USer Interface eXtensible Markup Language (UsiXML)
W3C Working Group Submission 1 February 2012

Editors:
Jean Vanderdonckt, UCL
François Beuvens, UCL
Jérémie Melchior, UCL
Ricardo Tesoriero, UCL, University of Castilla-La Mancha

Co-authors: The UsiXML Consortium
Ugo Sangiorgi, UCL
Vivian Genaro Motti, UCL
Nesrine Mezhoudi, UCL
Iyad Khaddam, UCL
Óscar Sanchez, UCL, University of Murcia
Thanh-Diane Nguyen, UCL
Sophie Dupuis, UCL
Benoit Michel, UCL
Vi Tran, UCL
Sascha Van Cauwelaert, UCL
Pascal Beaujeant, UCL
Efrem Mbaki, UCL
Philippe Thiran, University of Namur
Mohamed Boukhebouse, University of Namur
Waldemar Pires Ferreira Neto, University of Namur
Paul Fogarassy-Neszly, BAUM Engineering
Luc Ponsard, Defimedia
Olivier de Wasseige, Defimedia
Benoit Hambucken, Defimedia
Renaud Michel, Defimedia
Jean-Bernard Collet, Defimedia
Olivier Grisvard, Institut Télécom
Bart Vermeesch, Namahn
Costin Pribeanu, ICI
Javier Muñoz, ProDevelop
Patrick Raynal, PY Automation
Eric Delvaux, See and Touch
David Faure, Thales
Oscar Pastor, Universidad Politecnica de Valencia
Nathalie Aquino, Universidad Politecnica de Valencia, UCL
Ignacio Panach, Universidad Politecnica de Valencia
Joëlle Coutaz, Université Joseph Fourier
Gaëlle Calvary, Ensimag
Sophie Dupuis-Chessa, Université Joseph Fourier
Eric Ceret, Université Joseph Fourier
Alfonso Garcia, Université Joseph Fourier
Pascual Gonzalez, University of Castilla-La Mancha
Victor Lopez, University of Castilla-La Mancha
Francisco Montero, University of Castilla-La Mancha

Copyright © 2012 UCL

This document is available under the W3C Document License. See the W3C Intellectual Rights Notice and Legal Disclaimers for additional information.

Abstract

This is a submission to the W3C Model-Based UI Working Group and describes a metamodel and defining the models and meta-models of UsiXML.

Table of contents
1. Conformance
2. Introduction

3. Use cases
   3.1. Use Case 1
   3.2. Use Case 2
      3.2.1. Demonstrator

4. methodological framework
   4.1. Global rationale for meta-modelling
      4.1.1. Cameleon Reference Framework
      4.1.2. Multipath development of user interfaces
      4.1.3. Self-containment of user interfaces meta-models
   4.2. Methodological principles for meta-modelling
   4.3. Methodological principles for modelling

5. meta-Models
   5.1. User Interface
   5.2. Task
      5.2.1. Overview
      5.2.2. Main entities
   5.3. Context
      5.3.1. Overview
      5.3.2. Main entities
   5.4. Domain
      5.4.1. Overview
      5.4.2. Example
   5.5. Abstract User Interface
   5.6. Concrete User Interface
      5.6.1. Overview
      5.6.2. Main entities
      5.6.3. ConcreteGraphicalIU
      5.6.4. ConcreteGraphicalStyle
      5.6.5. ConcreteGraphicalListener
   5.7. Transformation
      5.7.1. Overview
      5.7.2. Structure of packages
      5.7.3. Package Context
      5.7.4. Package QOC
      5.7.5. Package Transformation
      5.7.6. Connections with other transformation languages
      5.7.7. A Lab Study of the Transformation Meta-model
      5.7.8. Analysis of the transformation meta-model against the taxonomy of model transformations proposed by Mens et al.
   5.8. Workflow
      5.8.1. Overview
      5.8.2. Example
   5.9. Quality
      5.9.1. Summary
      5.9.2. Quality perspectives
      5.9.3. The Meta-Model
      5.9.4. Objects, Methods and Results, Global Quality vs Local Quality
      5.9.5. Classes of the Quality Meta-model
      5.9.6. How to build a Quality Model
      5.9.7. Example
   5.10. Mapping
      5.10.1. Overview
   5.11. Adaptation to the context of use
   5.12. Interactor

6. Abstract Syntax
   6.1. Task
   6.2. Abstract User Interface

7. Concrete syntaxes
   7.1. XSD
      7.1.1. Task
      7.1.2. Abstract User Interface
1. CONFORMANCE

As well as sections marked as non-normative, all authoring guidelines, diagrams, examples, and notes in this specification are non-normative. Everything else in this specification is normative.

The key words must, must not, required, should, should not, recommended, may, and optional in this specification are to be interpreted as described in [RFC2119].

2. INTRODUCTION

User Interface eXtensible Markup Language (UsiXML) is a formal Domain-Specific Language (DSL) used in Human-Computer Interaction (HCI) and Software Engineering (SE) in order to describe any user interface of any interactive application independently of any implementation technology. A user interface may involve variations depending on: the context of use (in which the user is carrying out her interactive task), the device or the computing platform (on which the user is working), the language (used by the user), the organization (to which the user belongs), the user profile, the interaction modalities (e.g., graphical, vocal, tactile, haptics).

As such, UsiXML consists of a User Interface Description Language (UIDL) benefitting from the following properties:

- **Open:** UsiXML can be freely used by any organization by mentioning it.
- **Multi-model:** UsiXML provides means for conceptual modeling of task, domain abstract user interface, concrete user interface, and contexts of use as defined in the Cameleon Reference Framework (CRF). In addition, it covers transformation, mapping, adaptation, and interactor modeling.
- **Model-driven:** UsiXML is defined according to the principles of model-driven engineering for the following aspects:
  - Semantics express the context, meaning and intention of each abstraction captured by the underlying meta-models on which UsiXML is based on. Meta-Models are represented by means of UML 2.0 Class Diagrams (along with MOF language) and OWL 2.0 Full.
  - Abstract Syntax makes it possible to define UsiXML models in accordance with the semantics independently of any representation formalism. The abstract syntax consists of a simplified serialized version of UML 2.0 Class Diagrams.
  - Concrete Syntax is the concrete representation formalism intended to express syntactically UsiXML Models. XML Schema (XSD) are used to define the concrete syntax of UsiXML models.
  - Stylistics are graphical and textual representations of the UsiXML abstractions that maximize their representativity and meaningfulness in order to facilitate understanding and communication among different people. Stylistics are typically used by models editors and authoring tools.
  - Transformation-based: all transformations (e.g., reification, abstraction, translation, reflection, code generation) between UsiXML models are compliant with a transformation meta-model that establishes mappings across meta-models.
- **Multi-level of abstraction:** UsiXML is compliant with the four levels of abstraction of the Cameleon Reference Framework (CRF)
  - Multi-usage: UsiXML can be used during:
    - Requirements analysis: in order to gather and elicit requirements related to the user interface and related aspects
    - Systems analysis: in order to express specifications that address the aforementioned requirements
    - Design-time: in order to refine specifications depending on the context of use and design a user interface accordingly
    - Run-time: in order to realize a user interface via a rendering engine
- **Multi-path:** UsiXML supports multiple development paths depending on the development needs and constraints:
  - Forward engineering is a composition of reification and code generation enabling a transformation of a high-level viewpoint into a lower-level viewpoint
  - Reverse engineering is a composition of abstractions and code reverse engineering enabling a transformation of a low-level, viewpoint into a higher level viewpoint.
- Lateral engineering is a composition of reflections and translation at the same level of abstraction

- Context of use adaptation is a composition of a translation with another type of transformation enabling a viewpoint to be adapted in order to reflect a change in the context of use of a user interface

- Middle-out engineering: this term refers to a situation where a programmer starts a development by a specification of the UI (no task or concept specification is priorly built). Several contributions have shown that in reality, a development cycle is rarely sequential and even rarely begins by a task and domain specification. Literature in rapid prototyping converges with similar observations. Middle-out development shows a development path starting in the middle of the development cycle e.g., by the creation of a CUI or AUI model. After several iterations of this level (more likely until customer's satisfaction is reached) a specification is reverse engineered. From this specification the forward engineering path is followed.

Figure DF describes the workflow for managing the consistency between the semantics and the syntaxes. First, the UsiXML semantics are defined in UML 2.0 Class diagrams that are then source of two semi-automated transformations:

1. Derivation of an abstract syntax: initial UML 2.0 Class diagrams are automatically transformed by UML2Abs (e.g., simplification, restriction, serialization) in order to define an abstract syntax

2. Derivation of an ontology: initial UML 2.0 Class diagrams are automatically by UML2OWL in order to define an ontology expressed in OWL 2.0 Full format

Second, the abstract syntax is then automatically transformed into a concrete syntax expressed in terms of a XML Schema in XSD format by UML2XSD transformation. This XSD file is itself used to automatically generate classes for UsiXML import/export in model editors and rendering engines.

3. USE CASES

3.1. Use Case 1

The scenario used in this use case is a business trip process, which involves different users with different roles and contexts of use. This scenario outlines the main concepts of the UsiXML language: (1) user interface description; (2) multi-role user interfaces management; and (3) multi-contextualization (different user devices).

The business trip process consists in different activities and involves three different roles, namely an employee, a manager and an administrator. The employee interacts with the process through a mobile application while the manager and the administrator use the Web browser of their desktop. The business trip process is described as following. A process is initiated when an employee submits a travel request by providing data like arrival city, depart date, and return date (user input). On receiving travel request from the employee, the process initiates three sub-processes in parallel: selecting a hotel, selecting a main of transportation (flight or train) and selecting a car rental. For each sub-process, the employee has the option to select a hotel, a main of transportation or a car rental according to his/her preferences and the proposed prices. While the hotel, the main of transportation and a car are selected, the process asks a manager (e.g. director, head of the employee team) to approve the choice of the employee (user selection). If employee choice is not approved, the travel is canceled. Otherwise, the process asks an administrator (e.g. secretary, of a financial service) to take charge of the booking and paying of the selected hotel, main of transportation and car rental. In this case, a confirmation is sent to the employee (user output). Note that, at any time, the employee can cancel his/her travel request (user event).

Figure uc1 shows an excerpt of the process expressed in UI- BPEL language [BOU11]
3.2. Use Case 2

3.2.1. Demonstrator

See appendix 9.1.1 for more details.

4. METHODOLOGICAL FRAMEWORK

4.1. Global rationale for meta-modelling
4.1.1. Cameleon Reference Framework

The CAMLEON Unified Reference Framework was produced by the EU-funded CAMLEON FP5 Project (FP5-IST4-2000-30104). CAMLEON [CCB02] [CCTLBV03], describes a framework that serves as a reference for classifying user interfaces supporting multiple targets, or multiple contexts of use in the field of context-aware computing. The CAMLEON Framework provides a unified understanding of context-sensitive user interfaces rather than a prescription of various ways or methods of tackling different steps of development. Rather, the framework structures the development life cycle into four levels of abstraction, from the task specification to the running interface (see Figure 1):

- The **Task and Concepts level** (corresponding to the Computational-Independent Model–CIM–in MDE) which considers the logical activities that need to be performed in order to reach the users' goals. Often they are represented hierarchically along with indications of the temporal relations among them and their associated attributes.

- The **Abstract User Interface** (AUI) (corresponding to the Platform-Independent Model–PIM–in MDE) is an expression of the UI in terms of interaction spaces (or presentation units), independently of which interactors are available and even independently of the modality of interaction (graphical, vocal, haptic …). An interaction space is a grouping unit that supports the execution of a set of logically connected tasks.

- The **Concrete User Interface** (CUI) (corresponding to the Platform-Specific Model–PSM–in MDE) is an expression of the UI in terms of "concrete interactors", that depend on the type of platform and media available and has a number of attributes that define more concretely how it should be perceived by the user. "Concrete interactors" are, in fact, an abstraction of actual UI components generally included in toolkits.

- The **Final User Interface** (FUI) (corresponding to the code level in MDE) consists of source code, in any programming language or mark-up language (e.g. Java or HTML5). It can then be interpreted or compiled. A given piece of code will not always be rendered on the same manner depending on the software environment (virtual machine, browser …). For this reason, we consider two sublevels of the FUI: the source code and the running interface.

These levels are structured with a relationship of reification going from an abstract level to a concrete one and a relationship of abstraction going from a concrete level to an abstract one. There can be also a relationship of translation between models at the same level of abstraction, but conceived for different contexts of use. These relationships are depicted on the figure below.

![Figure 1 - Relationships between components in the Cameleon Reference Framework](image)

4.1.2. Multipath development of user interfaces

Concerning the level of abstraction where a user has to begin the sketching of its interface, we don't want to impose anything. The user should have the choice to start from any point: Task & domain level, Abstract UI level or Concrete UI level.

Beginning and thinking with the most abstract level (tasks) seems to be the best way to design the best interfaces. However, the wish of the user could be to start at another level of abstraction, because of other needs or constraints like time. In this case, we want to provide to the user the possibility of choosing.

4.1.3. Self-containment of user interfaces meta-models

This idea directly follows the previous point. In order to provide the possibility of choosing the starting level of abstraction, each meta-model must be independent of each other. At any level, all the information needed to derive a Final User Interface should be retrieved.

Nevertheless, each model can contain less or more details in function of their level of abstraction. For example, the concept of behaviour defined at Abstract User Interface level is also defined for the Concrete User Interface and can be specified at this level by taking into account a graphical interaction for instance.

4.2. Methodological principles for meta-modelling

Several principles have to be followed in order to define high-quality meta-models. We hereby define meta-model element by referring to any entity, attribute or
relationship of a meta-model. Each principle will be consistently described according to the following template:

- **P6: Extensibility.**
  - **General definition:** In software engineering, extensibility is a system design principle where the design takes into consideration future growth.
  - **Specific definition:** Regarding UsiXML, extensibility means the ability of each meta-model to be easily extended with new meta-model element. Further, it concerns also the whole UsiXML language for which new meta-model can be easily added in order to address specific need.
  - **Positive Example:** Add a new modality in the Concrete User Interface is easily done by refining the CuiContainer, the CuiInteractor and the CuiRelationship.
  - **Negative Example:** Let’s assume a meta-model describing cars and races, with these two concepts modeled on an entity car and an entity races, and linked through some relationship. To model a new engine would be really convenient because a new entity would be created (for example moto) with all the features needed. A better way to model that would be to create an entity engine with possibilities of refinement and for which all the features inherent

- **P5: Packaging.**
  - **General definition:** A package is a way of grouping entities together, which can be useful for understanding and managing a model.
  - **Specific definition:** Regarding UsiXML, packaging means the ability of each meta-model to be composed from several semantically related parts called packages.
  - **Positive Example:** In the Concrete User Interface meta-model, we have one package for each interaction modality.
  - **Negative Example:** Let’s assume a meta-model describing cars and races. Defining cars as a characteristic of races could work but would not be well separated for understanding and managing. A better way to represent should be to define cars and races separately and link them together through relationship.
  - **Exception:** Sometimes for any reason we do not want to model the real world.

- **P4: Separation of concerns.**
  - **General definition:** In computer science, separation of concerns is the process of separating a computer program into distinct features that overlap in functionality as little as possible.
  - **Specific definition:** Regarding UsiXML, separation of concerns means the ability of the meta-models to address only one specific point without overlapping with the other meta-models. The separation of concerns is also important inside the meta-models, where the different concepts must be well separated.
  - **Positive Example:** Task concepts are present only on the Task meta-model level and can not appear in other meta-models.
  - **Negative Example:** Let’s assume that the modality would taken into account in the Abstract User Interface meta-model, then it would overlap with the Concrete User Interface meta-model and avoid good separation of concerns.
  - **Exception:**
  - **Synonym:**

- **P3: Correction.**
  - **General definition:** State in which no mistake exists.
  - **Specific definition:** Regarding UsiXML, correction means the ability of a meta-model to produce an abstraction in a way that correctly reproduces the behaviour of the features of the domain.
  - **Positive Example:** In the Concrete User Interface meta-model, the car entity has an attribute numberRaces. Inconsistency can occur because the number of races can be retrieved through the relationship between the two entities and the numberRaces attribute adds unnecessary information that can be false.
  - **Negative Example:** In the Concrete User Interface meta-model, we can define a behaviour for a component. The way the behaviour is associated to a component is the same for another component. Furthermore, if a component is modified (for example, by changing its name), its behaviour is not impacted.
  - **Exception:**
  - **Synonym:**

- **P2: Consistency.**
  - **General definition:** Agreement or accordance with facts, form, or characteristics previously shown or stated.
  - **Specific definition:** Regarding UsiXML, consistency means the ability of a meta-model to produce an abstraction in a way that reproduce the behaviour of the features of the domain in the same way throughout the meta-model and that preserves this behaviour throughout any manipulation of the meta-model.
  - **Positive Example:** If we focus on the graphical refinement of the Concrete User Interface model, all the possible components have to be described in order to allow the derived model to describe any user interface.
  - **Negative Example:** Let’s assume a meta-model holding an entity car with no attribute or component allowing describing its color. Then the meta-model will not be complete because if a model has to describe the color of the car it will not be possible.
  - **Exception:**
  - **Synonym:**

- **P1: Completeness.**
  - **General definition:** In a general way, we can say that an object is complete if nothing needs to be added to it.
  - **Specific definition:** Regarding UsiXML, completeness means the ability of a meta-model to abstract all features of the domain via appropriate concepts and relations.
  - **Positive Example:** If we focus on the graphical refinement of the Concrete User Interface model, all the possible components have to be described in order to allow the derived model to describe any user interface.
  - **Negative Example:** Let’s assume a meta-model holding an entity car with no attribute or component allowing describing its color. Then the meta-model will not be complete because if a model has to describe the color of the car it will not be possible.
  - **Exception:**
  - **Synonym:**

These ones are summarized in the followings:

- **P6: Extensibility.**
  - **General definition:**
  - **Specific definition:**
  - **Positive Example:**
  - **Negative Example:**
  - **Exception:**
  - **Synonym:**

- **P5: Packaging.**
  - **General definition:**
  - **Specific definition:**
  - **Positive Example:**
  - **Negative Example:**
  - **Exception:**
  - **Synonym:**

- **P4: Separation of concerns.**
  - **General definition:**
  - **Specific definition:**
  - **Positive Example:**
  - **Negative Example:**
  - **Exception:**
  - **Synonym:**

- **P3: Correction.**
  - **General definition:**
  - **Specific definition:**
  - **Positive Example:**
  - **Negative Example:**
  - **Exception:**
  - **Synonym:**

- **P2: Consistency.**
  - **General definition:**
  - **Specific definition:**
  - **Positive Example:**
  - **Negative Example:**
  - **Exception:**
  - **Synonym:**

- **P1: Completeness.**
  - **General definition:**
  - **Specific definition:**
  - **Positive Example:**
  - **Negative Example:**
  - **Exception:**
  - **Synonym:**

- **P6: Extensibility.**
  - **General definition:**
  - **Specific definition:**
  - **Positive Example:**
  - **Negative Example:**
  - **Exception:**
  - **Synonym:**

- **P5: Packaging.**
  - **General definition:**
  - **Specific definition:**
  - **Positive Example:**
  - **Negative Example:**
  - **Exception:**
  - **Synonym:**

- **P4: Separation of concerns.**
  - **General definition:**
  - **Specific definition:**
  - **Positive Example:**
  - **Negative Example:**
  - **Exception:**
  - **Synonym:**

- **P3: Correction.**
  - **General definition:**
  - **Specific definition:**
  - **Positive Example:**
  - **Negative Example:**
  - **Exception:**
  - **Synonym:**

- **P2: Consistency.**
  - **General definition:**
  - **Specific definition:**
  - **Positive Example:**
  - **Negative Example:**
  - **Exception:**
  - **Synonym:**

- **P1: Completeness.**
  - **General definition:**
  - **Specific definition:**
  - **Positive Example:**
  - **Negative Example:**
  - **Exception:**
  - **Synonym:**
to an engine would be already present.

**Exception:**

**Synonym:**

- **P8: Pragmatism.**
  
  **General definition:** Pragmatism is the way of dealing with a problem in a realistic way rather than obeying fixed theories, ideas or rules.
  
  **Specific definition:** Designing the meta-models of UsiXML has to take into account the developers and their necessities.
  
  **Positive Example:** In the Concrete User Interface model, several graphical components are defined in order to provide a maximum powerfulness to the user. All these component are feasible.
  
  **Negative Example:** Some components are maybe not feasible in a generic way or would take too many resources. For example a component able to predict the future would be very convenient but totally impossible for the moment.
  
  **Exception:**
  
  **Synonym:**

- **P11: Pattern based meta-modelling.**
  
  **General definition:** Design pattern is a description or template for how to solve a problem that can be used in many different situations. Patterns typically show relationships and interactions between classes or objects, without specifying the final application classes or objects that are involved.
  
  **Specific definition:** Regarding UsiXML, pattern based meta-modelling means the ability for each meta-model to respect the following patterns: (i) The name of the root entity represents the name of the model; (ii) Fields of values are preferably represented explicitly through enumerations.
  
  **Positive Example:** In the Concrete User Interface meta-model, the root entity is CUIModel or the different possible actionType (attribute of the entity Action) are listed in an enumeration ActionType.
  
  **Negative Example:** Typing the attribute actionType as a string would let it free, without constraint.
  
  **Exception:**
  
  **Synonym:**

- **P15: Expressiveness.**
  
  **General definition:** Quality of being expressive, of being able to express any possibility.
  
  **Specific definition:** Regarding UsiXML, expressiveness means the ability of a meta-model to express via an abstraction any feature of the domain.
  
  **Positive Example:** In the Concrete User Interface meta-model, the CuiInteractor can abstract any interactor for a user interface.
  
  **Negative Example:**

4.3. Methodological principles for modelling

Several principles have to be followed in order to define high-quality models. We hereby define model element by referring to any entity, attribute or relationship of a model. Each principle will be consistently described according to the following template:

- **P[i]: [Principle name].**
  
  **General definition:** General definition of the principle according to computer science.
  
  **Specific definition:** Specific definition of the principle in human-computer interaction regarding UsiXML language. The principle should be stated in terms of ability.
  
  **Positive Example:** Appropriate application of the principle (synonym: do's).
  
  **Negative Example:** Inappropriate application of the principle (synonym: don't).
  
  **Exception:** Situation in which the principle may not be applied.
  
  **Synonym:** Any other name for the same principle.

These ones are summarized in the followings:

- **P12: Completeness.**
  
  **General definition:** In a general way, we can say that an object is complete if nothing needs to be added to it.
  
  **Specific definition:** Ability of a model to fulfill all the requirements needed to describe its features.
  
  **Positive Example:** If we focus on the graphical refinement of the Concrete User Interface model, it has to address all the possibilities for designing a graphical interface.
  
  **Negative Example:** Let’s assume an attribute color for a component representing a car. If only the red is possible, then all the choices are not represented and the model is not complete.
  
  **Exception:**
  
  **Synonym:**

- **P13: Consistency.**
  
  **General definition:** Agreement or accordance with facts, form, or characteristics previously shown or stated.
  
  **Specific definition:** Regarding UsiXML, consistency means the ability of a model to produce an abstraction in a way that reproduce the behaviour of the features of the domain in the same way throughout the model and that preserves this behaviour throughout any manipulation of the model.
  
  **Positive Example:**
  
  **Negative Example:**
  
  **Exception:**
  
  **Synonym:**

- **P14: Correction.**
  
  **General definition:** State in which no mistake exists.
  
  **Specific definition:** Regarding UsiXML, correction means the ability of a model to produce an abstraction in a way that correctly reproduces the behaviour of the features of the domain.
  
  **Positive Example:**
  
  **Negative Example:**
  
  **Exception:**
  
  **Synonym:** Soundness

- **P6: Extensibility.**
  
  **General definition:** In software engineering, extensibility is a system design principle where the design takes into consideration future growth.
  
  **Specific definition:** Regarding UsiXML, extensibility means the ability of each model to be easily extended with new model element.

**Exception:**

**Synonym:**
In the Concrete User Interface model, an interface can easily be extended by composing it with other interface (via the AuiContainer).

**P15: Expressiveness.**

*General definition:* Quality of being expressive, of being able to express any possibility.
*Specific definition:* Regarding UsiXML, expressiveness means the ability of a model to express via an abstraction any feature of the domain.
*Positive Example:* In the Concrete User Interface model, it is possible to add an entity to abstract any modality.

**P16: Concision.**

*General definition:* State or quality of being concise, to only provide needed information.
*Specific definition:* Regarding UsiXML, concision means the ability of a model to produce concise, compact abstractions to abstract real world aspects of interest.
*Positive Example:* In the Concrete User Interface model, enumerations are used in order to only propose possible choices.

**P17: Separability.**

*General definition:* The capability of being separated.
*Specific definition:* Regarding UsiXML, separability means the ability of models to univocally classify any abstraction of a real world aspect of interest into one single model.
*Positive Example:* Task model and Workflow model are both at the same level abstraction but they capture different information.

**P18: Correlability.**

*General definition:* The capability of being correlated, of being linked.
*Specific definition:* Regarding UsiXML, correlability means the ability of models to univocally and unambiguously establish relationships between models to represent a real world aspect of interest.
*Positive Example:* The Mapping model is used to link two models from two different layer of abstraction.

**P19: Integrability.**

*General definition:* The quality of being integrated.
*Specific definition:* Regarding UsiXML, integrability means the ability of models to concentrate and integrate abstractions of real world aspects of interest into a single model or a small list of them.
*Positive Example:* In the Concrete User Interface model, it is easy to add new modalities to an interface, or to add new features to existing modalities.

**P20: Usefulness.**

*General definition:* The quality of being of practical use.
*Specific definition:* Regarding UsiXML, usefulness means the ability of the models to avoid describing elements that does not convey any information to any problem feature.
*Positive Example:* In the Concrete User Interface model, the graphical refinement only describes existing and useful components.

**P21: Pragmatism**

*General definition:* Pragmatism is the way of dealing with a problem in a realistic way rather than obeying fixed theories, ideas or rules.
*Specific definition:* Designing the models of UsiXML has to take into account the developers and their necessities.
*Positive Example:* In the Concrete User Interface, describe a component that would be able to render the smell of the food.

5. **META-MODELS**

This section is normative.

5.1. **User Interface**
The UI meta-model is the UI model composed by all the models, as in Figure 2. UIModel Overview.

There can be any of the models and they can be combined together. Each model can have several instances. The following models will be described in details in this document: TaskModel, ContextModel, DomainModel, TransformationModel, AUIModel, CUIModel and WorkflowModel.

The MappingModel will be introduced in the document but not into details.

5.2. Task

5.2.1. Overview

The task metamodel defines the abstract syntax used to represent the task model of the system to be developed. It is part of the Tasks & Concepts layer of the UsiXML Framework. From the structural perspective, it is inspired by the HTA approach where complex tasks are decomposed into subtasks; from the behavioral perspective, it is inspired on CTT where temporal relationships are defined in terms of LOTOS operators.

The structure of a task model is defined as decomposition into subtasks that define the parts of a task. The temporal aspects of the tasks are defined in terms of temporal operations among tasks. The Figure 3 shows an overview of the task metamodel.

5.2.2. Main entities

- **TaskModel**: The TaskModel meta-class describes the interaction between the entities that are parts of the system, and the system itself, in terms of tasks that are performed by the entities of the system. It defines a set of tasks as a whole that are decomposed into sub-tasks as parts structuring the system into task hierarchies that define the behavior of the system.

  - **Attributes**:
    - name (String): Defines the name of the task model.
    - Example:
      - Location-aware remote control, Touristic Guide.
Task: The Task meta-class describes a task performed by one or more system entities. From the structural point of view, a task can be atomic or composed. While composed tasks represent complex tasks that can be decomposed into simpler sub-tasks; atomic tasks represent an indivisible action that is performed by a single entity of the system. Thus, collaborative tasks are represented as composed tasks that can be decomposed into atomic tasks performed by single entities. To define the temporal relationship among sub-tasks, the parent task defines a set of expressions that describes the temporal relationships among subtasks. The tasks can also be decorated in order to give information such as their criticity, optionality and other details.

Attributes:
- **id (int)**: Defines the task unique id.
  - Example:
    - Let Location-aware remote control be a TaskModel of a remote control application that mutates the control UI layout according to the nearest device to control.

- **name (String)**: Defines the task name.
  - Example:
    - NotifyPointOfInterest, NextPhoto, PreviousVideo, EnterComment, etc.

- **description (String)**: Explains what the task is about by a textual description as complete as possible.
  - Example:
    - NotifyPointOfInterest: a demand for a point of interest must have been created.

- **canonicalTaskType (TASKTYPE)**: The canonical type of a task describes the action of the task without going deeper on details on how it applies.
  - Possible value:
    - CONVEY, CREATE, REINITIALIZE, FILTER, DELETE, DUPLICATE, NAVIGATE, PERCEIVE, MOVE, MODIFY, MEDIATE, SELECT, TRIGGER, STOP, TOGGLE
  - Example:
    - The SignGuestbook task has as the attribute set to APPEND. It adds a sign to a guest book.

- **precondition (String)**: Describe the condition in which the task can be executed.
  - Example:
    - NotifyPointOfInterest: a demand for a point of interest must have been created.

- **postcondition (String)**: Describe how the task changes the system.
  - Example:
    - NotifyPointOfInterest: a message is shown to the user with the notification.

Example: Let SignGuestbook be a Task where the id is 1, with the name SignGuestbook, the description is to sign the guest book.

TaskExpression: The TaskExpression is an abstract meta-class that defines expressions on tasks. Task expressions allow developers to define: (a) task attributes that varies according to the situation (i.e. criticity, frequency, etc.), and (b) temporal relationships among tasks. To represent a TaskExpression we employ the Composite design pattern where the TaskExpression plays the role of Component.

TemporalRelationship: The TemporalRelationship meta-class is a TaskExpression that allows developers to define the temporal relationship among TaskExpressions. TemporalRelationships are temporal operations defined as LOTOS operators on TaskExpressions. The LOTOS temporal operation is defined as a TTEMPORALOPERATOR that defines the ENABLING, CONCURRENCY, DISABLING, SUSPEND, ORDERINDEPENDENCE and CHOICE temporal operations. The TemporalRelationship meta-class plays the role of Composite in the Composite design pattern rooted on TaskExpression.

Attributes:
- **type (TTEMPORALOPERATOR)**: Defines the type of the temporal operator in a TemporalRelationship.
  - Possible values:
    - ENABLING, CONCURRENCY, DISABLING, SUSPEND, ORDERINDEPENDENCE and CHOICE
  - Example:
    - Let suppose that EnterInformation and SubmitInformation are two tasks that are decorated, and we want to express that the EnterInformation task is performed before SubmitInformation task. Then, the TemporalRelationship defines EnterInformation task decoration as predecessor and the SubmitInformation task decoration as successor. The TTEMPORALOPERATOR used is ENABLING.

Decoration: The Decoration is an abstract meta-class of TaskExpression that allows developers to define TaskExpression attributes that vary according to the situation they are performed (i.e. criticity, frequency, etc.). The Decorator plays the role of abstract Component in the Composite design pattern rooted on the TaskExpression meta-class. It has two concrete meta-classes according to the type of element they are decorating (Task or TaskExpression).

Attributes:
- **minIteration (Integer)**: Defines the minimum cardinality of the iteration for the task or task expression. Note that if this attribute is set to 0 then the task or task expression it refers to is optional.
  - Example: If this attribute is set to 3, then the task, or the expression, it refers to must be repeated at least 3 times

- **maxIteration (Integer)**: Defines the maximum cardinality of the iteration for the task or task expression. Note that if this attribute is set to -1 then it means that the maximum cardinality is set to infinite.
  - Example: If this attribute is set to 5, then the task, or the expression, it refers to may be repeated at most 5 times.

- **criticity (Integer)**: Defines the criticity level of a the task (from 0 to 10). If a task is critic, the criticity level will be high.
  - Possible value: An integer from 0 to 5
  - Example: A very critical task will have a criticity level around 4 or 5.

- **frequency (Integer)**: Defines the frequency of the task (from 0 to 10). If a task happens often, the frequency will be high.
  - Possible value: An integer from 0 to 5
  - Example: A task that will often happen will have a high frequency (4 or 5).

- **centrality (Integer)**: Defines if the task is important for the current context. A log in task will not be central for visiting a website, but would become central when the user want to access to his account.
  - Possible value: An integer from 0 to 5

- **minExecutionTime (Integer)**: Set the minimum time for the execution.
  - Possible value: Any integer.
  - Example: The value 5 means that the task should at least last 5 seconds.

- **maxExecutionTime (Integer)**: Defines if the task is important for the current context. A log in task will not be central for visiting a website, but would become central when the user want to access to his account.
  - Possible value: An integer from 0 to 5
  - Example: The value 10 means that the task could not last more than 10 seconds.

➢ **TaskDecoration**: The TaskDecoration is a concrete meta-class of Decoration that allows developers to define Decorations on Tasks.

  - **Attributes**:
    - **nature (TNATURE)**: Defines the interaction needed for the task. This attribute depends on the type of entity that is available to perform the task. If it is a single user, the nature will be USER; if it is the system, the nature will be SYSTEM; if both are needed, the task will be INTERACTIVE; if the task is unknown or too complex to describe the nature, the UNDEFINED value is set.
      - Possible values: UNDEFINED, USER, SYSTEM, INTERACTIVE
      - Example: A user filling a form is an INTERACTIVE task because the user is interacting with the system in order to fill the form. A user that is thinking about a solution in his mind is an USER task. The printing of a document is a SYSTEM task because it only involved the system.

➢ **ExpressionDecoration**: The ExpressionDecoration is a concrete meta-class of Decoration that allows developers to define Decorations on TemporalRelationships.

### 5.3. Context

For details about this section and subsections, see deliverable [UsiXML11]

#### 5.3.1. Overview

#### 5.3.2. Main entities

### 5.4. Domain

The UsiXML domain model describes the various entities manipulated by a user while interacting with the system [UsiXML07, Sta08]. This model specifies the main concepts of a User Interface by identifying the relationships among all the entities within the scope of the User Interface, their attributes and the methods encapsulated within the entities. As such, the domain model can be modeled with an object-oriented model like the UML class diagram [OMG08]. A class diagram of UML is a type of static structure diagrams that provides a good expressiveness to describe the structure of a system by using the classes, their attributes, and the relationships between the classes [Seo06]. For this reason, the UsiXML domain model uses the UML class diagram to describe the different entities manipulated through a User Interface.

In the next section, we describe the UsiXML domain meta-model which is based on the UML class diagram meta-model [OMG09].

#### 5.4.1. Overview
Figure 4 - UsiXML domain Meta-model adopted from the UML class diagram [OMG09]

Figure 4 shows the UsiXML domain meta-model adopted from the UML class diagram. The UML Class is the main concept of this meta-model. It describes a User Interface (UI) entity, its attributes and its operations. A UI entity attribute and an operation are represented respectively by the property class and operation class. In turn, the relationship between entities can be an association that describes a semantic relationship between entities or a generalization that describes a taxonomic relationship between them.

The meta-model depicted in Figure 1 is a simplified view of the meta-model of UML 2.2 described by OMG specification in [OMG09]. The classes of this meta-model are defined in the following section. Note that, the definition of the main classes of the meta-model is from [OMG09]. Another note is that, the class Property, described in [OMG09], is specialized into two classes: Attribute and an Association End in order to provide a more understandable meta-model.

- **Abstraction**: An abstraction is a relationship that relates two elements or sets of elements that represent the same concept at different levels of abstraction or from different viewpoints.
  - **Attributes**: No specific attribute

- **AggregationKind**: is an enumeration type that specifies the kind of aggregation of a property. AggregationKind is an enumeration of the following values:
  - none: Indicates that the property has no aggregation.
  - shared: Indicates that the property has a shared aggregation.
  - composite: Indicates that the property is aggregated compositely, i.e., the composite object has responsibility for the existence and storage of the composed objects (parts).

- **AssociationClass**: An AssociationClass can be seen as an association that also has class properties, or as a class that also has association properties.
  - **Attributes**: No specific attribute

- **AssociationEnd**: This class represents the role played by one class within an association.
  - **Attributes**
    - aggregation (AggregationKind): Specifies the kind of aggregation that applies to the association. The default value is none.
    - isComposite (Boolean): This is a derived value, indicating whether the aggregation of the association is composite or not.
    - navigable (Boolean): This is a derived value, indicating whether the association is navigable or not.

- **Association**: An association specifies a semantic relationship that can occur between typed instances. It has at least two members end represented by properties, each of which is connected to the type of the end.
  - **Attributes**
    - isDerived (Boolean): Specifies whether the association is derived from other model elements such as other associations. The default value
is false.

- **Attribute**: This class represents an attributes of a class or an interface.
  - **Attributes**:
    - `default (String)`: A String that is evaluated to give a default value for the attribute when an object of the owning Class is instantiated.
    - `isReadOnly (Boolean)`: If true, the attribute may only be read, and not written. The default value is false.

- **Class**: A class describes a set of objects (entities) that share the same specifications of features, and semantics.
  - **Attributes**:
    - `name (String)`: The name of the class.
    - `isAbstract (Boolean)`: If this attribute is true, then the class does not provide a complete declaration and can typically not be instantiated. Default value is false.
    - `visibility (VisibilityKind)`: Determines the class accessibility.

- **Classifier**: A classifier is an abstract class that describes the common elements of interfaces and classes.
  - **Attributes**:
    - `isAbstract (Boolean)`: If this attribute is true, then the class does not provide a complete declaration and can typically not be instantiated. Default value is false.

- **Dependency**: A dependency is a relationship that signifies that a single or a set of model elements requires other model elements for their specification or implementation.
  - **Attributes**:
    - No specific attribute

- **DirectedRelationship**: A directed relationship references one or more source elements and one or more target elements.
  - **Attributes**:
    - No specific attribute

- **DomainModel**: Domain model is a description of the classes of objects manipulated by a user while interacting with a system. A domain model is made up of submodels.
  - **Attributes**:
    - `domainName (String)`: the name of the domain model.
    - `description (String)`: the textual description of the domain model.

- **Generalization**: A generalization is a taxonomic relationship between a more general class and a more specific class. Each instance of the specific class is also an indirect instance of the general class. Thus, the specific class inherits the features of the more general class.
  - **Attributes**:
    - `isSubstitutable (Boolean)`: Indicates whether the specific class can be used wherever the general class can be used. If true, the execution traces of the specific class will be a superset of the execution traces of the general class. The default value is true.

- **GeneralizationSet**: A GeneralizationSet defines a particular set of Generalization relationships that describe the way in which a general class (or superclass) may be divided using specific subtypes. For example, a GeneralizationSet could define a partitioning of the class Person into two subclasses: Male Person and Female Person.
  - **Attributes**:
    - `isCovering (Boolean)`: Indicates whether or not the set of specific classes are covering for a particular general class. When isCovering is true, every instance of a particular general Class is also an instance of at least one of its specific Classes for the GeneralizationSet. When isCovering is false, there are one or more instances of the particular general Class that are not instances of at least one of its specific Classes defined for the GeneralizationSet.
    - `isDisjoint (Boolean)`: Indicates whether or not the set of specific classes in a Generalization relationship have instance in common. If isDisjoint is true, the specific Classes for a particular GeneralizationSet have no members in common; that is, their intersection is empty. If isDisjoint is false, the specific Classes in a particular GeneralizationSet have one or more members in common; that is, their
Interface: An interface is a kind of classifier that represents a declaration of a set of coherent public features and obligations. Since interfaces are declarations, they are not instantiable. Instead, an interface specification is implemented by an instance of an instantiable class, which means that the instantiable class presents a public facade that conforms to the interface specification.

- Attributes:
  - No specific attribute

InterfaceRealization: An InterfaceRealization is a specialized Realization relationship between a class and an Interface. This relationship signifies that the realizing class conforms to the contract specified by the Interface.

- Attributes:
  - No specific attribute

ModelElement: A Model element is a constituent of a domain model. This class represents an abstract generalization class that is specialized in the meta-model.

- Attributes:
  - No specific attribute

ModelNamedElement: A named element represents elements that may have a name. This class represents an abstract generalization class that is specialized in the meta-model.

- Attributes:
  - name (String): The name of the element.
  - visibility (VisibilityKind): Determines the element accessibility.

MultiplicityElement: A multiplicity Element class is represent an inclusive interval of non-negative integers beginning with a lower bound and ending with a (possibly infinite) upper bound

- Attributes:
  - isOrdered (Boolean): For a multivalued multiplicity, this attribute specifies whether the values in an instantiation of this element are sequentially ordered. Default is false.
  - isUnique (Boolean): For a multivalued multiplicity, this attributes specifies whether the values in an instantiation of this element are unique. Default is true.
  - lower (Integer): Specifies the lower bound of the multiplicity interval, if it is expressed as an integer.
  - upper (UnlimitedNatural): Specifies the upper bound of the multiplicity interval, if it is expressed as an unlimited natural.

Operation: An operation specifies the name, type, and the parameters, of a class operation.

- Attributes:
  - isOrdered (Boolean): Specifies whether the return parameter is ordered or not, if present.
  - isUnique (Boolean): Specifies whether the return parameter is unique or not, if present.
  - lower (Integer): Specifies the lower multiplicity of the return parameter, if present.
  - upper (UnlimitedNatural): Specifies the upper multiplicity of the return parameter, if present. This is derive

Parameter: is a specification of an argument used to pass information into or out of an invocation of a operation.

- Attributes:
  - defaut (String): specifies a String that represents a value to be used when no argument is supplied for the Parameter.
  - direction (ParameterDirectionKind): Indicates whether a parameter is being sent into or out of a operation

ParameterDirectionKind: Parameter Direction kind is an enumeration type that defines literals used to specify direction of parameters.
Parameter Direction kind is an enumeration of the following values:

- In: indicates that parameter values are passed into the behavioral element by the caller.
- Inout: indicates that parameter values are passed into a behavioral element by the caller and then back out to the caller from the behavioral element.
- Out: indicates that parameter values are passed from a behavioral element out to the caller.
• Return: indicates that parameter values are passed as return values from a behavioral element back to the caller.

**Property:** This class represents a common property of all the instances of a class, interface or an Association. This class is a generalization of an Attribute and an Association End

  o **Attributes:**
    - **isDerived** *(Boolean):* Specifies whether the Property is derived, i.e., whether its value or values can be computed from other information. The default value is false.

**Realization:** Realization is a specialized abstraction relationship between two sets of model elements, one representing a specification and the other represents an implementation of the latter.

  o **Attributes:**
    - No specific attribute

**Relationship:** A relationship references one or more related elements.

  o **Attributes:**
    - No specific attribute

**Type:** This class is used as a constraint on the range of values represented by a typed element. Type is an abstract generalization class.

  o **Attributes:**
    - No specific attribute

**VisibilityKind:** is an enumeration of the following values:

  - public
  - private
  - protected
  - package

**Usage:** usage is a relationship in which one element requires another element (or set of elements) for its full implementation or operation.

  o **Attributes:**
    - No specific attribute

### 5.4.2. Example

![UsiXML Domain Model Example](image)

**Error! Reference source not found.** gives an example of a UsiXML domain model expressed using a UML class diagram. In this domain model, the various entities manipulated through a User Interface are represented by classes (e.g., Person, Flight, etc.). Each class can have a set of properties and set of operations (e.g., Person class has two properties: DepartureDate, and ArrivedDate, and two operations: OpenFlight and CloseFlight). In the UsiXML domain model, two kinds of relationships can exist between entities (classes): 1) Association relationship: it consists of two classes, each playing a specific role (e.g., a client has the role to make a reservation). Each role is characterized by a constraint on the number of instances of a class connected across an association (multiplicity). Note that, an
association can represent a composition relation between a class and another one (e.g. an airport is composed of several terminals). Another note is the fact that, an association can have a property (e.g. an air company manages a flight of a specific number). This kind of association can be modeled as a class if the association has itself a set of properties (e.g. Class Stopover). 2) Generalization relationship: it links a super-class to one or several other sub-classes (e.g. the super class Person is a generalization of Client class and Passenger class).

5.5. Abstract User Interface

The Abstract User Interface (AUI) (corresponding to the Platform-Independent Model–PIM– in MDE) is an expression of the UI in terms of interaction spaces (or presentation units), independently of which interactors are available and even independently of the modality of interaction (graphical, vocal, haptic ...). An interaction space is a grouping unit that supports the execution of a set of logically connected tasks.

The chosen modeling is to compose the AbstractUIModel by a set of AbstractCompoundIUs that are themselves sets of AbstractInteractionUnit: every abstract object is then such an interaction unit. Each interaction unit may have one or more AbstractListeners. These listeners allow defining the dynamic of the model, the way the interaction units will react to the different events. Additionally, the AbstractInteractionUnits are related to states and transitions enabling the possibility to comply with the State Chart XML (SCXML) defined by W3C. (http://www.w3.org/TR/scxml/)

- **AbstractUIModel**: This model describes canonically a user interface in terms of abstract interaction units, relationships and listeners in a way that is independent from the concrete interaction units available on the targets. The abstract user interface model is independent of the target device or modality.

- **AbstractInteractionUnit**: Main entity of the model, every abstract object is an AbstractInteractionUnit.
  - **Attributes**:
    - id (String): identification string of the AbstractInteractionUnit.
    - role (String): name of the role played by the AbstractInteractionUnit.
    - importance (String): importance level of the AbstractInteractionUnit.
      - 5 level scale can be used like “Very high”/”High”/”Medium”/”Low”/”Very low”.
    - repetitionFactor (int): number of times the AbstractInteractionUnit is repeated in the parent entity.
    - hierarchyNumber (int): allows to order the AbstractInteractionUnit in its parent entity, in function of the other AbstractInteractionUnits contained in the same parent.
    - securityType (AuthenticationType): defines the type of authentication for the AbstractInteractionUnit.
    - securityMechanism (SecurityMechanism): defines the type of security mechanism for the AbstractInteractionUnit.
AuthenticationType

- Attributes
  - NONE
  - USER_PASSWORD
  - INTEGRATED
    - Stands for an authentication that can be done by the system, not by the user. For example, through a MAC address or a certificate.

SecurityMechanism

- Attributes
  - NONE
  - CAPTCHA

AbstractLocalization: Describes text attributes for AbstractInteractionUnits. It is useful for internationalization.

- Attributes
  - lang (String[2]): international code (2 letters) for the language supported by the AbstractLocalization.
  - label (String): description label of the AbstractInteractionUnit.
    - Example: “department”.
  - longLabel (String): label as it appears in the interface.
    - Example: “Department”.
  - shortLabel (String): short version of the label.
    - Example: acronym, like “dept”.
  - descLabel (String): description label of the AbstractInteractionUnit.
    - Example: “the name of the department”.
  - help (String): textual help provided specifically for the AbstractInteractionUnit.

AbstractCompoundIU: Composition of one or several AbstractInteractionUnits.

AbstractSelectionIU: Special type of AbstractCompoundIU representing a way to interact with the interface by selecting an item in a list.

- Attributes:
  - orderCriteria (String): criteria used to sort the selection.
    - Example: “alphabetical”
  - isContinuous (Boolean): specifies if the selection is continuous.
    - Example: A selection that may be specialized at concrete level as a voice selection among any number between 0 and 1.
  - start (Float): starting number (for numerical selection).
  - end (Float): ending number (for numerical selection).
  - step (Float): interval between two successive items, by starting by start and ending by end (for numerical selection).
  - isExpandable (Boolean): specifies if the user can add item in the selection.
  - selectionType (SelectionType): specifies the type of selection of the unit.
    - Rating (Boolean): specifies if the unit is used for a rating.

SelectionType

- Attributes
  - UNDEFINED
  - NOSELECTION
  - SINGLESELECTION
- One item in the list is selectable.
- **SINGLEINTERVAL**
  - An interval of the list is selectable.
    - Example: for a list of digit between 0 and 9, an interval may be the digits between 2 and 5.
- **MULTIPLEINTERVAL**
  - Multiple intervals of the list are selectable.
    - Example: for a list of digit between 0 and 9, an interval may be the digits between 2 and 5 and a second one between 7 and 8.

- **AbstractElementaryIU**: Atomic **AbstractInteractionUnit** that can be of 2 types: **AbstractDataIU** or **AbstractTriggerIU**.

- **AbstractDataIU**: Interaction unit allowing to link data from the Domain Model with elements of the abstract user interface.
  - **Attributes**:
    - **domainReference** (String): reference allowing to link the Domain Model in order to populate the **AbstractDataIU**.
    - **maxCardinality** (Integer): maximum number of items in the **AbstractDataIU**.
    - **minCardinality** (Integer): minimum number of items in the **AbstractDataIU**.
    - **defaultValue** (String): default value or concatenation of default values that can be used if the required value is not available in the Domain Model.
    - **dataType** (Data): type of the data.
    - **dataLength** (Integer): size of the data, in bytes.
    - **dataFormat** (String): format of the data.
      - Example: .doc, .pdf, .xml, ...
    - **dataIUType** (**AbstractDatayIUType**): type of interaction.
    - **isDynamic** (Boolean): specifies if the content can evolve in the time.

- **DataType**
  - **Attributes**
    - **BOOLEAN**
    - **HOUR**
    - **DATE**
    - **NATURAL**
    - **INTEGER**
    - **REAL**
    - **TEXT**
    - **SECRET**
      - Stands for a type of data that we want to keep secret.
    - **ARRAY**
    - **MULTIMEDIA**

- **AbstractDataIUType**
  - **Attributes**
    - **INPUT**
    - **OUTPUT**
    - **INPUT_OUTPUT**

- **AbstractTriggerIU**: Interaction unit allowing triggering an event.
  - **Attributes**
    - **triggerIUType** (**AbstractTriggerIUType**): type of event triggered.

- **AbstractTriggerIUType**
  - **Attributes**
- NAVIGATION
- OPERATION
- OPERATION_NAVIGATION

- **AbstractListener**: Entity used to describe the behavior of the AbstractInteractionUnit by using Event-Condition-Action (ECA) rules.
  - **Attributes**
    - `id` (String): identification string of the AbstractListener.
    - `name` (String): name of the AbstractListener.

- **Rule**
  - **Attributes**
    - `name` (String): name of the AbstractListener.

- **Justification**: The justification is a kind of motivation for the ECA rules, it is not used for describing the interface itself but more for documentation purposes.
  - **Attributes**
    - `content` (String): text representing the justification itself.
    - `justificationType` (JustificationType): type of justification, the different possibilities are described in the AbstractJustificationType description.

- **JustificationType**
  - **Attributes**
    - `CLAIM`: assertion put forward publicly for general acceptance with the implication that there are underlying 'reasons' that could show it to be 'well founded' and therefore entitled to be generally accepted.
      - Example: "Our organization should cut costs next quarter"
    - `GROUND`: the term 'ground' refers to the specific facts relied on to support a given claim.
      - Example: "Our organization has lost money the last 3 quarters"
    - `WARRANT`: assertion that entitles you to interpret or link the grounds (facts) as support of the claim.
      - Example: "When losing money, organizations should cut expenses as much as possible"
    - `BACKING`: the 'backing' consists of a very general set of background assumptions which, in effect, legitimize the basis for believing in the warrant. That is, if the warrant is not accepted on its surface, then the backing is called into play to add deeper support to the argument.
      - Example: "A business can’t continue to lose money and stay in business"
    - `REBUTTAL`: the 'rebuttal' presents the possible exceptions or objections as to why the claim, the grounds, the warrants, or the backing may not hold for the situation under discussion.
      - Example: "That may be true in general, but not with the customers of our company. Besides that, times have changed; the economy as changed; the dollar has fallen in value."
    - `QUALIFIER`: word that indicates how strongly the claim is being asserted, or how likely that something might occur.
      - Example: "probably", "certainly", "very likely", etc.
    - `UNDEFINED`

- **EventExpression**: specifies a set of events with relationships between them. The events are represented by AtomicEvents and are related by TemporalEventExpressions.
  - **Attributes**
    - `name` (String): name of the EventExpression.

- **AtomicEvent**
  - **Attributes**
    - `type` (EventType): type of event, the different possibilities are described in the EventType description.
    - `Specification` (String): kind of argument that, used in conjunction with type, allows specifying Event to listen.

- **EventType**
  - **Attributes**
    - `onDataInput`: new data has been entered by the user
    - `onErroneousDataInput`: new erroneous data has been entered by the user
- **onDataOutput**: new data has been output through the interface
- **onDataSelection**: some data has been selected by the user
- **onOperationTrigger**: an AbstractOperationIU has been activated
- **onNavigationTrigger**: an AbstractNavigationIU has been activated
- **onIUFocusIn**: an AbstractInteractionUnit has been focused in
- **onIUFocusOut**: an AbstractInteractionUnit has been focused out
- **onModelUpdate**: the Domain Model has been updated

### TemporalEventExpression

- **Attributes**
  - **type** (*TemporalOperator*): type of relationship involved between two AtomicEvents, the different possibilities are described in the *TemporalOperator* description.

### EventType

- **Attributes**
  - **ENABLING**
  - **CHOICE**
  - **CONCURRENCY**
  - **SUSPEND**
  - **DISABLING**
  - **ORDERINDEPENDANCE**

### Condition

- **Attributes**
  - **specification** (*String*): logical formula representing the condition to evaluate, for the moment under the form of a String.

### ActionExpression

- **Attributes**
  - **name** (*String*): name of the *ActionExpression*.

### AtomicAction

- **Attributes**
  - **type** (*ActionType*): type of action, the different possibilities are described in the *ActionType* description.
  - **specification** (*String*): kind of argument that, used in conjunction with type, allows specifying the action to launch.
    - Example: for an *AbstractActionType* ‘modelUpdate’, the actionSpecification could specify which field to change and what is the new value.

### AbstractActionType

- **Attributes**
  - **modelSearch**: search for model elements based on logical formula
  - **modelCreate**: create a new model in the Domain Model
  - **modelRead**: read a specified field in the Domain Model
  - **modelUpdate**: update a field in the Domain Model
  - **modelDelete**: delete a field in the Domain Model
  - **modelInvoke**: perform a query external to the current Domain Model
  - **modelReset**: reset the Domain Model with the initial parameters
  - **modelCopy**: copy the Domain Model
  - **listenerCreate**: create a new listener on a specified AbstractInteractionUnit
  - **listenerDelete**: delete a listener of a specified AbstractInteractionUnit
- **eventDispatch**: dispatch the event to another AbstractInteractionUnit
- **IUOpen**: open a specified AbstractInteractionUnit
- **IUClose**: close a specified AbstractInteractionUnit
- **IUActivate**: activate a specified AbstractInteractionUnit
- **IUDesactivate**: desactivate a specified AbstractInteractionUnit
- **IUEmphasize**: focus in a specified AbstractInteractionUnit
- **IUDesemphasize**: focus out a specified AbstractInteractionUnit

*TemporaActionExpression*

  - **Attributes**
    - **type** (*TemporalOperator*): type of relationship involved between two AtomicActions, the different possibilities are described in the TemporalOperator description.

*State*: possible state of an AbstractInteractionUnit. An AbstractInteractionUnit may have many different states.

  - **Attributes**
    - **id** (*String*): identification string of the state.
    - **Description** (*String*): description of the state.

*Transition*: is related to State by two relationships meaning that a transition has source state and a target state. Additionally it is related to EventExpression and ActionExpression.

### 5.6. Concrete User Interface

For details about this section and subsections, see deliverable [UsiXML11]

#### 5.6.1. Overview

#### 5.6.2. Main entities

#### 5.6.3. ConcreteGraphicalIU

#### 5.6.4. ConcreteGraphicalStyle

#### 5.6.5. ConcreteGraphicalListener

### 5.7. Transformation

For details about this section and subsections, see deliverable [UsiXML11]

#### 5.7.1. Overview

#### 5.7.2. Structure of packages

#### 5.7.3. Package Context

#### 5.7.4. Package QOC

#### 5.7.5. Package Transformation

#### 5.7.6. Connections with other transformation languages

#### 5.7.7. A Lab Study of the Transformation Meta-model

#### 5.7.8. Analysis of the transformation meta-model against the taxonomy of model transformations proposed by Mens et al.

### 5.8. Workflow

For details about this section and subsections, see deliverable [UsiXML11]

#### 5.8.1. Overview

#### 5.8.2. Example
5.9. Quality

For details about this section and subsections, see deliverable [UsiXML11]

5.9.1. Summary

5.9.2. Quality perspectives

5.9.3. The Meta-Model

5.9.4. Objects, Methods and Results, Global Quality vs Local Quality

5.9.5. Classes of the Quality Meta-model

5.9.6. How to build a Quality Model

5.9.7. Example

5.10. Mapping

For details about this section and subsections, see deliverable [UsiXML11]

5.10.1. Overview

5.11. Adaptation to the context of use

For details about this section, see deliverable [UsiXML11]

5.12. Interactor

For details about this section, see deliverable [UsiXML11]

6. ABSTRACT SYNTAX

6.1. Task

The first UML Class diagram representing the Meta-Model is transformed into the second UML Class diagram simplified for obtaining the Abstract Syntax for Task
6.2. **Abstract User Interface**

The first UML Class diagram representing the Meta-Model is transformed into the second UML Class diagram simplified for obtaining the Abstract Syntax of Abstract User Interface.
7. CONCRETE SYNTAXES

7.1. XSD

7.1.1. Task

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema targetNamespace="http://UsiXML-XSD/TaskModel"
  xmlns:usi="http://UsiXML-XSD/TaskModel"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:package="http://UsiXML-XSD/Package" elementFormDefault="qualified">

  <xs:element name="TaskModel">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        The TaskModel meta-class describes the interaction between the entities that are parts of the system, and the system itself, in terms of tasks that are performed by the entities of the system. It defines a set of tasks as a whole that are decomposed into sub-tasks as parts structuring the system into task hierarchies that define the behavior of the system.
      </xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="name" type="xs:string" minOccurs="1" maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Defines the name of the task model.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element ref="usi:task" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="task">
    <xs:complexType>
      <xs:documentation xml:lang="en">
        The Task meta-class describes a task performed by one or more system entities. From the structural point of view, a task can be atomic or composed. While composed tasks represent complex tasks that can be decomposed into...
      </xs:documentation>
    </xs:complexType>
  </xs:element>
</xs:schema>
```
simpler sub-tasks; atomic tasks represent an indivisible action that is performed by a single entity of the system. Thus, collaborative tasks are represented as composed tasks that can be decomposed into atomic tasks performed by single entities. To define the temporal relationship among subtasks, the parent task defines a set of expressions that describes the temporal relationships among subtasks. The tasks can also be decorated in order to give information such as their criticity, optionality and other details.

</xs:documentation>
</xs:annotation>
</xs:sequence>

<xsd:element name="name" type="xs:string" minOccurs="0" maxOccurs="1">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      Defines the task name
    </xs:documentation>
  </xs:annotation>
</xsd:element>

<xsd:element name="description" type="xs:string" minOccurs="0" maxOccurs="1">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      Explains what the task is about by a textual description as complete as possible.
    </xs:documentation>
  </xs:annotation>
</xsd:element>

<xsd:element name="precondition" type="xs:string" minOccurs="0" maxOccurs="1">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      Describe the condition in which the task can be executed.
    </xs:documentation>
  </xs:annotation>
</xsd:element>

<xsd:element name="postcondition" type="xs:string" minOccurs="0" maxOccurs="1">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      Describe how the task changes the system.
    </xs:documentation>
  </xs:annotation>
</xsd:element>

<xsd:element ref="usi:task" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element name="expression" type="usi:TaskExpression" minOccurs="0" maxOccurs="unbounded"/>

</xs:sequence>

<xsd:attribute name="id" type="xs:ID" use="required"/>
<xsd:attribute name="canonicalTaskType" type="usi:TTYPE" use="optional">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      The canonical type of a task describes the action of the task without going deeper on details on how it applies.
    </xs:documentation>
  </xs:annotation>
</xsd:attribute>

</xsd:complexType>

<xsd:key name="id_task">
  <xs:selector xpath="/usi:task"/>
  <xs:field xpath="@id"/>
</xsd:key>

<xsd:keyref name="task_decoration" refer="usi:id_task">
  <xs:selector xpath="usi:expression"/>
  <xs:field xpath="@decoratedTask"/>
</xsd:keyref>

<xsd:key name="id_expression">
  <xs:selector xpath="/usi:expression"/>
  <xs:field xpath="@id"/>
</xsd:key>

<xsd:keyref name="expressionDecoration" refer="usi:id_expression">
  <xs:selector xpath="/usi:expression"/>
  <xs:field xpath="@expressionDecoration"/>
</xsd:keyref>
<xs:element>  
  
<xs:simpleType name="TTYPE">  
  <xs:restriction base="xs:string">  
    <xs:enumeration value="CREATE">  
      <xs:annotation>  
        <xs:documentation xml:lang="en">Not available</xs:documentation>  
      </xs:annotation>  
    </xs:enumeration>  
    <xs:enumeration value="CONVEY">  
      <xs:annotation>  
        <xs:documentation xml:lang="en">Not available</xs:documentation>  
      </xs:annotation>  
    </xs:enumeration>  
    <xs:enumeration value="REINITIALIZE">  
      <xs:annotation>  
        <xs:documentation xml:lang="en">Not available</xs:documentation>  
      </xs:annotation>  
    </xs:enumeration>  
    <xs:enumeration value="FILTER">  
      <xs:annotation>  
        <xs:documentation xml:lang="en">Not available</xs:documentation>  
      </xs:annotation>  
    </xs:enumeration>  
    <xs:enumeration value="DELETE">  
      <xs:annotation>  
        <xs:documentation xml:lang="en">Not available</xs:documentation>  
      </xs:annotation>  
    </xs:enumeration>  
    <xs:enumeration value="DUPLICATE">  
      <xs:annotation>  
        <xs:documentation xml:lang="en">Not available</xs:documentation>  
      </xs:annotation>  
    </xs:enumeration>  
    <xs:enumeration value="NAVIGATE">  
      <xs:annotation>  
        <xs:documentation xml:lang="en">Not available</xs:documentation>  
      </xs:annotation>  
    </xs:enumeration>  
    <xs:enumeration value="PERCEIVE">  
      <xs:annotation>  
        <xs:documentation xml:lang="en">Not available</xs:documentation>  
      </xs:annotation>  
    </xs:enumeration>  
    <xs:enumeration value="MOVE">  
      <xs:annotation>  
        <xs:documentation xml:lang="en">Not available</xs:documentation>  
      </xs:annotation>  
    </xs:enumeration>  
    <xs:enumeration value="MODIFY">  
      <xs:annotation>  
        </xs:annotation>  
    </xs:enumeration>  
  </xs:restriction>  
</xs:simpleType>  
</xs:element>
The TaskExpression is an abstract meta-class that defines expressions on tasks. Task expressions allow developers to define: (a) task attributes that vary according to the situation (i.e. criticity, frequency, etc.), and (b) temporal relationships among tasks. To represent a TaskExpression we employ the Composite design pattern where the TaskExpression plays the role of Component.

The TemporalRelationship meta-class is a TaskExpression that allows developers to define the temporal relationship among TaskExpressions. TemporalRelationships are temporal operations defined as LOTOS operators on TaskExpressions. The LOTOS temporal operation is defined as a TTEMPORALOPERATOR that defines the ENABLING, CONCURRENCY, DISABLING, SUSPEND, ORDERINDEPENDENCE and CHOICE temporal operations. The TemporalRelationship meta-class plays the role of Composite in the Composite design pattern rooted on TaskExpression.
<xs:extension base="usi:TaskExpression">
  <xs:sequence>
    <xs:element name="formula" type="xs:string" minOccurs="0" maxOccurs="1">
      <xs:annotation>
        <xs:documentation xml:lang="en">Not available</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="previous" type="usi:TaskExpression" minOccurs="1" maxOccurs="1" />
    <xs:element name="next" type="usi:TaskExpression" minOccurs="1" maxOccurs="1" />
  </xs:sequence>
  <xs:attribute name="type" type="usi:TTEMPORALOPERATOR" use="required">
    <xs:annotation>
      <xs:documentation xml:lang="en">Defines the type of the temporal operator in a TemporalRelationship.</xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:extension>

<xs:complexType name="Decoration">
  <xs:annotation>
    <xs:documentation xml:lang="en">The Decoration is an abstract meta-class of TaskExpression that allows developers to define TaskExpression attributes that vary according to the situation they are performed (i.e. criticity, frequency, etc.). The Decorator plays the role of abstract Component in the Composite design pattern rooted on the TaskExpression meta-class. It has two concrete meta-classes according to the type of element they are decorating (Task or TaskExpression).
    </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:attribute type="xs:IDREF" name="decoratedTask" use="required" />
    <xs:attribute type="xs:int" name="minIteration" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">Defines the minimum cardinality of the iteration for the task or task expression. Note that if this attribute is set to 0 then the task or task expression it refers to is optional.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute type="xs:int" name="maxIteration" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">Defines the maximum cardinality of the iteration for the task or task expression. Note that if this attribute is set to -1 then it means that the maximum cardinality is set to infinite.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute type="xs:int" name="criticity" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">Defines the criticity level of a the task (from 0 to 10). If a task is critic, the criticity level will be high.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute type="xs:int" name="frequency" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">Defines the frequency of the task (from 0 to 10).</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexContent>
</xs:complexType>
If a task happens often, the frequency will be high.

```xml
<xs:attribute type="xs:int" name="centrality" use="optional">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      Defines if the task is important for the current context.
      A log in task will not be central for visiting a website, but would become central when the user want to access to his account.
    </xs:documentation>
  </xs:annotation>
</xs:attribute>

<xs:attribute type="xs:int" name="minExecutionTime" use="optional">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      Set the minimum time for the execution.
    </xs:documentation>
  </xs:annotation>
</xs:attribute>

<xs:attribute type="xs:int" name="maxExecutionTime">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      Defines if the task is important for the current context. A log in task will not be central for visiting a website, but would become central when the user want to access to his account.
    </xs:documentation>
  </xs:annotation>
</xs:attribute>
```

The ExpressionDecoration is a concrete meta-class of Decoration that allows developers to define Decorations on TemporalRelationships.

```xml
<xs:simpleType name="TTEMPORALOPERATOR">
  <xs:restriction base="xs:string">
    <xs:enumeration value="ENABLING">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Not available
        </xs:documentation>
      </xs:annotation>
    </xs:enumeration>
    <xs:enumeration value="CONCURRENCY">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Not available
        </xs:documentation>
      </xs:annotation>
    </xs:enumeration>
    <xs:enumeration value="DISABLING">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Not available
        </xs:documentation>
      </xs:annotation>
    </xs:enumeration>
    <xs:enumeration value="SUSPEND">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Not available
        </xs:documentation>
      </xs:annotation>
    </xs:enumeration>
  </xs:restriction>
</xs:simpleType>
```
<xs:annotation>
  <xs:documentation xml:lang="en">
    Not available
  </xs:documentation>
</xs:annotation>
</xs:enumeration>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
</xs:complexType>
7.1.2. Abstract User Interface

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://UsiXML-XSD/AbstractUIModel"
elementFormDefault="qualified">
<xs:element name="abstractUIModel">
<xs:annotation>
<xs:documentation xml:lang="en">
This model describes canonically a user interface in terms of abstract
interaction units, relationships and listeners in a way that is
independent from the concrete interaction units available on the
targets. The abstract user interface model is independent of the
target device or modality.
</xs:documentation>
</xs:annotation>
<xs:complexType>
<xs:sequence>
<xs:element name="abstractCompoundIUs" type="usi:AbstractCompoundIU" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>
<!-- #PTH:le id est maintenant un string -->
<xs:element name="id" type="xs:string" use="required"/>
<!-- #PTH:fin -->

<!-- --><xs:attribute name="role" type="xs:string"/>
<!-- --><xs:attribute name="importance" type="xs:int"/>
<!-- --><xs:attribute name="repetitionFactor" type="xs:int"/>

<!-- #PTH:le est maintenant un string -->
<xs:element name="transition" type="usi:Transition" minOccurs="0" maxOccurs="unbounded"/>
<!-- #PTH:fin -->

<xs:complexType name="AbstractCompoundIU">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      Composition of one or several AbstractInteractionUnit
    </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="usi:AbstractInteractionUnit">
      <xs:sequence>
        <xs:element name="abstractInteractionUnit" type="usi:AbstractInteractionUnit" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="AbstractInteractionUnit">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      Main entity of the model, every abstract object is an AbstractInteractionUnit.
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="name" type="usi:AbstractLocalization" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="listener" type="usi:AbstractListener" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="state" type="usi:State" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
Number of times the AbstractInteractionUnit is repeated in the parent entity.

</xs:documentation>
</xs:annotation>
</xs:attribute>
<x:s:attribute name="hierarchyNumber" type="xs:int"/>

Allows to order the AbstractInteractionUnit in its parent entity, in function of the other AbstractInteractionUnits contained in the same parent.
</xs:documentation>
</xs:annotation>
</xs:attribute>
<x:s:attribute name="securityType" type="usi:AuthenticationType"/>

Defines the type of authentication for the AbstractInteractionUnit.
</xs:documentation>
</xs:annotation>
</xs:attribute>
<x:s:attribute name="securityMechanism" type="usi:SecurityMechanism"/>

Defines the type of security mechanism for the AbstractInteractionUnit.
</xs:documentation>
</xs:annotation>
</xs:attribute>
</xs:complexType>

<x:s:complexType name="State"/>

Possible state of an AbstractInteractionUnit.
An AbstractInteractionUnit may have many different states.
</xs:documentation>
</xs:annotation>
</xs:complexType>

<x:s:complexType name="Transition">

Is related to State by two relationships meaning that a transition has source state and a target state. Additionally it is related to EventExpression and ActionExpression
</xs:documentation>
</xs:annotation>
</xs:complexType>

<x:s:simpleType name="SecurityMechanism">

<x:s:restriction base="xs:string">

<x:s:enumeration value="NONE" />
<x:s:enumeration value="CAPTCHA" />
</xs:restriction>
</xs:simpleType>
<xs:simpleType name="AuthenticationType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="NONE"/>
    <xs:enumeration value="USER_PASSWORD"/>
    <xs:enumeration value="INTEGRATED"/>
  </xs:restriction>
</xs:simpleType>

<xs:complexType name="AbstractListener">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      Entity used to describe the behavior of the AbstractInteractionUnit by using Event-Condition-Action (ECA) rules.
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="name" type="xs:string" minOccurs="0" maxOccurs="1">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Name of the AbstractListener.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="rules" type="usi:Rule" minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="Rule">
  <xs:sequence>
    <xs:element name="name" type="xs:string" minOccurs="0" maxOccurs="1">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Name of the AbstractListener.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="justification" type="usi:Justification" minOccurs="0" maxOccurs="1"/>
    <xs:element name="event" type="usi:EventExpression" minOccurs="1" maxOccurs="1"/>
    <xs:element name="condition" type="usi:Condition" minOccurs="0" maxOccurs="1"/>
    <xs:element name="action" type="usi:ActionExpression" minOccurs="1" maxOccurs="1"/>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="Justification">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      The justification is a kind of motivation for the ECA rules, it is not used for describing the interface itself but more for documentation purposes.
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="content" type="xs:string" minOccurs="0" maxOccurs="1">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Text representing the justification itself.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="type" type="usi:JustificationType" minOccurs="0" maxOccurs="1">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Type of justification, the different possibilities are described in the AbstractJustificationType description.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string" use="required"/>
<xs:complexType>
  <xs:simpleType name="JustificationType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="CLAIM"/>
      <xs:enumeration value="GROUND"/>
      <xs:enumeration value="WARRANT"/>
      <xs:enumeration value="BACKING"/>
      <xs:enumeration value="REBUTTAL"/>
      <xs:enumeration value="QUALIFIER"/>
      <xs:enumeration value="UNDEFINED"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="EventExpression">
    <xs:annotation>
      <xs:documentation xml:lang="en">Specifies a set of events with relationships between them. The events are represented by AtomicEvents and are related by TemporalEventExpressions.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="name" type="xs:string" minOccurs="0" maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">Name of the EventExpression.</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:complexType>
<xs:annotation/>
<xs:element/>
<xs:sequence/>

<!-- Jb: plus besoin -->
<!--
<xs:attribute name="id" type="xs:string" use="required" /> -->
<!--
Jb: fin -->
<xs:complexType>
<xs:complexType name="TemporalEventExpression">
<xs:complexContent>
<xs:extension base="usi:EventExpression">
<xs:sequence>
<xs:element name="type" type="usi:TemporalOperator" minOccurs="0" maxOccurs="1">
<xs:annotation>
<xs:documentation xml:lang="en">Type of relationship involved between two AtomicEvents, the different possibilities are described in the TemporalOperator description.
</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="previous" type="usi:EventExpression" minOccurs="1" maxOccurs="1" />
<xs:element name="next" type="usi:EventExpression" minOccurs="1" maxOccurs="1" />
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>

<xs:complexType name="TemporalOperator">
<xs:restriction base="xs:string">
<xs:enumeration value="ENABLING" />
<xs:enumeration value="CHOICE" />
<xs:enumeration value="CONCURRENCY" />
<xs:enumeration value="SUSPEND" />
<xs:enumeration value="DISABLING" />
<xs:enumeration value="ORDERINDEPENDANCE" />
</xs:restriction>
</xs:simpleType>

<xs:complexType name="ActionExpression">
<xs:annotation>
<xs:documentation xml:lang="en">Specifies a set of actions with relationships between them. The actions are represented by AtomicActions and are related by TemporalActionExpressions.
</xs:documentation>
</xs:annotation>
<xs:sequence>
<xs:element name="name" type="xs:string" minOccurs="0" maxOccurs="1">
<xs:annotation>
<xs:documentation xml:lang="en">Name of the ActionExpression.
</xs:documentation>
</xs:annotation>
</xs:element>
</xs:sequence>

<!-- PTH: ai changer ID en String -->
<!--
Jb : plus besoin -->
<!--
<xs:attribute name="id" type="xs:string" use="required" /> -->
<!--
Jb: fin -->
<!-- PTH: fin de chgt -->
</xs:complexType>

<xs:complexType name="TemporalActionExpression">
<xs:complexContent>
<xs:extension base="usi:ActionExpression">
<xs:sequence>
<xs:element name="type" type="usi:TemporalOperator" minOccurs="0" maxOccurs="1">
<xs:annotation>
<xs:documentation xml:lang="en">Type of relationship involved between two AtomicActions, the different possibilities are described in the TemporalOperator description.
</xs:documentation>
</xs:annotation>
</xs:element>
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="Condition">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      Logical test that, if satisfied or evaluates to true, causes the action to be carried out.
    </xs:documentation>
  </xs:annotation>
  <xs:all>
    <xs:element name="specification" type="xs:string" minOccurs="0" maxOccurs="1">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Logical formula representing the condition to evaluate, for the moment under the form of a String.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:all>
  <xs:attribute name="id" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="AbstractLocalization">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      Describes text attributes for AbstractInteractionUnits. It is useful for internationalization.
    </xs:documentation>
  </xs:annotation>
  <xs:all>
    <xs:element name="label" type="xs:string" minOccurs="0" maxOccurs="1">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Description label of the AbstractInteractionUnit.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="longLabel" type="xs:string" minOccurs="0" maxOccurs="1">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Label as it appears in the interface.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="shortLabel" type="xs:string" minOccurs="0" maxOccurs="1">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Short version of the label.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="descLabel" type="xs:string" minOccurs="0" maxOccurs="1">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Description label of the AbstractInteractionUnit.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="help" type="xs:string" minOccurs="0" maxOccurs="1">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Textual help provided specifically for the AbstractInteractionUnit.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:all>
  <xs:attribute name="id" type="xs:string" use="optional"/>
</xs:complexType>
<xs:attribute name="lang" type="xs:string" use="required">
  <xs:annotation>
    <xs:documentation xml:lang="en">International code (2 letters) for the language supported by the AbstractLocalization.</xs:documentation>
  </xs:annotation>
</xs:attribute>

<xs:complexType name="AbstractDataIU">
  <xs:annotation>
    <xs:documentation xml:lang="en">Interaction unit allowing to link data from the Domain Model with elements of the abstract user interface.</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="usi:AbstractElementaryIU">
      <xs:sequence>
        <xs:element name="domainReference" type="xs:string" minOccurs="0" maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">Reference allowing to link the Domain Model in order to populate the AbstractDataItem.</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="defaultValue" type="xs:string" minOccurs="0" maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">Default value or concatenation of default values that can be used if the required value is not available in the Domain Model.</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="dataFormat" type="xs:string" minOccurs="0" maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">Format of the data.</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="maxCardinality" type="xs:int">
        <xs:annotation>
          <xs:documentation xml:lang="en">Maximum number of items in the AbstractDataIU.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="minCardinality" type="xs:int">
        <xs:annotation>
          <xs:documentation xml:lang="en">Minimum number of items in the AbstractDataIU.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="dataType" type="usi:DataTypeType">
        <xs:annotation>
          <xs:documentation xml:lang="en">Type of the data.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="dataLength" type="xs:int">
        <xs:annotation>
          <xs:documentation xml:lang="en">Size of the data, in bytes.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="isDynamic" type="xs:boolean">\n        <xs:annotation>
          <xs:documentation xml:lang="en">Maximum number of items in the AbstractDataIU.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:annotation>
  <xs:documentation xml:lang="en">
    Specifies if the content can evolve in the time.
  </xs:documentation>
</xs:annotation>

<xs:attribute name="dataType" type="usi:AbstractDataIUType">
  <xs:documentation xml:lang="en">
    Type of interaction.
  </xs:documentation>
</xs:attribute>

<xs:simpleType name="DataTypeType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="BOOLEAN"/>
    <xs:enumeration value="HOUR"/>
    <xs:enumeration value="DATE"/>
    <xs:enumeration value="NATURAL"/>
    <xs:enumeration value="INTEGER"/>
    <xs:enumeration value="REAL"/>
    <xs:enumeration value="TEXT"/>
    <xs:enumeration value="SECRET"/>
    <xs:enumeration value="ARRAY"/>
    <xs:enumeration value="MULTIMEDIA"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="AbstractDataIUType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="INPUT"/>
    <xs:enumeration value="OUTPUT"/>
    <xs:enumeration value="INPUT_OUTPUT"/>
  </xs:restriction>
</xs:simpleType>

<xs:complexType name="AbstractElementaryIU">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      Atomic AbstractInteractionUnit that can be of 2 types: AbstractDataIU or AbstractTriggerIU.
    </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="usi:AbstractInteractionUnit"/>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="AbstractSelectionIU">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      Special type of AbstractCompoundIU representing a way to interact with the interface by selecting an item in a list.
    </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="usi:AbstractCompoundIU"/>
    <xs:sequence>
      <xs:element name="orderCriteria" type="xs:string" minOccurs="0" maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Criteria used to sort the selection.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="isContinuous" type="xs:boolean"/>
  </xs:complexContent>
</xs:complexType>
Specifies if the selection is continuous.

Starting number (for numerical selection).

Ending number (for numerical selection).

Interval between two successive items, by starting by start and ending by end (for numerical selection).

Specifies if the user can add item in the selection.

Specifies the type of selection of the unit.

Specifies if the unit is used for a rating.

One item in the list is selectable.

An interval of the list is selectable.

An interval of the list is selectable.
Multiple intervals of the list are selectable.

Interaction unit allowing triggering an event.

Type of event triggered.

Search for model elements based on logical formula

Create a new model in the Domain Model

Read a specified field in the Domain Model

Update a field in the Domain Model

Delete a field from the Domain Model
Delete a field in the Domain Model

```xml
<xs:enumeration
    value="modelInvoke">
    <xs:annotation
        xml:lang="en">
        Perform a query external to the current Domain Model
    </xs:annotation>
</xs:enumeration>

<xs:enumeration
    value="modelReset">
    <xs:annotation
        xml:lang="en">
        Reset the Domain Model with the initial parameters
    </xs:annotation>
</xs:enumeration>

<xs:enumeration
    value="modelCopy">
    <xs:annotation
        xml:lang="en">
        Copy the Domain Model
    </xs:annotation>
</xs:enumeration>

<xs:enumeration
    value="listenerCreate">
    <xs:annotation
        xml:lang="en">
        Create a new listener on a specified AbstractInteractionUnit
    </xs:annotation>
</xs:enumeration>

<xs:enumeration
    value="listenerDelete">
    <xs:annotation
        xml:lang="en">
        Delete a listener of a specified AbstractInteractionUnit
    </xs:annotation>
</xs:enumeration>

<xs:enumeration
    value="eventDispatch">
    <xs:annotation
        xml:lang="en">
        Dispatch the event to another AbstractInteractionUnit
    </xs:annotation>
</xs:enumeration>

<xs:enumeration
    value="IUOpen">
    <xs:annotation
        xml:lang="en">
        Open a specified AbstractInteractionUnit
    </xs:annotation>
</xs:enumeration>

<xs:enumeration
    value="IUClose">
    <xs:annotation
        xml:lang="en">
        Close a specified AbstractInteractionUnit
    </xs:annotation>
</xs:enumeration>

<xs:enumeration
    value="IUActivate">
    <xs:annotation
        xml:lang="en">
        Activate a specified AbstractInteractionUnit
    </xs:annotation>
</xs:enumeration>

<xs:enumeration
    value="IUDesactivate">
    <xs:annotation
        xml:lang="en">
        Desactivate a specified AbstractInteractionUnit
    </xs:annotation>
</xs:enumeration>
```
<xs:simpleType name="EventType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="IUEmphasize">
      <xs:annotation>
        <xs:documentation xml:lang="en">Focus in a specified AbstractInteractionUnit</xs:documentation>
      </xs:annotation>
    </xs:enumeration>
    <xs:enumeration value="IUDesemphasize">
      <xs:annotation>
        <xs:documentation xml:lang="en">Focus out a specified AbstractInteractionUnit</xs:documentation>
      </xs:annotation>
    </xs:enumeration>
  </xs:restriction>
</xs:simpleType>

<xs:complexType name="AtomicAction">
  <xs:complexContent>
    <xs:extension base="usi:ActionExpression">
      <xs:sequence>
        <xs:element name="type" type="usi:ActionType" minOccurs="0" maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">Type of action, the different possibilities are described in the ActionType description.</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="specification" type="xs:string" minOccurs="0" maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">Kind of argument that, used in conjunction with type, allows specifying the action to launch.</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="AtomicEvent">
  <xs:complexContent>
    <xs:extension base="usi:EventExpression">
      <xs:sequence>
        <xs:element name="type" type="usi:EventType" minOccurs="0" maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">Type of event, the different possibilities are described in the EventType description.</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="specification" type="xs:string" minOccurs="0" maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">Kind of argument that, used in conjunction with type, allows specifying Event to listen.</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:simpleType name="EventType"/>
<xs:restriction base="xs:string"/>
<xs:enumeration value="onDataInput">
    <xs:annotation>
        <xs:documentation xml:lang="en">
            New data has been entered by the user
        </xs:documentation>
    </xs:annotation>
</xs:enumeration>
<xs:enumeration value="onErroneousDataInput">
    <xs:annotation>
        <xs:documentation xml:lang="en">
            New erroneous data has been entered by the user
        </xs:documentation>
    </xs:annotation>
</xs:enumeration>
<xs:enumeration value="onDataOutput">
    <xs:annotation>
        <xs:documentation xml:lang="en">
            New data has been output through the interface
        </xs:documentation>
    </xs:annotation>
</xs:enumeration>
<xs:enumeration value="onDataSelection">
    <xs:annotation>
        <xs:documentation xml:lang="en">
            Some data has been selected by the user
        </xs:documentation>
    </xs:annotation>
</xs:enumeration>
<xs:enumeration value="onOperationTrigger">
    <xs:annotation>
        <xs:documentation xml:lang="en">
            An AbstractOperationIU has been activated
        </xs:documentation>
    </xs:annotation>
</xs:enumeration>
<xs:enumeration value="onNavigationTrigger">
    <xs:annotation>
        <xs:documentation xml:lang="en">
            An AbstractNavigationIU has been activated
        </xs:documentation>
    </xs:annotation>
</xs:enumeration>
<xs:enumeration value="onFocusIn">
    <xs:annotation>
        <xs:documentation xml:lang="en">
            An AbstractInteractionUnit has been focused in
        </xs:documentation>
    </xs:annotation>
</xs:enumeration>
<xs:enumeration value="onFocusOut">
    <xs:annotation>
        <xs:documentation xml:lang="en">
            An AbstractInteractionUnit has been focused out
        </xs:documentation>
    </xs:annotation>
</xs:enumeration>
<xs:enumeration value="onModelUpdate">
    <xs:annotation>
        <xs:documentation xml:lang="en">
            The Domain Model has been updated
        </xs:documentation>
    </xs:annotation>
</xs:enumeration>
</xs:simpleType>

7.2. OWL 2 Full
7.2.1. Task

7.2.1.1. Graphic representation
7.2.1.2. Task OWL 2 Text content

<?xml version="1.0"?>

<!DOCTYPE rdf:RDF [
<!ENTITY owl "http://www.w3.org/2002/07/owl#" >
<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
<!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
<!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
<!ENTITY taskui "http://www.usixml.org/metamodel/taskui.owl#" >
<!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
<!ENTITY taskui "http://www.usixml.org/metamodel/taskui.owl#" >
]

<rdf:RDF xmlns="http://www.usixml.org/metamodel/taskui.owl#"
  xml:base="http://www.usixml.org/metamodel/taskui.owl"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:taskui="http://www.usixml.org/metamodel/taskui.owl#"
  xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <owl:Ontology rdf:about=""/>
</rdf:RDF>

<!--

// Object Properties

-->
<!-- http://www.usixml.org/metamodel/taskui.owl#Composition_Task_isCompositedOf_Task -->

<owl:ObjectProperty rdf:about="#Composition_Task_isCompositedOf_Task">
    <rdfs:range rdf:resource="#Task"/>
    <rdfs:domain rdf:resource="#Task"/>
    <owl:inverseOf rdf:resource="#Task_task_Task"/>
</owl:ObjectProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Composition_TemporalRelationship.taskExpression_next -->

<owl:ObjectProperty rdf:about="#Composition_TemporalRelationship.taskExpression_next">
    <rdfs:subPropertyOf rdf:resource="#Composition_TemporalRelationship_next_TaskExpression"/>
    <rdfs:range rdf:resource="#TaskExpression"/>
    <owl:inverseOf rdf:resource="#TaskExpression.temporalRelationship_next"/>
    <rdfs:domain rdf:resource="#TemporalRelationship"/>
</owl:ObjectProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Composition_TemporalRelationship.taskExpression_previous -->

<owl:ObjectProperty rdf:about="#Composition_TemporalRelationship.taskExpression_previous">
    <rdfs:subPropertyOf rdf:resource="#Composition_TemporalRelationship_previous_TaskExpression"/>
    <rdfs:range rdf:resource="#TaskExpression"/>
    <owl:inverseOf rdf:resource="#TaskExpression.temporalRelationship_previous"/>
    <rdfs:domain rdf:resource="#TemporalRelationship"/>
</owl/ObjectProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Composition_TemporalRelationship_next_TaskExpression -->

<owl:ObjectProperty rdf:about="#Composition_TemporalRelationship_next_TaskExpression">
    <rdfs:range rdf:resource="#TaskExpression"/>
    <rdfs:domain rdf:resource="#TemporalRelationship"/>
</owl/ObjectProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Composition_TemporalRelationship_previous_TaskExpression -->

<owl:ObjectProperty rdf:about="#Composition_TemporalRelationship_previous_TaskExpression">
    <rdfs:range rdf:resource="#TaskExpression"/>
    <rdfs:domain rdf:resource="#TemporalRelationship"/>
</owl/ObjectProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Decoratoin.task -->

<owl:ObjectProperty rdf:about="#Decoratoin.task">
    <rdfs:domain rdf:resource="#Decoration"/>
    <rdfs:range rdf:resource="#Task"/>
    <owl:inverseOf rdf:resource="#Task.decoration"/>
</owl/ObjectProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Decoratoin_decoratedTask_Task -->

<owl:ObjectProperty rdf:about="#Decoratoin_decoratedTask_Task">
    <rdfs:domain rdf:resource="#Decoration"/>
    <rdfs:range rdf:resource="#Task"/>
    <owl:inverseOf rdf:resource="#Task.decoration"/>
</owl/ObjectProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Task.decoration -->

<owl:ObjectProperty rdf:about="#Task.decoration">
    <rdfs:domain rdf:resource="#Decoration"/>
    <rdfs:range rdf:resource="#Task"/>
</owl/ObjectProperty>
<owl:ObjectProperty rdf:about="#TaskExpression_next_TemporalRelationship">
  <owl:inverseOf rdf:resource="#Composition_TemporalRelationship_next_TaskExpression"/>
  <rdfs:domain rdf:resource="#TaskExpression"/>
  <rdfs:range rdf:resource="#TemporalRelationship"/>
</owl:ObjectProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#TaskExpression_previous_TemporalRelationship -->

<owl:ObjectProperty rdf:about="#TaskExpression_previous_TemporalRelationship">
  <owl:inverseOf rdf:resource="#Composition_TemporalRelationship_previous_TaskExpression"/>
  <rdfs:domain rdf:resource="#TaskExpression"/>
  <rdfs:range rdf:resource="#TemporalRelationship"/>
</owl:ObjectProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Task_decoredTask_Decoration -->

<owl:ObjectProperty rdf:about="#Task_decoredTask_Decoration">
  <rdfs:range rdf:resource="#Decoration"/>
  <owl:inverseOf rdf:resource="#Decoratoin_decoratedTask_Task"/>
  <rdfs:domain rdf:resource="#Task"/>
</owl:ObjectProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Task_taskOf_TaskModel -->

<owl:ObjectProperty rdf:about="#Task_taskOf_TaskModel">
  <owl:inverseOf rdf:resource="#Composition_TaskModel_isComposedOf_Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#TaskModel"/>
</owl:ObjectProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Task_task_Task -->

<owl:ObjectProperty rdf:about="#Task_task_Task">
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
</owl:ObjectProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#TemporalRelationship.expressionDecoration -->

<owl:ObjectProperty rdf:about="#TemporalRelationship.expressionDecoration">
  <rdfs:range rdf:resource="#ExpressionDecoration"/>
  <rdfs:domain rdf:resource="#TemporalRelationship"/>
  <rdfs:subPropertyOf rdf:resource="#TemporalRelationship_expressionDecoration_ExpressionDecoration"/>
</owl:ObjectProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#TemporalRelationship_expressionDecoration_ExpressionDecoration -->

<owl:ObjectProperty rdf:about="#TemporalRelationship_expressionDecoration_ExpressionDecoration">
  <owl:inverseOf rdf:resource="#Composition_ExpressionDecoration_expressionDecoration_TemporalRelationship"/>
  <rdfs:range rdf:resource="#ExpressionDecoration"/>
  <rdfs:domain rdf:resource="#TemporalRelationship"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="#TemporalRelationship_expressionDecoration_ExpressionDecoration">
  <owl:inverseOf rdf:resource="#Composition_ExpressionDecoration_expressionDecoration_TemporalRelationship"/>
  <rdfs:range rdf:resource="#ExpressionDecoration"/>
  <rdfs:domain rdf:resource="#TemporalRelationship"/>
</owl:ObjectProperty>

<!-- Data properties -->

// Data properties
<!-- http://www.usixml.org/metamodel/taskui.owl#Decoration.centrality -->
<owl:DatatypeProperty rdf:about="#Decoration.centrality">
  <rdfs:domain rdf:resource="#Decoration"/>
  <rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Decoration.criticity -->
<owl:DatatypeProperty rdf:about="#Decoration.criticity">
  <rdfs:domain rdf:resource="#Decoration"/>
  <rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Decoration.frequence -->
<owl:DatatypeProperty rdf:about="#Decoration.frequence">
  <rdfs:domain rdf:resource="#Decoration"/>
  <rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Decoration.maxExecutionTime -->
<owl:DatatypeProperty rdf:about="#Decoration.maxExecutionTime">
  <rdfs:domain rdf:resource="#Decoration"/>
  <rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Decoration.maxInteraction -->
<owl:DatatypeProperty rdf:about="#Decoration.maxInteraction">
  <rdfs:domain rdf:resource="#Decoration"/>
  <rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Decoration.minExecutionTime -->
<owl:DatatypeProperty rdf:about="#Decoration.minExecutionTime">
  <rdfs:domain rdf:resource="#Decoration"/>
  <rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Decoration.minInteraction -->
<owl:DatatypeProperty rdf:about="#Decoration.minInteraction">
  <rdfs:domain rdf:resource="#Decoration"/>
  <rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/taskui.owl#Task.canonicalTaskType -->

<owl:DatatypeProperty rdf:about="#Task.canonicalTaskType">
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="#Task"/>
  <rdfs:domain rdf:resource="#Task"/>
<owl:DatatypeProperty rdf:about="#Task.description">
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="&xsd:string;"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="#Task.id">
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="&xsd:string;"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="#Task.name">
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="&xsd:string;"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="#Task.postcondition">
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="&xsd:string;"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="#Task.precondition">
  <rdfs:domain rdf:resource="#Task"/>
  <rdfs:range rdf:resource="&xsd:string;"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#TaskDecoration.nature">
  <rdfs:domain rdf:resource="#TaskDecoration"/>
  <rdfs:range rdf:resource="&owl;DataRange"/>
  <owl:oneOf>
    <rdf:Description>
      <rdf:rest rdf:resource="&rdf;List"/>
      <rdf:Description>
        <rdf:first rdf:datatype="&xsd;string">INTERACTIVE</rdf:first>
      </rdf:Description>
    </rdf:Description>
    <rdf:Description>
      <rdf:rest rdf:resource="&rdf;List"/>
      <rdf:Description>
        <rdf:first rdf:datatype="&xsd;string">SYSTEM</rdf:first>
      </rdf:Description>
    </rdf:Description>
    <rdf:Description>
      <rdf:rest rdf:resource="&rdf;List"/>
      <rdf:Description>
        <rdf:first rdf:datatype="&xsd;string">UNDEFINED</rdf:first>
      </rdf:Description>
    </rdf:Description>
    <rdf:Description>
      <rdf:rest rdf:resource="&rdf;nil"/>
      <rdf:Description>
        <rdf:first rdf:datatype="&xsd;string">USER</rdf:first>
      </rdf:Description>
    </rdf:Description>
  </owl:oneOf>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="#TaskExpression.formula">
  <rdfs:domain rdf:resource="#TaskExpression"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="#TaskExpression.id">
  <rdfs:domain rdf:resource="#TaskExpression"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="#TaskModel.name">
  <rdfs:domain rdf:resource="#TaskModel"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="#TemporalRelationship.type">
  <rdfs:domain rdf:resource="#TemporalRelationship"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>
<rdf:Description>
  <rdf:Description>
    #Decoration
  </rdf:Description>
</owl:Class>

<owl:Class rdf:about="#Decoration">  
  <rdfs:subClassOf rdf:resource="#TaskExpression"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:OnProperty rdf:resource="#Decoration.task"/>
      <owl:Class rdf:resource="#Task"/>
      <owl:qualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:qualifiedCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#TaskExpression"/>
<owl:Restriction>
  <owl:onProperty rdf:resource="#TaskExpression.task"/>
  <owl:Class rdf:resource="#Task"/>
  <owl:minQualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">0</owl:minQualifiedCardinality>
</owl:Restriction>
<owl:subClassOf/>
<owl:Restriction>
  <owl:onProperty rdf:resource="#TaskExpression.temporalRelationship_next"/>
  <owl:Class rdf:resource="#TemporalRelationship"/>
  <owl:minQualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">0</owl:minQualifiedCardinality>
</owl:Restriction>
<owl:subClassOf/>
<owl:Restriction>
  <owl:onProperty rdf:resource="#TaskExpression.temporalRelationship_previous"/>
  <owl:Class rdf:resource="#TemporalRelationship"/>
  <owl:minQualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">0</owl:minQualifiedCardinality>
</owl:Restriction>
<owl:subClassOf/>
<owl:Restriction>
  <owl:onProperty rdf:resource="#TemporalRelationship.expressionDecoration"/>
  <owl:Class rdf:resource="#ExpressionDecoration"/>
  <owl:minQualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">0</owl:minQualifiedCardinality>
</owl:Restriction>
<owl:subClassOf/>
<owl:Restriction>
  <owl:onProperty rdf:resource="#Composition_TemporalRelationship.taskExpression_next"/>
  <owl:Class rdf:resource="#TaskExpression"/>
  <owl:qualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:qualifiedCardinality>
</owl:Restriction>
<owl:subClassOf/>
<owl:Restriction>
  <owl:onProperty rdf:resource="#Composition_TemporalRelationship.taskExpression_previous"/>
  <owl:Class rdf:resource="#TaskExpression"/>
  <owl:qualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:qualifiedCardinality>
</owl:Restriction>
<owl:subClassOf/>
<owl:Restriction>
  <owl:onProperty rdf:resource="#TemporalRelationship.expressionDecoration"/>
  <owl:Class rdf:resource="#ExpressionDecoration"/>
  <owl:maxQualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:maxQualifiedCardinality>
</owl:Restriction>
<owl:subClassOf/>
<owl:Restriction>
  <owl:onProperty rdf:resource="#Composition_TemporalRelationship.taskExpression_previous"/>
  <owl:Class rdf:resource="#TaskExpression"/>
  <owl:maxQualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:maxQualifiedCardinality>
</owl:Restriction>
<owl:subClassOf/>
</owl:Class>

<!-- http://www.usixml.org/metamodel/taskui.owl#TaskModel -->
<owl:Class rdf:about="#TaskModel"/>

<!-- http://www.usixml.org/metamodel/taskui.owl#TemporalRelationship -->
<owl:Class rdf:about="#TemporalRelationship"/>
<owl:subClassOf rdf:resource="#TaskExpression"/>
<owl:Restriction>
  <owl:onProperty rdf:resource="#TemporalRelationship.expressionDecoration"/>
  <owl:Class rdf:resource="#ExpressionDecoration"/>
  <owl:minQualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">0</owl:minQualifiedCardinality>
</owl:Restriction>
<owl:subClassOf/>
<owl:Restriction>
  <owl:onProperty rdf:resource="#Composition_TemporalRelationship.taskExpression_next"/>
  <owl:Class rdf:resource="#TaskExpression"/>
  <owl:qualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:qualifiedCardinality>
</owl:Restriction>
<owl:subClassOf/>
<owl:Restriction>
  <owl:onProperty rdf:resource="#TemporalRelationship.expressionDecoration"/>
  <owl:Class rdf:resource="#ExpressionDecoration"/>
  <owl:maxQualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:maxQualifiedCardinality>
</owl:Restriction>
<owl:subClassOf/>
<owl:Restriction>
  <owl:onProperty rdf:resource="#Composition_TemporalRelationship.taskExpression_previous"/>
7.2.2. Abstract User Interface

7.2.2.1. Graphic representation

![Diagram of Abstract User Interface]

Figure 7 - ???
7.2.2.2. Abstract User Interface OWL 2 Text content

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<catalog prefer="public" xmlns="urn:oasis:names:tc:entity:xmlns:xml:catalog">
  <group id="Folder Repository, directory=", recursive=true, Auto-Update=true, version=2" prefer="public" xml:base="">
    <uri id="Automatically generated entry, Timestamp=1323512857959" name="http://www.usixml.org/metamodel/abstractui.owl" uri="uiAbstractV3.owl"/>
  </group>
</catalog>

<?xml version="1.0"?>

<!DOCTYPE rdf:RDF []
<!ENTITY owl "http://www.w3.org/2002/07/owl#" >
<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
<!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
<!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
<!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
<!ENTITY abstractui "http://www.usixml.org/metamodel/abstractui.owl#" >
>
  <owl:Ontology rdf:about="http://www.usixml.org/metamodel/abstractui.owl#"/>

```
<owl:ObjectProperty rdf:about="&abstractui;EventExpression_next_TemporalEventExpression">
  <rdfs:domain rdf:resource="&abstractui;EventExpression"/>
  <rdfs:range rdf:resource="&abstractui;TemporalEventExpression"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="&abstractui;EventExpression_previous_TemporalEventExpression">
  <rdfs:domain rdf:resource="&abstractui;EventExpression"/>
  <rdfs:range rdf:resource="&abstractui;TemporalEventExpression"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="&abstractui;Justification.rule">
  <owl:inverseOf rdf:resource="&abstractui;COMPOSITION_Rule.justification"/>
  <rdfs:domain rdf:resource="&abstractui;Justification"/>
  <rdfs:subPropertyOf rdf:resource="&abstractui;Justification_justify_Rule"/>
  <rdfs:range rdf:resource="&abstractui;Rule"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="&abstractui;Justification_justify_Rule">
  <rdfs:domain rdf:resource="&abstractui;Justification"/>
  <rdfs:range rdf:resource="&abstractui;Rule"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="&abstractui;Rule_abstractListener">
  <owl:inverseOf rdf:resource="&abstractui;COMPOSITION_AbstractListener.rule"/>
  <rdfs:domain rdf:resource="&abstractui;Rule"/>
  <rdfs:subPropertyOf rdf:resource="&abstractui;Rule_trigger_AbstractListener"/>
  <rdfs:range rdf:resource="&abstractui;AbstractListener"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="&abstractui;Rule_trigger_AbstractListener">
  <rdfs:domain rdf:resource="&abstractui;Rule"/>
  <rdfs:subPropertyOf rdf:resource="&abstractui;Rule_trigger_AbstractListener"/>
  <rdfs:range rdf:resource="&abstractui;AbstractListener"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="&abstractui;State.abstractInteractionUnit">
  <owl:inverseOf rdf:resource="&abstractui;AGGREGATION_AbstractInteractionUnit.state"/>
  <rdfs:range rdf:resource="&abstractui;AbstractInteractionUnit"/>
  <rdfs:domain rdf:resource="&abstractui;State"/>
  <rdfs:subPropertyOf rdf:resource="&abstractui;State_stateIU_AbstractInteractionUnit"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="&abstractui;State.transition_source">
  <owl:inverseOf rdf:resource="&abstractui;AGGREGATION_AbstractInteractionUnit.state"/>
  <rdfs:range rdf:resource="&abstractui;AbstractInteractionUnit"/>
  <rdfs:domain rdf:resource="&abstractui;State"/>
  <rdfs:subPropertyOf rdf:resource="&abstractui;State_stateIU_AbstractInteractionUnit"/>
</owl:ObjectProperty>
<!-- http://www.usixml.org/metamodel/abstractui.owl#State_transition_target -->

<owl:ObjectProperty rdf:about="&abstractui;State.transition_target">
  <rdfs:domain rdf:resource="&abstractui;State"/>
  <rdfs:subPropertyOf rdf:resource="&abstractui;State_target_Transition"/>
  <rdfs:range rdf:resource="&abstractui;Transition"/>
</owl:ObjectProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#State_transition_source -->

<owl:ObjectProperty rdf:about="&abstractui;State.transition_source">
  <rdfs:domain rdf:resource="&abstractui;State"/>
  <rdfs:range rdf:resource="&abstractui;Transition"/>
</owl:ObjectProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#State_target_Transition -->

<owl:ObjectProperty rdf:about="&abstractui;State_target_Transition">
  <rdfs:domain rdf:resource="&abstractui;State"/>
  <rdfs:range rdf:resource="&abstractui;Transition"/>
  <owl:inverseOf rdf:resource="&abstractui;Transition_targetOf_State"/>
</owl:ObjectProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#State_stateIU_AbstractInteractionUnit -->

<owl:ObjectProperty rdf:about="&abstractui;State_stateIU_AbstractInteractionUnit">
  <owl:inverseOf rdf:resource="&abstractui;AGGREGATION_AbstractInteractionUnit_hasStates_State"/>
  <rdfs:range rdf:resource="&abstractui;AbstractInteractionUnit"/>
  <rdfs:domain rdf:resource="&abstractui;State"/>
</owl:ObjectProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#State_transition_target -->

<owl:ObjectProperty rdf:about="&abstractui;State.transition_target">
  <rdfs:domain rdf:resource="&abstractui;State"/>
  <rdfs:range rdf:resource="&abstractui;Transition"/>
  <owl:inverseOf rdf:resource="&abstractui;Transition_targetOf_State"/>
</owl:ObjectProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#State_transition_source -->

<owl:ObjectProperty rdf:about="&abstractui;Transition.state_source">
  <rdfs:range rdf:resource="&abstractui;State"/>
  <owl:inverseOf rdf:resource="&abstractui;State.transition_source"/>
  <rdfs:domain rdf:resource="&abstractui;Transition"/>
  <rdfs:subPropertyOf rdf:resource="&abstractui;Transition_sourceOf_State"/>
</owl:ObjectProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#State_transition_target -->

<owl:ObjectProperty rdf:about="&abstractui;Transition.state_target">
  <rdfs:range rdf:resource="&abstractui;State"/>
  <owl:inverseOf rdf:resource="&abstractui;State.transition_target"/>
  <rdfs:domain rdf:resource="&abstractui;Transition"/>
</owl:ObjectProperty>
<!-- http://www.usixml.org/metamodel/abstractui.owl#Transition_sourceOf_State -->

<owl:ObjectProperty rdf:about="&abstractui;Transition_sourceOf_State">
  <rdfs:range rdf:resource="&abstractui;State"/>
  <rdfs:domain rdf:resource="&abstractui;Transition"/>
</owl:ObjectProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#Transition_targetOf_State -->

<owl:ObjectProperty rdf:about="&abstractui;Transition_targetOf_State">
  <rdfs:range rdf:resource="&abstractui;State"/>
  <rdfs:domain rdf:resource="&abstractui;Transition"/>
</owl:ObjectProperty>

<!--
///////////////////////////////////////////////////////////////////////////////////////
// Data properties
///////////////////////////////////////////////////////////////////////////////////////
-->

<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractDataIU.dataFormat -->

<owl:DatatypeProperty rdf:about="&abstractui;AbstractDataIU.dataFormat">
  <rdfs:domain rdf:resource="&abstractui;AbstractDataIU"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractDataIU.dataIUType -->

<owl:DatatypeProperty rdf:about="&abstractui;AbstractDataIU.dataIUType">
  <rdfs:domain rdf:resource="&abstractui;AbstractDataIU"/>
  <rdfs:range rdf:resource="&abstractui;AbstractDataIU"/>
  <rdfs:Datatype>
    <owl:oneOf>
      <rdfs:Datatype>
        <rdfs:Description>
          <rdfs:Type rdf:resource="&rdf;List"/>
          <rdfs:first rdf:datatype="&xsd;string">INPUT</rdfs:first>
          <rdfs:rest>
            <rdfs:Datatype>
              <rdfs:Description>
                <rdfs:Type rdf:resource="&rdf;List"/>
                <rdfs:first rdf:datatype="&xsd;string">INPUT_OUTPUT</rdfs:first>
                <rdfs:rest>
                  <rdfs:Datatype>
                    <rdfs:Description>
                      <rdfs:Type rdf:resource="&rdf;List"/>
                      <rdfs:first rdf:datatype="&xsd;string">OUTPUT</rdfs:first>
                      <rdfs:rest rdf:resource="&rdf;nil"/>
                    </rdfs:Datatype>
                    <rdfs:Description>
                      <rdfs:rest>
                        <rdfs:Description>
                          <owl:oneOf>
                            <rdfs:Datatype>
                              <rdfs:range/>
                            </rdfs:Datatype>
                          </owl:oneOf>
                        </rdfs:Description>
                      </rdfs:Description>
                    </rdfs:Datatype>
                  </rdfs:rest>
                </rdfs:Datatype>
              </rdfs:Description>
            </rdfs:rest>
          </rdfs:Datatype>
        </rdfs:Description>
      </rdfs:Datatype>
    </owl:oneOf>
  </rdfs:Datatype>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractDataIU.dataLength -->

<!--
-->
<owl:DatatypeProperty rdf:about="&abstractui;AbstractDataIU.defaultValue">
   <rdfs:domain rdf:resource="&abstractui;AbstractDataIU"/>
   <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="&abstractui;AbstractDataIU.domainReference">
   <rdfs:domain rdf:resource="&abstractui;AbstractDataIU"/>
   <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="&abstractui;AbstractDataIU.isDynamic">
   <rdfs:domain rdf:resource="&abstractui;AbstractDataIU"/>
   <rdfs:range rdf:resource="&xsd;boolean"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="&abstractui;AbstractDataIU.maxCardinality">
   <rdfs:domain rdf:resource="&abstractui;AbstractDataIU"/>
   <rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="&abstractui;AbstractDataIU.minCardinality">
   <rdfs:domain rdf:resource="&abstractui;AbstractDataIU"/>
   <rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="&abstractui;AbstractIU.id">
   <rdfs:domain rdf:resource="&abstractui;AbstractIU"/>
   <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="&abstractui;AbstractIU.securityMechanism">
   <rdfs:domain rdf:resource="&abstractui;AbstractIU"/>
   <rdfs:range rdf:resource="&abstractui;AbstractIU"/>
</owl:DatatypeProperty>
<rdf:List rdf:resource="&xsd:string">
  <CAPTCHA/>
</rdf:List>

<owl:oneOf>
  <rdf:List rdf:resource="&xsd:string">
    <NONE/>
  </rdf:List>
</owl:oneOf>

<owl:DatatypeProperty rdf:about="&abstractui;AbstractInteractionUnit.securityType"/>

<owl:oneOf>
  <rdf:List rdf:resource="&xsd:string">
    <INTEGRATED/>
  </rdf:List>
</owl:oneOf>

<owl:DatatypeProperty rdf:about="&abstractui;AbstractListener.id"/>

<owl:DatatypeProperty rdf:about="&abstractui;AbstractListener.name"/>

<owl:DatatypeProperty rdf:about="&abstractui;AbstractLocalization.desLabel"/>
<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractLocalization.help -->

<owl:DatatypeProperty rdf:about="&abstractui;AbstractLocalization.help">
  <rdfs:domain rdf:resource="&abstractui;AbstractLocalization"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractLocalization.hierarchyNumber -->

<owl:DatatypeProperty rdf:about="&abstractui;AbstractLocalization.hierarchyNumber">
  <rdfs:domain rdf:resource="&abstractui;AbstractInteractionUnit"/>
  <rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractLocalization.id -->

<owl:DatatypeProperty rdf:about="&abstractui;AbstractLocalization.id">
  <rdfs:domain rdf:resource="&abstractui;AbstractLocalization"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractLocalization.importance -->

<owl:DatatypeProperty rdf:about="&abstractui;AbstractLocalization.importance">
  <rdfs:domain rdf:resource="&abstractui;AbstractInteractionUnit"/>
  <rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractLocalization.label -->

<owl:DatatypeProperty rdf:about="&abstractui;AbstractLocalization.label">
  <rdfs:domain rdf:resource="&abstractui;AbstractLocalization"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractLocalization.lang -->

<owl:DatatypeProperty rdf:about="&abstractui;AbstractLocalization.lang">
  <rdfs:range rdf:resource="&xsd:string"/>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractLocalization.longLabel -->

<owl:DatatypeProperty rdf:about="&abstractui;AbstractLocalization.longLabel">
  <rdfs:domain rdf:resource="&abstractui;AbstractLocalization"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractLocalization.repetitionFactor -->

<owl:DatatypeProperty rdf:about="&abstractui;AbstractLocalization.repetitionFactor">
  <rdfs:domain rdf:resource="&abstractui;AbstractInteractionUnit"/>
  <rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>
<rdfs:range>
  <rdfs:Datatype>
    <owl:oneOf>
      <rdf:Description>
        <rdf:Datatype rdf:resource="&rdf;List"/>
        <rdf:Datatype rdf:resource="&xsd;string">MULTIPLEINTERVAL</rdf:Datatype>
      </rdf:Description>
      <rdf:Datatype rdf:resource="&rdf;List"/>
      <rdf:Datatype rdf:resource="&xsd;string">NOSELECTION</rdf:Datatype>
      <rdf:Datatype rdf:resource="&rdf;List"/>
      <rdf:Datatype rdf:resource="&xsd;string">SINGLEINTERVAL</rdf:Datatype>
      <rdf:Datatype rdf:resource="&rdf;List"/>
      <rdf:Datatype rdf:resource="&xsd;string">SINGLESELECTION</rdf:Datatype>
      <rdf:Datatype rdf:resource="&rdf;List"/>
      <rdf:Datatype rdf:resource="&xsd;string">UNDEFINED</rdf:Datatype>
      <rdf:Datatype rdf:resource="&rdf;nil"/>
    </owl:oneOf>
    <owl:DatatypeProperty>
      <rdfs:range rdf:resource="&xsd;float"/>
    </owl:DatatypeProperty>
  </rdfs:Datatype>
</rdfs:range>

<owl:DatatypeProperty rdf:about="&abstractui;AbstractSelectionIU.start">
  <rdfs:domain rdf:resource="&abstractui;AbstractSelectionIU"/>
  <rdfs:range rdf:resource="&xsd;float"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="&abstractui;AbstractSelectionIU.step">
  <rdfs:domain rdf:resource="&abstractui;AbstractSelectionIU"/>
  <rdfs:range rdf:resource="&xsd;float"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="&abstractui;AbstractTriggerIU.triggerIUType">
  <rdfs:domain rdf:resource="&abstractui;AbstractTriggerIU"/>
  <rdfs:range>
    <owl:oneOf>
      <rdf:Description>
        <rdf:Datatype rdf:resource="&rdf;List"/>
        <rdf:Datatype rdf:resource="&xsd;string">NAVIGATION</rdf:Datatype>
      </rdf:Description>
      <rdf:Datatype rdf:resource="&rdf;List"/>
      <rdf:Datatype rdf:resource="&xsd;string">OPERATION</rdf:Datatype>
    </owl:oneOf>
  </rdfs:range>
</owl:DatatypeProperty>
<rdf:rest>
  <rdf:Description>
    <rdf:type rdf:resource="&rdf;List"/>
    <rdf:first rdf:datatypext=&xsd:string>OPERATION_NAVIGATION</rdf:first>
    <rdf:rest rdf:resource="&rdf;nil"/>
  </rdf:Description>
  <rdf:rest>
    <rdf:Description>
      <rdf:type rdf:resource="&rdf;List"/>
      <rdf:first rdf:datatypext=&xsd:string>EVENTDispatch</rdf:first>
      <rdf:rest>
        <rdf:Description>
          <rdf:type rdf:resource="&rdf;List"/>
          <rdf:first rdf:datatypext=&xsd:string>IUAckivate</rdf:first>
          <rdf:rest>
            <rdf:Description>
              <rdf:type rdf:resource="&rdf;List"/>
              <rdf:first rdf:datatypext=&xsd:string>IUClose</rdf:first>
              <rdf:rest>
                <rdf:Description>
                  <rdf:type rdf:resource="&rdf;List"/>
                  <rdf:first rdf:datatypext=&xsd:string>IUDeactivate</rdf:first>
                  <rdf:rest>
                    <rdf:Description>
                      <rdf:type rdf:resource="&rdf;List"/>
                      <rdf:first rdf:datatypext=&xsd:string>IUDeemphasisize</rdf:first>
                      <rdf:rest>
                        <rdf:Description>
                          <rdf:type rdf:resource="&rdf;List"/>
                          <rdf:first rdf:datatypext=&xsd:string>EVENTDispatch</rdf:first>
                          <rdf:rest>
                            <rdf:Description>
                              <rdf:type rdf:resource="&rdf;List"/>
                              <rdf:first rdf:datatypext=&xsd:string>IUAckivate</rdf:first>
                              <rdf:rest>
                                <rdf:Description>
                                  <rdf:type rdf:resource="&rdf;List"/>
                                  <rdf:first rdf:datatypext=&xsd:string>IUClose</rdf:first>
                                  <rdf:rest>
                                    <rdf:Description>
                                      <rdf:type rdf:resource="&rdf;List"/>
                                      <rdf:first rdf:datatypext=&xsd:string>IUDeactivate</rdf:first>
                                      <rdf:rest>
                                        <rdf:Description>
                                          <rdf:type rdf:resource="&rdf;List"/>
                                          <rdf:first rdf:datatypext=&xsd:string>IUDeemphasisize</rdf:first>
                                          <rdf:rest>
                                            <rdf:Description>
                                              <rdf:type rdf:resource="&rdf;List"/>
                                              <rdf:first rdf:datatypext=&xsd:string>EVENTDispatch</rdf:first>
                                              <rdf:rest>
                                                <rdf:Description>
                                                  <rdf:type rdf:resource="&rdf;List"/>
                                                  <rdf:first rdf:datatypext=&xsd:string>IUAckivate</rdf:first>
                                                  <rdf:rest>
                                                    <rdf:Description>
                                                      <rdf:type rdf:resource="&rdf;List"/>
                                                      <rdf:first rdf:datatypext=&xsd:string>IUClose</rdf:first>
                                                      <rdf:rest>
                                                        <rdf:Description>
                                                          <rdf:type rdf:resource="&rdf;List"/>
                                                          <rdf:first rdf:datatypext=&xsd:string>IUDeactivate</rdf:first>
                                                          <rdf:rest>
                                                            <rdf:Description>
                                                              <rdf:type rdf:resource="&rdf;List"/>
                                                              <rdf:first rdf:datatypext=&xsd:string>IUDeemphasisize</rdf:first>
                                                              <rdf:rest>
                                                                <rdf:Description>
                                                                  <rdf:type rdf:resource="&rdf;List"/>
                                                                  <rdf:first rdf:datatypext=&xsd:string>EVENTDispatch</rdf:first>
                                                                  <rdf:rest>
                                                                    <rdf:Description>
                                                                      <rdf:type rdf:resource="&rdf;List"/>
                                                                      <rdf:first rdf:datatypext=&xsd:string>IUAckivate</rdf:first>
                                                                      <rdf:rest>
                                                                        <rdf:Description>
                                                                          <rdf:type rdf:resource="&rdf;List"/>
                                                                          <rdf:first rdf:datatypext=&xsd:string>IUClose</rdf:first>
                                                                          <rdf:rest>
                                                                            <rdf:Description>
                                                                              <rdf:type rdf:resource="&rdf;List"/>
                                                                              <rdf:first rdf:datatypext=&xsd:string>IUDeactivate</rdf:first>
                                                                              <rdf:rest>
                                                                                <rdf:Description>
                                                                                  <rdf:type rdf:resource="&rdf;List"/>
                                                                                  <rdf:first rdf:datatypext=&xsd:string>IUDeemphasisize</rdf:first>
                                                                                  <rdf:rest>
                                                                                    <rdf:Description>
                                                                                      <rdf:type rdf:resource="&rdf;List"/>
                                                                                      <rdf:first rdf:datatypext=&xsd:string>EVENTDispatch</rdf:first>
                                                                                      <rdf:rest>
                                                                                    <rdf:Description>
                                                                                      </rdf:Description>
                                                                                  </rdf:rest>
                                                                                </rdf:Description>
                                                                              </rdf:rest>
                                                                            </rdf:Description>
                                                                          </rdf:rest>
                                                                        </rdf:Description>
                                                                      </rdf:rest>
                                                                    </rdf:Description>
                                                                  </rdf:rest>
                                                                </rdf:Description>
                                                              </rdf:rest>
                                                            </rdf:Description>
                                                          </rdf:rest>
                                                        </rdf:Description>
                                                      </rdf:rest>
                                                    </rdf:Description>
                                                  </rdf:rest>
                                                </rdf:Description>
                                              </rdf:rest>
                                            </rdf:Description>
                                          </rdf:rest>
                                        </rdf:Description>
                                      </rdf:rest>
                                    </rdf:Description>
                                  </rdf:rest>
                                </rdf:Description>
                              </rdf:rest>
                            </rdf:Description>
                          </rdf:rest>
                        </rdf:Description>
                      </rdf:rest>
                    </rdf:Description>
                  </rdf:rest>
                </rdf:Description>
              </rdf:rest>
            </rdf:Description>
          </rdf:rest>
        </rdf:Description>
      </rdf:rest>
    </rdf:Description>
  </rdf:rest>
</rdf:Description>
<owl:DatatypeProperty rdf:about="&abstractui;State.id">
  <rdfs:domain rdf:resource="&abstractui;State"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="&abstractui;TemporalActionExpression.type">
  <rdfs:domain rdf:resource="&abstractui;TemporalActionExpression"/>
  <rdfs:range>
    <rdf:Datatype>
      <owl:oneOf>
        <rdf:Description>
          <rdfs:domain rdf:resource="&rdf;List"/>  
          <rdfs:range rdf:resource="&xsd;string"/>
          <owl:DatatypeProperty>
          </owl:DatatypeProperty>
        </rdf:Description>
        <rdf:Description>
          <rdfs:domain rdf:resource="&rdf;List"/>  
          <rdfs:range rdf:resource="&xsd;string"/>
          <owl:DatatypeProperty>
          </owl:DatatypeProperty>
        </rdf:Description>
        <rdf:Description>
          <rdfs:domain rdf:resource="&rdf;List"/>  
          <rdfs:range rdf:resource="&xsd;string"/>
          <owl:DatatypeProperty>
          </owl:DatatypeProperty>
        </rdf:Description>
      </owl:oneOf>
    </rdf:Datatype>
  </rdfs:range>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="&abstractui;Transition.transitionExpression">
  <rdfs:domain rdf:resource="&abstractui;Transition"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>
<owl:Class rdf:about="&abstractui;AbstractCompoundIU">
  <rdfs:subClassOf rdf:resource="&abstractui;AbstractInteractionUnit"/>
  <owl:Restriction>
    <owl:onProperty rdf:resource="&abstractui;AbstractCompoundIU_belong_AbstractUIModel"/>
    <owl:onClass rdf:resource="&abstractui;AbstractUIModel"/>
    <owl:minQualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">0</owl:minQualifiedCardinality>
  </owl:Restriction>
  <owl:Restriction>
    <owl:onProperty rdf:resource="&abstractui;AbstractCompoundIU_belong_AbstractUIModel"/>
    <owl:onClass rdf:resource="&abstractui;AbstractUIModel"/>
    <owl:maxQualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">0</owl:maxQualifiedCardinality>
  </owl:Restriction>
  <owl:Restriction>
    <owl:onProperty rdf:resource="&abstractui;AbstractCompoundIU.abstractUIModel"/>
    <owl:onClass rdf:resource="&abstractui;AbstractUIModel"/>
    <owl:minQualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">0</owl:minQualifiedCardinality>
  </owl:Restriction>
  <owl:Restriction>
    <owl:onProperty rdf:resource="&abstractui;AbstractCompoundIU.abstractUIModel"/>
    <owl:onClass rdf:resource="&abstractui;AbstractUIModel"/>
    <owl:maxQualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:maxQualifiedCardinality>
  </owl:Restriction>
</owl:Class>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractDataIU -->

<owl:Class rdf:about="&abstractui;AbstractDataIU">
  <rdfs:subClassOf rdf:resource="&abstractui;AbstractElementaryIU"/>
</owl:Class>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractElementaryIU -->

<owl:Class rdf:about="&abstractui;AbstractElementaryIU">
  <rdfs:subClassOf rdf:resource="&abstractui;AbstractInteractionUnit"/>
</owl:Class>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractInteractionUnit -->

<owl:Class rdf:about="&abstractui;AbstractInteractionUnit">
</owl:Class>
<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractSelectionIU -->
<owl:Class rdf:about="&abstractui;AbstractSelectionIU">
  <rdfs:subClassOf rdf:resource="&abstractui;AbstractCompoundIU"/>
</owl:Class>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractTriggerIU -->
<owl:Class rdf:about="&abstractui;AbstractTriggerIU">
  <rdfs:subClassOf rdf:resource="&abstractui;AbstractElementaryIU"/>
</owl:Class>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AbstractUIModel -->
<owl:Class rdf:about="&abstractui;AbstractUIModel">
  <rdfs:subClassOf rdf:resource="&owl;Thing"/>
</owl:Class>

<!-- http://www.usixml.org/metamodel/abstractui.owl#ActionExpression -->
<owl:Class rdf:about="&abstractui;ActionExpression">
  <rdfs:subClassOf rdf:resource="&abstractui;ActionExpression"/>
  <owl:Restriction>
    <owl:onProperty rdf:resource="&abstractui;ActionExpression_actionOf_Rule"/>
    <owl:onClass rdf:resource="&abstractui;Rule"/>
    <owl:qualifiedCardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:qualifiedCardinality>
  </owl:Restriction>
</owl:Class>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AtomicAction -->
<owl:Class rdf:about="&abstractui;AtomicAction">
  <rdfs:subClassOf rdf:resource="&abstractui;AtomicAction"/>
</owl:Class>

<!-- http://www.usixml.org/metamodel/abstractui.owl#AtomicEvent -->
<owl:Class rdf:about="&abstractui;AtomicEvent">
  <rdfs:subClassOf rdf:resource="&abstractui;EventExpression"/>
</owl:Class>
8. **WORKFLOW FOR MANAGING META-MODELS**

Each meta-model inevitably evolves over time depending on the change requests introduced by any partner. In order to effectively and efficiently ensure the process of following these change requests over time, a workflow management system has been designed and developed that defines the following roles:

- Each UsiXML meta-model is led by a specific project member called leader. A leader is the only member that can modify the meta-model for which he is responsible. The leader is named owner. In order to avoid a conflict situation, the owner of a meta-model should be a specific member. In other words, a co-ownership is not permitted.

- Even if a member is not the owner of a meta-model, she can participate in its design by subscribing to a meta-model and issuing modification requests to the owner of meta-model to be modified. This member is called participant.

- An owner can accept or reject a request modification issued by a participant. A modification of a meta-model by its owner must be validated by a third-party. A member named reviewer handles this validation.

The link provides a URL towards a demonstration of the UsiXML Extranet that gives access to the workflow management system based on the above state charts.

**UsiXML Extranet system for managing meta-models**

The following state charts defines the possible states of the workflow for managing a UsiXML meta-model over time.
9. APPENDIX

9.1.1. Use Case 2 détails

User Choose Track
User Case Descriptions

Principal
Super Use Case

Author: jbcollet
Date: 1 févr. 2012 16:22:10

Brief Description

Preconditions

Post-conditions: The user has chosen a track and the track is saved into memory
### Flow of Events

1. The user selects a category of a track, a place to search from (or its GPS localisation) and a maximum search radius.
2. The system displays the tracks matching the given parameters.
3. The user selects a track.
4. The system displays the track's detail.
5. The user saves the track into its favorites for further usage.
6. The system saves the track into memory.

### Choose Language

**Use Case Descriptions**

**Principal**

**Super Use Case**

**Author**

**Date**

**Brief Description**

**Preconditions**

**Post-conditions**

The display uses the user's language.

**Flow of Events**

1. The user chooses its language.
2. The system refresh the display with the selected language.

### Request Travel

**Use Case Descriptions**

**Principal**

**Super Use Case**

**Author**

**Date**

**Brief Description**

**Preconditions**

**Post-conditions**

The User has filled its travel's preferences and booked its travel's elements.

**Flow of Events**

1. The user fills its travel's preferences.
2. The system saves the user's preferences.

### Rent the Car

**Use Case Descriptions**

**Principal**

**Super Use Case**

**Author**

**Date**

**Brief Description**

**Preconditions**

**Post-conditions**

The car is rent and payed.

**Flow of Events**

1. The user rents and pay a car.
2. The car's reservation system confirms the reservation.

### Choose a car

**Use Case Descriptions**
## Principal

**Super Use Case**

**Author**  jbcollet  
**Date**  2 févr. 2012 09:28:39  
**Brief Description**  
**Preconditions**  -  
**Post-conditions**  The user has filled its car's preferences and has chosen one  
**Flow of Events**  
1. The user fill its car's preferences  
2. The system computes which cars match user's preferences  
3. The user selects a car  

## Pay the car

**Use Case Descriptions**

**Principal**

**Super Use Case**

**Author**  jbcollet  
**Date**  8 févr. 2012 13:09:36  
**Brief Description**  
**Preconditions**  The user has selected a car  
**Post-conditions**  The user has payed its car  
**Flow of Events**  
1. The user pay its car  
2. The car reservation system confirms the payment.  

## Book the Hotel

**Use Case Descriptions**

**Principal**

**Super Use Case**

**Author**  jbcollet  
**Date**  1 févr. 2012 15:47:53  
**Brief Description**  
**Preconditions**  -  
**Post-conditions**  The hotel is booked and payed  
**Flow of Events**  
1. The user books and pay the hotel with the given parameters  
2. The hotel's systems confirms the reservation  

## Choose a hotel

**Use Case Descriptions**

**Principal**

**Super Use Case**

**Author**  jbcollet  
**Date**  1 févr. 2012 16:25:06  
**Brief Description**  
**Preconditions**  -  
**Post-conditions**  The user has filled it's hotel preferences and has chosen one
Flow of Events
1 The user selects the hotel rating
2 The user selects the arrival date
3 The user selects the departure date
4 The user selects the room type
5 The user selects if he wants the breakfast or not
6 The system computes which hotel match user's preferences.
7 The user select an hotel

Pay the hotel
Use Case Descriptions
Principal
Super Use Case
Author jbcollet
Date 8 févr. 2012 13:08:37
Brief Description
Preconditions The user has selected a hotel
Post-conditions The user has payed its hotel
Flow of Events
1 The user pay its hotel
2 The hotel booking service confirms the payment.

Book the Train
Use Case Descriptions
Principal
Super Use Case
Author jbcollet
Date 1 févr. 2012 15:45:10
Brief Description
Preconditions
Post-conditions The train is booked and payed
Flow of Events
1 The user books and pay the train with the given parameters
2 The train's ticketing system confirms the reservation

Select train
Use Case Descriptions
Principal
Super Use Case
Author jbcollet
Date 1 févr. 2012 16:27:58
Brief Description
Preconditions
Post-conditions The user has filled its train’s preferences and has chosen one.
Flow of Events
1 The user fill its train departure date, train station and destination
2 The system computes which trains match user's preferences
3 The user selects a train

Pay the train
Use Case Descriptions
### Book the Flight

**Use Case Descriptions**

<table>
<thead>
<tr>
<th>Principal</th>
<th>Super Use Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author</td>
<td>bcollet</td>
</tr>
<tr>
<td>Date</td>
<td>1 févr. 2012 14:37:38</td>
</tr>
</tbody>
</table>

**Brief Description**

**Preconditions** -

**Post-conditions** The flight is booked and payed

**Flow of Events**

1. The user books and pay the flight.
2. The flight's ticketing systems confirms the reservation

---

### Pay the flight

**Use Case Descriptions**

<table>
<thead>
<tr>
<th>Principal</th>
<th>Super Use Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author</td>
<td>bcollet</td>
</tr>
<tr>
<td>Date</td>
<td>8 févr. 2012 13:04:45</td>
</tr>
</tbody>
</table>

**Brief Description**

**Preconditions** The user has selected a flight

**Post-conditions** The user has payed its flight

---

### Select flight

**Use Case Descriptions**

<table>
<thead>
<tr>
<th>Principal</th>
<th>Super Use Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author</td>
<td>bcollet</td>
</tr>
<tr>
<td>Date</td>
<td>2 févr. 2012 09:30:41</td>
</tr>
</tbody>
</table>

**Brief Description**

**Preconditions** -

**Post-conditions** The user has filled it's flight preferences and has chosen one

**Flow of Events**

1. The user fill the flight's departure date, airport and destination
2. The system computes which flights match user's preferences
3. The user selects a flight

---

### Principal

**Use Case**

<table>
<thead>
<tr>
<th>Author</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>bcollet</td>
<td>8 févr. 2012 13:07:34</td>
</tr>
</tbody>
</table>

**Brief Description**

**Preconditions** The user has selected a train

**Post-conditions** The user has payed its train

**Flow of Events**

1. The user pay its train
2. The train's ticketing service confirms the payement

---

### Principal

**Use Case**

<table>
<thead>
<tr>
<th>Author</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>bcollet</td>
<td>1 févr. 2012 14:37:38</td>
</tr>
</tbody>
</table>

**Brief Description**

**Preconditions** -

**Post-conditions** The flight is booked and payed

**Flow of Events**

1. The user pay its train
2. The train's ticketing service confirms the payement
| Flow of Events | 1  | The user pay its flight                  |
|               | 2  | The flight ticketing service confirms the payment. |

### Being guided on the track

**Use Case Descriptions**

**Principal**

Super Use Case

Author: bcollet

Date: 1 févr. 2012 16:33:39

Brief Description

**Preconditions**

The user has to have chosen and saved a track in its device memory

**Post-conditions**

The user has been guided through the track

**Flow of Events**

1  | The user start the navigation mode
2  | The system displays a map with the track, an arrow directing to the first point of interest, the distance to that point and an arrow to indicate the next direction to take there.
3  | The user walk along the track.
4  | When the user arrives close to a point of interest, the devices vibrates and display available media if any.
5  | The user can enjoy those medias or continue its walk.

### Create a new POI

**Use Case Descriptions**

**Principal**

Super Use Case

Author: bcollet

Date: 2 févr. 2012 15:42:30

Brief Description

**Preconditions**

-

**Post-conditions**

The user has created a new POI

**Flow of Events**

1  | The user gets its current geoposition
2  | The user fill the POI's name
3  | The user takes some pictures of the POI
4  | The user shots some movies of the POI
5  | The user write a description of the POI
6  | The system saves the POI into memory

### Create a new track

**Use Case Descriptions**

**Principal**

Super Use Case

Author: bcollet

Date: 2 févr. 2012 15:44:59

Brief Description

**Preconditions**

-

**Post-conditions**

The user has created a new track

**Flow of Events**

1  | The user fill the track’s name
2  | The user fill the track’s description
### Save User Preferences

**Use Case Descriptions**

<table>
<thead>
<tr>
<th>Principal</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Super Use Case</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Author</strong></td>
<td>jbcollet</td>
</tr>
<tr>
<td><strong>Date</strong></td>
<td>1 févr. 2012 16:58:22</td>
</tr>
</tbody>
</table>

**Brief Description**

- 

**Preconditions**

- 

**Post-conditions**

- User's preferences are saved

**Flow of Events**

1. The system saves the user's preferences on a given webservice

### Make Suggestions

**Use Case Descriptions**

<table>
<thead>
<tr>
<th>Principal</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Super Use Case</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Author</strong></td>
<td>jbcollet</td>
</tr>
<tr>
<td><strong>Date</strong></td>
<td>1 févr. 2012 16:59:15</td>
</tr>
</tbody>
</table>

The user takes some pictures of the track

The user shots some movies of the track

The user defines POI along the track

The system save the track into memory

<table>
<thead>
<tr>
<th>Demonstator</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Enfants</strong></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Nom</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Book the Flight</td>
<td></td>
</tr>
<tr>
<td>Book the Train</td>
<td></td>
</tr>
<tr>
<td>Book the Hotel</td>
<td></td>
</tr>
<tr>
<td>Rent the Car</td>
<td></td>
</tr>
<tr>
<td>Save User Preferences</td>
<td></td>
</tr>
<tr>
<td>Make Suggestions</td>
<td></td>
</tr>
<tr>
<td>Being guided on the track</td>
<td></td>
</tr>
<tr>
<td>Choose a car</td>
<td></td>
</tr>
<tr>
<td>Select flight</td>
<td></td>
</tr>
<tr>
<td>Select train</td>
<td></td>
</tr>
<tr>
<td>Choose a hotel</td>
<td></td>
</tr>
<tr>
<td>Choose Track</td>
<td></td>
</tr>
<tr>
<td>Choose Language</td>
<td></td>
</tr>
<tr>
<td>Request Travel</td>
<td></td>
</tr>
<tr>
<td>Create a new track</td>
<td></td>
</tr>
<tr>
<td>Create a new POI</td>
<td></td>
</tr>
<tr>
<td>Pay the flight</td>
<td></td>
</tr>
<tr>
<td>Pay the train</td>
<td></td>
</tr>
<tr>
<td>Pay the hotel</td>
<td></td>
</tr>
<tr>
<td>Pay the car</td>
<td></td>
</tr>
</tbody>
</table>
Based on the user's preferences, suggestions are made.

Flow of Events

1. Based on the user's preferences, the systems computes suggestions.

10. REFERENCES

10.1.Normatives references


10.2. Informative references


[KSC09]


[UsiXML11]

[Van93]

[Van05]

[Bod95]

[Bod95b]

[Bod94]

[Van94]

[Bod94b]

[Bod84]

[Bod89]

[Aqui11]

[Van10]

[Van08]