Expert Systems with Applications 40 (2013) 3145–3159

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Transportation ontology definition and application for the content personalization of user interfaces

Káthia Marçal de Oliveira [a,*], Firas Bacha [a], Houda Mnasser [b], Mourad Abed [a]

[a] University of Valenciennes and Hainaut-Cambrésis, LAMIH, CNRS UMR 8201, Le Mont Houy, Cedex 9, F-59313 Valenciennes, France
[b] Mir@cl Lab, FSEGS, University of Sfax, BP 1088, 3018 Sfax, Tunisia

ARTICLE INFO

Keywords:
Ontology
User interface
Human-computer interaction
Model-driven architecture
Transportation

ABSTRACT

Ontologies have been largely exploited in many domains and studies. In this paper, we present a new application of a domain ontology for generating personalized user interfaces for transportation interactive systems. The concepts, relationships and axioms of transportation ontology are exploited during the semi-automatic generation of personalized user interfaces. Personalization deals with the capacity of adaptation of a user interface, reflecting what is known about the user and the domain application. It can be performed on the interface container presentation (e.g., layout, colors, sizes) and in the content provided in their input/output (e.g., data, information, document). In this paper, the transportation ontology is used to provide the content personalization. This paper presents the ontology and how it is used for the personalization of user interfaces for developing transportation interactive systems by model-driven engineering.

1. Introduction

A user from the transportation system is, for instance, a passenger who seeks assistance about moving from one place to another. This simple daily activity deals with the complexity of multimodality of transportation (i.e., the possibility of using multiple transportation modes, such as bus and subway, for a single itinerary between the origin and the final destination). For a better assistance to the passenger, it is also important to present all the possible services related to the itinerary (e.g., information about restaurants and banks that the user has access on his/her journey). All this knowledge must be considered for developing useful transportation systems.

Several kinds of computing platforms (e.g., tablets, desktop, mobile phones) and medias (e.g., audio, video, image, text) simultaneously were explored for developing transportation systems to support user needs. In this daily activity, a typical scenario is, for example, a person who plans her/his trip by train at home with a PC; then, while going to the train station, she/he gets stuck in a traffic jam and checks train schedules to find another departure time, using a PDA, and changes his/her reservation. When arriving at the train station, she/he goes to the kiosk and uses terminal to get the ticket. In this simple example, three kinds of platforms were used and different context of use have to be taken into account. This flexibility makes the user even more demanding. The solution is to conquer the users by developing customizable systems adapted to any electronic device and providing particularized information to them in a way that she/he can save time and be more productive, thus providing personalized software systems.

Scenarios like that motivated us to work on the definition of personalized interactive systems for the transportation domain. Model-driven architecture (MDA) (OMG, 2003) has been shown to be an appropriate approach for the design and code of the software system and their user interfaces (UI) to address those challenges. In MDA, models play a more direct role in software production, manipulation and transformation by machines. UI can be specified with a high level of abstraction, and the final UI are transformed for different platforms.

Using MDA, we developed an approach for generating personalized UI (Bacha, Oliveira, & Abed, 2011a). To that end, we defined a transportation ontology that organizes the knowledge of the domain and a context model to capture the information about the user. This ontology is mapped with context elements to allow personalization. Both the ontology and the context are used in the approach for generating personalized UI. In this paper, we detail how the ontology is applied (i.e., concepts, relationships and axioms) to make it possible to personalize UI. Details about the UI generation using MDA can be found in other papers (Bacha, Oliveira, & Abed, 2011b; Bacha, Oliveira, & Abed, 2011c; Bacha et al., 2011a; Bacha et al., 2011b).

* Corresponding author. Tel.: +33 3 27 51 14 65; fax: +33 3 27 51 13 16.
E-mail addresses: kathia.oliveira@univ-valenciennes.fr (K.M. de Oliveira), firas.bacha@univ-valenciennes.fr (F. Bacha), ing_mnasserhouda@yahoo.fr (H. Mnasser), mourad.abed@univ-valenciennes.fr (M. Abed).

0957-4174/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2012.12.028

This paper is organized as follows. Section 2 briefly defines ontology and presents its application in the transportation domain. Then, Section 3 describes the fundamentals of personalization. Section 4 presents our approach for generating personalized UI to highlight the content personalization proposed in this paper. Section 5 provides the transportation ontology, and Section 6 shows how it is used for personalization. Finally, Section 7 offers our conclusions.

2. Ontology: definition and application in the transportation domain

Ontology is a description of entities and their properties, relationships, and constraints expressed via axioms (Grüninger & Fox, 1995). Domain ontologies (Guarino, 1998) express conceptualizations that are specific for a particular domain (e.g., medicine or transportation). They put constraints on the structure and contents of domain knowledge. For example, in the medical domain, ontologies would describe the concept, ‘‘symptom’’, and that symptoms are a manifestation of a disease. Ontologies have been exploited in many studies and domains (e.g., medicine (Arsene, Dumitrache, & Mihu, 2011; Rodriguez-Gonzalez et al.,2012; Zhou et al., 2004)), education (Chu, Lee, & Tsai 2011; Jia et al., 2011; Macrics & Georgakellos, 2006; Versin, Ivanovic, Klašnja-Milicévic, & Budimac, 2012), and logistics (Anand, Yang, van Duin, & Tavasszy, 2012; Giménez, Vegetti, Leone, & Henning, 2008; Grubic & Fan, 2010)) using their capacity to promote sharability of knowledge bases, knowledge organization, and interoperability among systems. Many studies can also be found in the transportation domain with various goals.

Becker & Smith (1997) defined an ontology for multi-modal military transportation planning and scheduling. Their ontology focuses on concepts about transportation services, activities, resources (i.e., vehicles, crews, terminal facilities) and constraints, which dictate how, when, by whom and where transportation activities (e.g., deployment, evacuation) can be executed. This ontology considers different transport modes, but it is not complete enough to support the development of travel planning systems since it deals only with military transportation activities.

Timpf (2002) described two ontologies of ‘‘wayfinding’’ with multiple transport modes in an urban area based on two perspectives: the traveler and the public transportation system. His work identified the concepts to define ontology from the description of directions given verbally by five people. At the end, he obtained a list of concepts of both perspectives and showed that one is a subset of the other. The research of seeking for concepts from the direction descriptions was very detailed; however, the list of concepts obtained are only a part of the ontology definition, since proprieties, relationships and axioms were not defined.

In another similar work, Wang, Ding, & Jiang (2005) developed a system based on public transportation ontology. Based on user inputs (i.e., origin, destination and priorities), the software system searches for bus stops, using a spatial radius search. The algorithm finds journeys based on origin-destination pairs by bus route identification, using a relationship matrix between route and station. Therefore, this work does not consider either the transportation multi-modality or the possible associated services that can be offered to the user.

Niaraki & Kim (2009) proposed a road segment ontology to determine an impedance model of road geographic information system and intelligent transportation system. This impedance model computes the amount of cost, or resistance, expected to pass through a link from its origin node to destination node. They defined a road segment ontology based on the user preferences criteria and context (or environmental) criteria. From this ontology, they defined an hierarchical structure divided into two branches: one related to the user criteria (e.g., information about tourist attractions and preferences) and the other related to the context criteria (e.g., weather and safety). The impedance is calculated using the ontology information.

Yang & Wang (2012) introduce an urban traffic ontology into information integration, describing the semantic rules and relationships, as well as the regulations of semantic mergence and the selection and verification of semantic fusion. They show that the semantic fusion based on ontology, increases the effect and efficiency of the urban traffic information integration, reduces the storage quantity and improves query efficiency and information completeness. They look for how to rapidly build ontology with massive real time traffic information and how to effectively analyze required information with the ontology.

Barrachina et al. (2012) define a vehicle accident ontology, which combines the information collected when an accident occurs and the data available in the general estimates system's accident database. Their goal is to define a common vocabulary for the interoperability of transportations systems that will be used in the future to support vehicle accidents. The ontology organizes the knowledge about the accident (e.g., coordinates, speed), the vehicle (e.g., chassis, make, model), the occupant (e.g., identity, age, sex), and the environment (e.g., speed limit, surface condition, weather).

This paper proposes a different application of transportation ontology to support the personalized UI generation, working on the definition of a travel planning ontology since this is a common task in which several applications have been developed for different platforms to support their users.

3. Personalization

Personalization is defined as ‘‘the ability to provide content and services that are tailored to individuals based on knowledge about their preferences and behavior’’ (Hagen, Manning, & Souza, 1999). ‘‘Delivering to a group of individuals, relevant information that is retrieved, transformed, and/or deduced from information sources’’ (Won, 2002). Garía-Barrios, Mödritscher, & Gütl (2005) define personalization as ‘‘adaptation towards a named user for which an internal and individual model is needed’’. Simonin & Carbonell (2006) describe personalization as ‘‘the dynamic adaptation of the interface to the profile’’. In general, personalization deals with the ability of adapting a UI considering some information related to this user.

Personalization can take many features into account and can be applied to many levels (Chevalier & Julien, 2003; Kobsa, 2001), as follows:

- Personalization of the presentation – this category of personalization tries to adapt the style and format of interaction interface components (e.g., buttons, text fields) based on the user needs and their context. Anli (2006) and Brossard, Abed, & Kolski (2011) call this ‘‘container personalization’’, and consider similar to ‘‘interface plasticity’’ defined by Thevenin & Coutaz (1999).
- Personalization of the structure – this category of personalization is applied to the links between a website's pages. For example, some links may be proposed with special notation or included in the first position in a list of links, according to their relevance.
- Personalization of functionalities – this category of personalization makes available only the functions necessary for a specific user to answer a task by automatically adapting the system.
- Personalization of the navigation – this category of personalization guides the user to the right information, by avoiding irrelevant pages.

Fig. 1. Examples of user interface personalization in a transportation system.

- Personalization of the content – this category of personalization works on the selection and adaptation of the input/output information, according to the user, his/her preferences and context.

Fig. 1 illustrates examples of the personalization of the content and containers of a travel planning system. For instance, the adaptation of the size of interface elements, such as fonts and widgets, represents a personalization of the container presentation based on the specific platform/device (i.e., iPhone) used by the user. Information about the departure city and departure date (Fig. 1a) are examples of personalization of content. Departure city is automatically filled based on the location where the user at the moment of the use of the system. Departure date is the current day. Another example of personalization of content is presented in Fig. 1b. In this example, the user is unable to walk; thus, the system will propose direct itineraries with a reduced price, according to his/her age, or indirect itineraries with the connection in the same train station.

In this paper, we are interested in the personalization of the content using domain ontology. Several methods are proposed in literature for content personalization (Doucet et al., 2004; Ioannidis and Abbas, 2008; Ioannidis & Koutrika, 2005). Some of the most used are:

- *Information filtering*, which eliminates the data that is not pertinent for the user based on a set of criteria defined by the user;
- *Recommendation*, which proposes a set of information in different formats (e.g., videos, texts, images, links) based on the system's use history by several users (Burke, 2002; Miller, Konstan, & Riedl, 2004);
- *Personalized information query*, in which the query results is ordered based on the user's profile, or the query is personalized with specific conditions based on the user's criteria and context information (Ioannidis & Koutrika, 2005; Koutrika & Ioannidis, 2005) or synonyms semantically related to the terms of a thesaurus or ontology (Manning, Raghavan, & Schtze, 2008), called *query enrichment*;
- *Auto-fill forms*, in which the input information is filled automatically by the system based on the user's context, commonly used by web-based systems.

4. An overview of our model-driven architecture approach for generating personalized UI

According to OMG (2003), model-driven architecture (MDA) is an approach for specifying a system independently of the platform that supports it: specifying platforms, choosing a particular platform for the system, and transforming the specification into a software system for a particular platform. To do so, three levels of abstraction system models are defined: computation independent model (CIM), which focuses on the specific system requirements; platform independent model (PIM), which specifies a degree of platform independence appropriate to be used with different specific platforms; and platform specific model (PSM), which combines the specifications in the PIM with the details that specify how this system uses a particular type of platform. Transformations are used to convert a model to another model of the same system from CIM to PIM and from PIM to PSM, and then the generation of final code.

Several MDA approaches and tools for UI design have been proposed (Berti, Mori, Paternò, & Santoro, 2004; Jespersen & Linvald, 2003; Sottet, Calvary, Coutaz, & Favre, 2008; Vanderdonkt, 2005). They automatically generate user interface adapted to specific platforms, considering the container presentation issues, such as fields, screen resolution or screen size. However, to the best of our knowledge, content personalization is not considered in these approaches/tools.

To generate personalized UI for transportation interactive systems, we use a MDA approach by integrating specific knowledge about transportation system and information about the user and the context when she/he uses the system. Thus, the UI is developed using the three MDA levels of abstraction (Bacha et al., 2011a): the CIM to establish the user's interaction with a high level of abstraction, represented as a UI task model; the PIM to introduce the UI structure using a UI language and content personalization; and PSM to specify the UI for a specific platform. To define the CIM model, we use Business Process Modeling Notation (BPMN) (OMG, 2006) because of its ability to represent the flow of information through the tasks, which is particularly important to present which knowledge about the transportation domain should be used in the tasks. PIM and PSM models are specified in User Interface Markup Language (UIML) (Helms et al., 2009), a general language for describing UI that facilitates the work with dynamic information. This language provides several tools for creating platform-independent interfaces and the conversion to code in different platforms (e.g., the toolkit, LiquidApps, which makes it possible to convert to Java, HTML, WML & VoiceXML).

While defining the UI task model at the CIM level, all the existing approaches for UI specification and generation use a vocabulary usually specified in a domain model. In our case, this vocabulary is established by a transportation domain ontology. To take into account the user profile and his/her context, we use a context model that is mapped for the transportation ontology. The transportation ontology and the context model are used for generating personalized UI of different transportation interactive systems.

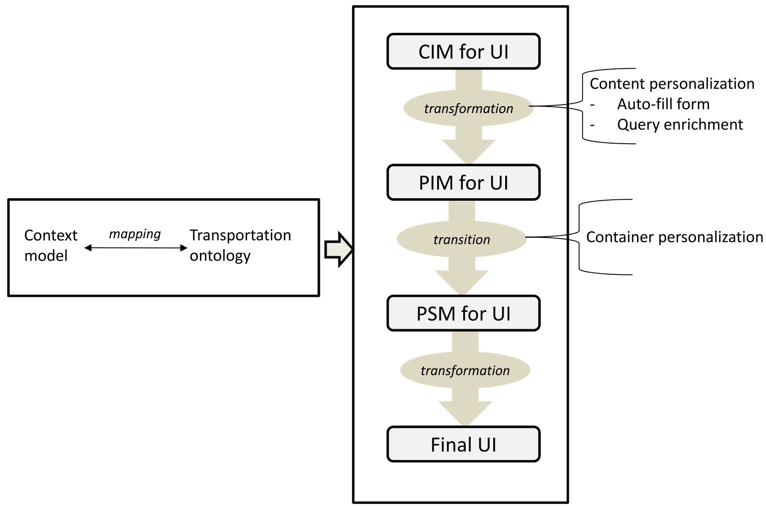

Fig. 2. Main components for the generation of UI for transportation interactive system.

Fig. 2 summarizes our MDA approach. The transportation ontology and the context model are used for generating UI. As Bacha et al. (2011c) stated, transformations from CIM to PIM generate all the UIML parts that are independent of the platform and integrate the specific information from the context, which should be considered to provide personalized information. In this transformation, we used the transportation ontology for content personalization by applying two of the methods described in the previous section: the *auto-fill form* for the input information and the *query enrichment*, including the arguments to be used in the search for the information. The generation from PIM to PSM includes or modifies the platform's specific characteristics in the generated PIM code. Thus, we preferred to call this "transition" instead of "transformation". At this level, the information of the context is used to provide the personalization of the presentation (i.e., the containers). Finally, the transformation from PIM to the final code is done using existing tools (e.g., LiquiApps (http://liquidapps.harmonia.com/features/). This final code is compiled, linked with the rest of the application, and executed.

All the details of the automatic generation of the final interface by transformations can be found in Bacha et al. (2011a), Bacha et al. (2011b). In the next sections, we will focus on how we provide the content personalization using the domain ontology. Section 5 presents the transportation ontology itself and its mapping with the context information, and then Section 6 explains how this ontology is used to generate personalized UI for transportation systems.

5. The transportation ontology and mapping the context

The purpose of this transportation ontology is to facilitate information retrieval for transportation systems used by passengers (Mnasser, Khemaja, Oliveira, & Abed, 2010). Based on this purpose, and as recommended by ontology methodologies (Grüninger & Fox, 1995; Noy & McGuinness, 2001), we defined the ontology by setting its competency questions (i.e., requirements in the form of questions that the ontology must answer), as follows:

i. What is the "transportation multi-modality"?
ii. How is a transportation journey characterized?
iii. How are the public transportation stop points organized?
iv. What are the associated services to a journey?
v. How good is the public transportation infrastructure?
vi. Which kinds of journeys can be offered to a passenger?

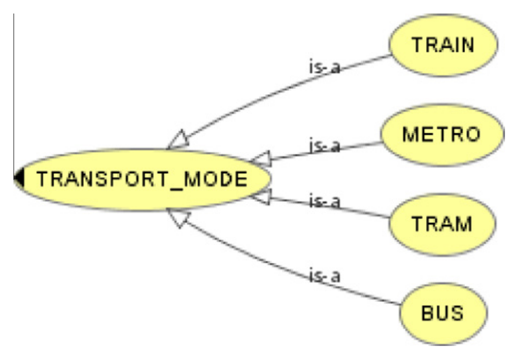

Fig. 3. Hierarchical structure of class, "Transport mode".

To answer the first competency question (i.e., What is the transportation multi-modality?), four transport modes were selected from the literature (Predim, 2009a; Timpf, 2002) (Fig. 3): *metro*, *tram*, *train* and *bus*. They are used by several *transport lines* from different *operators* of the *transportation network* (Predim, 2009a). A transport mode has a *vehicle type* with relevant properties for a journey planning (e.g., number of seats). Several object properties were set to represent the connection between classes.

We have the following (software-specific) first-order logic[1] expressions[2]:

```
TRANSPORT_LINE is_exploited exactly 1 TRANSPORT_MODE
TRANSPORT_LINE is_part_of only TRANSPORTATION_NETWORK
TRANSPORT_LINE is_served exactly 1 OPERATOR
TRANSPORT_MODE is_classified exactly 1 VEHICLE_TYPE
```

- The word, *only*, is equivalent to the universal, denoted by $\forall$. It indicates that any instance of the class, TRANSPORT_LINE has a relationship along the specified property `is_part_of` only to instances that are member the class, TRANSPORTATION_NETWORK.

[1] In this paper, we use the same format found in (Anand et al., 2012) to present the axioms.

[2] Each object property had its inverse object property defined. For example, `is_exploited` has as inverse property `used_by` (`TRANSPORT_MODE used_by only TRANSPORT_LINE`).

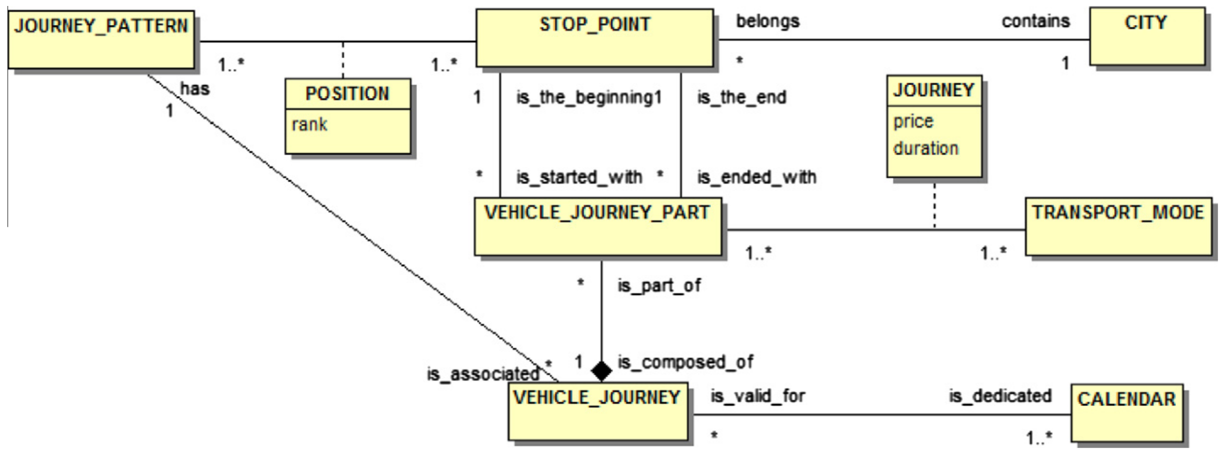

Fig. 4. Object properties related to the concepts that characterize a transportation journey.

- The word, *exactly*, represents a cardinality restriction. It specifies the exact number that an individual must participate in for a given property. For example, an instance of the class, TRANSPORT_LINE, participates in one relationship along the specified object property `is_exploited` to exactly one instance of the class, TRANSPORT_MODE. In other words, "a TRANSPORT_LINE is exploited by exactly one TRANSPORT_MODE".

To answer the second competency question (i.e., How is transportation journey characterized?), we found in the literature that a transport line offers different trip itineraries, or as defined by Transmodal guidelines (Predim, 2009a), different journey patterns. Each *journey pattern* is a collection of ordered *stop points* from an origin to a destination. A *journey pattern* is associated to a *vehicle journey*, which defines the depart time based on the *calendar* that should consider a planning for each weekday and its particularities (e.g., holidays). From one stop point to another, *a part of a vehicle journey* describes a public service vehicle's journey, which is one kind of *transport mode*. The price and duration of a *journey* depends on the transport mode associated (e.g., a bus journey may be cheaper but longer than a TGV train journey). A *stop point* belongs to a *city*. Those classes have several object properties. To better understand them, we used a UML diagram notation (Fig. 4).

To formalize the association classes in the UML diagram (*Position* and *Journey* shown in Fig. 4), we used the pattern proposed by W3C (2006) and Hoekstra (2009). Using these patterns, new ontology classes and object properties are defined for each attribute of UML association class (i.e., for *price* and *duration* and *rank*). To formalize the composition association (*is_part_of*), we used the proposition defined by Antoniou & van Harmelen (2004). Thus, we can assert that:

```
POSITION is_defined_for only JOURNEY_PATTERN
POSITION is_defined_for some JOURNEY_PATTERN
POSITION hasRank only RANK
STOP_POINT is_designated only POSITION
JOURNEY is_set_to only TRANSPORT_MODE
JOURNEY is_set_to some TRANSPORT_MODE
JOURNEY has_Price only PRICE
JOURNEY has_Duration only DURATION
VEHICLE_JOURNEY_PART is_characterized_by only JOURNEY
VEHICLE_JOURNEY_PART is_part_of only VEHICLE_JOURNEY
VEHICLE_JOURNEY is_associated exactly 1 JOURNEY_PATTERN
```

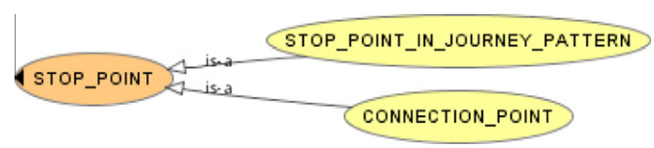

Fig. 5. Hierarchical structure of class, "Stop point".

```
VEHICLE_JOURNEY is_valid_for only CALENDAR
VEHICLE_JOURNEY is_valid_for some CALENDAR
STOP_POINT belongs exactly 1 CITY
```

- The word, *some*, here is equivalent to the existential restriction, denoted by ∃. It indicates that any instance of the class POSITION participates at least of one relationship along the specified object property is_defined_for to instances that are members of the class, JOURNEY_PATTERN. In other words, "each POSITION is defined for at least one JOURNEY_PATTERN".
- The word, *only*, is equivalent to the universal, denoted by ∀. It indicates that any instance of the class POSITION has a relationship along the specified property `hasRank` only to instances that are member the class, RANK.
- The word, *exactly*, represents a cardinality restriction. It specifies the exact number that an individual must participate in for a given property. For example, an instance of the class, STOP_POINT, participates in one relationship along the specified object property `belongs` to exactly one instance of the class, CITY.

As shown in Fig. 5, in order to answer the third competency question (i.e., How are the public transportation stop points organized?), two kinds of stop points were identified: a *stop point in a journey pattern*, which represents a stop point where the passenger does not change vehicles to arrive at the destination, even if the vehicle stops; and a *connection point*, in which the passenger needs to change the vehicles and may also change the transport mode associated to them. A *connection point* is characterized by a *connection link* in which the passenger has available vehicles of different *transport modes*. Different *walking* times may be necessary to cover this link, depending on the passenger. However, an average of the minutes is defined to provide the journey plan to the passenger.

The object properties set to represent the connection between classes provide these expressions:

```
CONNECTION_LINK is_started_with exactly 1 CONNECTION_POINT
```

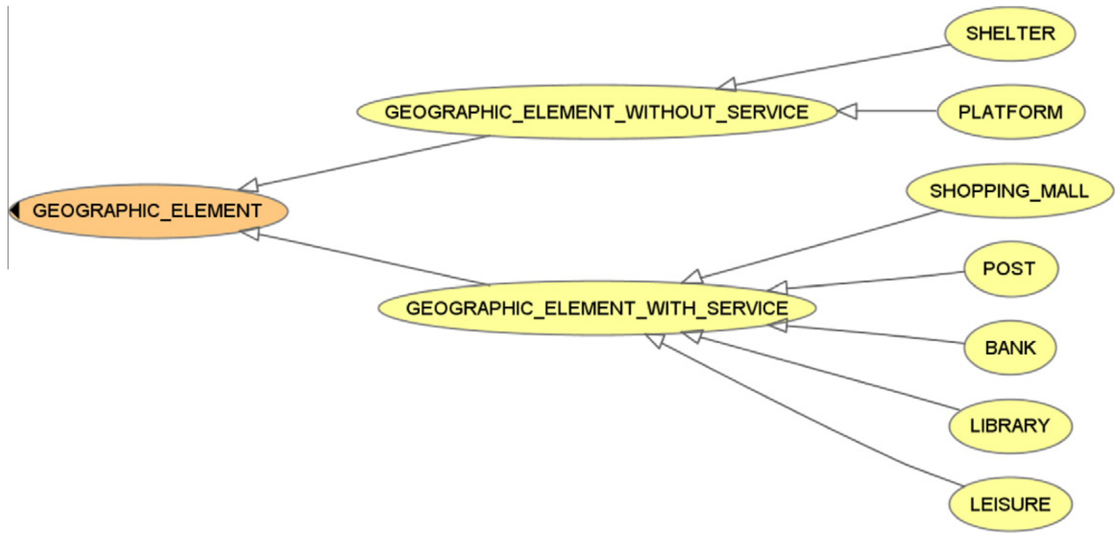

Fig. 6. Hierarchical structure of class, "Geographic Element".

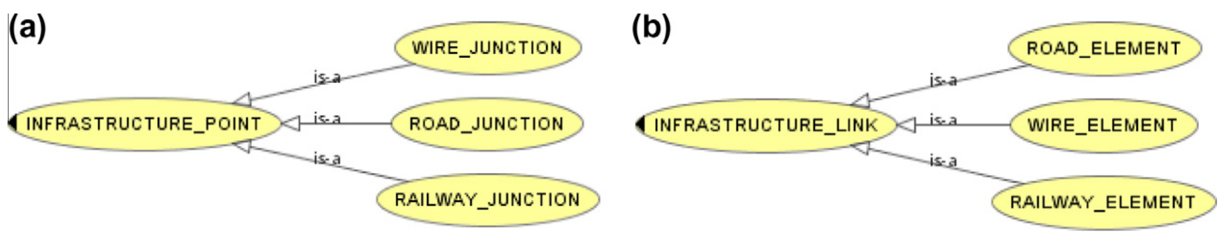

Fig. 7. Hierarchical structure of classes, "Infrastructure point" and "Infrastructure link".

```
CONNECTION_LINK is_endeed_with exactly 1 CONNECTION_POINT
CONNECTION_LINK hasWalking exactly 1 WALKING
CONNECTION_LINK is_used_by only TRANSPORT_MODE
CONNECTION_LINK is_used_by some TRANSPORT_MODE
```

The kind of a journey's associated services (i.e., Which are the associated services to a journey?) are presented as different geographic elements, which are located at the connection point (Fig. 6). The *geographic elements* are any place, location or site, physically situated in the connection stop point. Geographic elements can be located next to another one. These geographic elements, called *geographic elements with services*, may provide services that can interest the user, since the user will change vehicles at these stop points and may spend some time there before taking the other vehicle. The ontology included the main geographic elements in this case. It is also possible to have a geographic element that does not provide services, called *geographic elements without services*. It important to inform the users when there is some kind of protection for the rain (e.g., *platform* or *shelter*) at the connection point. The geographic elements can also be organized in *exchange poles* with more than one service possibilities.

With these object proprieties, we can assert that:

```
CONNECTION_POINT is_encercled only GEORGRAPHIC_ELEMENT
GEOGRAPHIC_ELEMENT is_part_of only EXCHANGE_POLE
GEOGRAPHIC_ELEMENT is_next only GEOGRAPHIC_ELEMENT
```

The inverse property for the object property, is_part_of, imposes a minimum cardinality restriction, represented by the word *min*:

```
EXCHANGE_POLE is_composed_of min 2 GEOGRAPHIC_ELEMENT.
```

An instance of class EXCHANGE_POLE participates in at least to (<=2) relationships along the specified object property, is_composed_of, in instances of class, GEOGRAPHIC_ELEMENT.

As shown in Fig. 7, in order to answer the fifth competency question (i.e., How good is the public transportation infrastructure?), the literature shows the stop points are located in some i*nfrastructure points*: *railway junction*, *wire junction* and *road junction* (MEDAD., 2008; Predim, 2009a) (Fig. 7a). These junctions are joined by *infrastructure links* (Fig. 7b). Each vehicle journey part describes the displacement in the infrastructure link.

Those connections between the classes are expressed as follows:

```
INFRASTRUCTURE_LINK is_started exactly 1 INFRASTRUCTURE_POINT
INFRASTRUCTURE_LINK is_ended exactly 1 INFRASTRUCTURE_POINT
VEHICLE_JOURNEY_PART is_described exactly 1 INFRASTRUCTURE_LINK
```

Based on the organization of transportation described, several kinds of journey patterns can be offered to the passenger for a better planning (Fig. 8). We organized them in a taxonomy of journey patterns. In this way a journey pattern is classified as a *direct journey pattern*, when the journey pattern is composed only of stop points in journey pattern; or as an *indirect journey pattern*, when it has some connection stop points. Direct journey patterns were divided in two kinds: *fast journey pattern*, when the vehicle of a journey uses only wire and road elements, and *non-fast journey*

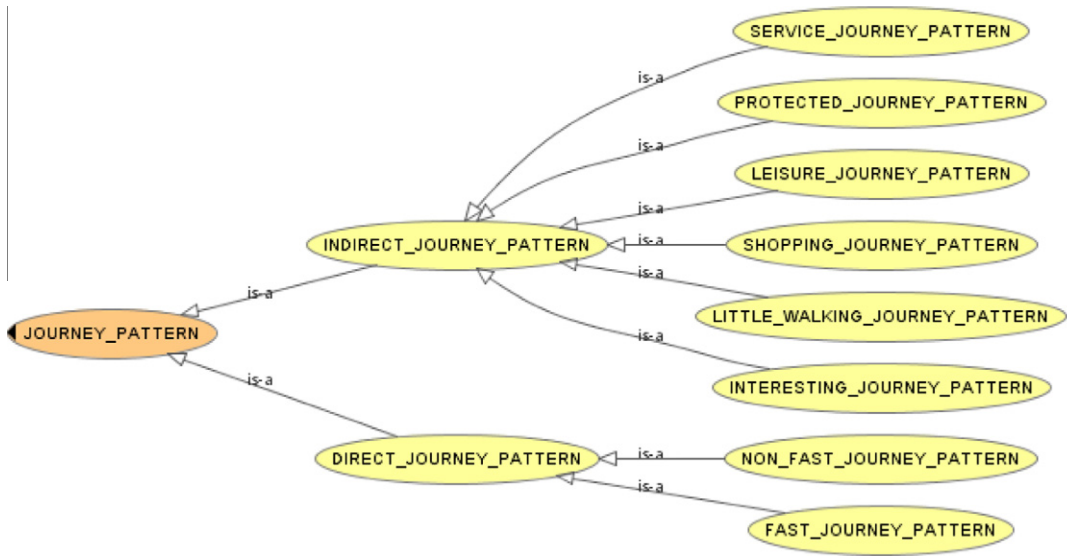

Fig. 8. Hierarchical structure of the class, "Journey pattern".

Table 1
Indirect journey patterns.

Journey pattern	Definition	SWRL expressions
Protected journey pattern	The journey pattern in which some shelter or platform is available with the associated connection stop point.	CONNECTION_POINT(?e) ∧ PLATFORM(?b) ∧ is_encercled (?e, ?b) ∧ POSITION (?a) ∧ is_designated(?e, ?a) ∧ JOURNEY_PATTERN(?q) ∧ is_defined_for(?a, ?q) → PROTECTED_JOURNEY_PATTERN(?q) CONNECTION_POINT(?d) ∧ SHELTER(?a) ∧ is_encercled (?d, ?a) ∧ POSITION (?e) ∧ is_designated(?d, ?e) ∧ JOURNEY_PATTERN(?p) ∧ is_defined_for(?e, ?p) → PROTECTED_JOURNEY_PATTERN(?p)
Leisure journey pattern	In the associated connection stop points, the journey pattern contains some libraries or other leisure centers.	CONNECTION_POINT(?z) ∧ LIBRARY(?f) ∧ is_encercled(?z, ?f) ∧ POSITION (?b) ∧ is_designated(?z, ?b) ∧ JOURNEY_PATTERN(?v) ∧ Is_defined_for(?b, ?v) → LEISURE_JOURNEY_PATTERN(?v) CONNECTION_POINT(?x) ∧ LEISURE(?e) ∧ is_encercled(?x, ?e) ∧ POSITION (?a) ∧ is_designated(?x, ?a) ∧ JOURNEY_PATTERN(?p) ∧ is_defined_for(?a, ?p) → LEISURE_JOURNEY_PATTERN(?p)
Journey pattern with little walking	The journey pattern in which the walking distances associated to any stop point does not exceed 5 min.	WALKING(?x) ∧ walking_duration(?x, ?d) ∧ swrlb:lessThan(?d, 5) ∧ CONNECTION_LINK(?y) ∧ is_relative(?x, ?y) ∧ CONNECTION_POINT(?m) ∧ is_started_with(?y, ?m) ∧ CONNECTION_POINT(?k) ∧ is_ended_with(?y, ?k) ∧ POSITION (?r) ∧ is_designated(?m, ?r) ∧ POSITION (?h) ∧ is_designated(?k, ?h) ∧ JOURNEY_PATTERN(?p) ∧ is_defined_for(?r, ?p) ∧ is_defined_for(?h, ?p) → LITTLE_WALKING_JOURNEY_PATTERN(?p)
Interesting journey pattern	The interesting journey pattern in which the exchange pole is available with the associated connection stop point.	EXCHANGE_POLE(?e) ∧ GEOGRAPHIC_ELEMENT(?a) ∧ is_composed_of(?e, ?a) ∧ CONNECTION_POINT(?c) ∧ corresponds(?a, ?c) ∧ POSITION (?m) ∧ is_designated(?c, ?m) ∧ JOURNEY_PATTERN(?f) ∧ is_defined_for(?m, ?f) → INTERESTING_JOURNEY_PATTERN(?f)
Service journey pattern	The journey pattern in which banks or post offices are available with the associated connection stop point.	CONNECTION_POINT(?j) ∧ BANK(?k) ∧ is_encercled(?j, ?k) ∧ POSITION (?d) ∧ is_designated(?j, ?d) ∧ JOURNEY_PATTERN(?e) ∧ is_defined_for(?d, ?e) → SERVICE_JOURNEY_PATTERN(?e). CONNECTION_POINT(?i) ∧ POST(?s) ∧ is_encercled(?i, ?s) ∧ POSITION (?q) ∧ is_designated(?i, ?q) ∧ JOURNEY_PATTERN(?b) ∧ is_defined_for(?q, ?b) → SERVICE_JOURNEY_PATTERN(?b)
Shopping journey pattern	The journey pattern in which shopping malls are available with the associated connection stop point.	CONNECTION_POINT(?m) ∧ SHOPPING_MALL(?a) ∧ is_encercled(?m, ?a) ∧ POSITION(?k) ∧ is_designated(?m, ?k) ∧ JOURNEY_PATTERN(?g) ∧ is_defined_for(?k, ?g) → SHOPPING_JOURNEY_PATTERN(?g)

pattern, when it uses only railway elements. Six indirect journey patterns were defined considering the different kinds of geographic elements defined in the ontology and the interest of taking a shorter duration journey. Although direct and indirect journey patterns are disjoint, these kinds of indirect journey patterns are not disjoint. For example, a journey pattern can be classified, as a protected journey pattern and as a leisure journey pattern.

Table 1 presents the definition of the indirect journey patterns and their formalization with Semantic Web Rule Language (SWRL) expressions. For example, the first rule from Table 1 states that a *journey pattern* named *q* (JOURNEY_PATTERN(?q)) is classified as a *protected journey pattern* (PROTECTED_JOURNEY_PATTERN(?q)) since it has a *connection point*, named (CONNECTION_POINT(?e) ∧ POSITION (?a) ∧ is_designated(?e, ?a) ∧ is_defined_for(?a, ?q)) that is encircled by a *platform*, named n (CONNECTION_POINT(?e) ∧ PLATFORM(?b) ∧ is_encercled(?e, ?b)). In these rules, the concepts are written with uppercase letters and the relationships between concepts with lowercase letters.

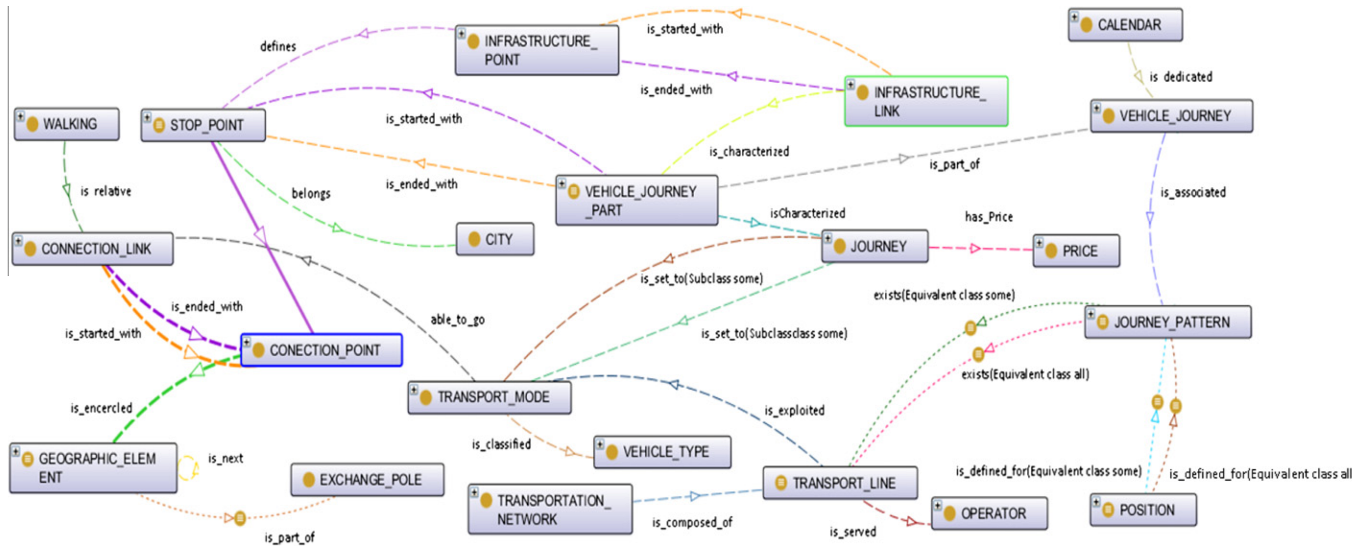

Fig. 9. The transportation ontology global view.

Table 2
Glossary of concepts.

Concepts	Definition
Calendar	Defines a valid schedule period for a vehicle journey
City	A center of population, commerce, and culture.
Connection link	The physical or spatial possibility for a passenger to change from one public transportation vehicle to another to continue the journey
Connection point	A stop point in which passengers change vehicles to the same or different transport mode
Exchange pole	A place that facilitates intermodal practices between different passenger transport modes; the interchanges are distinguished by the variety of transport modes gathered in one place
Geographic element	Location, place, position, site or corner
Geographic element with services	Geographic element in which various stores offer some services to the passenger or some physical protective structure
Geographic element without services	Geographic element in which the passenger waits for a public transportation
Infrastructure link	A link between two points in a physical network
Journey	A trip from an origin to a destination, using a specific transport mode
Journey pattern	An ordered list of stop points defining one single path through the road or rail network
Operator	Institutions that offer public transportation
Railway element	A type of infrastructure link used to describe a railway network
Railway junction	A type of infrastructure point used to describe a railway network
Road element	A type of infrastructure link used to describe a road network
Road junction	A type of infrastructure point used to describe a road network
Stop point	A point in which passengers can board vehicles or descend from vehicles
Stop point in journey pattern	A stop point in which the passenger does not change vehicles
Transport line	A group of journey patterns that is generally known to the public by the same name or number
Transport mode	A characterization of the operation according to the means of transportation
Transportation network	A set of transport lines to guarantee public transportation
Vehicle journey	The planned movement of a public transportation vehicle on a weekday from the start point to the end point of a journey pattern on a specified infrastructure
Vehicle journey part	A part of a vehicle journey created according to a specific functional purpose
Vehicle type	A classification for public transportation vehicles, according to the vehicle scheduling requirements in mode and capacity (e.g., standard bus, double-decker bus).
Wire element	A type of infrastructure link used to describe a wire network
Wire junction	A type of infrastructure point used to describe a wire network

Fig. 9 shows the top-level transportation ontology with some relationships between the main classes. This ontology was formalized in OWL1.0 and Protégé.

Table 2 shows all the concepts and attributes are clearly defined in a glossary. The definitions of the concepts were based on MEDAD (2008) and Predim (2009b).

5.1. Mapping the ontology with the context model

In general, the research in UI design considers the context is composed of three classes of entities (Calvary et al., 2003): the system's *user*; the *platform* (i.e., hardware and software), which is used for interacting with the system; and the physical *environment* in which the interaction takes place. To define the information about those elements, we have done a literature review (Bacha et al., 2011b). Although eighteen context model propositions were found, none of them was considered complete. Some propositions considered only one of the context dimensions: user, platform or environment. Others were particular for a specific domain (e.g., smart phones, e-commerce) and/or not detailed enough. Thus, we decided to integrate the main information from all propositions in a context model (Fig. 10) and to specify, where possible, the information for the transportation domain (see Bacha et al. (2011b) for details about our context model's definition).

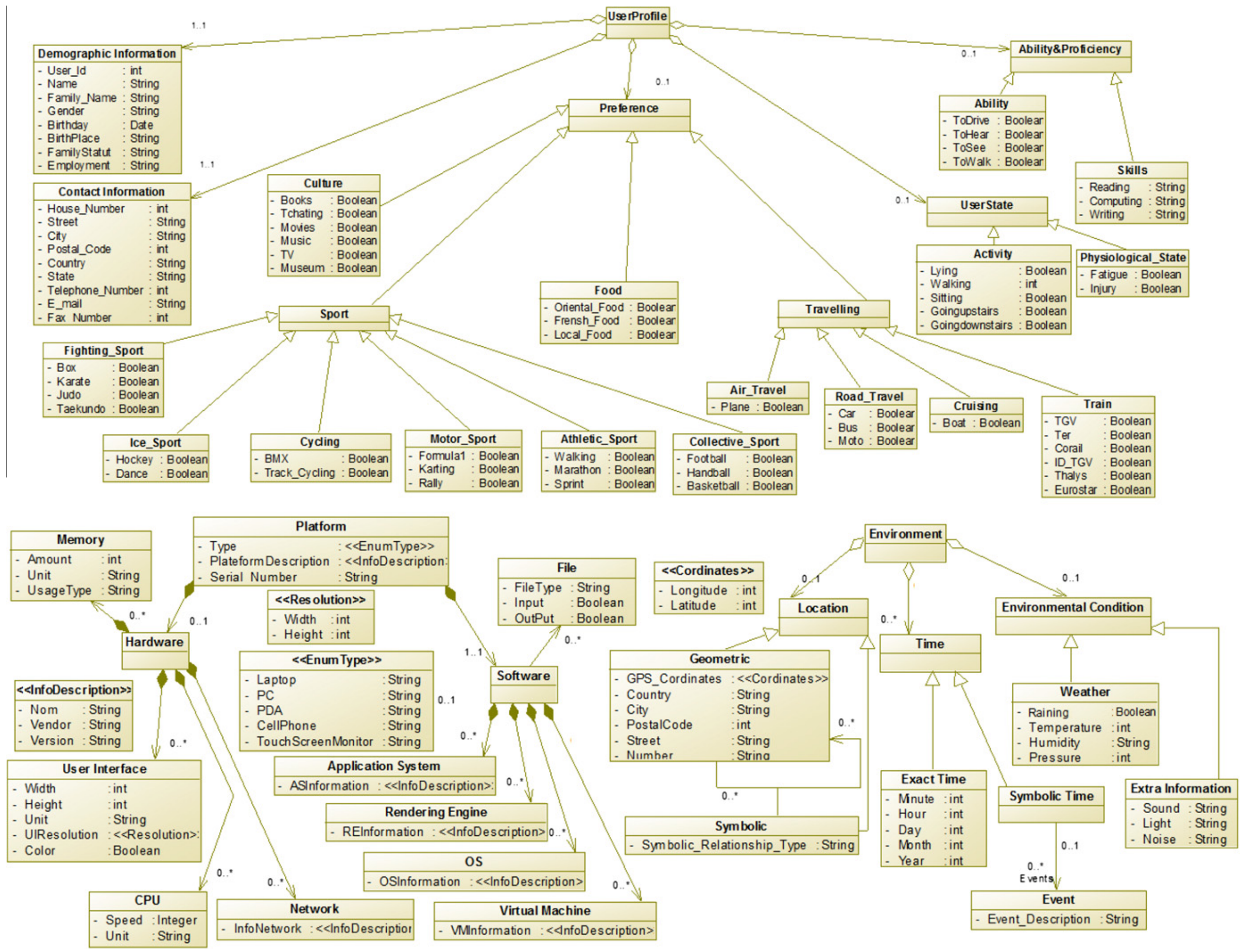

Fig. 10. The context model.

The central element of the context model is the *user profile*. To implement this model component, we divided it into five major parts, specifying the user when interacting with the final interface: *Contact information*, which contains personal data; *Demographic information*, which contains basic and unchanged user data; *Preference*, which describes user interests and preferences; *User State*, which describes the physiological user state and user activity; and *Ability* and *Proficiency*, which specifies the user skills and abilities. The second context dimension is the *platform*, which describes the final interaction platform, divided into hardware and software parts.

The third context model component is the *environment*. It contains all information about the interaction between the user and the platform. Most of the information related to this model are dynamic and influence the content personalization. This model is composed of three main parts: location, time and environmental condition. The first part, *Location*, refers to the place where the user is located at the time of the interaction. Location could be either *geometric*, using exact and deterministic information, or *symbolic*, relative to another geometric location. The second part, *Time*, describes the interaction moment. By analogy with the location dimension, the time could be described by the *exact time* (e.g., year, month, day) or by a *symbolic* one (e.g., summer, school holidays). The last part, *Environmental Condition*, describes other information that makes it possible to specify features about the environment (e.g., it is raining) at the moment of the user's interaction.

With the domain ontology, we can define the UI task model at the CIM level using this ontology's concepts, thus using the domain knowledge about the transportation domain. However, in order to provide personalized UI, we need to develop the UI in a way that we could set for each domain concept from the domain ontology used in the task model, in which context information must be considered. For example, the information about departure city in an input interface can be previously filled in by the system, depending on the context information (i.e., location) where the user is. In the same way, when providing the possible itineraries, the system must take the user's abilities (e.g., the user is handicapped) or the user's age to display reduced prices into account.

To allow these types of UI personalization, we defined that an ontology concept that should be mapped with the context model's element(s) in relation to this concept. This concept must be analyzed against the elements of the context model, looking for each context element that could influence this concept. Once an element is found, this ontology concept is mapped. The information provided in the UI will be personalized, depending on the context.

Analyzing the transportation ontology, we identified three main mapping cases. The first case refers to concepts that are exactly the same information present in the context model, although sometimes with a different name. For example, the transportation ontology concept, "city", represents the same information provided in the contact information class (i.e., the "city" attribute), which means the context information has a direct influence on the

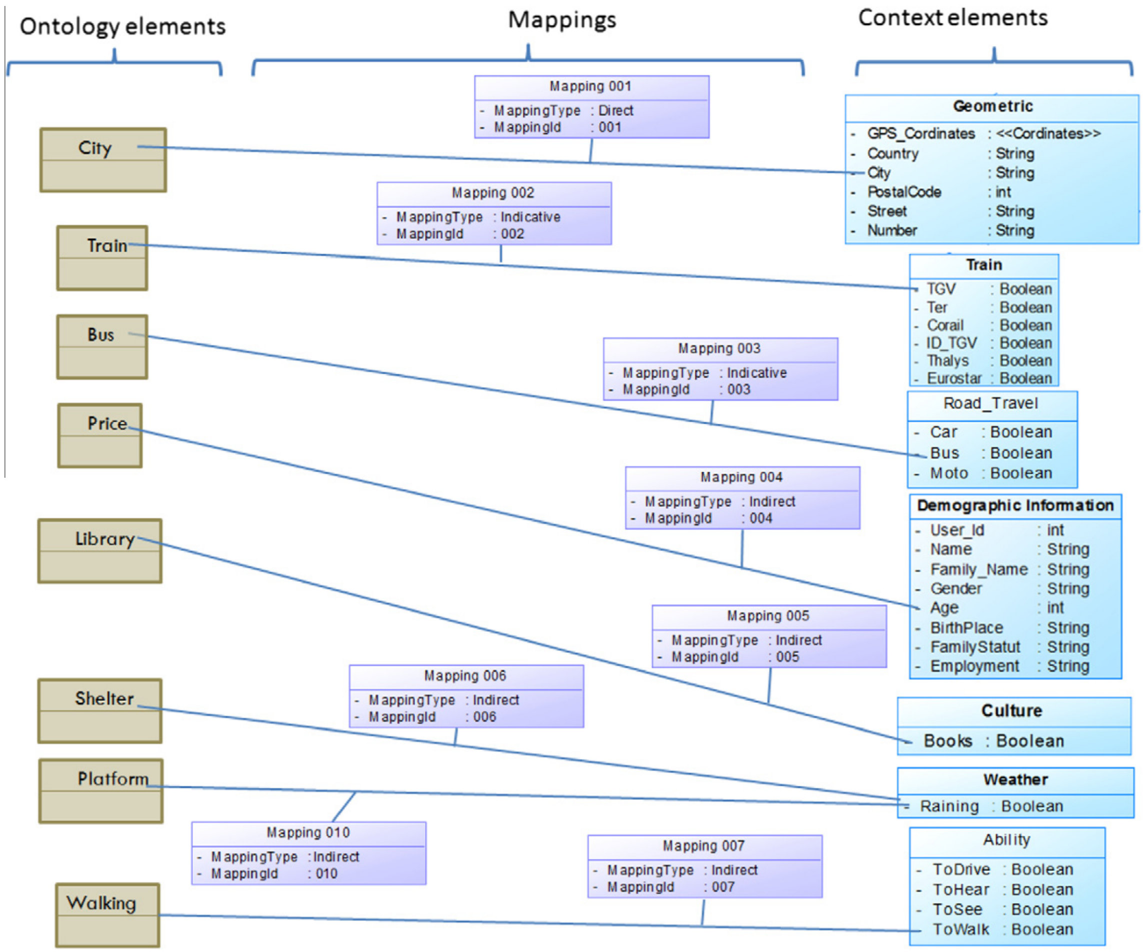

Fig. 11. Mapping examples.

content of the same information of this ontology. This is called a *direct mapping* (i.e., the concepts from the domain model are directly associated to the context model). The second case refers to the class attributes from the context model, which indicates a user profile particularity (e.g., all attributes from user preferences, abilities and state classes). These attributes can influence some domain concept by defining the existence or absence of that information. This is called an *indicative mapping*. For example, the TGV attributes indicates the user preference for some kind of train, which is an ontology concept.

The third case refers to some of the concepts from the context model that can have an indirect influence in the ontology concepts, which means it can impact indirectly the content of the domain concept. To define an *indirect mapping*, the designer must verify whether or not there is any information in the ontology that may change, depending on some personnel data modeled in the context. For example, the information about age of the user in the context model can influence the choice of the ticket price. Another example is the information of the user's ability to walk that is indirectly associated with a promenade for an itinerary in this ontology. Fig. 11 shows some of the mapping examples from our transportation ontology and the context model elements.

6. The use of the transportation ontology for the UI content personalization

Let's suppose we need to develop a system for planning a journey. In this scenario, the user connects to the system to allow the user to be identified. Thus, all information about the user's context

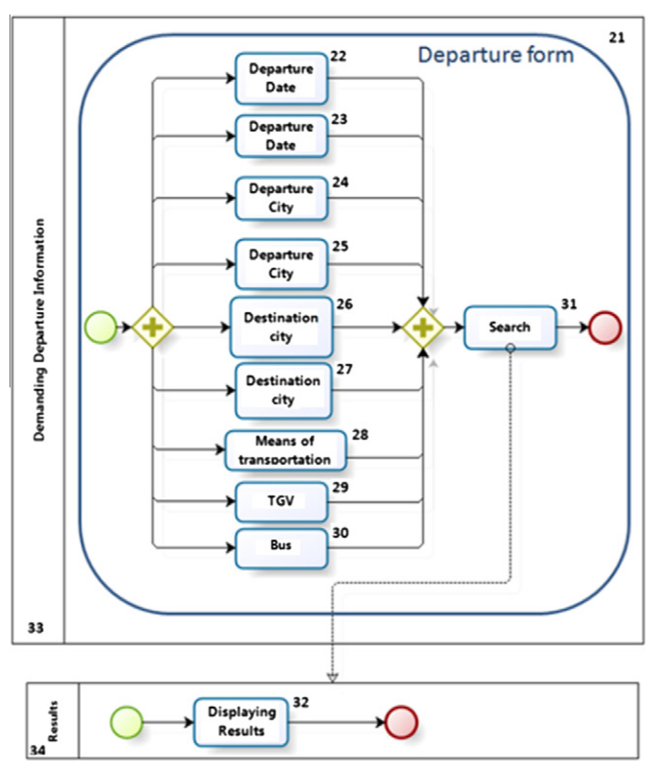

Fig. 12. Example of a Business Process Model.

is already registered. Next, the system shows the form with the required information (e.g., departure city, destination city, dates, and user preferences of transport mode) and, then, the result of the user's query. Fig. 1 shows an example of UI for this system. Generating personalized UI means dynamically generating an UI for the platform, which the user is using the system with all the user information and the domain already filled in the correct input fields. Fig. 12 shows the CIM, i.e., the Business Process Model (BPM) related to this part of the process.

As shown in Fig. 13, each BPM element is defined by a ID, its type (i.e., "user task" if the element represents an user interaction; "system task" if the element represents the result of an execution of the system functionalities; or "subprocess" if the element will be decomposed in others BPM elements until the elementary elements that represent the "user task" and the "system task"), the kind of interaction element is presented (e.g., the field, "Departure-City", will be presented in a field that should be filled in manually by the user), the domain vocabulary related to this field (i.e., the ontology concept associated), and the kind of mapping, if applicable.

To provide content personalization, the departure city and user preferences for the transport mode should be already filled in by the system. Although the user can change this information, the system should provide the form with all content collected based on information collected from the context. The results of the query should also consider the domain knowledge and the context. In this example, the user is unable to walk. So, the system should propose direct itineraries with a reduced price, according to the user's age. In case of indirect itineraries, the system should propose only protected journey patterns containing shelters since it is raining, information obtained from the context. After defining the annotated BPM, the designer can launch the model transformation process from CIM to PIM (see Bacha et al. 2011b for details about the transformation rules). In this transformation, the content personalization of the UI is done by two methods: auto-fill forms and query enrichments.

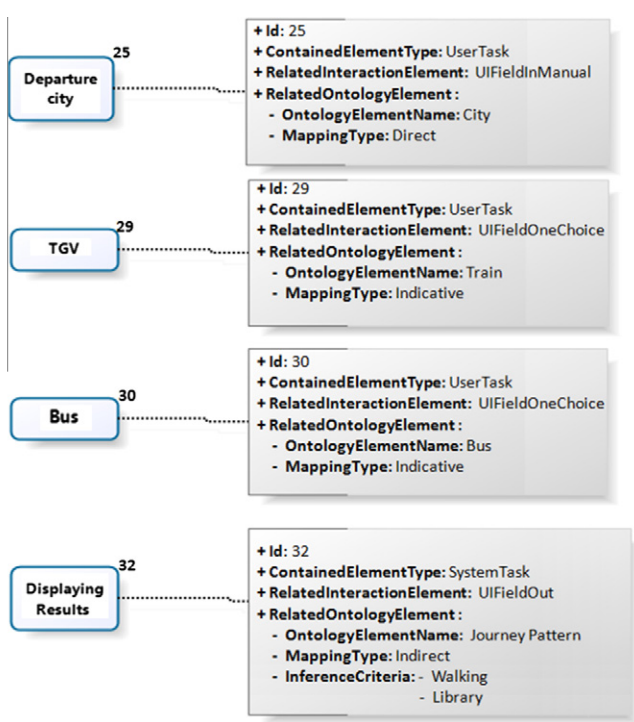

Fig. 13. Annotation of task model elements (Bacha, Oliveira, & Abed, 2011d).

Auto-fill forms are performed for all fields that was defined the direct or indicative mapping in the CIM model for UI (Figs. 12 and 13). We search for what concept from the transportation ontology was used and what context element it was mapped with. For direct mapping, the UIML property that contains the content (`g:text`) will be filled in automatically by the value taken from the instance of the context model by invoking a method that has as parameter the name of the ontology element that is related to the task (see `methodId` and `param` name in the code presented below). The following part of UIML code shows the code generated for the "City" in the first UI:

```
...
  <action>
    <whenTrue>
      <property name=''g:text''
  partName=''0025''>
      <call componentId=''0025Context''
  methodId=''0025GetValueFromCcontext''>
    <param name=''City''/>
  </call>
</property>
...
```

In the same way, it is verified the information in relation to the ontology concept used, which had an indicative mapping. In order to decide, or not decide, the selection of the concerned element, the value of the context element (i.e., true/false) is verified in the condition part in the UIML code. The following part of the code is generated for the transport mode (i.e., TGV), which will have the preselection of the right alternative based on the user preferences:

```
...
  <UIML:Behavior id="Main Behavior">
    <variable name="variable0080"
      <property>
        <call componentId="0029Context"
        methodId="0029GetValueFromContext">
      <param name="TGV"/>
      </call>
    </property>
  </variable>
...
  <whenTrue>
  <property name="g:selected" partName="0029">
    <constant value="true"/>
  </property>
  <property name="g:visible" partName="0029">
    <constant value="true"/>
  </property>
...
```

For an indirect mapping, the transformation generates a call UIML statement in PIM. The call element is an abstraction of any type of invocation of external method. It represents a method that is coded by the software designer to search the required information based on the given parameters. This method is equivalent to a searching query, in which the *DomainElementName* annotation attribute presents the searched element and the method's first argument, and the *InferenceCriteria* presents the parameters to take into account when searching them. Since we were interested in providing personalized content to the user, this query was enriched by

Table 3
Using ontology properties in path finding.

Relationship characteristic	Definition	How to be used
Functional property	If a property, P, is tagged as functional, then for all class C, C', and C'': P(C,C') and P(C,C'') implies C' = C''. In other words If a property is functional, for a given individual, there can be at most one individual that is related to the individual via the property.	Assuming that "B" is an *InferenceCriteria* for searching "A". Each ontology class that has an indirect mapping and that is connected threw a functional relationship, to a class that belongs to the path from A to B, will be integrated among the *InferenceCriteria* since this class is considered strongly pertinent for searching "A". A class related with a functional property means that there is at most one instance that is related to the other class, therefore, it restricts the search.
Symmetric Property	If a property, P, is tagged as symmetric, then for any class C and C': P(C,C') = P(C',C)	While searching the path, one can go from C to C' and vice versa, which means the transition from one class to another would be always enabled in both directions.
Transitive Property	When a property P is defined to be a transitive property, this means that if a pair (x,y) is an instance of P, and the pair (y,z) is also an instance of P, then the pair (x,z) can be inferred as an instance of P.	Assuming that "B" is an *InferenceCriteria* for searching "A" and that "X" is a class related to "B" that has a transitive relationship. Each ontology class that has an indirect mapping and was related to "X" would be integrated as the *InferenceCriteria* of "A" since it is closely related to "B".

Table 4
Using ontology properties in transportation ontology path finding.

Relationship characteristic	How is being used
Functional property	To search the "Journey_Pattern" class, the designer defined the "Walking" ontology class as an *InferenceCriteria* (Fig. 13). Among classes in the path between these two classes, there is the "Journey" class. This class has an object property "has_Price" as functional and "Price" class has an indirect mapping defined. Thus, the "Price" class, will be automatically added to *InferenceCriteria* during the "Journey_Pattern" search
Symmetric property	To search the "Journey_Pattern" class, the designer defines the "Walking" and "Library". Looking the object properties that connects those classes, the only simetric property found is *is_next* that is an auto-relationship "Geographic_Element". Therefore, in this case no new classes are added in the *InferenceCriteria*
Transitive property	To search for "Journey_Pattern " class, the designer defines "Library" as an *InferenceCriteria*. As "Shelter" has an indirect mapping, and as the class "Geographic_Element" is a hierarchical (Super) class for both classes: "Library" and "Shelter", and it has a property *is_next* which is transitive, then "Shelter" will be automatically added to *InferenceCriteria* during the "Journey_Pattern" search. Similarly, "Platform" class has an indirect mapping and is a "Geographic_Element". Therefore, "Platform" is also added to *InferenceCriteria* during the "Journey_Pattern" search

including other context elements automatically during the search process. We used the ontology relationships and axioms.

The idea is to exploit the ontology classes that implement an indirect mapping, which the designer has not explicitly specified as inference criteria in the UI design. Assuming that the *DomainElementName* is named "A" and that "B" belongs to its *InferenceCriteria*, each ontology class that has an indirect mapping and that is connected, threw a functional relationship, to a class on the path from "A" to "B" will be integrated among the *InferenceCriteria*. As a result, it will enrich the query.

We need to find an oriented path from "A" to "B" that has some pertinent class "C" to enrich the query. A class "C" is pertinent when:

- There is a path between "A" and "B", with a set of relationships "R",
- "C" is connected, threw a functional relationship, to a class that belongs to this path,
- "C" has an indirect mapping, and/or
- "C" is a different form from "B"

To find the right path between classes, "A" and "B", the relationships between them are verified looking for the object properties that allow the orientation between the classes. As illustrated in the Table 3, we check the pertinence of a class using its properties. Our goal is to find class to enrich the query to search "A". In general, all classes are potential classes. We use ontology properties only to choose those classes that we consider more pertinent in the search of "A".

Table 4 applies these rules in the transportation ontology.

The PIM code generated for the indirect mapping includes a method, called `Get-Element`, which is invoked by the *call statement*. Its parameters are the searched element (i.e., Journey Pattern), followed by the parameters that the system should consider during the search process (i.e., Walking, Shelter, Platform, Library and Price):

```
...
  <action>
    <whenTrue>
      <property name="g:text" partName="0032">
      <call componentId="0032Context"
  methodId="0032Get-Element">
        <param name="Journey Pattern"/>
        <param name="Library"/>
        <param name="Walking"/>
        <param name="Platform"/>
        <param name="Price"/>
        <param name="Shelter"/>
      </call>
      </property>
    </whenTrue>
  </action>
...
```

The implemented method may use the context information and the ontology inferences to provide the results. This method is coded by the UI designer, using the parameters automatically generated by the MDA approach. In our example, the inferences for

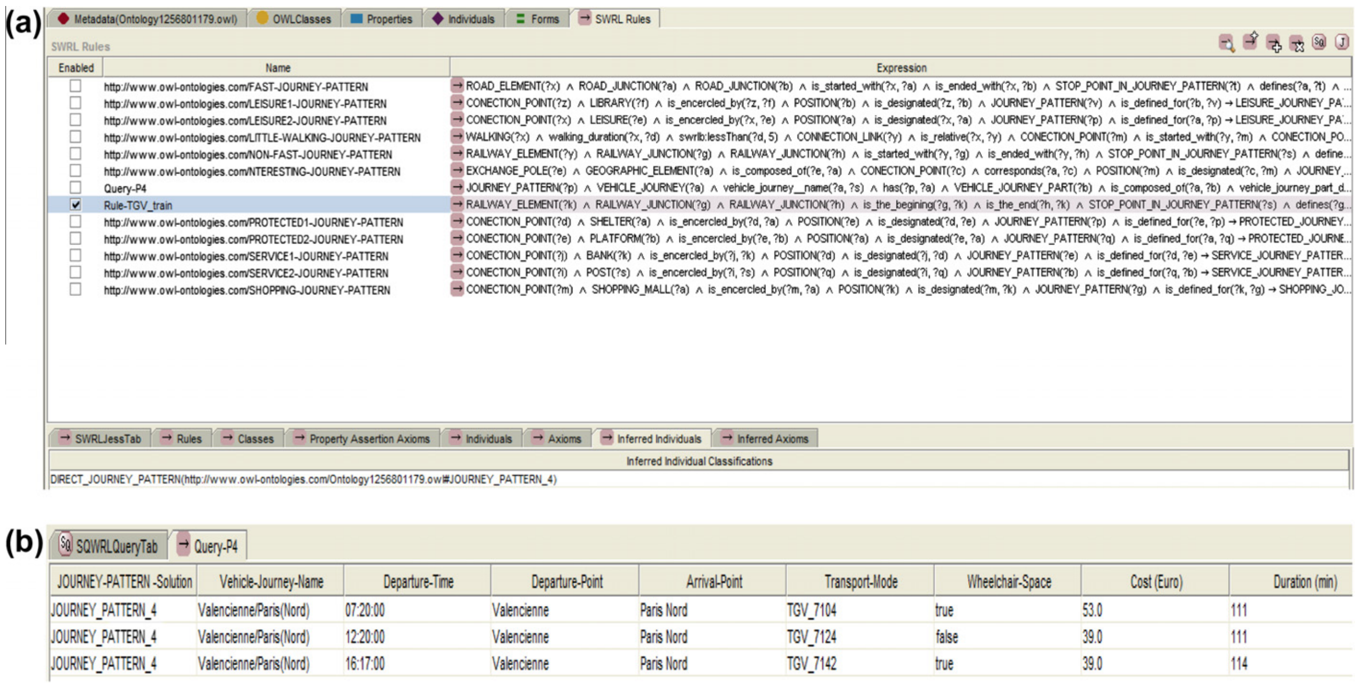

Fig. 14. Execution of (a) SWRL rules for the results presented on Fig. 1 (direct itineraries) and (b) SQWRL query to show the content of the selected Journey_Pattern.

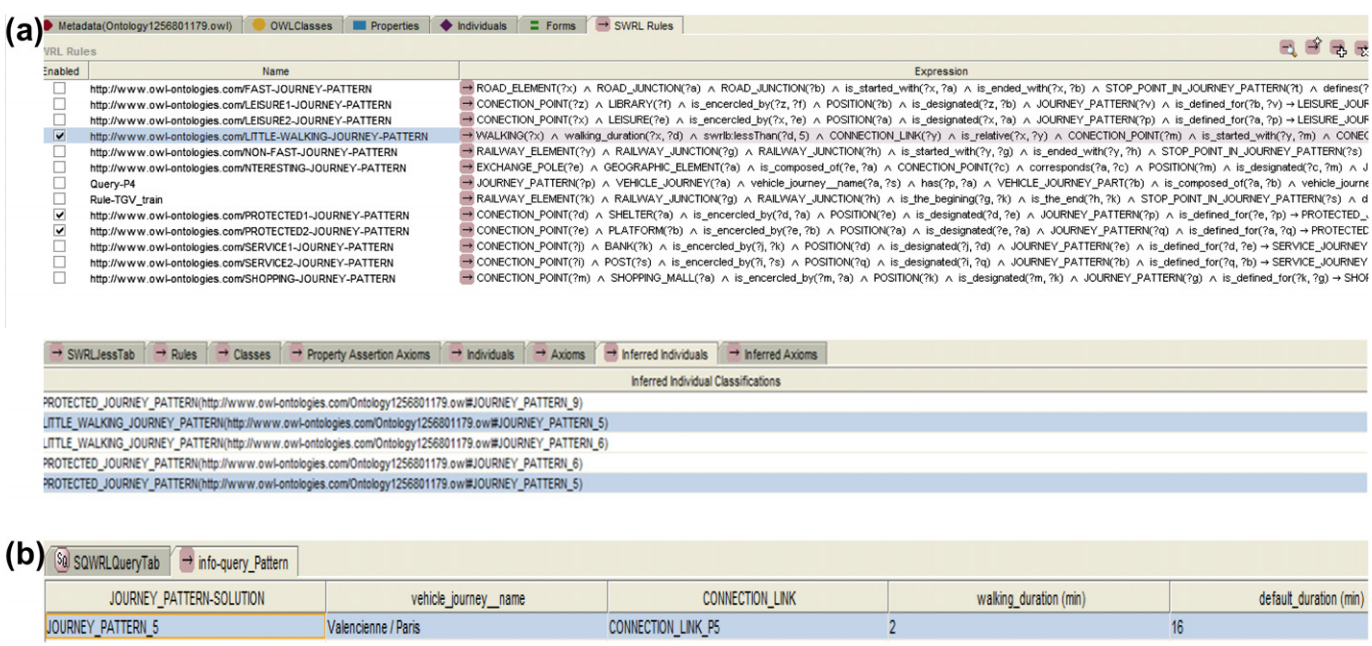

Fig. 15. Execution of (a) SWRL rules for the results presented on Fig. 1 (indirect itineraries) and (b) SQWRL query to show the content of the selected Journey_Pattern.

"Journey pattern with little walking" and "Protected journey pattern" (see Table 1)are used in this method since in this case, indirect journey patterns with little walking and protected journey patterns are looked for since the user is disabled and we consider it is raining (information collected from the context).

For directed journey patterns, we used the following inferences rule that considers direct itineraries using train:

RAILWAY_ELEMENT(?k) ∧ RAILWAY_JUNCTION(?g) ∧ RAILWAY_JUNCTION(?h) ∧ is_the_begining(?g, ?k) ∧ is_the_end(?h, ?k) ∧ STOPPING_POINT_IN_JOURNEY_PATTERN(?s) ∧ defines(?g, ?s) ∧ POSITION(?f) ∧ is_designated(?s, ?f) ∧ JOURNEY_PATTERN(?p) ∧ is_defined_for(?f, ?p) ∧ STOPPING_POINT_IN_JOURNEY_PATTERN(?v) ∧ defines(?h, ?v) ∧ POSITION(?n) ∧ is_designated(?v, ?n) ∧ is_defined_for(?n, ?p) ∧ TRANSPORT_LINE(?b) ∧ includes(?b, ?p) ∧ TRAIN (?c) ∧ exploited_by (?b, ?c) → DIRECT_JOURNEY_PATTERN(?p)

The results presented in Fig. 1b are generated in runtime by executing this method. The ontology was instantiated in *Protégé*, with real examples collected from SNCF train agency in France. The inference rules are executed, using Jess engine for the SWRL rules and SQWRL queries. Figs. 14 and 15 show the results used in the final UI.

After generating the UIML code of the PIM, the designer should do the transition in order to obtain a complete UIML code that may be tailored to a specific target platform (i.e., perform the container adaptation). The adaptation of the size of interface elements (e.g., fonts and widgets) represents a personalization of the container presentation based on the specific platform/device. In Fig. 1, it is a Smart phone. When using another device, the Human–Computer Interaction (HCI) elements have to be adapted to it according to its characteristics. For example, since the screen of the Smart phone is smaller than the Mac's screen, the arrangement and the size of the HCI elements were adapted to the new device.

7. Conclusion

Nowadays, with the diversity of technologies, the software systems are designed to be executed for different platforms with different configurations. In this context, one of the main components that has to be adapted is the interface between user and the system. This is very common for several domains, such as internet banking, e-commerce, and, in particular, for transportation systems, mainly used by users from different nationalities, cultures and physical disabilities in different situations. Scientific research is searching for the automatic generation of different user interfaces for the same software application. In this case, UI are personalized not only about aesthetic qualities but also the consideration of the personal information the system knows about the user.

This paper presented a study about UI personalization in the development of transportation interactive systems. To personalize the content presented in the UI, we used the information about the user, his/her context and the knowledge of the domain application. For this purpose, this knowledge was organized as transportation ontology and its concepts were mapped with the context information. One of the weaknesses of our proposal is that this mapping requires a deep knowledge of the application domain in order to choose which concept should be mapped with which context element and how it should be mapped. However, once this mapping has been done, this knowledge can be reused for the development of several software applications.

The ontology defined and used in this proposition was limited to deal with the problem of travel planning generically. It could be better explored to contemplate a large knowledge of the domain, for instance, other kinds of transport modes (e.g., boats, plane) provided by different kind of operators, other kinds of geographic elements and services that can be of interest to the passenger (e.g., different kind of geographic elements that offers leisure services), and real time data (e.g., real time data provided for buses and trains).

We can confirm that a domain ontology is very useful in the process of personalization since the design time when it was used to automatically identify the information to perform query enrichment and auto-fill forms to the runtime when the inference rules can be integrated in traditional methods to provide the required result of the software application. However, in order to be better applied, support tools could be developed to help the software designers to use this approach. As future work, we plan to use this approach for developing other transportation systems using the same transportation ontology.

References

Abbas, K. (2008). *Système d'accès personnalisé à l'information: Application au domaine medical.* Ph.D. thesis, INSA de Lyon.

Anand, N., Yang, M., van Duin, J. H. R., & Tavasszy, L. (2012). GenCLOn: An ontology for city logistics. *Expert Systems with Applications, 39*, 11944–11960.

Anli, A. (2006). *Méthodologie de développement des systèmes d'information personnalisés.* Ph.D. thesis, University of Valenciennes et du Hainaut-Cambrésis, France.

Antoniou, G., & van Harmelen, F. (2004). Web ontology language: OWL. In Staab & Studer (Eds.), *Handbook on ontologies*. Springer-Verlag.

Arsene, O., Dumitrache, I., & Mihu, I. (2011). Medicine expert system dynamic Bayesian network and ontology based. *Expert Systems with Applications, 38*, 15253–15261.

Bacha, F., Oliveira, K., & Abed, M. (2011a). A model driven architecture approach for user interface generation focused on content personalization. In Proceedings of the fifth IEEE international conference on research challenges in information science, Gosier, Guadeloupe, France, pp. 1–6.

Bacha, F., Oliveira, K., & Abed, M. (2011c). Supporting models for the generation of personalized user interfaces with UIML. In Software support for user interface description language, interact'2011 workshop, Lisbon, Portugal.

Bacha, F., Oliveira, K., & Abed, M. (2011d). Providing personalized information in transport systems: A model driven architecture approach. In first IEEE international conference on mobility, security and logistics in transport, Hammamet, Tunisia, pp. 452–459.

Bacha, F., Oliveira, K., & Abed, M. (2011b). Using context modeling and domain ontology in the design of personalized user interface. *International Journal on Computer Science and Information Systems (IJCSIS), 6*, 69–94.

Barrachina, J., Garrido, P., Fogue, M., Martinez, F. J., Cano, J.-C., Calafate, C. T., et al. (2012). VEACON: A vehicular accident ontology designed to improve safety on the roads. *Journal of Network and Computer Applications*.

Becker, M., & Smith, S. F. (1997). An ontology for multi-modal transportation planning and scheduling. Technical Report CMU-RI-TR-98-15, Robotics Institute, Carnegie Mellon University, Pittsburgh, USA.

Berti, S., Mori, G., Paternò, F., & Santoro, C. 2004. TERESA: A transformation-based environment for designing multi-device interactive applications. In: Proceedings of CHI – extended abstracts on human factors in computing systems, Springer, Berlin, pp. 793–794.

Brossard, A., Abed, M., & Kolski, C. (2011). Taking context into account in conceptual models using a model driven engineering approach. *Information and Software Technology, 53*, 1349–1369.

Burke, R. (2002). Hybrid recommender systems : Survey and experiments. *User Modeling and User-Adapted Interaction, 12*(4), 331–370.

Calvary, G., Coutaz, J., Thevenin, D., Limbourg, Q., Bouillon, L., & Vanderdonckt, J. A. (2003). Unifying reference framework for multi-target user interfaces. *Interacting with Computers, 15*(3), 289–308.

Chevalier, M., & Julien, C. (2003). Interface coopérative et adaptative pour la recherche d'information sur le web. In C. Chrisment (Ed.) *BDA*.

Chu, C., Lee, C., & Tsai, R. (2011). Ontology technology to assist learners' navigation in the concept map learning system. *Expert Systems with Applications, 38*, 11293–11299.

Doucet, A., Lumineau, N., Berrut, C., Denos, N., Rumpler, B., Rocacher, D., Boughanem, M., Soule-Dupuy, C., Mouaddib, N., & Kostadinov, D. (2004). Action Specifique sur la Personnalisation de l'Information. Rapport de fin de contrat, Groupe MRIM – CLIPS-IMAG.

Garía-Barrios, V., Mödritscher, F., & Gütl, C. (2005). Personalisation versus Adaptation? A user-centred model approach and its application. In K. Tochtermann, & H. Maurer (Eds.), Proceedings of the international conference on knowledge management (pp. 120-127). Graz: Austria.

Giménez, D. M., Vegetti, M., Leone, H. P., & Henning, G. P. (2008). Product ontology: Defining product-related concepts for logistics planning activities. *Computers in Industry, 59*, 231–241.

Grubic, T., & Fan, I. (2010). Supply chain ontology: Review, analysis and synthesis. *Computers in Industry, 61*, 776–786.

Grüninger, M., & Fox, M. S. (1995). Methodology for the design and evaluation of ontologies, Technical Report, University of Toronto, Toronto, Canada.

Guarino, N. (Ed.). (1998). Formal ontology in information systems. In Proceedings of FOIS'98, Trento, Italy.

Hagen, P., Manning, H., & Souza, R. (1999). *Smart personalization*. USA: Forrester Research.

Helms, J., Schaefer, R., Luyten, K., Vermeulen, J., Abrams, M., Coyette, A., & Vanderdonckt, J. (2009). Human-centered engineering with the user interface markup language. In: A. Seffah, J. Vanderdonckt, M. Desmarais, (Eds.), Human-centered software engineering, chap. 7, HCI Series, pp. 141–173.

Hoekstra, R. (2009). *Ontology representation-design patterns and ontologies that make sense*. Ph.D. thesis, University of Amsterdam (<http://dare.uva.nl/document/144859> (last access 15/09/2012).

Ioannidis, Y. E., & Koutrika, G. (2005). Personalized systems: Models and methods from an ir and db perspective. In K. Bohm, C. S. Jensen, L. M. Haas, M. L. Kersten, P. -_A. Larson, B. C. et Ooi (Eds.) VLDB, p. 1365.

Jespersen, J., & Linvald, J. (2003). Investigating user interface engineering in the model driven architecture. In Proceedings of second IFIP TC 13, workshop on integrating software engineering and, human-computer interaction, pp. 63–66.

Jia, H., Wang, M., Ran, W., Yang, S., Liao, J., & Chiu, D. (2011). Design of a performance-oriented workplace e-learning system using ontology. *Expert Systems with Applications, 38*, 3372–3382.

Kobsa, A. (2001). Generic user modeling systems. *User Modeling and User-Adapted Interaction, 11*(1), 49–63.

Koutrika, G., & Ioannidis, Y. E. (2005). Constrained optimalities in query personalization. In F. Ozcan (Ed.) Proceedings of the ACM SIGMOD international conference on management of data, pp. 73–84.

Macrics, A., & Georgakellos, D. A. (2006). A new teaching tool in education for sustainable development: Ontology-based knowledge networks for environmental training. *Journal of Cleaner Production, 14*, 855–867.

Manning, C. D., Raghavan, P., & Schtze, H. (2008). *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press.

MEDAD. (2008). Quelles données échanger avec l'application CHOUETTE ? Une implémentation des spécifications TRIDENT Projet – Profil d'échange – Concepts et contexte normatif. Ministère de l'écologie, du Développement, et de l'Aménagement durables.

Miller, B., Konstan, J., & Riedl, J. (2004). PocketLens: Toward a personal recommender system. *ACM Transactions on Information Systems, 22*(3), 437–476.

Mnasser, H., Khemaja, M., Oliveira, K., & Abed, M. (2010). A public transportation ontology to support user travel planning, In IEEE proceedings of research challenges in information science, pp. 127–136.

Niaraki, A. S., & Kim, K. (2009). Ontology based personalized route planning system using a multi-criteria decision making approach. *Expert Systems with Applications, 36*, 2250–2259.

Noy, N. F., & McGuinness, D. L. (2001). Ontology development 101: A guide to creating your first ontology''. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical, Report SMI-2001-0880, March 2001.

OMG, (2003). *MDA Guide, Version 1.0.1*, <http://www.omg.org/cgi-bin/doc?omg/03-06-01>.

OMG, (2006). Business process modeling notation specification.

Predim, (2009a). *Transmodel. Guide d'apprentissage de Transmodel.* <http://www.predim.org/spip.php?article2688>. (last access 15/09/2012).

Predim, (2009b). *Transmodel – english and french definitions for transportation.* <http://www.predim.org/spip.php?article2502>. (last access 15/09/2012).

Rodriguez-Gonzalez, A., Torres-Nino, J., Hernandez-Chan, F., Jimenez-Domingo, E., & Alvarez-Rodriguez, J. (2012). Using agents to parallelize a medical reasoning system based on ontologies and description logics as an application case. *Expert Systems with Applications, 39*, 13085–13092.

Simonin, J., & Carbonell, N. (2006). Interfaces adaptatives: adaptation dynamique à l'utilisateur courant. In I. Saleh, & D. Regottaz, Interfaces numériques, Paris: Hermes Lavoisier (coll. Information, hypermédias et communication).

Sottet, J-S., Calvary, G., Coutaz, J., & Favre, J-M. (2008). A model-driven engineering approach for the usability of plastic user interfaces. In Proceedings of EICS, LNCS 4940, pp. 140–157.

Thevenin, D., & Coutaz, J. (1999). Plasticity of user interfaces: Framework and research agenda. In Proc Interact99 Edinburgh A Sasse C Johnson Eds IFIP IOS Press Publ, 99, pp. 110–117.

Timpf, S. (2002). Ontologies of Wayfinding: a Traveler's perspective. *Networks and Spatial Economics, 2*, 9–33.

Vanderdonkt, J. (2005). A MDA-compliant environment for developing user interfaces of information systems. *CAISE, LNCS, 3520*, 16–31.

Versin, B., Ivanovic, M., Klašnja-Milicévic, A., & Budimac, Z. (2012). Protus 2.0: Ontology-based semantic recommendation in programming tutoring system. *Expert Systems with Applications, 39*, 12229–12246.

W3C (2006). *Defining N-ary relations on the semantic web*, W3C Working Group Note 12 April 2006 (<http://www.w3.org/TR/swbp-n-aryRelations/>) (last access 15/09/2012).

Wang, J., Ding, Z., & Jiang, C. (2005). An ontology-based public trasnport query system, In first international conference on semantincs and grid.

Won, K. (2002). Personalization: Definition, status, and challenges ahead. *Journal of Object Technology, 1*, 29–40.

Yang, W-D., & Wang, T. (2012). The fusion model of intelligent transportation systems based on the urban traffic ontology, In International conference on solid state devices and materials science, Physics Procedia, 25, pp. 917-923.

Zhou, X., Wu, Z., Yin, A., Wu, L., Fan, W., & Zhang, R. (2004). Ontology development for unified traditional Chinese medical language system. *Artificial Intelligence in Medicine, 32*, 15–27.