Taking Context into Account in Conceptual Models Using a Model Driven Engineering Approach

Arnaud Brossard[1,*,a], Mourad Abed[2,3,4] and Christophe Kolski[2,3,4]

[1] Aplon France, F-62223 Anzin St Aubin, France

[2] Univ Lille Nord de France, F-59046 Lille, France
[3] UVHC, LAMIH, F-59313 Valenciennes, France
[4] CNRS, UMR 8530, F-59313 Valenciennes, France

Context: In public transport, travelers (considered as information systems users) do not have the same objectives and/or concerns at the same time. For this reason it is not always easy to provide them with the right information at the right time. If personalizing the information to the user allows to do this to some extent it is not enough since the information could also depend on the use of the context and the environment (e.g., place, time…).

Objective: This paper proposes a solution allowing the context to be managed inside an application's conceptual model in order to provide a more flexible web application from the users' point of view.

Method: Our work is based on a modeling method using the Model Driven Engineering (MDE) approach and on practical field experiences permitting us to validate our solution. Our domain of application is personalized transport information.

Results: Introducing the notion of context into rules and decision trees that are used inside conceptual models allowed us to incorporate context as important information for personalizing web applications.

Conclusion: The context should be integrated into an application during the modeling phase to allow a smooth integration inside the application and to facilitate the evolution over time. Our solution offers a better users' experience through an extended personalization of web applications.

Keywords: Business Process, Model-Driven Engineering, Human-Computer Interaction

1. Introduction

Nowadays, developing applications for end users is becoming increasingly complex as they want to access these applications everywhere, at every moment of the day and the night and with many different devices [1] [2]. In addition, users want to access the right application and the right information only when needed, and they do not want to be overwhelmed by information [3] [4]. These users' requirements are even more critical in the field of public transportation where users are, by their very nature, mobile and do not have the same goals (e.g., waiting for a train, looking for a taxi or a restaurant, planning a trip, etc.) [5]. To be able to offer this kind of services, designers of interactive applications must design applications that are less dependent on the technical aspects and that can

[*] Corresponding author: Tel.: +33 321077155 , Email address: a.brossard@aplon.org

[a] This research was conducted while Dr. Brossard was at the *Université de Valenciennes et du Hainaut-Cambraisis* (UVHC, LAMIH, Valenciennes, F-59313, France). At present, he is the CEO of *Aplon France*.

exploit the user's context to provide the user with useful information only. The rising use of mobile devices only adds to the complexity of designing new applications. Although the discussion in this paper is limited to web applications, the above mentioned observation is generic to application design.

Currently on its way to becoming the new paradigm for developing applications [6] [7], Model-Driven Engineering (MDE) allows applications to be created using conceptual models, in particular by combining already existing models. Although the notion of *context* was defined long ago [8], only a few studies have tried to integrate this notion into an application modeling approach, and even fewer have tried to integrate it into an MDE modeling approach [9] [10].

Nowadays, the paradigm of applications factories is becoming increasingly important in today world [11]. Its long term goal is to achieve a genuine industrialization of IT developments that would allow a significant improvement in the development phase in terms of cost, quality and deadlines. In order to reach this goal, the development of an application would be in the hands of business experts who would then be able to create their own applications from an assembly of existing components. One of the important steps to achieve this goal was the development of service oriented architectures (SOA) [12] and more particularly approaches based on business processes [13] [14]. So today it is important to consider this trend in IT development of the creation and/or establishment of a MDE process type.

According to Tariq &. Akhter [15] and our own analysis of the existing methods and tools [16], only a few MDE methods exist that allow interactive applications to be modeled using conceptual models. For this reason, we developed PERCOMOM (PERsonalization and CONceptual MOdeling Method) [16], which allows the semi-automatic creation of personalized WIMP (Window, Icon, Mouse and Pointing device) applications, using reusable conceptual models based on business processes. However, knowing how to model an application is only the first stage of an MDE approach. Reusing models can create new problems related to their evolution over time; these problems can become particularly important if the models are shared by several applications.

In this paper, our contribution to provide a solution to these problems is focused on the validation of the capacity to manage the context inside an MDE approach at the conceptual level. For that, we developed a set of decision-tree rules that define the type of conceptual models that must be modified. We tested these rules with PERCOMOM, and more specifically on the conceptual models related to the business processes, but they were intended to be used with any modeling approach based on MDE.

The next section focuses on what led us to create these decision-tree rules. The third section presents these rules within the framework of context awareness. The fourth section provides actual examples in the field of public transportation along with a discussion of the associated limits and constraints. The fifth section summarizes our conclusions and outlines future research directions.

2. Modelling Business Processes that integrate interactive applications

In this section, we provide a brief review of the literature on conceptual modeling of computer applications and present different approaches for modeling an interactive application. Then, we summarize the new orientations for the conceptual modeling of interactive applications and introduce the notion of *context* and examine the complexity associated with it. Finally, we present the existing solutions for adapting computer applications to their context of use.

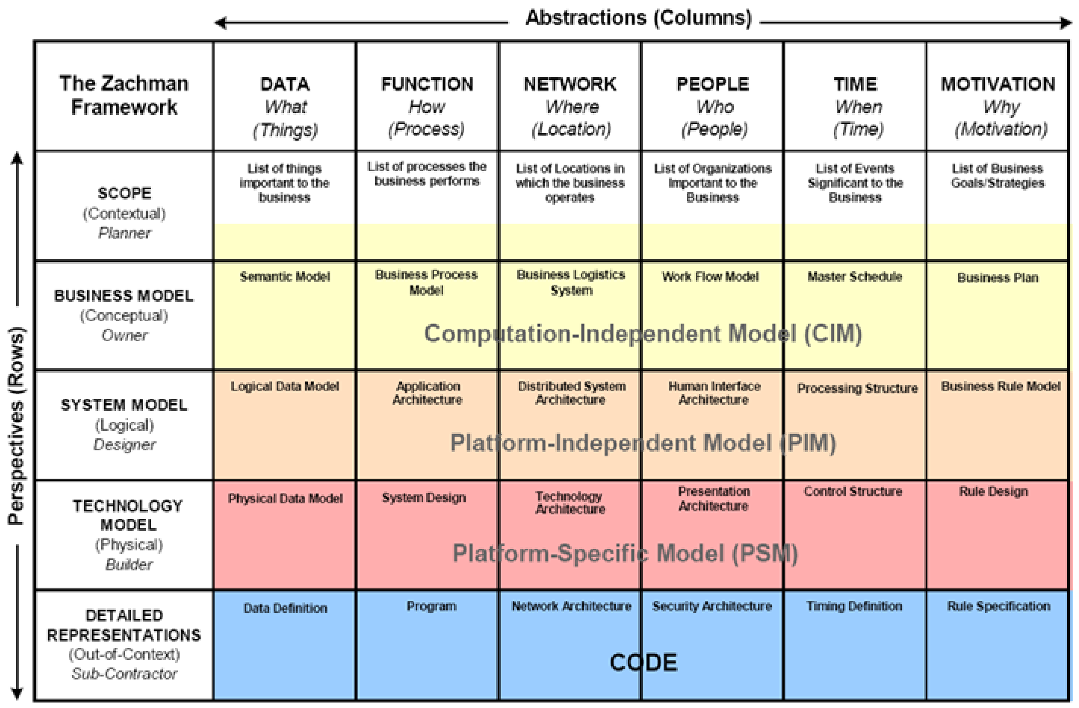
2.1 Model Driven Engineering: Principles

A Model-Driven Engineering (MDE) approach to application design is an approach that makes use of several conceptual models so that each model manages one or more well-defined part(s) of the application. Because of the conceptual nature of such models, they would not have to address the technological problems associated with the final application handled by the users [17]. Within an MDE modeling framework, the main focus is on the design and management of conceptual models. In this paper, we retain the definition of a conceptual model given by the Object Management Group [18] for the Computation Independent Model (CIM); namely, "*A computation independent model is a view of a system from the computation independent viewpoint. A CIM does not show details of the structure of systems. A CIM is sometimes called a domain model and a vocabulary that is familiar to the practitioners of the domain in question is used in its specification.*"

The objective of such conceptual models is to make it possible for business domain experts – who are not data processing specialists but rather specialists in their own domain, to define the models of the business processes that they use and that they want to see included in an application. The passage from a conceptual model to an actual application is accomplished through a succession of model transformations based on a Model-Driven Architecture (MDA). We studied these model transformations from the point of view of Human-Computer Interaction (HCI).

Through its use of three abstraction levels, MDA separates the logic of the business processes associated with the application from the technology that is used to execute these processes. This is done with the objective of removing the direct link between the applications and the coding, thus facilitating their interaction and making them less sensitive to technological changes. The transition from one level to another is done through model transformations that, at each stage (from CIM to PIM – Platform Independent Model – and from PIM to PSM – Platform Specific Model), makes it possible to enrich the models with the necessary and sufficient technical information needed. Given the numerous model transformation tools that exist today, we refer the readers who desire more information to the review done by Czarneski *et al.* [20] for a more detailed presentation and classification of these tools.

From the functional point of view, one of the advantages of the MDE approach is that it covers all the domains used in an information system architecture, as defined in the Zachman Framework [19] (see Figure 1). At the CIM level, this is done through a whole set of models each covering a specific abstraction domain (Things, Processes, and so on). In the field of interactive applications, these models, linked together, allow to cover all the different aspects of an application.


Abstractions (Columns)
Perspectives (Rows)
Computation-Independent Model (CIM)
Platform-Independent Model (PIM)
Platform-Specific Model (PSM)
CODE

The Zachman Framework	DATA *What (Things)*	FUNCTION *How (Process)*	NETWORK *Where (Location)*	PEOPLE *Who (People)*	TIME *When (Time)*	MOTIVATION *Why (Motivation)*
SCOPE (Contextual) *Planner*	List of things important to the business	List of processes the business performs	List of Locations in which the business operates	List of Organizations Important to the Business	List of Events Significant to the Business	List of Business Goals/Strategies
BUSINESS MODEL (Conceptual) *Owner*	Semantic Model	Business Process Model	Business Logistics System	Work Flow Model	Master Schedule	Business Plan
SYSTEM MODEL (Logical) *Designer*	Logical Data Model	Application Architecture	Distributed System Architecture	Human Interface Architecture	Processing Structure	Business Rule Model
TECHNOLOGY MODEL (Physical) *Builder*	Physical Data Model	System Design	Technology Architecture	Presentation Architecture	Control Structure	Rule Design
DETAILED REPRESENTATIONS (Out-of-Context) *Sub-Contractor*	Data Definition	Program	Network Architecture	Security Architecture	Timing Definition	Rule Specification

Figure 1. MDA Model used in software development for the Zachman Framework [19]

After this brief description of the general principles of MDE, we will present, in the next section, the existing methods and approaches for defining and managing the conceptual models.

2.2 Existing methods for the conceptual modeling of applications

Although MDA has been around for several years [21] [22], it is seldom used to create large interactive applications. When used, it is frequently incomplete in spite of the abundant research showing the advantages of using it [23] [24] [25], particularly in HCI [26] [27]. Nowadays, its use is limited today partially due the lack of tools and methods for modeling at the CIM level [14] [16] with truly independent models from the computer techniques used to create the applications. In fact, most of the existing solutions proposed use only the PIM and PSM levels of MDA.

Today, one of the most advanced HCI approach is probably UsiXML, a XML-based markup language for defining human-computer interfaces [28] (see Figure 2), developed on the framework CAMELEON [29] [30] (see Figure 3). By defining a level called "*Tasks and concepts*", this framework not only allows user tasks to be handled, but also contains an entire set of additional concepts allowing the users or the environment to be modeled [31]. In UsiXML, user tasks are modeled using ConcurTaskTrees (CTT) [32]. This method uses a hierarchical structure for user action tasks, temporal operators and the handling of objects that communicate between the tasks.

Using UsiXML, user interfaces can be defined at such a high abstraction level so that it is possible to free the model from the constraints related to the technical platforms that will actually be used by the final application. The passage from the abstract interface to the concrete interface is accomplished through a succession of transformations and enrichments of the initial abstract models. Due to its approach, UsiXML is compatible with MDA. For a detailed presentation of UsiXML, the reader could refer to Florin's doctoral thesis [33] and to the UsiXML Consortium website, http://www.usixml.org.

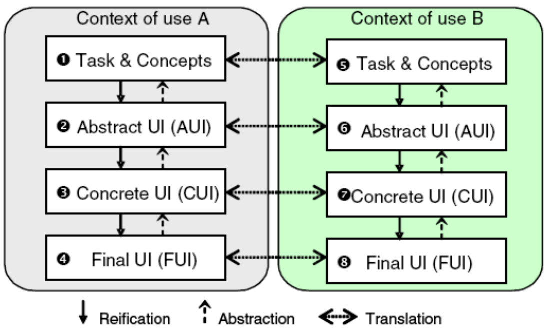


Figure 2. Simplified Model of the framework associated to graphical interfaces in UsiXML [28]

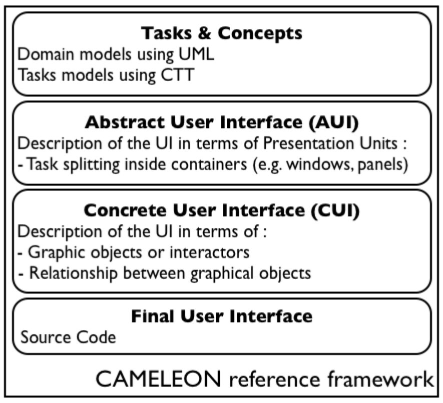

Tasks & Concepts
Domain models using UML
Tasks models using CTT
Abstract User Interface (AUI)
Description of the UI in terms of Presentation Units :
- Task splitting inside containers (e.g. windows, panels)
Concrete User Interface (CUI)
Description of the UI in terms of :
- Graphic objects or interactors
- Relationship between graphical objects
Final User Interface
Source Code
CAMELEON reference framework

Figure 3. Global diagram of the CAMELEON reference framework used by UsiXML [29] [30]

The CUP 2.0 [34] approach, centered on the creation of contextual graphical interfaces, models an application by using UML 2.0 stereotypes (see Figure 4), which facilitates the use of existing UML 2.0 tools for modeling applications. However, it also has the drawback of being directly linked to the UML language, and thus to the Object Oriented paradigm. CUP 2.0 is based on a direct translation of the contextual ConcurTaskTrees [35] inside a UML profile to express all the processes used inside an application.

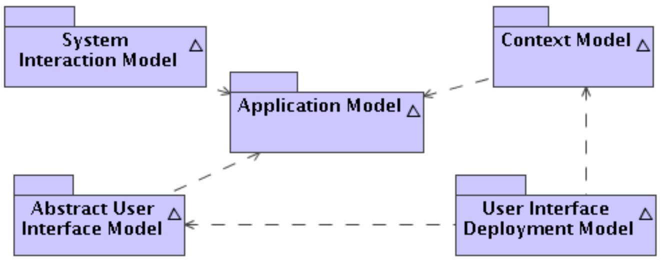

System Interaction Model
Context Model
Application Model
Abstract User Interface Model
User Interface Deployment Model

Figure 4. Overview of the models supported by UML Profile CUP 2.0

Other approaches have also been proposed in the HCI literature. For example, Schattkowsky *et al.* [36] introduced an approach that strictly separates the interface's visual aspects from the other elements of the overall model, particularly from the processing elements. The Wisdom approach, introduced by Nunes [37], separates the analysis and design models for the user interface from the functional processes associated to an application by using different models. Proposed by Pinheiro da Silva *et al*. [38], UMLi extends the UML task model in a proprietary way and uses a specific interface model to model interactive applications. However, all these approaches are less advanced and less complete than UsiXML or CUP 2.0.

In the field of public transportation, the various presented approaches have a certain number of limitations that make them unable to take all the complexity of the field into account. For example, they are unable to take the problems related to content personalization as well as those related to secure access to the applications into account. Moreover, they only take very partially the problems related to the context into account.

Today, the tools and methods for creating applications containing models are not limited to the research world, and there are a certain number of commercial products on the market that are able to apply an MDE approach relatively well. Among these products, we can mention:

- Leonardi by Lyria (http://www.lyria.com), which allows fast GUI generation for database applications through models and model transformations connected to MDA.
- Acceleo by Obeo (http://www.acceleo.org), which is a code generator that allows PIM models to be transformed directly into executable code.
- Arcstyler by Interactive Objects (http://www.arcstyler.com), which is also a code generator that allows PIM models to be transformed directly into executable code.
- Rational Software Architect by IBM (http://www.ibm.com), which is another code generator that allows PIM models to be transformed, while also taking into account the characteristics of various functional fields through the use of different modeling languages.

The last method that we will mention is not strictly a commercial product, being developed with the support of the European Commission. Comet (http://www.modelbased.net/comet), developed primarily by SINTEF ICT, is an incremental method for modeling and designing interactive applications. This method is itself based on functional business processes models and uses the UML language to generate the applications, with the models progressing towards the final application through a series of transformations.

Even if these "commercial" methods and tools represent an unquestionable advance in the use of MDE, they nevertheless have limitations. Leonardi deals with database applications but only with aspects related to the GUI and not business processes associated with the GUI. Acceleo, Arcstyler and Rational Software Architect are all tools for generating code, using a PIM model; they do not offer or use any real conceptual models. As for Comet, though the method and the tools are closest to a true MDE approach, the product orientation towards internet applications based on a Service Oriented Architecture (SOA) greatly limit its potential.

To address the problem related to the difficulty of managing the CIM models in the different approaches presented, new solutions are proposed in the next section.

2.3 New orientations for the conceptual modeling of interactive applications

Today, the Unified Modeling Language (UML) [39] is the most used language for modeling applications. However, it is too close to the Object-Oriented Programming paradigm to be a good candidate for modeling applications at the conceptual level. In addition, UML is not always easily readable by the various stakeholders in complex projects, particularly non-computer scientists [40] [41] [42] [43] [44] [45].

A new paradigm emerged a few years ago, called the business process paradigm. For this new paradigm, new modeling languages have been proposed, such as Business Process Modeling Notation (BPMN) [46] as well as new principles for creating web applications using Service-Oriented Architectures (SOA) [47]. However, this business process paradigm makes it possible to solve only part of the time evolution problems. Although BPMN takes functional task models into account, it does not provide specific modeling solutions for interactive tasks involving the user, as the ConcurTaskTree (CTT) notation does [48] but CTT is not without its limitations in that it only partially takes into account functional tasks.

Such deficiencies, specifically in the domain of Human-Computer Interaction (HCI) [27], led us to propose PERCOMOM [16] (described in section 3.1). PERCOMOM allows functional and interactive tasks to be managed by creating conceptual models used in specific business processes. First used in the field of web applications for public transportation (see section 4), PERCOMOM provides an initial solution to the problem of managing context, by making web applications context-aware while designing the application or after it has been designed through a reengineering process.

2.4 Notion of context

In this paper, we use Dey's definition for context [8]:

> "*Context is any information that can be used to characterize the situation of entities (i.e., whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves. Context is typically the location, identity and state of people, groups and computational and physical objects.*"

Thus, context in the domain of artificial intelligence could be defined in terms of time, location, culture, topic, granularity and modality [49]. In relation to workflow, context could be defined in terms of function, behaviour, causality, organization, information, operation and history [50]. In relation to business processes, context could be defined in terms of business processes; sender and receiver roles; third-party roles; industry, product or service classifications; geopolitical information, legal and contractual constraints; system or application constraints; or compliance requirements [51]. Given these definitions, it is important to point out that the notion of context is dependent on both the applications and the business domain in which it is used. In short, the context elements could be defined as all the relevant information:

- that are not processing data (e.g., they do not come from a database),
- that are not directly linked to the business processes (i.e., each business process could be executed without knowing the context), and
- that could have an influence on the business processes (e.g., the result of a business process or the task sequence inside a business process could depend on a context element).

In the framework of context-aware web applications, which is the subject of this article, it is possible to define the context as being composed of:

- the web application used, which represents a specific context of use for a model;

- the technical aspects related to using the web application, which represents a context that can produce constraints and limitations for the application; and
- the "physical" external environment in which the web application is used, which is specific to the place where the user is located [52].

Each group of elements can itself contain a significant number of sub-elements. Thus, the "physical" environment can be composed of the noise level, the luminosity level and so on. One of the common characteristics of these sub-elements is that they must be quantifiable and measurable to be used in an application.

In PERCOMOM, we define three types of contextual elements, which are extracted from the context elements identified by Kaltz et al. [9]:

- The *application* context, which makes it possible to know the application or the application category used. The category is defined by its place in an ontology; a web application can belong to one or more categories.
- The *geographical* context, which makes it possible to know the place where the web application is being used. This context is based on a geographical ontology that identifies the links between the various types of spaces (e.g., country, city) and the various places (e.g., France, Wales, New York)
- The *temporal* context, which makes it possible to locate the moment that the web application is used, compared to temporal intervals (e.g., public holidays) that can be either fixed (e.g., Christmas) or repetitive (e.g., every first day of the month).

It is important to note here that while some elements of context can be more or less easily modeled or integrated into models of applications, taking into account all the elements representing the context is almost unthinkable. Thus, a user of public transportation, located at a station, taking into account the full context would require to include in the models the physical environment (location, noise level, atmospheric pressure, and so on), the other users present in the station (attitudes, expectations, needs, behavior, and so on) and the user himself or herself. This represents a significant amount of information that could not be taken into account in a single model without making it very difficult to understand and use. Therefore, taking into account the context in applications makes it imperative not to take into account all aspects of the context of use but only the most relevant from the viewpoint of the application.

2.5 Adaptation to the context

For Han [53], speaking about interactive applications on the internet, the adaptation to the context is defined as *"the process of selection, generation or modification of content (e.g., text, images, animations) to suit the user's computing environment and context of use"*. Adaptation can be done in three ways [54]:

- *manually*, in which case the adaptation is only done if the user has explicitly asked for it.
- *semi-automatically*, in which case the adaptation is done in relation to user choices.
- *automatically*, in which case the context adaptation was determined in the design phase and the user cannot express any choice.

From the business processes point of view, the adaptation to the context could be [10]:

- an *extrinsic* adaptation process, which is triggered from outside the business process but is nonetheless used to define and characterize the context, or
- an *intrinsic* adaptation process, which allows the business process to adapt itself to the context.

If, at a first glance, the process seems easy to manage, many problems remain unsolved [10], for example: 1) the conceptualization of the business process context, 2) the integration of context elements in the business process design, and 3) the support of context-aware business processes.

Conceptualization of the business process context: The solution that seems to be emerging today is the use of context ontologies [55]. These ontologies allow each context to be defined in relation to a specific business domain. They also define the semantic relationships between the context elements, which allows the expression of how the context state could evolve over time (i.e., only some context elements are considered to be valid for a given context situation).

Integration of context elements in the business process design: This problem is related to the problem of how enough flexibility can be introduced into a business process to allow it to adapt itself to the context of use. One of the possible solutions could be to create a specific business process for each context state; in this case, the context information would be managed directly inside the business process. Although this solution would allow a great deal of control of the context adaptation process, it would not be easy to manage over time since a great number of business processes would have to be defined. On the other hand, the solution proposed by Andersson *et al.* [56] takes the business processes' adaptation to the context into account during the design phase and run-time. This is done by using the business process patterns, which could then be automatically used during runtime to allow each process to adapt itself to the context.

Support of context-aware business processes: This type of support could be done by using ad-hoc processes, which are subject to change during execution, and knowledge processes, which allow expansible process data and process models to be defined for specific domains [57]. It could also use case-based reasoning to allow the adaptation to be processed during run-time [58]. Or, it could be managed though such tools as ProM [59], which allows different kinds of solutions to be combined.

The context elements now being defined, the next step is to determine how to generate context-aware web application models and how to allow already existing models to manage new context elements.

3. Context of use adaptation for interactive applications: contribution focussed at the conceptual level

Before presenting our solution, it is important to note that the research described in this paper does not involve a global reflection on business process lifecycles. The paper focuses on defining rules that will allow existing models to evolve to become context aware. In fact, it solves only one small part of the reengineering problem related to existing business processes in a context-aware environment, as defined by Saidani et al. [60] and Bessai et al. [61].

It is also important to point out that the solution presented in this paper is not focused on modeling contextual interfaces, which is a dense, rich research field that is far beyond the scope of this paper. The reader interested in this field of research could refer to Kris Lyuten's thesis [62] or the different articles by Breiner *et al.* [63], Forbrig *et al.* [64], Hinz *et al.* [65], Kjedov *et al.* [66] and Jacquet *et al.* [67].

The main contribution of our works resides on the validation of the capacity to manage the context inside an MDE approach based on business processes at the conceptual level.

Roseman et al. [68] have proposed using a context framework, based on a context classification, to adapt the business processes. Kapitsaki et al. [69] have proposed solving the problem by separating the business functions from the context adaptation function; in their method, the context adaptation is made during the model transformation between the PIM and the PSM level, for a specific context. Our method allows functional (i.e., business tasks) and interactive tasks to be managed by creating conceptual models that could have different behaviors depending on the use context. In this section, we begin with a quick presentation of our method, PERCOMOM, which was used as the basis of our research on integrating the context of use into conceptual models. Then, we present how the notion of context is used inside the conceptual models defined by PERCOMOM. Finally, we examine how the notion of context could be managed inside a modeling method like PERCOMOM.

3.1 What is PERCOMOM?

Based on Model-Driven Architecture (MDA) [70] and inspired by previous research in our lab [5] [71] [72] [73], PERCOMOM [16] allows business experts themselves to model personalized interactive applications. As shown in Figure 4, PERCOMOM defines 14 types of conceptual models. These models, which were designed to cover a large range of applications, are not mandatory for each application.

In fact, each model represents, for a specific kind of business experts, a view of the application in the sense used in architectural modeling as described by Rozanski *et al* [74]. For example, a business expert in business rules will only work with models dealing with business rules used in the application whereas a business expect in security problems will work with models dealing with the security of the application. And, if a model is not relevant with a specific application, it will be not used during the modeling phase of the application. For example, a geographical model, which deals with geographical information, will not be useful for a simple textpad application. This allows, during the modeling phase of the application, business experts to fully express their knowledge and skills without having to deal with problems related to other business problems. Otherwise, to be easily manipulated by business experts all kinds of models use a specific modeling language that could be assimilated to a Domain Specific Language. As the computers are still unable today to understand plain English, the modeling languages used in our models are defined as a balance between ease of understanding by domain experts and their ability to be processed by a computer. Models in PERCOMOM are grouped in 5 categories as presented in Figure 5.

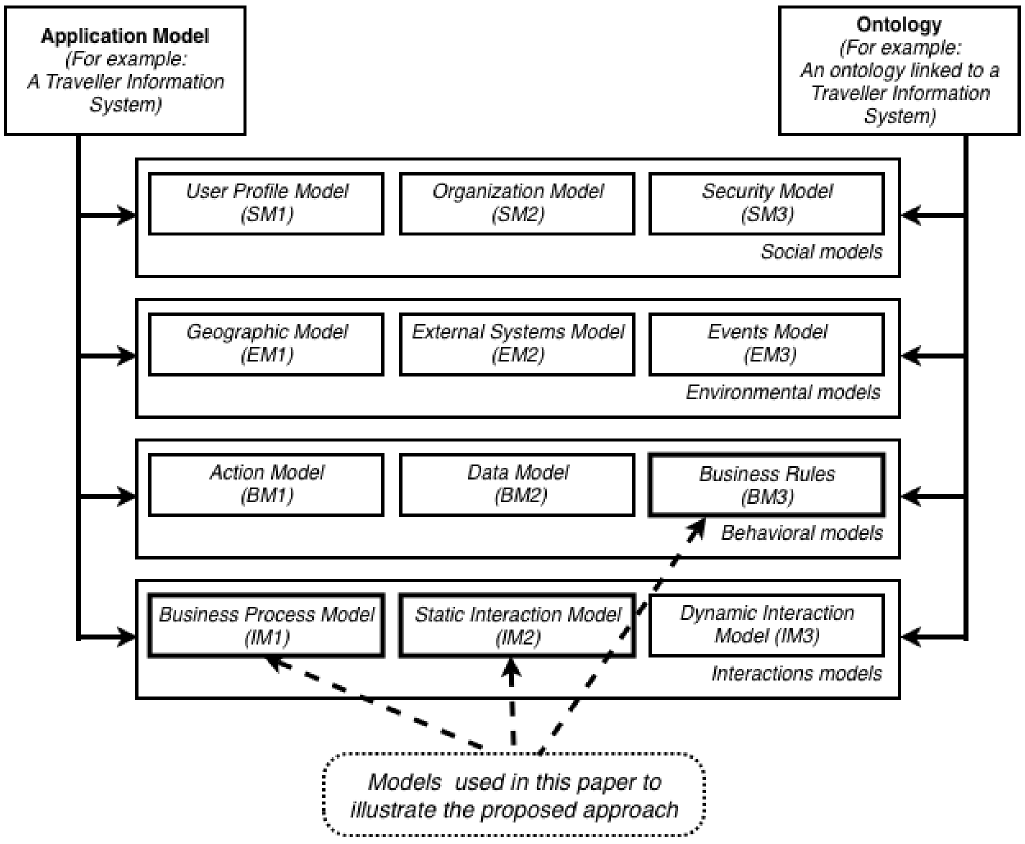


Figure 5. Conceptual models in PERCOMOM

Although PERCOMOM is not a central focus in this article, presenting a brief overview the PERCOMOM models is useful for understanding the approach proposed in this article. The following categories of models are considered in the current version of PERCOMOM:

- *Cross models:*

The ***application model*** makes it possible to define the properties and elements that are specific to the application, such as the name of the application or the default language used to display the contents to the user. For each application, we associate an application class defined within an application ontology, contained in the complete ontology. This application class makes it possible to use the concept of the application range within the other models. In a simplified way, this allows each application to be associated to one or more application "categories". Thus, an application providing information about train schedules could be associated to the generic category, "transport information", whereas an application providing information about TV programs would be associated to the generic category, "leisure-activity information". Within the other models, it is thus possible to define validity constraints according to the application categories and/or to associate elements and/or behaviors according to the membership of the application to such or such application "category". On the level of the contextual adaptation of the applications, this defining of validity constraints allows different behaviors to be defined for applications according to their membership category, with respect to modifying their use context. To give an example, if validity constraints were so defined, it would be possible to consider that the adaptation of the ambient luminosity would be done only for the applications in the generic "transport information" category and not for the applications in the generic "leisure-activity information" category.

This ***ontology***, which is in fact a business domain ontology, makes it possible to handle only business domain concepts in the models, thus providing a total abstraction with the data that will be used in the concrete application. This ontology also makes it possible to use a single vocabulary and semantic for all the people involved in the modeling of the application, which provides semantic coherence between all the applications. This coherence should facilitate the comprehension of the applications and their maintenance over time.

- *Social models:*

  The social models model the users using three different ways:

  The ***User Profile model (SM1)*** manages all information specific to the user: for example, name or date of birth, but also user preferences or the history of the user's interactions with the application. If this specific information is structured hierarchically, the preferences are associated to the business ontology, thus making it possible to associate one preference level to each class of the ontology but also to each individual in the ontology. For example, users can indicate that they would prefer to take the bus rather than to go by foot (a class preference) and that when they take the bus, they prefer the transportation company X rather than the transportation company Y (an individual preference).

  The ***Organization model (SM2)*** represents the social structure of the human society in which the users evolve/move, allowing the social links between the various application users to be highlighted by defining the hierarchical relationships between the user groups (e.g., "directed by", "managed by", "belonged to"). This in turn allows specific rules to be defined in the other models according to the user's social group membership. SM2 makes it possible to model several different human structures: for example, a structure defining the organization of external customers in organized social groups or a structure defining the hierarchical organization of an industrial or commercial business. Each user can be a part of one or more different structures and, within each structure, can belong to one or more different social groups. Thus, in an application in the domain of public transportation, a single traveler can belong to several groups: student, tourist, etc.

  The ***Security model (SM3)*** defines the user roles for each application according to the Organization model, thus allowing access to be authorized or forbidden to various parts of the application according to the user roles that have been defined. It is important to note here that, though both are defined in the social models, there is difference between *user groups*, which influence the behavior of the applications and the information reported back to the user, and *user roles*, which control user access to the different parts of an application.

- *Environmental models:*

  The environmental models model the elements of the external environment of the application that could interfere with its operation:

  > The ***Geographical model (EM1)*** models the geographical environment in which the application evolves/moves. This model is defined through an ontology of geographical classes (e.g., Continent, Country, State, Town/City, Street, House, Office, Desk) connected to one another through properties of inclusion and exclusion and a database containing the individuals associated with each geographical class, for example, the country of France or the continent of North America. Each individual is associated to a geographical definition compatible with GIS (Geographic Information System) cartographic systems.

  > The ***External Systems model (EM2)*** models the external systems with which the application can communicate (e.g., printers, software packages). To do so, it defines the external systems as black boxes with inputs and outputs, command elements and either a synchronous or asynchronous relationship to the application. In order to take into account the fact that external systems (e.g., a printer) are not always available depending on the localization of the user, we associate each external system with one or more membership classes, using an external system ontology. Each class groups external systems that behave identically (e.g., printers), making it possible to define associations either with specific external systems or with general classes of external systems within the application. Thus, when a user needs to print information while traveling, the information will be printed on the printer that is physically the closest and not on a specified printer. To allow this, it is necessary to define the possibility of a geographical and/or temporal validity clause for each external system, which permits up-to-date knowledge of a user's valid external systems, depending on the user's location, the application, the time and the date.

  > The ***Events model (EM3)*** defines all the events that can be managed by the applications. Each event has a unique name, an applicative range indicating whether or not it is valid for one or more application classes, a social range indicating the social groups for which an event is valid, a geographical range indicating its zone of geographical validity using information from the Geographical model, and a temporal range indicating its period of validity defined in minutes, hours, days. For instance, a technical problem preventing the subway from circulating on a subway line could generate an event whose the lifespan would be related to the duration of the technical problem. This event could be used to announce the problem to the users and as information for a route calculation system that could offer substitute routes to the user.

- *Behavioral models:*

  The behavioral models model the elements that influence the application's behavior but are not related to human-computer interactions.

  > The **Action model (BM1)** allows series of business tasks to be modeled without any interaction with the user. Each action can have input values that are provided by the user interface and can modify the output values displayed on the user interface. To define the actions, a pseudo specific language derived from PASCAL language is used within each task. An action could be the printing of a user travel card at an information terminal. The user initiates the action, but the various tasks composing the action are carried out automatically without any user intervention.

  > The **Data model (BM2)** uses a mapping file to connect the concepts of the business domain ontology to a conceptual model of the entity-relationship data. In order to manage several different data sources according to their context, it is possible to associate each data source to a geographical and temporal validity clause, as well as to an application validity clause, which allows the different databases to be used not only with different contents but also with different structures depending on the contextual constraints of the application. For example,

the same application for consulting bus schedules could be connected to a specific database in city A and to another database in city B without changing the application model.

The **Business Rules model (BM3)** permits certain parts of the business logic to be taken out of the business processes, thus making the processes easier to modify. The OMG [75] and the Business Rules Group [76] defined business rules as: "[...] *a statement that defines or constrains some aspect of the business. It is intended to assert business structure or to control or influence the behavior of the business. The business rules that concern the project are atomic - that is, they cannot be broken down further*". We defined three types of business rules in PERCOMOM: *validation rules* whose evaluation returns a Boolean value (true or false); *selection rules* whose evaluation returns a value that can be a character string, a date or a number; and *action rules* whose evaluation triggers the execution of a particular action. The definition of these rules can involve all of the elements defined in the models presented previously. For example, a validation rule could be defined to govern the distribution of ads on the user platform; this rule would make it possible to know whether or not a person was a senior citizen, defined as a person over 60 years of age who does not have a job. In

- *Interaction models:*

The interaction models model the interactions between the user and the computer system:

The ***Business Process model*** (IM1) defines, for each business goal, all the tasks that must be accomplished to achieve this goal, within the limitations described by Dijkman et al. [77]. These tasks can be interactive tasks (e.g., inputting data, selecting options), system tasks (e.g., closing or opening a group of interaction elements), or action tasks (e.g., the tasks defined in the action model). Based on the information contained in all the other models, IM1 defines the order of the tasks to be carried out as well as the constraints associated with them. For each business process, it is possible to associate the geographical, temporal, social and application validity constraints that will permit several different business processes to be defined for the same business goal, though only one business process can be active at the same time for a given user and a given context.

The ***Static Interaction model*** (IM2) represents all the elements proposed to the user for interacting with the application. Each model represents either a logical group of interaction elements or a physical group of interaction elements. A logical group of interaction elements is composed of elements whose interaction is linked logically, and thus could not be separated when they are presented to the user (e.g., two different data areas that can be presented on the same screen on a PC and on two screens on a PDA). A physical group of interaction elements is composed of elements that cannot be physically separated when they are presented at the user (e.g., a data entry form).

The ***Dynamic Interaction model*** (IM3) defines how the various elements of each static model interact temporally and spatially. This model fuses a static model with all the business process models that use this static model at one time or another, taking into account only the elements related to the interactions. This model serves as a validation model, allowing the global functions of a specific static model to be visualized. For example, for a Web application, a static interaction model would correspond to the Web page and the dynamic interaction model would correspond to all of the interactions that could be executed on and from this Web page, though each business process may not always use all the possibilities of interaction for each Web page.

The interaction models, especially IM1, rely on all the other models for their definition. Thus, in a business process model, it is possible to define a particular behavior based on location, social group membership or user profile. Similarly, it is also possible to define a particular task using business rules, actions or external systems. In addition, it is possible to manage application events by defining business processes that do not necessarily require user interaction.

We used 14 types of models in order to have a good balance between the necessity of defining models appropriate for the different kinds of business experts involved in designing an application and the necessity to obtain something that can be managed at the conceptual level. This balance was validated during our work with business experts on applications in the field of transportation [16]. If the small number of business experts involved did not allow to draw a definitive conclusion, it nevertheless served to validate the ease of handling of the different types of models used in PERCOMOM[1]. These first tests also identified the important role of a coordinator who must be able to:

- Have a global view of functional problems treated by the application
- Define types of models that are necessary or not to model the application
- Distribute the different types of models between business experts
- Help business experts in creating models
- Check the integration of different models created by the business experts
- Make the link between business experts

The coordinator's role is therefore a key role in PERCOMOM. And much of the success of the modeling phase is based on him or her. If the coordinator must be a modeling expert, this person must also have good communication skills and, more especially, must have knowledge about all aspects covered by the application. Otherwise, the coordinator will not be able to distribute models between business experts and validate the relevance and completeness of the models created. In the present state of the research, this task is performed manually. In the future, we intend to automate it, to some extend, to facilitate collaboration between the different types of business experts.

In this paper, PERCOMOM is used as an example of a MDE modeling method, and thus the concepts proposed here can be used in other MDE methods.

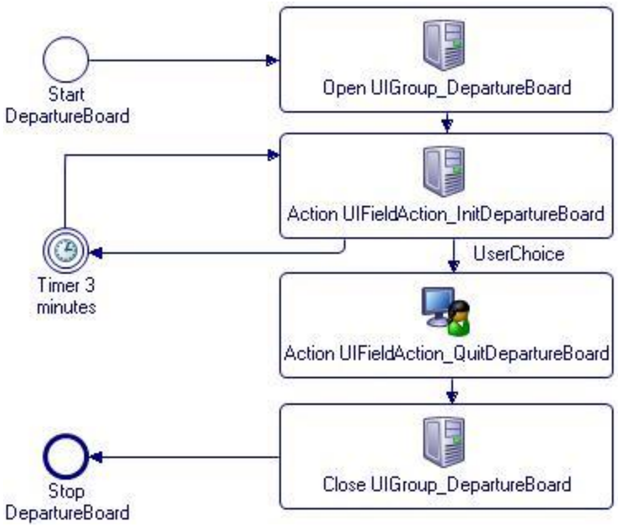
3.2 Taking context into account in PERCOMOM

Before examining how PERCOMOM manages context in the conceptual models, we want to present, through an example, two of the “Interaction Models” used in PERCOMOM [78]

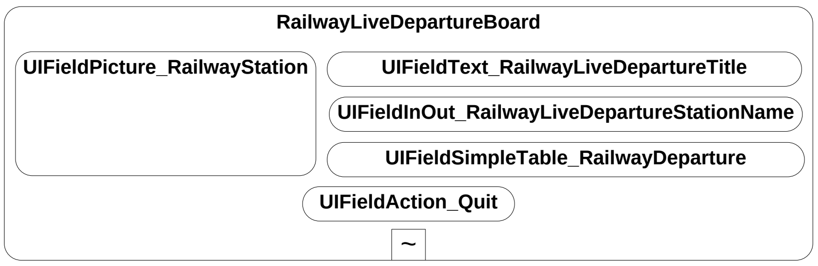
- The *business process* model, whose notation is derived from Business Process Modeling Notation (BPMN).
- The *static interaction* model, whose notation is specific to PERCOMOM and could be seen as a simplified BPMN model.

Figure 6 presents an example of a business process model (a) and a static interaction model (b), defined at the conceptual level. These models are used to create an application that displays the next train departures in a railway station (c). In this example, the conceptual models are created by transportation experts and the application is generated semi-automatically. All the information about the final look and feel of the user interface are managed manually because PERCOMOM is currently only able to generate basic user interfaces.

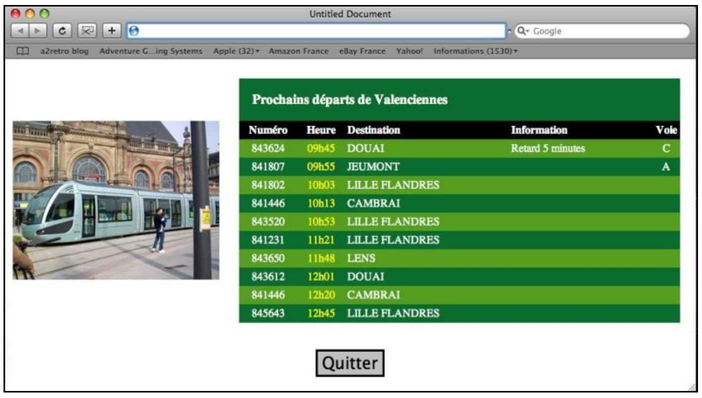
[1] One of the problems of using as many as 14 models in a modeling method is clearly its usability from the perspective of business experts. Up to a certain point, this problem could be resolved by using helper applications and wizards from the main elements of an application. For the details, the business experts will probably have to manipulate each model directly. This problem, which also exists when using the 13 models of the UML 2 notation, is a real difficulty for the Software Engineering community [40] [41] [42] [43] [44] [45]. The answer is not easy and could be, on its own, the subject of several papers, going far beyond the scope of this paper.


Start DepartureBoard
Open UIGroup_DepartureBoard
Action UIFieldAction_InitDepartureBoard
UserChoice
Timer 3 minutes
Action UIFieldAction_QuitDepartureBoard
Close UIGroup_DepartureBoard
Stop DepartureBoard

(a)


RailwayLiveDepartureBoard
UIFieldPicture_RailwayStation
UIFieldText_RailwayLiveDepartureTitle
UIFieldInOut_RailwayLiveDepartureStationName
UIFieldSimpleTable_RailwayDeparture
UIFieldAction_Quit
~

(b)



Prochains départs de Valenciennes

Numéro	Heure	Destination	Information	Voie
843624	09h45	DOUAI	Retard 5 minutes	C
841807	09h55	JEUMONT		A
841802	10h03	LILLE FLANDRES		
841446	10h13	CAMBRAI		
843520	10h53	LILLE FLANDRES		
841231	11h21	LILLE FLANDRES		
843650	11h48	LENS		
843612	12h01	DOUAI		
841446	12h20	CAMBRAI		
845643	12h45	LILLE FLANDRES		

(c)

Figure 6. Example of a business process model (a) and a static interaction model (b) that allow a web interface, used to display the next train departures in a railway station, to be generated for a French-speaking traveler/user of the transportation information system (c)

It is interesting to note here that the use of BPMN notation overcomes some problems of comprehension models that are generally associated with the use of UML [40] [41]. Thus, the business process model is presented as a series of tasks to be akin to a workflow application in which each task is described in sub-tasks and can be performed either by computer or by a human being. This mix of manual tasks and computing tasks allow a business domain expert to express a problem completely in terms of knowledge he/she has on it without being limited by a specific level of granularity on the definition of each task. So if the concerned expert does not know the exact content of a specific task, it can probably be described by another expert job that might not need to know the content of the other tasks. The use of BPMN notation can also overcome the challenge issued by the UML to be too close to the object oriented approach. Indeed, in BPMN models, the user does not manipulate objects in terms of computer but only business tasks. Finally, as BPMN has been designed to be used by business users, its ownership and its use require little computer skills. This allows, in theory, business experts to use more quickly and more easily this type of modeling than UML.

In this paper, we only deal with how the context can be managed simply at the conceptual level; we do not deal with how context could be managed during run-time. (Although, as a representative MDE approach, PERCOMOM offers a general framework to explain context management during run-time). In PERCOMOM, context is managed through restrictions [79], which are specific business rules, returning a boolean value, that allow one or more contextual elements to be evaluated simultaneously. These restrictions are used in the business process models and in the static interaction models to define specific behaviors based on the result of this evaluation.

In PERCOMOM, a restriction is defined as a statement, which is either *true* or *false* and is composed of one or more other statements linked together by logical connectives (NAND, AND, OR, NOR or XOR). The smallest kind of statement (a statement without any connective) is called a literal. In terms of context, a literal is mainly defined from information in the conceptual models. Each literal is defined as a logical expression composed of a left member, generally a property of a conceptual model, a comparator and a comparator value used for the comparison. For example, to define a restriction called *UserIsInParis* that is only true if the user is in Paris, information from the User Profile Model will be used to determine where the user is (defined by the *UserProfile.Context.CurrentLocation* property) and from the Geographic Model, where the term "Paris" is defined from the geographic point of view. Thus, the restriction is defined as follows:

```
// A restriction is always defined by its name
<restriction name="UserIsInParis">
    // A statement is always defined by its name so that it can be reused
     <statement name="UserIsInParis">
           // A literal is always defined by its name so that it can be reused
          <literal name="UserIsInParis">
              // Definition of the left member of the literal
               <element model="UserProfile" property="Context.CurrentLocation"/>
               // Definition of the comparator to be used (here an inclusion)
               <comparator type="inclusion"/>
               // Definition of the comparator value
               <comparatorvalue model="GeographicModel" value="Paris"/>
          </literal>
     </statement>
</restriction>
```

It is interesting to note here that the language used to express some of the models used in PERCOMOM is an XML-based language. From the technical point of view it can relatively easily integrate the models in different computer's tools and different types of platforms. But this facilitates also the reuse of models though the ease of sending and processing of XML files. Thus, in the example above, the restriction could be used by several different applications and possibly by several different companies if they agree to exchange some of their models. In PERCOMOM, reusing a restriction is possible if the restriction is not directly linked to the application, as it is the case in this example. But it also involves a limitation, in that the elements used for defining the context must be shared between the applications. In our example, the restriction "*UserIsInParis*" could only be used by two applications if they are able to manipulate the same geographical information. One of the solutions for reaching this goal is for the two applications to use the same Geographical Model (EM1).

If, for the moment, these models must be created by PERCOMOM experts with business experts, in the near future we intend to provide tools to manage all the models in a much simpler manner through the use of graphical tools. As of today, only the models IM1 (Business Process Model) and IM2 (Static Interaction Model) have these kinds of tools.

In PERCOMOM, restrictions can be composed of more than one literal and may include other restrictions, which are then considered to be literals. To facilitate the domain experts' use of restrictions, PERCOMOM has three sub-categories of restrictions:

- *Application restrictions* that only take into account contextual information about the application.
- *Geographical restrictions* that only take into account contextual information about the user's location. In our approach, the geographical context allows to define geographical spaces. So, it is possible to define a shop or an office as a geographical space that could be used as a context for an application. In that way, we could even define a specific behavior for an application when it is used outside the office because a geographical context could be used as an inclusive or as an exclusive condition. This allows us to use the notion of geographical context in a broader way.
- *Time restrictions* that only take into account contextual information about time.

It is important to point out here that each kind of restriction is not useful for all kinds of applications, since some applications do not need to be aware of the context. So, depending of the application, it is possible to use only one type of restriction, or two types or all the types.

With this definition of a restriction in PERCOMOM, the question remains how can already existing conceptual models be used to manage new contextual elements in an MDE approach.

The response to this question is: it could be done in one of two ways:

- By implementing an *a priori* mode in which the context elements are taken into account in the initial models of the web application. In this case, the specific behaviors associated to the business process models and to the static interaction models will only be activated only when the web application is able to “physically” take the context information into account. For example, it is possible to design a web application that is aware of the geographic context, but if the users do not have a physical system to geolocalize them, this context awareness is serves no purpose. However, when the users obtain a geolocalization system, the web application models will not need to be modified.
- By implementing an *a posteriori* mode in which the web application models must be modified after the fact to take a new context element into account. For example, the web application was not initially designed to take temporal information into account, but a new development makes temporal context-awareness desirable. In this case, the initial models have to be modified.

In PERCOMOM, from the user point of view, all the information regarding the context is saved inside its profile that could be compared to a configuration file. The information contained in the user profile are updated periodically at runtime as follows:

- For the application context, the information is updated each time the user launches or quits an application. All applications used are arranged so that the last application used is the first in the list of open applications.
- For the geographical context, information is periodically updated by a specific application that runs on the technical platform used by the user to access it. The geographical context is then defined as the smallest element of the ontology to determine the geographical position of the user.
- For the temporal context, the information on the current context is determined as runtime each time the application needs it. In simplified terms, the current temporal context matches the current time on the technical platform of the user consultation.

This means that, in PERCOMOM, these three elements of the context always have values and can therefore be used in an application through the use of restrictions.

For most applications, the user is, normally, never aware of the current context used by the application. But, as the information regarding the context is saved inside the user profile, it could be, if the use case related to the business process requires it, displayed or it could be changed by the user himself or herself. This could be very useful in some situation where the choice of the current context could not be done automatically by the application with a high level of confidence.

This also means that an application can respond at runtime to information on the application context, the geographical context or the temporal context only if the application has been designed so that it has a particular behavior for one or more specific elements of the context.

3.3 Managing context in PERCOMOM

In the *a posteriori* mode, the first question that arises is related to the choice of the conceptual model that should be modified. If the modification will change the behavior of the application without changing anything in the User Interface (UI), it will be realized in the business processes, according to the decision tree shown in Figure 7.

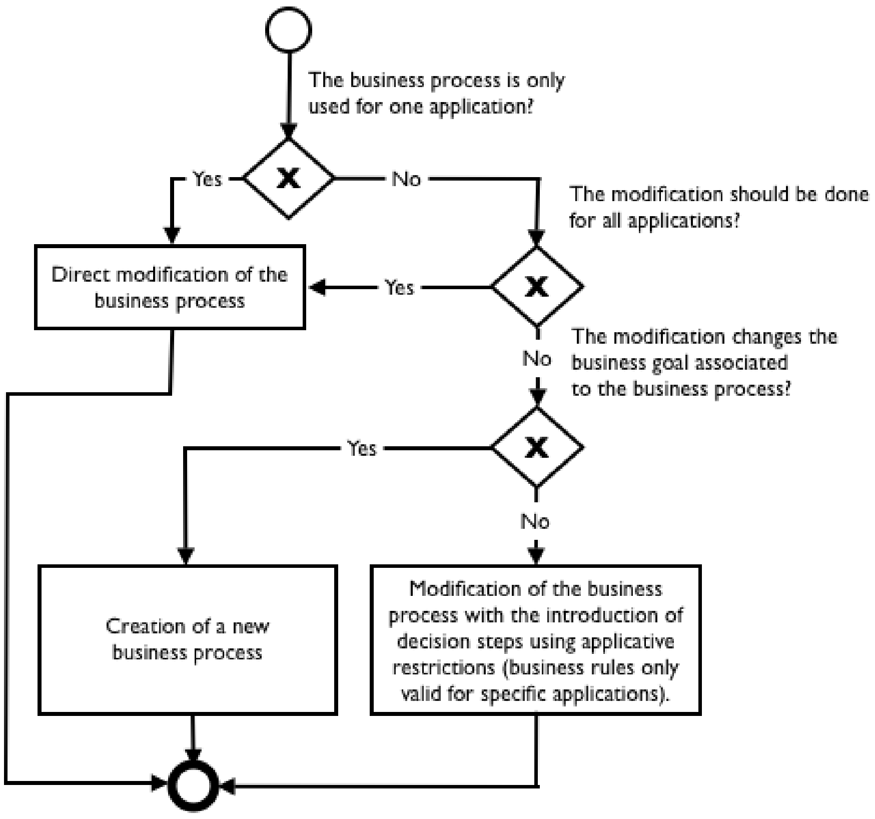

The business process is only used for one application?
Yes
No
The modification should be done for all applications?
Direct modification of the business process
Yes
No
The modification changes the business goal associated to the business process?
Yes
No
Creation of a new business process
Modification of the business process with the introduction of decision steps using applicative restrictions (business rules only valid for specific applications).

Figure 7. Decision tree for modifying a business process model in PERCOMOM

If the modification changes the behavior of the UI, there are two possibilities:

- Modifying the business process models, which supposes that the UI are designed to be *generic*, with the context-awareness being managed by tasks in the business processes.
- Modifying the UI, which allows the business processes and UI to be kept separate, thus avoiding mixing tasks of different natures excessively in the business processes.

This second possibility has the advantage of being coherent with the traditional MVC (Model-View-Controller) approaches to web applications [80], given that, in a simplified way, the *View* can be seen as corresponding to the static interaction model and the *Controller* to the business process model. This separation allows making the business processes as independent as possible from problems directly linked with the UI. It also facilitates the evolution of the UI without necessarily changing the associated business processes.

Since a user interface can be used by several business processes in several web applications, any modification of the static interaction model can result in two situations, depending on whether or not the modifications can be made compatible with all the business processes associated with the static interaction model:

- the modification can be made compatible with all the business processes Compatibility can be achieved by adding new valid interaction elements only for certain business processes or by adapting the properties of the elements contained in the static interaction model. Figure 8 presents an example of a decision tree using a geographical restriction in a static interaction model to change the interaction element properties associated to the user location (a static text value type). By taking the geographical restriction into account, the interaction element can be displayed as a static text that is visible if the user is in France and hidden if the user is not in France.
- the modification can not be made compatible with all the business processes Incompatibility results in the creation of a new UI, which implies a modification of all the business processes associated with this UI, thus leading to a possible creation of new business processes. Figure 9 presents the decision tree used for any modification to the static interaction model.

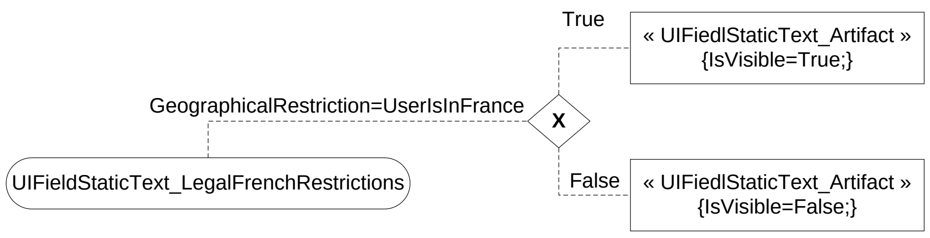

True
« UIFiedlStaticText_Artifact » {IsVisible=True;}
GeographicalRestriction=UserIsInFrance
X
UIFieldStaticText_LegalFrenchRestrictions
False
« UIFiedlStaticText_Artifact » {IsVisible=False;}

Figure 8. Example of use of a geographical restriction in a static interaction model to change the properties associated to the interaction element regarding the geographical localization of the user

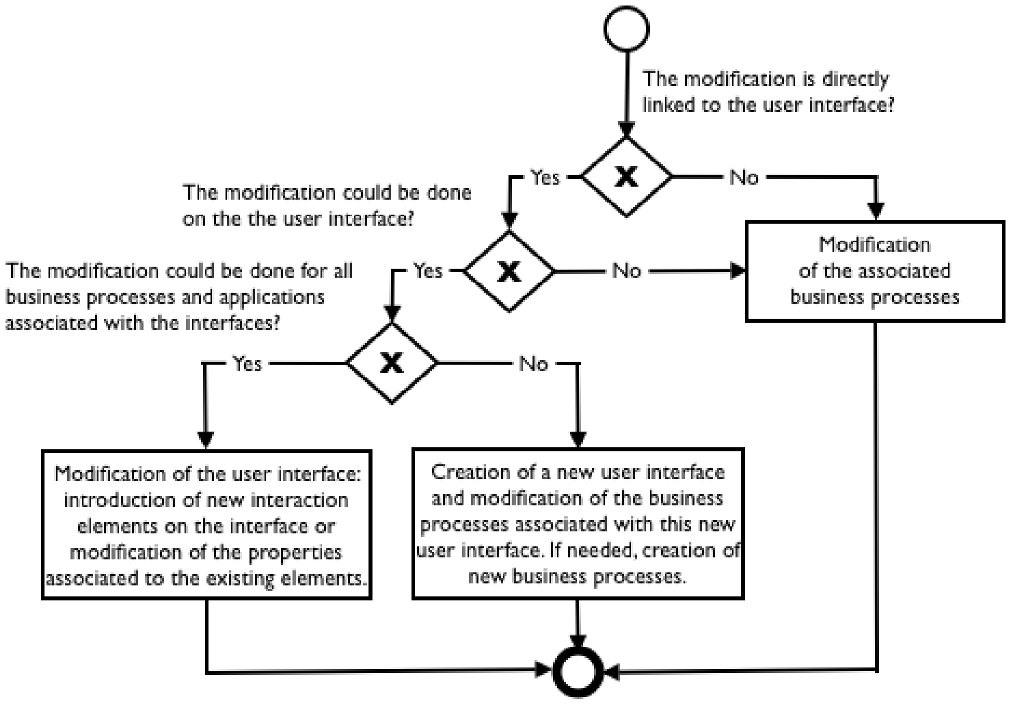

The modification is directly linked to the user interface?
Yes
No
The modification could be done on the the user interface?
Yes
No
Modification of the associated business processes
The modification could be done for all business processes and applications associated with the interfaces?
Yes
No
Modification of the user interface: introduction of new interaction elements on the interface or modification of the properties associated to the existing elements.
Creation of a new user interface and modification of the business processes associated with this new user interface. If needed, creation of new business processes.

Figure 9. Decision tree modifying a static interaction Model in PERCOMOM

In PERCOMOM, the adaptation of the models being in the posterior mode, it is not performed during the execution of the program but in the modeling phase. This means that taking into account new elements of the context or the adaptation of existing processes need to go through a modeling phase that will use both models of decision trees defined above. To validate the adequacy of the good choices made during the modeling with the actual use of the application and especially to consider some type of change in use of the application, field studies should be conducted. These, through observation of practice, will:

- validate or invalidate the choices made during the modeling phase,
- detect new user needs in relation to issues of use of the application related to the context.

This means that in PERCOMOM, a user application can respond to the context only if this reaction has been defined during the modeling phase. So, regarding the context, the evolution of the business processes could be see as a continuous loop between a modeling phase and, what we could name, an execution phase (handling of the application by end users).

Now that they have been defined with their scope of use, we will use these two decision trees in concrete examples in the next section.

4. Case study in the field of transportation

Our research was done as part of the *Viatic.Mobilité* project [81], in association with the French *i-Trans* competitiveness pole dedicated to transport [82]. A total of fifteen partners, both private companies and research laboratories, were involved in this wide-range project. The project goal was to set up new interactive services for public transportation users, who were considered as nomadic users of information systems. These services are designed to provide not only mobility information, but also general information, by logically articulating all the potential informational needs of travelers. They are also designed to be accessible throughout the user's journey (e.g., at home, in the stations and on board the vehicles (train, bus, tram)) through the use of web applications.

The concrete examples given in this section came from this project. The first example shows how the business processes can be modified to take context into account. The second example shows how it is possible to manage context within the static interaction models. The third example presents a web application in the field of public transportation, which uses the two types of context adaptation to provide an enhanced user experience.

The models used in this paper were designed with the help of business experts in order to validate not only the model’s contents but also to validate the approach proposed in this paper. Since the tools were not finished when the models were designed with the help of business experts, they were created on paper and then introduced in the first version of the tools to validate them. In order to allow the reader to concentrate on the main aspects of managing the context inside models, we present in detail only the two main PERCOMOM models from the business experts’ point of view: the *business process* and the *static interaction* models.

4.1 Adapting business processes

By applying the decision trees defined in section 3.3, this first example shows how it is possible to modify a business process to make it context-aware.

Let us suppose that, for an English traveler, we need to modify a business process used to print an e-receipt for a payment. Depending the nature of the web application used (service, transport or leisure), the modification will allow different e-tickets to be e-printed. The business process is composed of a succession of what are called *actions* in PERCOMOM: the first one, called *Action_PaymentValidation*, makes it possible to validate the e-payment and to prepare the text to be e-printed on the e-receipt; the second one, called *Action_Receipt*, allows to e-print the e-receipt (Figure 10).

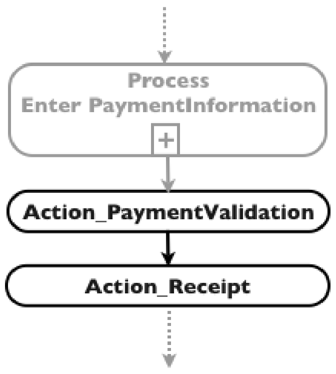


Figure 10. Extract of the global business process associated to e-printing an e-receipt payment

To make this process context-aware, it is possible to apply the decision tree specifically designed for modifying business processes. To do this, it is necessary to determine whether or not the business process is used by several applications. Because the printing of an e-receipt is supposed to be identical for all web applications, the response doesn't matter. Then, it is necessary to determine whether or not the modification to be executed should be done for every web application. This time, the response is positive because if the modification is to make the business process context-aware, it does not change the business goal associated with e-printing process. Thus, to modify this business process, two possibilities exist:

- Modify *Action_Receipt* to make it context-aware and allow it to print two types of e-receipts depending on the type of the web application, or
- Create specific tasks for each e-payment type for each type of web application.

From an MDE perspective, the second solution is preferable because it allows the creation of distinct business sub-goals that can be reused in other business processes. Thus, an action called *Action_TransportReceipt* will be created to add a specific text for the "transport" applications and an action called *Action_ServiceReceipt* will be created to add the specific text for the other applications. One or the other action will be selected after evaluating an application restriction identifying the nature of the user's application (Figure 11).

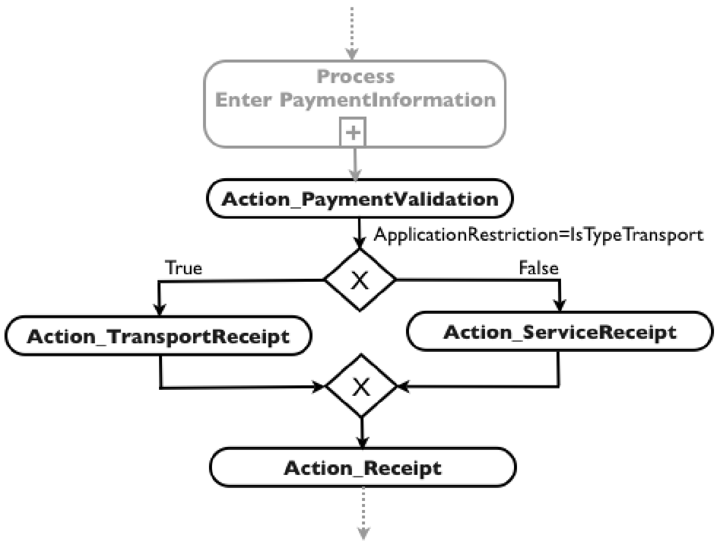
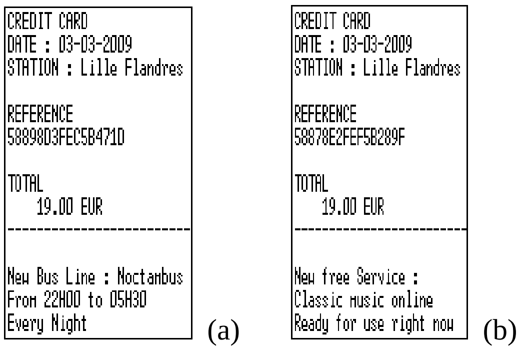


Figure 11. Adapting the business process shown in Figure 9 to the context of use (i.e., application type)

Figure 12 provides examples of e-receipts e-printed by executing the two actions.


```
CREDIT CARD
DATE : 03-03-2009
STATION : Lille Flandres

REFERENCE
58898D3FEC5B471D

TOTAL
    19.00 EUR
-------------------------

New Bus Line : Noctambus
From 22H00 to 05H30
Every Night
```
(a)

```
CREDIT CARD
DATE : 03-03-2009
STATION : Lille Flandres

REFERENCE
58878E2FEF5B289F

TOTAL
    19.00 EUR
-------------------------

New free Service :
Classic music online
Ready for use right now
```
(b)

Figure 12. Example of an e-printed e-receipt resulting from the execution of the action Action_TransportReceipt (a) and an e-receipt resulting from the execution of the action « Action_ServiceReceipt » (b)

It is interesting to note here that the sub-process outlined in our example used to validate a payment and print a receipt can be defined as a business sub-process re-usable inside other business processes. This sub-process then uses the information necessary for its proper operation (amount paid, payment type, and on on) using information supplied by the processes and sub-processes that have called it. For this, we use an interesting property of the business process that lies in the ability of each process and sub-process, receive and exchange information with other processes and sub-business processes [46]. This means that each process and sub-business process is always defined by the tasks and sub-processes that compose it, but also by the information that can be exchanged with other processes and sub-processes; and this by specifying for each information, if it is an input only, an output only or an input/output information.

Another advantage of using sub-processes and business processes in the context of reusability is that it is possible to give them some genericity for use in various applications. Thus, in our example, the sub-process of validating an e-payment and print an e-ticket could be used for an application to purchase subscriptions for transport but also for an application to purchase concert tickets.

Modifying a web application to make it context-aware can require modifying the static interaction models of the application. In this case, as explained in the following section, the decision tree for the static interaction models will be used.

4.2 Static interaction model adaptation

For this example, let us suppose that a transportation company decides to set up an information space on its website in order to identify all the delays on its transport lines. This information space service will be accessible in the morning from the general menu of the website. This modification supposes changing the general menu's user interface (UI) to provide access to this service and then adapting the business processes associated with the application so that the application can deal with this new service.

For the moment, we limit our discussion to the changes concerning the UI. The setting up of the information space will necessitate the creation of new interaction element, which could be an option in a selection menu to provide access to the new service. This new element will allow a business process specific to the new service, called *ShowDelay*, to be executed. This interaction element, called *UIAction_ShowDelay*, will be visible only in the morning. To accomplish this, the value of the *IsVisible* property is changed according to the result of the evaluation of the restriction, *IsMorning*. To simplify the notation, the restriction created by using a particular property called *IsVisibleRule* is used. This property allows the interaction element to directly assign the value of the *IsVisible* property, based on the result of the business rule evaluation (Figure 13).

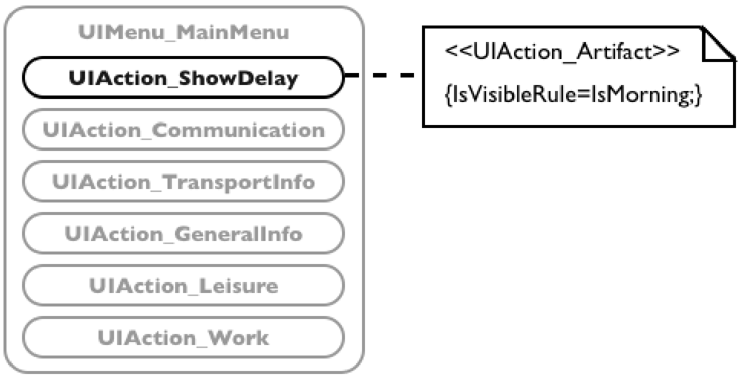


Figure 13. Association of a visibility rule to the *UIAction_ShowDelay* interaction element

It is then necessary to modify the business processes associated with the application's static interaction model to allow the business process associated with the new service to be executed. This is done by adding a new action to the business process associated with selecting an option in the general menu (Figure 14).

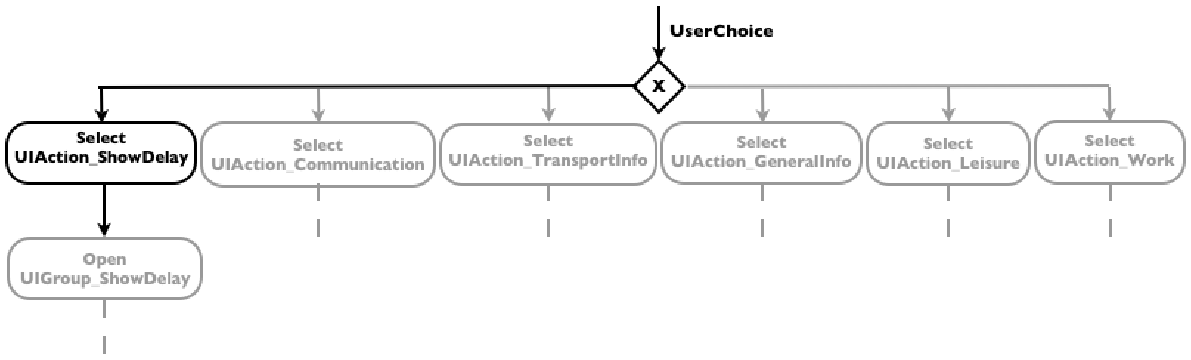

UserChoice
X
Select UIAction_ShowDelay
Select UIAction_Communication
Select UIAction_TransportInfo
Select UIAction_GeneralInfo
Select UIAction_Leisure
Select UIAction_Work
Open UIGroup_ShowDelay

Figure 14. Extract of the business process associated with the application's static interaction model

After taking the modification into account at the UI level by using the *IsMorning* restriction, it is necessary to know whether or not it is also necessary to modify all the application's business processes. The decision tree tells us that this is not necessary because the modification, which is valid for all the business processes, only concerns the user interface (i.e., access to the option, or not, depending on the time of day).

Now, defining a different behavior in the morning and in the afternoon would require that the menu option be always accessible, whatever the time of day. Therefore, it would be necessary to define specific business sub-processes based on the temporal context. According to our decision tree, this would be a modification of the static interaction model not directly related to the UI, and thus require a modification of all the associated business processes.

4.3 Adapting public transportation web applications

One of the functional problems that has been raised was that web applications, because they are used worldwide, should be able to offer information and services related to the user's location. This third example will illustrate how the adaptations to business processes and static interaction models could be combined to create a web application that can react to its own context of use.

In our example, which is quite simple, two specific services are offered to users if they are located in or around Lille (France). The first one will allow them to buy merchandise through a web application and pick it up at a specific corner of the Lille train station. The second one will give users the possibility of consulting information about nearby shops that are less than one kilometer from their location. Because the main service provided by the application is to give travel information, according to our decision tree, the two new services correspond to two new business processes. They could be launched through a user choice on the main menu of the web application, which means that the user interface needs to be modified too. A geographic restriction is added to the *IsVisible* property of the *UIAction* element (represented as a menu option in Figure 14) that will give access to the two business processes only if the user is near Lille.

To show the users some specific advertisements that are directly linked to the user's location, a specific *UIElement*, an *UIAdvert*, is attributed a different behavior according to whether the user is near Lille or elsewhere, since the modification is not directly linked to a business process but to the user interface. In both cases, the advertisements are randomly chosen from a pool of valid ads in terms of the user location and will change every 20 seconds if the user is near Lille and every 60 seconds if the user is elsewhere (Figure 15).

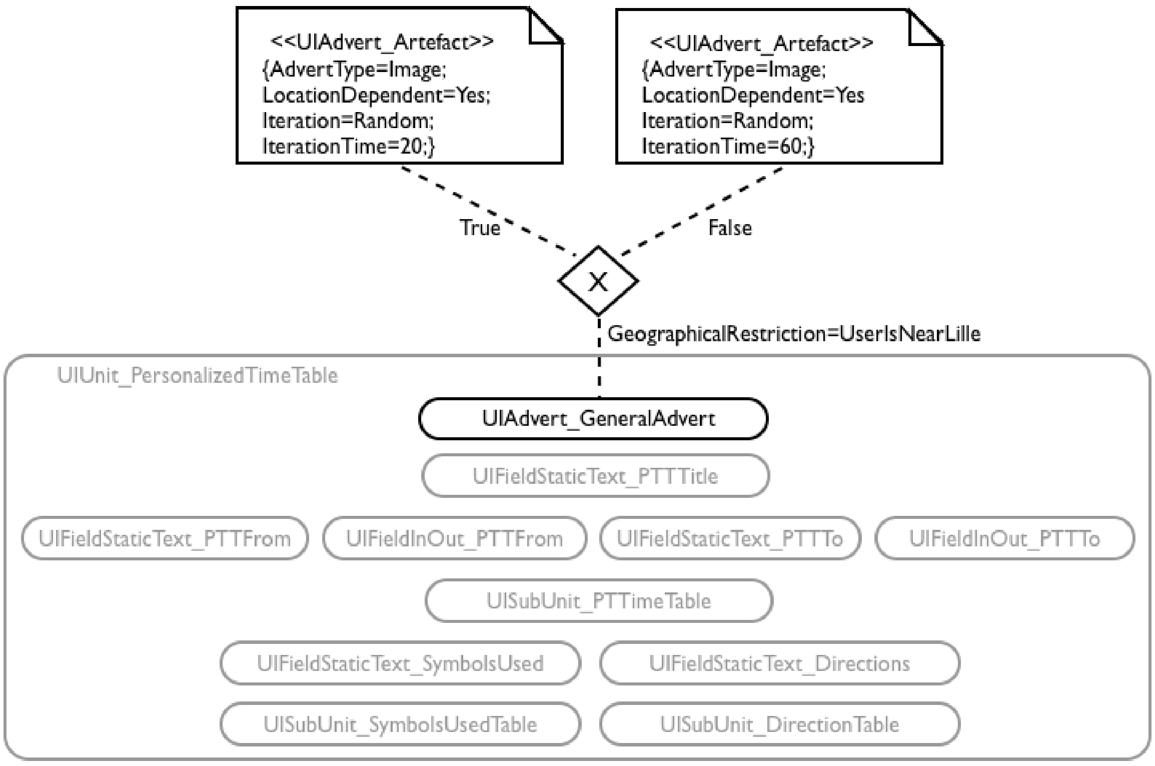
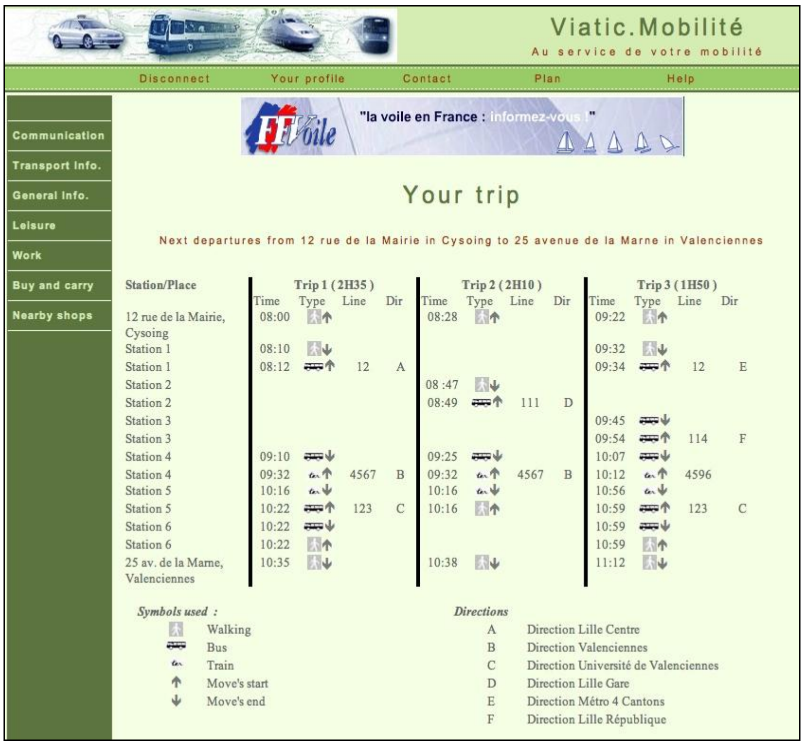


Figure 15. Use of a geographic restriction associated with the *UIAdvert* element, attributing a different behavior based on the user's location (near Lille or elsewhere). Only the main information is shown in this figure.

Figure 16 present two views of the application: one when the user is near Lille (a) and one when the user is in London (b).



Your trip

Next departures from 12 rue de la Mairie in Cysoing to 25 avenue de la Marne in Valenciennes

Station/Place	Trip 1 (2H35)				Trip 2 (2H10)				Trip 3 (1H50)			
	Time	Type	Line	Dir	Time	Type	Line	Dir	Time	Type	Line	Dir
12 rue de la Mairie, Cysoing	08:00	🚶↑			08:28	🚶↑			09:22	🚶↑		
Station 1	08:10	🚶↓							09:32	🚶↓		
Station 1	08:12	🚌↑	12	A					09:34	🚌↑	12	E
Station 2					08 :47	🚶↓						
Station 2					08:49	🚌↑	111	D				
Station 3									09:45	🚌↓		
Station 3									09:54	🚌↑	114	F
Station 4	09:10	🚌↓			09:25	🚌↓			10:07	🚌↓		
Station 4	09:32	🚆↑	4567	B	09:32	🚆↑	4567	B	10:12	🚆↑	4596	
Station 5	10:16	🚆↓			10:16	🚆↓			10:56	🚆↓		
Station 5	10:22	🚌↑	123	C	10:16	🚶↑			10:59	🚌↑	123	C
Station 6	10:22	🚌↓							10:59	🚌↓		
Station 6	10:22	🚶↑							10:59	🚶↑		
25 av. de la Marne, Valenciennes	10:35	🚶↓			10:38	🚶↓			11:12	🚶↓		

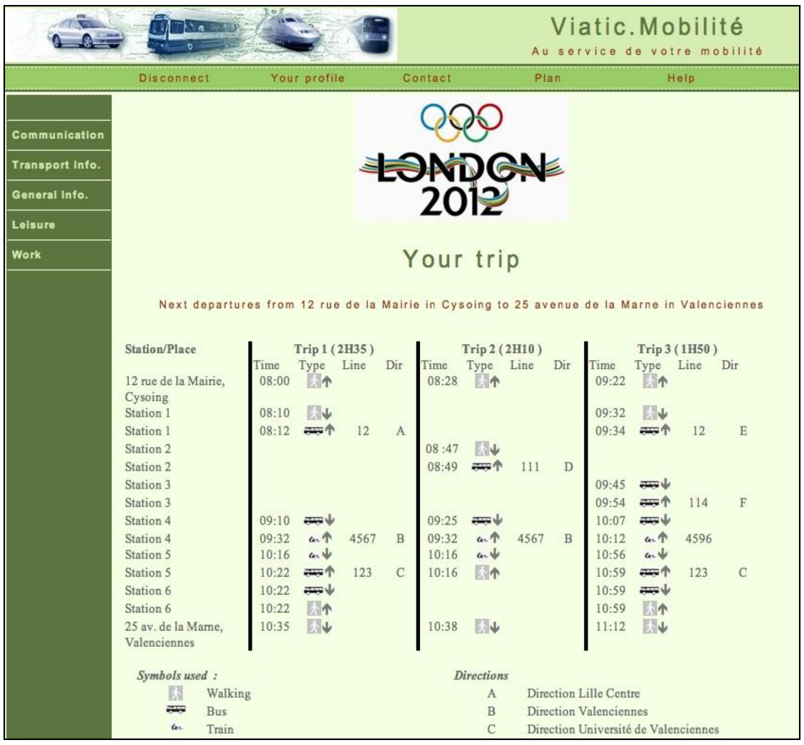
Symbols used :

🚶	Walking
🚌	Bus
🚆	Train
↑	Move's start
↓	Move's end

Directions

A	Direction Lille Centre
B	Direction Valenciennes
C	Direction Université de Valenciennes
D	Direction Lille Gare
E	Direction Métro 4 Cantons
F	Direction Lille République

(a)



Your trip

Next departures from 12 rue de la Mairie in Cysoing to 25 avenue de la Marne in Valenciennes

Station/Place	Trip 1 (2H35)				Trip 2 (2H10)				Trip 3 (1H50)			
	Time	Type	Line	Dir	Time	Type	Line	Dir	Time	Type	Line	Dir
12 rue de la Mairie, Cysoing	08:00	🚶↑			08:28	🚶↑			09:22	🚶↑		
Station 1	08:10	🚶↓							09:32	🚶↓		
Station 1	08:12	🚌↑	12	A					09:34	🚌↑	12	E
Station 2					08 :47	🚶↓						
Station 2					08:49	🚌↑	111	D				
Station 3									09:45	🚌↓		
Station 3									09:54	🚌↑	114	F
Station 4	09:10	🚌↓			09:25	🚌↓			10:07	🚌↓		
Station 4	09:32	🚆↑	4567	B	09:32	🚆↑	4567	B	10:12	🚆↑	4596	
Station 5	10:16	🚆↓			10:16	🚆↓			10:56	🚆↓		
Station 5	10:22	🚌↑	123	C	10:16	🚶↑			10:59	🚌↑	123	C
Station 6	10:22	🚌↓							10:59	🚌↓		
Station 6	10:22	🚶↑							10:59	🚶↑		
25 av. de la Marne, Valenciennes	10:35	🚶↓			10:38	🚶↓			11:12	🚶↓		

Symbols used :

🚶	Walking
🚌	Bus
🚆	Train

Directions

A	Direction Lille Centre
B	Direction Valenciennes
C	Direction Université de Valenciennes

(b)

Figure 16. Example of a public transportation web application that takes the geographic context into account, showing different advertisements and offering different services depending the user's location: (a) near Lille (France) or (b) in London (England)

This third example allows us to show how taking the context into account could have an impact both on the business processes and on the user interfaces associated with the application. This example demonstrates that the process is coherent, given that the modifications done on the UI level to add the new business processes are quite simple: just linking a geographic restriction to the *IsVisible* property of the two *UIAction*, which allows the new business processes to be launched.

Using our method, the application could evolve very smoothly if user needs evolve:

- The content of the business processes could change without modifying the UI (at least for the UI showed in this example).
- It would be very easy to change the geographic restrictions associated to the *IsVisible* property if needed.

Of course, there could be a more complex evolution, which would require changing both the business processes and the UI. But, if users needs were correctly determined during the application design phase, complex evolutions should be quite rare.

Another advantage of our method is that it allows tiny links between business processes and UI to be created, which should make it easier to reuse both the business processes and the user interface. Reusability is one of the main goals of the MDE approach [85]. In the example shown in Figure 15, the Static Interaction Model (IM2) associated to the web page, called *UIGroup_ViaticApplication*, is composed of three sub-models defined as *UIUnit*. The first one, *UIUnit_MainMenu*, allows all the components of the upper level of the web page to be defined. The second one, *UIUnit_LeftMenu*, allows all the components of the menu on the left of the web page to be defined. The last one, *UIUnitGroup_Content*, is a container associated to a list of *UIUnit* and can only be presented to the user on the *UIUnit* list, depending on the value of the *UIUnitGroup*'s condition. The *UIUnit* presented in Figure 14 is one of the *UIUnits* in *UIUnitGroup_Content*. Figure 17 shows the model associated to *UIGroup_ViaticApplicatio*n.

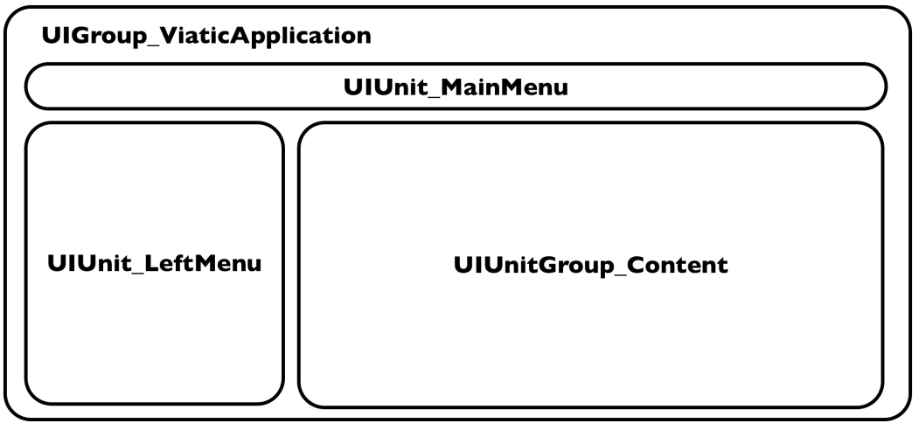


Figure 17. The Static Interaction Model associated with the web page presented in Figure 15

The different elements of the *UIGroup_ViaticApplication* could be reused for different applications by changing their properties and adding different restrictions. If needed, the business processes can easily be reused in other applications with different UIs. In our example, *UIUnit_MainMenu* is not strongly linked to the application: each element in the menu is generic (Disconnect, Your Profile, Contact, Plan, Help) and the pictures are linked to the application context. This allows it to be reused for different applications without needing to redefine a new upper menu each time.

4.4 PERCOMOM compared to similar approaches

In the current stage of our work, we mainly focused on conceptual models for modeling interactions and more specifically the business process model (IM1) and static model of interaction (IM2). For both models, PERCOMOM offers a number of elements distinct from existing approaches.

As the number of approaches to model an interactive application is rather large (see paragraph 2.2), we decided to select only the two most relevant approaches related to the issues addressed by PERCOMOM and UML 2.0, which is now a standard as well as the most widely used language for modeling applications. Thus, in the remainder of this section, we will compare PERCOMOM to aspects directly related to HCI (IM1 and IM2 models) and to its ability to take the context into account. The comparison will be done with:

- UML 2.0, the reference notation currently promoted by the OMG for modeling applications.
- UsiXML, a language for defining interfaces for interactive applications and for which there are tools and modeling methods to deal with many aspects of an application.
- CUP 2.0, an extension that uses "UML Profiles" and UML 2.0 notation, which can be used to model contextual interactive applications.

We used CUP 2.0 for our comparison as we did not find UML profiles that had been validated by the OMG for modeling business processes, HCI or the context.

As shown in Table 1, the main differences between the four approaches are:

- PERCOMOM offers support for business processes from the viewpoint of interactive applications (IM1 models) and allows considering the reuse of models within an MDE approach. Regarding the modeling of interfaces, PERCOMOM offers almost the same advantages as UsiXML allowing more access to data through the use of business concepts that allows a true abstraction from the data handled in the interface.
- UML 2.0, in its standard version, is not appropriate for supporting business process models or the templates of tasks associated with interactive applications; this is only possible by using specific UML 2.0 profiles. Neither is it appropriate for modeling the interfaces of an interactive application.
- UsiXML, through the use of CTT, is able to take over the task models associated with interactive applications. By cons, it was not created with the reuse of models in mind and does not systematically identify the type of interactions. Otherwise, UsiXML can fully define the interfaces of interactive applications. But, it does take into account only WIMP interfaces (Window, Icon, Mouse, Pointing device). In addition, it allows only direct links to data models without going through a higher level of abstraction, which does not allow a true conceptual approach at the CIM level.
- CUP 2.0, like UML 2.0, is quite linked with the object-oriented approach and for this reason, it cannot be considered as allowing the creation of real conceptual models. Like UML 2.0, it has not been designed to be easily manipulated by business users. On the other hand, CUP 2.0 allows interactive applications to be modeled by using specific UML profiles. It also allows the context to be taken into account to some extent.

Regarding the Business Processes Models (IM1 for PERCOMOM), our comparison shows that UML 2.0 is not the ideal tool for modeling tasks and/or business processes associated with interactive applications and that, in fact, PERCOMOM, UsiXML and CUP 2.0 are very close to each other. One of the great differences resides in PERCOMOM’s management of the business process model (IM1) and its ability to integrate elements from other conceptual models. Thus, in PERCOMOM, a single business process can have inside of itself several different behaviors, depending on the result of the evaluation of a business rule linked to the context, for example.

Concerning the static interaction models (IM2), UML 2.0 is not the most appropriate language for modeling an interactive application, especially at the conceptual level. CUP 2.0, UsiXML and PERCOMOM are very close to each other. However, PERCOMOM adds a number of additional benefits besides the fact that it gives a greater abstraction at the conceptual level. Thus, by using properties for each interaction element and using business rules or restrictions, PERCOMOM allows properties that can react to context to be defined.

	PERCOMOM	UML 2.0	UsiXML	CUP 2.0
Allows conceptual models to be created, which are not linked to technical elements	Yes, as it is fully compliant with the MDA approach	No, since UML is based on an object-oriented approach	Yes, since it is fully compliant with the MDA approach	No, since CUP 2.0 is based on UML 2.0 and thus has the same limitations
Uses a standardized notation for modeling business processes	Yes, by using an adapted BPMN notation	Not without using specific non-standard profiles	Yes, by using CTT	Yes, by using a System Interaction Model
Allows users, social organizations and security aspects to be modeled	Yes, by using the social models	Not without using specific non-standard profiles	Can take into account only a limited model of the user	No
Allows the geographical context to be modeled	Yes, by using a geographical models based on an ontology	Not without using specific non-standard profiles	No. UsiXML does not define a specific model for the geographical context	No. CUP 2.0 does not define a specific model for the geographical context
Allows personalization to be integrated inside the models	Yes, since PERCOMOM was created to model personalized applications [16]	Not without using specific non-standard profiles	No. UsiXML does not integrate the notion of personalization	No. Only the context is taken into account.
Allows HCI to be modeled	Yes, by using Static Interaction Models	Not without using specific non-standard profiles	Yes, by using an Abstract User Interface Model	Yes, by using an Abstract User Interface Model
Handles all types of applications	Yes, since using Static Interaction Model and Dynamic Interaction Model are optional	Yes	Not easily, since UsiXML was specifically developed to handle interactive applications	Yes, by using the UML 2.0 models
Allows user tasks and system tasks to be distinguished	Yes, by using specific keywords	Not without using specific non-standard profiles	Yes, by using CTT	Yes, by using a System Interaction Model
Facilitates the reuse of existing models	Yes, for models not directly linked to an application	Yes, but only for specific models	Only possible for HCI models, since CTT does not allow reusing models	Not specifically
Can be extended	Yes	Yes, by using profiles	Yes	Yes, by using new profiles or modifying existing profiles
Allows the context to be taken into account	Yes, by using restrictions. Can directly use any context element in a restriction.	Not without using specific non-standard profiles	Yes, but only for aspects of the context linked to the technical characteristics of the platform used by the application (e.g., screen size, colors)	Yes, by using a Context Model (i.e., an UML profile). Limited by the fact the context is linked to a class and not to context elements.
Models were designed to be used by business experts	Yes. Each model represents a view of the application adapted to a type of business expert	No.	Yes, but only for the models directly linked to the user interface	No, CUP 2.0, like UML, was not designed to be used by business experts
Offers tools for creating applications	Has only a limited number of tools	Offers a lot of commercial and non-commercial tools	Offers different tools suitable for different application types (see http://www.usixml.org)	Yes, by using UML 2.0 tools that can be extended using UML profiles. CUP 2.0 can only be used to model the application and can only generate them in a limited way
Maturity of the approach	Experimental	Standardized for many years and is widely used	On the way to being standardized by the OMG.	Experimental

Table 1: Comparison of PERCOMOM, UML 2.0, UsiXML and CUP 2.0

As presented in Table 1, in its standard version, UML 2.0 has too many limitations to be a good candidate for modeling context-aware interactive applications, at the conceptual level. PERCOMOM, UsiXML and CUP 2.0 were all designed for modeling interactive applications, which is why they share many similarities.

However, there are differences between the three. A major difference between PERCOMOM and UsiXML and CUP 2.0 lies in its ability to take into account the concept of personalization, which is not a part of UsiXML [16] or CUP 2.0. On the other hand, PERCOMOM was designed with the high model reusability in mind. Each model can be used for several different applications and, if necessary, with different behaviors by using restrictions. UsiXML and CUP 2.0, meanwhile, were mainly designed to handle applications one by one. And, although their models are reusable, they are, to the best of our knowledge, only aware of some aspects of the context. Finally, PERCOMOM, unlike UsiXML or CUP 2.0, was not designed to support only the modeling of user interfaces but was also designed to support interactive applications as a whole. This explains the existence of Social models (SM1, SM2, and SM3), environmental models (EM1, EM2 and EM3), and behavioral patterns (BM1, BM2, BM3) at the CIM level.

4.5 Discussion

These examples, as well as the other case studies carried out during the *Viatic.Mobilité* project [72], allowed us to validate our method for adapting the conceptual models of a web application to the context. The use of decision trees was also validated with the domain experts responsible for modeling the application. Thus, our method provides a solution that could be easily understood and used by domain experts. From their perspective, it also offers them a solution that allows the management of all the information needed to adapt applications to their contexts of use.

Nonetheless, our method has limitations, as certain adaptations lead to different types of interpretations. Thus, at the business process level, the choice between adaptating the process and creating a new process is subject to the interpretation of the decision tree by the person in charge of modifying the business process. In the same way, the person in charge could decide to modify the actions contained in the business process and not to modify the business process itself, even if this is not recommended in an MDE approach. It would be necessary to propose a stricter definition for the decision trees to avoid problems of interpretation. This stricter definition would also make it possible to extend the use of this type of decision tree to other types of modifications.

It is important to point out that in an MDE approach, the management of the context-awareness can be done not only at the conceptual level but also at the others two levels, the Platform Independent Model (PIM) level and the Platform Specific Model (PSM) level. At these latter levels, as described in Saidani *et al.* [60], the adaptation is done independently of the business processes carried out by the application, which allows only the context information to be managed at the conceptual level. This is really relevant in the business domains associated with the web applications.

In PERCOMOM, the use of decision trees in the business processes models allows:

- facilitates the process of making the conceptual models context-aware,
- limits the number of new models that have to be created during the adaptation process,
- standardizes context management when the existing models of a web application are evolving, and
- facilitates the reuse of already defined models (business processes or static interaction models that are directly linked to UI).

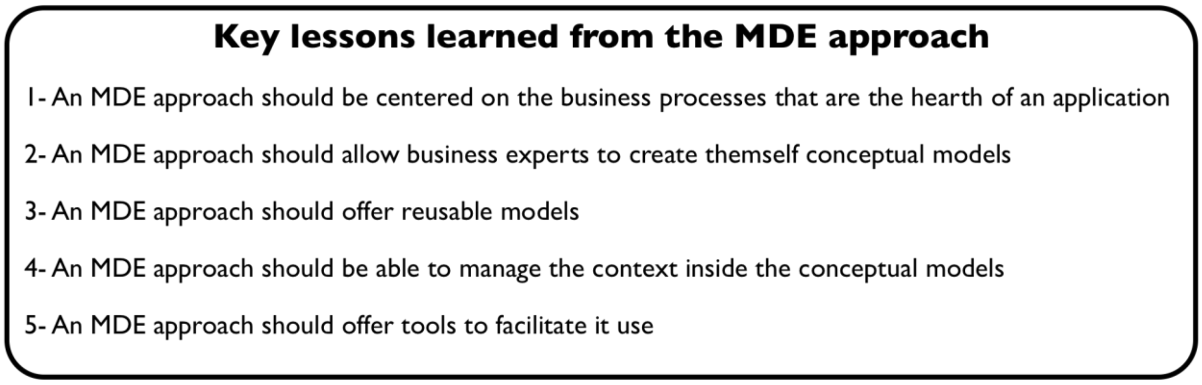
For this reason, using decision trees represents, in our opinion, a relevant and coherent solution to the problems involving the evolution of existing models.

In the current stage of our research, the possibility of extending the use of our approach to other business areas has been validated, in a limited way, through the creation of some very simple test applications in the field of online sales on the web.

If this does not allow us to make a definitive conclusion, however, we have identified a number of limitations to consider when using this approach in other business domains:

- The modeled application should be a WIMP application.
- The modeled application could be modeled using business processes.
- The modeled application does not take the context into account continuously but discretely (the application has only specific behaviors for specifics values of the context that can be determined).

In a more general way, Figure 18 presents the key lessons learned from the MDE approach when creating our case studies with the business experts.



Key lessons learned from the MDE approach

1- An MDE approach should be centered on the business processes that are the hearth of an application

2- An MDE approach should allow business experts to create themself conceptual models

3- An MDE approach should offer reusable models

4- An MDE approach should be able to manage the context inside the conceptual models

5- An MDE approach should offer tools to facilitate it use

Figure 18. Key lessons learned from the MDE approach using case studies

5. Conclusion

In the Model-Driven Engineering (MDE) approach, one of the principal expected benefits is the possibility of easily reusing already created models to design new applications. Since models have to be used in more than one application, one problem that needs to be resolved is how to make the models evolve coherently over time by limiting the number of modifications carried out.

However, today, with the multiplication of applications and the rapid development of mobile devices, users can access a large number of applications everywhere. It is no longer only important to give them information; they must be given the right information at the right time. To do so, it is necessary to take the context of use into account as a filter for the applications and the information provided to users. This can be done without linking the application to the models, or in an MDE approach, to the conceptual models.

Our proposition to resolve these problems is to use decision trees during the application design phase, which allows us to define in which types of models the modifications must be carried out in the context of an evolving application. These decision trees, which were designed to be independent of the modeling method used, separate the interactive tasks and the non-interactive tasks using an approach similar to the MVC approach.

Since the conceptual models are defined to be understood and manipulated by business domain experts, who are seldom computer engineers, the decision trees are defined to be easily used by these experts, providing only the information needed to correctly manage some aspects of the application context of use. By using business rules, these decision trees allow the models to be made context-aware. Thus, it is possible to create business processes with different behaviors according to the application, geographic or temporal context.

Our method was applied in the field of public transportation (*Viatic.Mobilité* project) to create web applications dedicated to travelers. This application allowed us to validate the use of decision trees to allow the evolution of a multi-application modeling environment that is able to take context into account. Our MDE method was implemented coherently by limiting the creation of new models to take the context into account.

In our future research, we hope to be able to go one step further in defining the notion of context, which, in our opinion, must be able to evolve in terms of the application type and the user needs. For example, the notion of geographic context is not the same when users employ an application allowing them to read a newspaper at home and when they employ an application allowing them to find the closest bakery when they are walking in the street. We will try also to integrate more context elements than the application, geographic and/or temporal context. For example, our work on the perception of time during journey [83, 84] has already allowed us to introduce some aspects of traveler behavior during a trip, as a contextual element that allows us to offer a new way for personalizing applications. This work has also introduced the question of the possibility to detect the future context of the users for which we have defined a first approach, with the use of a global level of confidence, that needs to be improved. For the moment, this first approach allows us to manage easily the most obvious changes of context states and need an adjustment period to manage the other cases.

We plan also to create all the necessary tools to allow business experts to create, manipulate and verify the consistency of all the models related to an application when using PERCOMOM. To be efficient these tools would have to be able to manage the evolution of the models during time. And particularly, they will have to be able to do a consistency check of all the existing models, for all known applications, each time a model is modified. For PERCOMOM to be used by business experts, we will need to provide, in a near future, a solution, at the coding level, to allow, at runtime, management of business processes according to the context.

During our research, we found that it is not always relevant in an MDE approach to manage the notion of context only at the conceptual level. However, we still have not been able to define rules that will allow us to clearly define which context element should be managed at each level (i.e., conceptual level, platform independent level or platform specific level). This is one of the points that we have to work on to be able to clearly characterize the context. In the near future, we will also test our decision trees with other modeling methods and other business domains to make sure that they can be generalized.

Acknowledgement

This research was supported by International Campus on Safety and Intermodality in Transportation (CISIT), the *Nord/Pas-de-Calais* Region, the European Community, the Regional Delegation for Research and Technology, the French Ministry of Higher Education and Research, the French National Research Agency (Viatic.Mobilité project) and the French National Center for Scientific Research. The authors gratefully acknowledge the support of these institutions. The authors also thank the journal’s anonymous reviewers for their many constructive remarks.

References

[1] U. Hansmann, L. Merk, MS. Nicklous, T. Stober, Pervasice Computing: The Mobile World Second Edition , Springer Professional Computing, 2003.

[2] B. Gates, Bill Gates Keynote: Microsoft Tech-Ed 2008 – Developpers. Available at: http://www.microsoft.com/presspass/exec/billg/speeches/2008/06-03teched.mspx

[3] ChoiceStream, 2008 ChoiceStream Personalization Survey, Available at: http://www.choicetream.com.

[4] J. Vesanen, What is personalization: A Litterature Review And Framework, Helsinki School of Economics, Working Papers, W-391, 2005.

[5] C. Petit-Rozé, E. Grislin-Le-Strugeon, MAPIS, a multi-agent system for information personalization, Information and Software Technology 48 (2005) 107-120.

[6] J.M. Favre, J. Estublier, Blay-Fornarino, L'ingénierie dirigée par les modèles, Hermès Lavoisier, Paris, 2006.

[7] B. Hailpern, P. Tarr, Model-driven development: The good, the bad and the ugly, IBM System Journal 45 (2006) 451-461.

[8] A. Dey, Understanding and Using Context, Personal and Ubiquitous Computing 5 (2001) 4-7.

[9] J.W. Kaltz, J. Ziegler, S. Lohmann, S. Context-aware web engineering: Modeling and applications", Revue d'Intelligence Artificielle 19 (2005) 439-458.

[10] M. Rosemann, J. Recker, Context-aware Process Design: Exploring the Extrinsic Drivers for Process Flexibility, In proceedings of the Caise'06 Workshop on Business Process Modeling Development and Support, BPMD'06, Luxemburg, 2006.

[11] J. Greenfield, K. Short, S. Cook, S. Kent, J. Crupi, Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools, Wiley, USA, 2004.

[12] T. Erl, SOA Principles of Service Design, Prentice Hall, USA, 2007.

[13] S. R. Magal, J. Word, Essentials of Business Processes and Information Systems, Wiley, USA, 2009.

[14] T. Debevoise, Business Process Management with a Business Rules Approach: Implementing The Service Oriented Architecture, BookSurge Publishing, 2007.

[15] N.A. Tariq, N. Akhter, Comparison of Model Driven Architecture (MDA) based tools, In Proceedings of the 13th Nordic Baltic Conference on Biomedical Engineering and Medical Physics, IFMBE, Umea, Sweden, pp. 43-44, 2004.

[16] A. Brossard, PERCOMOM : Une méthode de modélisation des applications interactives personnalisées appliquée à l'information voyageur dans le domaine des transports collectifs, Ph.D. Dissertation, University of Valenciennes and Hainaut-Cambrésis, 2008. Available at:
http://tel.archives-ouvertes.fr/docs/00/36/32/56/PDF/uvhc_these_abrossard_2008.pdf

[17] D.C. Schmidt, Model-Driven Engineering, Computer 39 (2006) 25-31.

[18] OMG, MDA Guide v1.0.1, 2003.

[19] D. Frankel, P. Harmon, J. Mukerji, J. Odell, M. Owen, P. Rivit, M. Rosen, R.M. Soley, The Zachman Framework and the OMG's Model Driven Architecture, White Paper, Buiness Process Trend, 2003.

[20] K. Csarneski, S. Heksen, Feature-based Survey of model transformation approaches, IBM Journal 45 (2006) 621-645.

[21] OMG, Model Driven Architecture, Model Driven Architecture (MDA), Document number ormsc/2001-07-01, Technical report, 2001.

[22] M. Oya, MDA and System Design, MDA Information Day, OMG Technical Meeting, 2002.

[23] L.J. Eyermann, A.B. Smith A.B., Roberts E.F., What Senior Management Needs to Know about the Value of MDA, OMG, 2004.

[24] D. Frankel, Reflections on the State of MDA, MDA Journal, April (2005) 1-9.

[25] P.-A. Muller, P. Studer, F. Fondement, J. Bezivin, Platform independent Web application modeling and development with Netsilon, Software & System Modeling 4 (2005) 424-442.

[26] J.-S. Sottet, G. Calvary, J.-M. Favre, J. Coutaz, IHM & IDM : Un tandem prometteur, Ergo'IA 2006, Biarritz, France, 2006.

[27] J.L Pérez-Medina, S. Dupuy-Chessa, A. Front, A Survey of Model Engineering Tools for User Interface Design, In Proceedings of 6th International Workshop on TAsk Models and DIAgrams TAMODIA'2007, Toulouse, 2007.

[28] J. Vanderdonckt, A MDA-Compliant Environment for Developing User Interfaces of Information Systems, Proceedings of the 17th Conference on Advanced Information Systems Engineering, CAiSE 2005, Porto, Portugal, 2005, pp. 16-31.

[29] G. Calvary, J. Coutaz, D. Thevenin, A Unifying Reference Framework for the Development of Plastic User Interfaces, Proceedings of 8th IFIP International Conference on Engineering for Human-Computer Interaction EHCI'2001 (Toronto, 11-13 May 2001), R. Little and L. Nigay (eds.), Lecture Notes in Computer Science, Vol. 2254, Springer-Verlag, Berlin, 2001, pp. 173-192.

[30] G. Calvary, J. Coutaz, O. Daassi, L. Balme, A. Demeure, Towards a new Generation of widgets for supporting software plasticity: the comet, 9th IFIP Working Conference on Engineering for Human-Computer Interaction, Hambourg, Germany, 2003.

[31] G. Calvary, J. Coutaz, D. Thevenin, Q. Limbourg, L. Bouillon, J. Vanderdonckt, A unifying reference framework for multi-target user interfaces, *Interacting With Computers* 3 (2003) 289-308.

[32] F. Paterno, *Model-Based Design and Evaluation of Interactive Applications*, Springer-Verlag, Londres, England, 1999.

[33] M. Florins, Graceful Degradation: a Method for Designing Multiplatform Graphical User Interfaces, Ph. D. Thesis, Université Catholique de Louvain, Belgium, 2006.

[34] Van den Bergh J., High-Level User Interface Models for Model-Driven Design of Context-Sensitive User Interfaces, Phd Thesis, Hasselt University, 2006

[35] Van den Bergh J., Coninx K., Contextual ConcurTaskTrees: Integrating dynamic contexts in task based design. In PerCom Workshops, 2004, 13–17

[36] T. Schattkowsky, M. Lohmann, Towards Employing UML Model Mappings for Platform Independent User Interface Design, Proceedings of Satellite Events at the MoDELS 2005 Conference: MoDELS 2005 International Workshops OCLWS, MoDeVA, MARTES, AOM, MTiP, WiSME, MODAUI, NfC, MDD, WUsCAM, Montego Bay, Jamaica, Volume 3844, 2006, pp. 201 - 209

[37] Nunes J.D.N., *Object Modeling for User-Centered Development and User Interface Design: The Wisdom Approach.* Ph.D. Thesis, University of Madeira, Portugal, 2001.

[38] Pinheiro da Silva P., Paton N. W., User Interface Modelling in UMLi, IEEE Software, vol. 20, no. 4, 2003, 62–69

[39] OMG, Introduction To OMG's Unified Modeling Language (UML), 2005.

[40] S. Cook, Domain-Specific Modeling and Model Driven Architecture, MDA Journal, pp. 2-10, January 2004.

[41] B. Henderson-Sellers, S. Cook, S. Mellor, J. Miller, B. Selic, UML the Good, the Bad or the Ugly, Perspective from a panel of experts, Software and Systems Modeling 4 (2005) 4-13.

[42] R. Palano, A. Pandurino, A.L. Guido, Conceptual design of web application families: the BWW approach, In Proceedings of 6th Workshop on Domain Specific Modeling, Portland, USA, pp. 23-32, 2006.

[43] S. Bernonville, C. Kolski, N. Leroy, M. Beuscart-Zéphir, Integrating the SE and HCI models in the human factors engineering cycle for re-engineering Computerized Physician Order Entry systems for medications: basic principles illustrated by a case study, International Journal of Medical Informatics, in press, doi:10.1016/j.ijmedinf.2008.04.003.

[44] S. Bernonville, C. Kolski, M. Beuscart-Zéphir, and limits of UML models for task modelling in a complex organizational context: case study in the healthcare domain, In K.S. Soliman (Ed.), Internet and Information Technology in Modern Organizations: Challenges & Answers, Proceedings of The 5th International Business Information Management Association Conference (December 13 - 15, 2005, Cairo, Egypt), IBIMA, pp. 119-127, décembre, ISBN 0-9753393-4-6.

[45] A. Nugroho, Level of detail in UML models and its impact on model comprehension: A controlled experiment, Information and Software Technology 51 (2009) 1670-1685.

[46] BPMI, Business Process Modeling Notation version 1.0, 2004.

[47] T. Erl, Service-oriented Architecture: Concepts, Technology, and Design, Prentice Hall, 2005.

[48] F. Paternò, C. Mancini and S. Meniconi, ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models, In Proceedings of INTERACT 1997, Sydney, pp. 362-369, 1997.

[49] D. Lenat, The Dimensions of Context-Space, CycCorp, 1998.

[50] H. Maus, Workflow Context as a Means for Intelligent Information Support**,** In proceeding of Third International and Interdisciplinary Conference on Modeling and Using Context, CONTEXT 2001, pp. 261-274, 2001.

[51] http://www.ebxml.org

[52] M. Van Setten, Personalized Information Systems, Giga CE project part of Gigaport Project, Telematica Instituut, 2001.

[53] R. Han, P. Bhagwat, R. Lamaire, T. Mummert, V. Perret, J. Rubas, Dynamic adaptation in an image trans-coding proxy for mobile www browsing, IEEE Personal Communication 5 (1998).

[54] J.M. Pawlowski, M. Bick, P. Veith, Context Metadata to adapt Ambient Learning Environment, In proceedings of 7th ieee conference on Polymers and Adhesives in Microelectronics and Photonics, pp. 1-6, 2008.

[55] R. Holten, A. Dreiling, J. Becker, Ontology-Driven Method Engineering for Information Systems Development, Business System Analysis with Onthologies, IDEA Group, pp. 174-217, 2005.

[56] B. Andersson, M. Bergholtz, A. Edirisuriya, T. Ilayperuma, P. Johannesson, A declarative Foundation of Process Model, In proceedings of 17th Conference on Advanced Information Systems Engineering, CAISE 2005, Porto, Portugal, pp.243-247, 2005.

[57] A. Freßmann, R. Bergmann, B. Taylor, B. Diamond, G. Carr-Smith. Mobile Knowledge Management Support in Fire Service Organisations, 8. Internationalen Tagung der Wirtschaftsinformatik, Karlsruhe, Germany, 2007.

[58] M. J. Adams, Ter, D. Edmond, W. M. Van der Aalst, Facilitating Flexibility and Dynamix Exception Handling in Workflows through Worklets, In proceedings of 17th Conference on Advanced Information Systems Engineering, CAISE 2005, Porto, Portugal, 2005.

[59] http://prom.win.tue.nl/tools/prom/

[60] O. Saidani, S. Nurcan, Towards Context Aware Business Process Modelling, The 8th Workshop on Business Process Modelling, Development, and Support (BPMDS'07), Trondheim, Norway, 2007.

[61] B. Bessai, B. Claudepierre, O. Saidani, S. Nurcan, Context-aware Business Process Evaluation and Redesign, In proceedings of the 9th Workshop on Business Process Modelling, Development and Support, BPMDS'08, Montpellier, France, pp. 86-95, 2008.

[62] K. Luyten, Dynamic User Interface Generation for Mobile and Embedded Systems with Model-Based User Interface Development, Phd Thesis, Transnationale Universiteit Limburg, 2004

[63] K. Breiner, D. Görlich, O. Maschino, G. Meixner, D. Zühlke, Run-Time Adaptation of a Universal User Interface for Ambient Intelligent Production Environments HCI 4 (2009) 663-672

[64] P. Forbrig, A. Dittmar, D. Reichart, D. Sinnig, From Models to Interactive Systems Tool Support and XIML. In Proceedings of the First International Workshop on Making model-based user interface design practical: usable and open methods and tools, MBUI 2004, Funchal, Portugal, 2004

[65] M. Hinz, Z. Fiala, AMACONT: A System Architecture for Adaptive Multimedia Web Applications. In Proceedings of the Berliner XML Tage 2004, Berlin, Allemagne, 2004, 65-74

[66] J. Kjeldskov, S. Howard, J. Murphy, J. Carroll, F. Vetere, C. Graham, Designing TramMatena: a context-aware mobile system supporting use of public transportation. *In Proceedings of the 2003 conference on Designing for User Experiences*, 2003, 1-4

[67] C. Jacquet, Y. Bellik, Y. Bourda, PRIAM : Affichage dynamique d'informations par des écrans coopérants en environnement mobile. In Proceedings of Conférence francophone ubiquité et mobilité, Ubimob 2006, France, 2006, 33-40

[68] M. Roseman, J. Recker, C. Flender, Contextualisation of business processes, International Journal of Business Process Integration and Management 3 (2008) 47-60.

[69] G. M. Kapisaki, D. A. Dimitrios, A. Kateros, G. N., Prezerakos, L. S. Iakovos, S.Venieros, Model-driven development of composite context-aware web applications, Information and Software Technology 51 (2009) 1244-1260.

[70] OMG, Model Driven Architecture (MDA), Document number ormsc/2001-07-01, Technical report, 2001.

[71] C. Petit-Rozé, A. Anli, E. Grislin-Le Strugeon, M. Abed, G. Uster, C. Kolski, Système d'information transport personnalisée à base d'agents logiciels, Génie Logiciel 70 (2004) 29-38.

[72] A. Anli, E. Grislin-Le Strugeon, M. Abed, A Generic Personalization Tool based on a Multi-agent Architecture, In C. Stephanidis (Ed.), Proceedings HCI International (Las Vegas, Nevada, July 22-27, 2005), Volume 7, Universal Access in HCI: Exploring New Interaction Environments, Lawrence Erlbaum Associates, Mahwah, New Jersey, 2005.

[73] H. Ezzedine, T. Bonte, C. Kolski, C. Tahon, Integration of traffic management and traveller information systems: basic principles and case study in intermodal transport system management, International Journal of Computers, Communications & Control (IJCCC) 3 (2008) 281-294.

[74] N. Rozanski, E. Woods, Software Systems Architecture: Working with Stackeholders using Viewpoints and Perspective, Addison-Wesley Professional, USA, 2005.

[75] OMG, Semantic of Business Vocabulary and Business Rules (SBVR), 2005, Available at: http://www.omg.org/technology/documents/bms_spec_catalog.htm

[76] The Business Rules Group, *Defining Business Rules ~ What Are They Really?*, Final Report Version 1.3, USA, 2000.

[77] M. R. Dijkman, M. Dumas, C. Ouyang, Semantics and analysis of business process models in BPMN, Information and Software Technology 50 (2008) 1281-1294.

[78] A. Brossard, M. Abed and C. Kolski, Modélisation conceptuelle des IHM : Une approche globale s'appuyant sur les processus métier, Ingénierie des Systèmes d'Information 5 (2007) 69-108.

[79] A. Brossard, M. Abed and C. Kolski, Context Awareness and Model Driven Engineering: A Multi-Level Approach for the Development of Interactive Applications in Public Transportation", 27th European Annual Conference on Human Decision-Making and Manual Control, EAM'08, Delft, The Netherlands, 2008.

[80] T. Reenskaug, Models-View-Controller, Report, Xerox-Parc, USA, 1979.

[81] http://viatic.inrets.fr

[82] http://www.i-trans.org/en/

[83] A. Brossard, M. Abed, C. Kolski, G. Uster, User modelling: the consideration of the experience of time during journeys in public transportation. ACM Mobility Conference, 6th International Conference on Mobility Technology, Applications and Systems (2-4 September, 2009, Nice), ACM Press, 2009.

[84] A. Brossard, M. Abed, C. Kolski, G. Uster, Prise en compte de l'expérience des temps de déplacement dans les modèles conceptuels d'applications interactives personnalisées. In C. Kolski (Ed.), *Interaction homme-machine dans les transports - information voyageur, personnalisation et assistance*, Hermes Science Publications, Paris, pp. 223-251, ISBN 978-2-7462-3010-1, 2010.

[85] F. Fondement, R. Silaghi, Defining Model Driven Engineering Processes. Third Workshop in Software Model Engineering, Wisme @ UML 2004, (October, 2003, Lisbon, Portugal), 2004.