considered either in a PIM or PSM model. This could be achieved also on the platform, e.g. a UML Source Model (with or without merged context) can be transformed to a web services platform represented by a WSDL (W3C-WSDL, 2001) target model (contextualized or not).

4 APPLICATION TO WEB SERVICES PLATFORM

Web Services (W3C-WSA, 2004) have been used to build software applications on account of their distributed and interoperable properties.

There are some works building context-aware applications on web services platform. We intend in a modelling context to provide relevant services by conceptualizing and structuring context information.

In Web Services platform context information could be modelled as XML Schema.

In this platform the context can be supplied by a service that accesses the XML context information using a SOAP (W3C-SOAP, 2001) message.

This web service can also be published in a UDDI (UDDI, 2002) registry and linked to others services.

However, as we said before our objective in this work is showing that separating context properties from application code improves a best practice in ubiquitous software development.

This web service can bind user's identification, profiles, actions or preferences described in an XML archive. There are some XML databases and relational databases that support XML.

The application *servlet* based on this context information can invoke other specific services to provide effective solutions.

The context service can also append the context information about this user in a repository or database. This action improves real time reaction by context information.

5 CONCLUSIONS

The objective of this work is to propose the MDD approach to develop context-aware applications. Nowadays, most ubiquitous applications developers put context information into application code.

This way keeps the application and context from being reusable. As context is particular of each user, situation and environment we propose the separation of the context characteristics from application code. To do this and to improve reuse and interoperability we have suggested the Model Driven Development Approach.

Context can be modelled independently of the application model and can be integrated with the last one by merging techniques.

Contextual Platform Independent Model (CPIM) is a solution proposed to integrate Context Model and OMG`s MDA PIM. Moreover the CPSM can be built by transformation from CPIM.

Using this approach the abstraction level can be increased and different formalisms can be used to represent context.

The use of traditional transformations can change the models formalisms and refine each model. Here we have presented a brief discussion about context and merging techniques to improve context in model level.

The web service is a target platform used to support context implementation. Its interoperability and distributed characteristics become the best platform to develop contextual applications. For future works we suggest the development of a context model framework to guide the designer in modelling context and a semantic for merging operation.

REFERENCES

Dey A. K., *Understanding and Using Context*, Personal and Ubiquitous Computing, 2001: 4-7.

Bezivin J, Hammoudi S, Lopes D, Jouault F, *An experiment in Mapping Web Services to Implementation Platforms*, Research Report, RR-IRIN2004-01, France, 2004.

Bottoni P, Fulvio D'Antoni O, Missikoff M, *Towards a unified view of model mapping and transformation*. EMOI Workshop, 2006. CAISE Conference.

Coutaz J, Crowley J L, Dobson S, Garlan D, *Context is key,* ACM, 2005: 49-53.

Lopes D, Hammoudi S, Bézivin J and Jouault F, *Mapping Specification in MDA: From Theory to Practice*, INTEROP-ESA 2005, Geneva, Switzerland, 21–25.

OMG-MDA., *Model Driven Architecture (MDA*)-document number ormsc/2001-07-01. (2001)

UDDI. *Universal, Description, Discovery and Integration (UDDI) 3.0*, 2002. http://www.uddi.org.

Weiser M. *The computer for the 21st century*, Scientific American, 1991. p. 94-104.

W3C-SOAP. *Simple Object Access Protocol (SOAP) 1.1*, 2001.http://www.w3.org/TR/SOAP.

W3C-WSDL., *Web Services Description Language (WSDL)1.1*, 2001.http://www.w3c.org/tr/wsdl.

W3C-WSA. *Web Services Architecture (WSA)*, 2004. http://www.w3c.org/TR/2004/NOTE-ws-arch-20040211/.