reviewed paper

Ontology-based Model of a Smart City

Oksana Smirnova, Tatiana Popovich

(PhD Oksana Smirnova, SPIIRAS - Hi Tech Research and Development Office Ltd., St. Petersburg, Russia, sov@oogis.ru)
(PhD Tatiana Popovich, SPIIRAS - Hi Tech Research and Development Office Ltd., St. Petersburg, Russia, t.popovich@oogis.ru)

1 ABSTRACT

The city environment is a complex, dynamic and distributed system. Effective management of the city environment presents a major challenge for a smart city. Effective city management must seek to improve urban infrastructure while minimising the costs, to stimulate innovations in different industries and to improve the quality of life for citizens. To create an adequate model of city's environment structure it is necessary to analyse all the city's components that influence the city environment management and to define data flows between these components. The most effective tool for representation of knowledge about the city environment, its components and relationships between them is the organisation of information in the form of an ontology-based model.

In this paper we would like to introduce a novel ontology-based model for the smart city that includes definitions of entities and their properties, classes and their attributes and relationships between them. Such ontology enables us to use a knowledge base for the smart city as a basis of the decision-making support system for smart city management.

Keywords: decision-making support system, smart city, city environment, city management, ontology

2 INTRODUCTION

Over the recent years application of ontologies in information systems' design has become more and more popular in various spheres of human activities: from biology to history and, naturally, in traditional computer science.

The notion "ontology" was first introduced by R. Goclenius and initially indicated a branch of philosophy that studied the fundamental basis of existence. Different definitions of ontology in computer science are given in (Breitman et al., 2007), among which we may single out the definition by T. Gruber: "An ontology is a formal, explicit specification of a shared conceptualisation" (Gruber, 1993). In this definition, the term conceptualisation means an abstract model, the term formal indicates that specification is suitable for machine processing. Therefore, from Gruber's point of view, ontology is a representation of knowledge about the subject domain, in which a set of objects and relations between them are described by a hierarchical model.

At the present time, there exist several different types of ontologies in computer science. Ontology can include various objects from computer science domain, depending on the concept. For example, data dictionary is used for search and transmission of information, OWL model of data representation is used for representation of connected data, XML-schemes are used for representation of data from the database.

Therefore, in order to develop a complete and universal ontology of the city environment suitable for further application in the creation of an effective decision-making making support system for city environment management, it is necessary to identify what city environment ontology implies and which city environment's objects are to be considered.

By ontology model of city environment we understand a description of basic notions (classes) of the city environment, description of objects of the city environment, their specific properties and relations between these objects (Popovich et al., 2013). By city environment's objects we mean engineering communication, transport infrastructure, objects of cultural heritage, industrial and social facilities, services and people.

In this paper we present a new approach to the development of an ontology of city environment. The presented approach implies the creation of two interconnected ontology models: ontology of the subject domain and scenario ontology. The combination of these two ontologies will permit not only to structure the information and knowledge about the subject domain (city environment) but also to create scenarios of possible situations connected to city environment's objects. Existence of such scenarios will increase the quality of services provided by city environment at all management levels.

3 STATE OF THE ART

The Fig. 1 depicts one of the existing variants of an ontological framework of the Smart City (Ramaprasad et al., 2017). This model includes two components: Smart and City, each of which is also a complex construct. The City component determines its subcomponents: Stakeholders and Outcomes. Authors of this ontology framework believe that major Stakeholders in the city are its Citizens, Professionals, Communities, Institutions, Businesses and Governments. Key indicators of Smartness of the city are "citizens' quality of life", "equity of communities", "sustainability of business" and other possible combinations of Stakeholders and Outcomes.

Smart						City		
Structure		**Functions**		**Focus**		**Semiotics**	**Stakeholders**	**Outcomes**
Architecture	[to]	Sense	[+]	Cultural	[+]	Data	Citizens	Sustainability
Infrastructure		Monitor		Economic		Information	Professionals	QoL
Systems		Process		Demographic		Knowledge	Communities	Equity
Services		Translate		Environmental			Institutions	Livability
Policies		Communicate		Political			Businesses	Resilience
Processes				Social			Governments	
Personnel				Technological				
				Infrastructural				

[by/from/to] [for]

Fig. 1: Ontological framework of a Smart City.

The Fig. 2 depicts a top-level ontology proposed by the European Urban Knowledge Network (https://www.eukn.eu/). The main purpose of this ontology is accumulation and distribution of knowledge about the city environment among municipal governing bodies. As we can see from Fig. 2, the dictionary has a rather large volume and a rich content. It includes five levels and 254 notions. This ontology aims to help solve only two major tasks: crime prevention and bringing together different social groups. At the present time, it is only available in English.

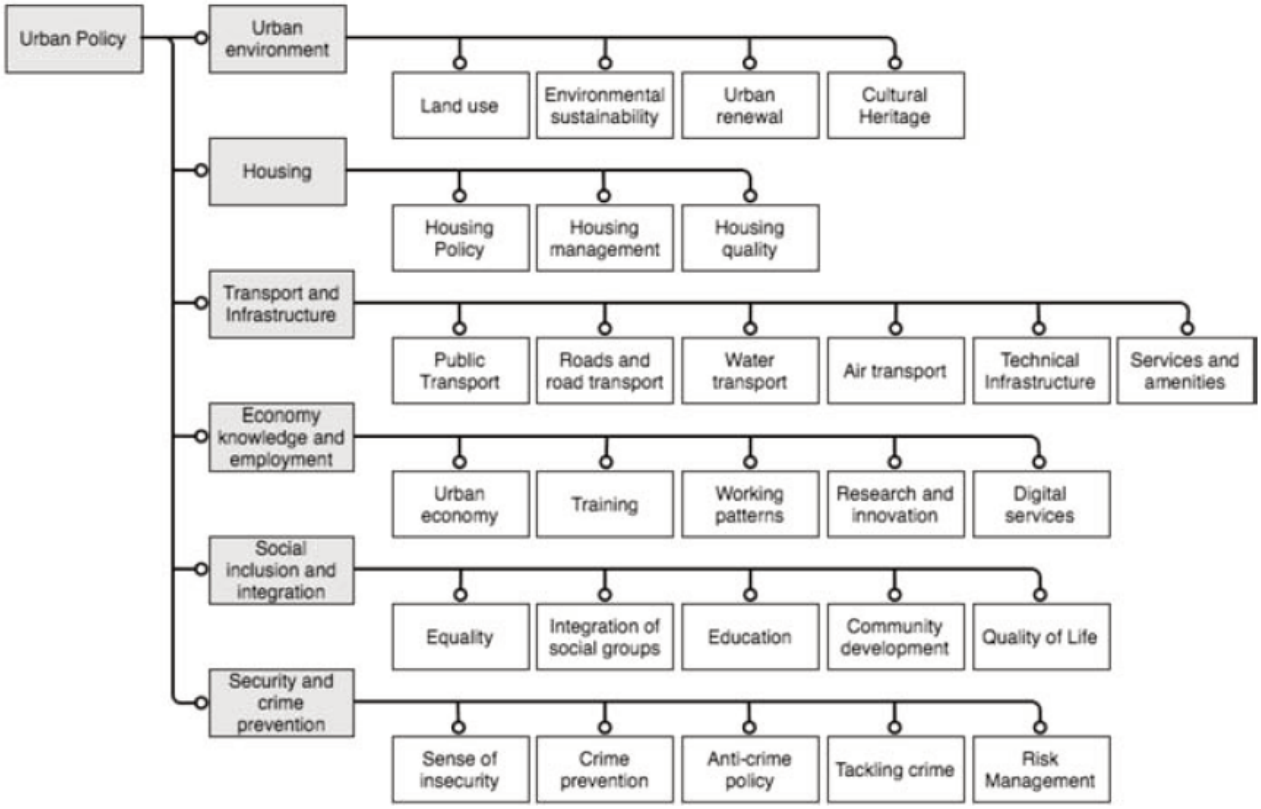

Fig. 2: The European Urban Knowledge Network top-level ontology.

Open311 is an ontology (http://ontology.eil.utoronto.ca/open311.html) developed by the University of Toronto. Its aim is to interpret and to visualise events that occur in the city environment and that have no connection to emergency situations. These events are presented in a form of semantic structure. Moreover,

this ontology model formed the basis for a service that provides feedback between citizens and the city government. Among advantages of this model, we should emphasise its universality. It is tuned to obtain heterogeneous information about the city environment and does not require a special configuration when applied to different cities. Furthermore, this model is oriented on users without special training in computer science and permits to send requests in understandable natural language without special machine languages.

International non-profit association of regional and national building SMART chapters proposes a Building Information Model (BIM) data (http://www.buildingsmart-tech.org/) aimed at organisation of data exchange between participants from the construction sector or from buildings' (objects') infrastructure management. This model is implemented as EXPRESS schema specification and alternatively as XML-schema specification. BIM can be used for data description and exchange and for the solution of related tasks at all stages of the life-cycle in the following areas: architecture, building service, structural engineering, procurement, construction planning, facility management, project management, client requirement management, building authority for permits and approval.

Among disadvantages of this model we can name the following: limited range of application (only construction and buildings' infrastructure management), the model contains only classes appropriate for description of the process of building construction from the engineering point of view (building engineering) and lack of means for the description of civil engineering; means for objects' behaviour modelling in the construction sector are also absent.

In accordance with the brief review given above, we can point out the main disadvantages of the existing city ontologies. Firstly, most of the reviewed ontologies have a narrow scope of application and do not fully cover all the important elements and components of the city environment. Secondly, these ontological models do not describe the behaviour of objects in the city environment.

4 DESIGN OF ONTOLOGY-BASED MODEL FOR THE SMART CITY

As was already mentioned above, ontology of the city environment should include two interconnected ontologies: ontology of city environment as a subject domain and scenario ontology that describes the behaviour of objects in the city environment.

The process of ontology creation for any information system includes the following steps:

(1) identification of classes (concepts): basic notions from city environment domain

(2) setting up a hierarchy of classes (basic class → subclass)

(3) identification of properties of classes: slots and their possible values

(4) filling the slots for class instances.

Typical ontology components are the following:

(1) Classes (notions, concepts) represent a holistic set of assertions that contain some observations about distinctive features of the object under examination; the core of such a set is formed by assertions (statements) about the most general but at the same time essential features of this object. While developing the ontology of the subject domain, one must determine the set of notions from subject domain and the logical connections (relations) between them, and then identify and formalise them. The result of the development of an ontology of subject domain is the hierarchy of classes that contain the notions from the subject domain and relations between them. Each notion is characterised by its scope and content. The scope and content of the notion are two interconnected parts of the notion. The scope is a class of objects generalised in the notion, and content is a set of (significant) properties that are used to generalise and single out objects in this notion.

(2) Relations tie together classes and describe them. The most common type of relations used in ontologies is categorisation which means distribution to a specific category. This type of relations has other names: taxonomic relation, IS-A relation, class-subclass, hyponym-hyperonym, subsumption relation, a-kind-of relation.

(3) Axioms set the rules of distribution to categories and relations. They represent the obvious statements that link notions and relations. By axiom we understand a statement that is taken to be true. Axioms could be entered into ontology in the completely finished state. In turn, these axioms can be used to obtain new

ISBN 978-3-9504173-6-4 (CD), 978-3-9504173-7-1 (print)
Editors: M. SCHRENK, V. V. POPOVICH, P. ZEILE, P. ELISEI, C.BEYER, J. RYSER

statements. They allow to convey the information that cannot be reflected in the ontology by building the hierarchy of notions (classes) and by setting relations between them. Axioms permit to make deductions within the ontology. They can provide the decision-making support system with information about the rules that regulate automatic adding of context information. Axioms can also act as restrictions on any relations which makes it possible to make deductions later.

Along with the listed elements, ontology also includes “instances”, e.g. individual members of entity or event classes, in other words concrete elements of a category.

Basic requirements for the ontology of the city environment, according to (Korpipää et al., 2003), are the following:

(1) simplicity: expressions and relations must be simple to use

(2) flexibility and scalability: adding of new notions and relations to the ontology must be clear and accessible

(3) universality: ontology must support different types of information (textual, graphical, etc.) and of knowledge about the city environment

(4) expressiveness: ontology must support description of the necessary number of attributes in order to adequately reflect the information and knowledge about the city environment.

Technologies for representation of knowledge about the city environment that are based on ontologies permit to speed up the creation of new databases and to keep the existing ones up to date in accordance with information available.

One of the most convenient ways of ontology representation is a tree of classes which permits to describe the ontology in a knowledge representation language using such tools as CLIPS, JEES, etc. For that, special knowledge representation languages, understandable for the inference machine of the expert system, are used. Notation of said languages permits to describe each class (ontology’s notions) with its corresponding slots (ontology’s relations).

Among the major components of city environment’s ontology we can highlight the following:

- information about constructions and buildings
- information about the transport system
- information about the energy system
- information about the environment
- information about citizens
- information about urban space use.

5 SCENARIO ONTOLOGY FOR CITY ENVIRONMENT

The second type of ontology for the city environment is scenario ontology. The scenario notion (Popovich et al., 2008) generalises the algorithm notion. This generalisation serves to enable us to represent a set of parallel interconnected processes in the form of a scenario, since complex situations in the city environment or users’ (citizens) actions usually play out by scenario rules.

Formally, a scenario can be defined as a sequence of phases and decisions. A phase is a set of elementary actions performed sequentially or in parallel. The decision is an element of a scenario in which the process can change direction depending on prevailing conditions at the time. Formally, the decision is defined as a combination of branches.

Actions act as “building” blocks for scenarios. They represent specific elements of actions of the participants in scenarios realised in different ways. Actions lead to various events or changes in the environment. It is assumed that actions take place over time.

In contrast to algorithms, actions that belong to a phase of a scenario can be executed not only sequentially but also in parallel. The second important difference from the algorithm is that actions have duration in time while algorithms usually do not take duration into account (sometimes the number of phases is considered).

Different duration of actions coupled with the possibility of their parallel execution in scenarios result in the need to synchronise the parallel branches of a scenario.

Any object's action can be described by a particular scenario, elemental actions of which, in turn, can have their own particular programs and so on. Consequently, in scenario decomposition, it is necessary to estimate the level of detail of elemental actions.

Graphical primitives are used for creation of scenarios and their phases. Graphical primitives have a certain contextual interpretation. In general, a scenario is represented in the form of a two-dimensional oriented graph, in which actions serve as nodes and lines of transition from one action to another are arcs. Both types of graphs can be cyclic.

The more illustrative instrument for the representation and modelling of user's actions within the given scenario is a consolidated tool based on interference machine of RETE type.

RETE algorithm (Forgy et al., 1982) has been successfully applied for over a decade for computer implementation of interference machine in knowledge representation and processing systems based on rules in expert systems in particular. This algorithm, which has already proven to be effective, is a network algorithm in a sense that before the beginning of algorithm's operation all rules are transmitted into network. Facts that describe states and changes in the outside world act as initial information for this algorithm. During the algorithm's operation, facts move along the network from entry nodes to exit nodes and are stored in intermediate nodes. Due to such structure and organisation, effectiveness of RETE algorithm stays independent of the number of rules. The main reason for its high operation speed is that only new facts are subject to verification.

In order for the RETE algorithm to operate effectively, the set of factors must change slowly. Current realisations of the algorithm operate reasonably well with hundreds of facts and ten thousands of rules.

The modern approach based on object-oriented ontologies is used for representation of knowledge that constitutes the scenarios and the rule set. Development of scenario ontology includes the following steps:

(1) plotting scenario schemes for the simulated situations

(2) plotting schemes for scenario phases

(3) implementation of decision blocks into scenarios

(4) implementation of concrete "elemental" actions from which the scenarios of simulated situations are built.

It is worth noting that some steps, in the general case, may be missing, e.g. when we create a scenario that consists only of already implemented actions. The order of execution of these steps may also vary, e.g. when we already know beforehand what phases will be included in the scenario we can begin from step 4. Moreover, a visual scenario design environment permits to switch from the design of a scenario or a phase to the design of a needed action and back at any time.

The most general notion in scenario ontology is the "activity" notion, represented by an abstract class Activity. Concrete subclasses of this class are listed below:

- Scenario
- Phase
- Decision
- Action.

Main attributes of the Activity class and of its subclasses are listed below:

- status: the state of the given activity
- run: concrete realization of scenario execution
- context: the context or the "environment" of the given realization of the scenario
- period: limitation on action's execution time.

ISBN 978-3-9504173-6-4 (CD), 978-3-9504173-7-1 (print)
Editors: M. SCHRENK, V. V. POPOVICH, P. ZEILE, P. ELISEI, C.BEYER, J. RYSER

The status attribute can have various values depending on the status of a concrete type of action. For example:

(1) START: the beginning (initialisation) of a given type of activity

(2) DOING: a given type of activity is being performed

(3) REPEAT: a repeated action

(4) DONE: the activity is fulfilled, etc.

Scenarios can be general (universal) which means they can suit different users' actions. To represent such scenarios, one must use variables that designate objects and values contained in the scenario. Upon launching of such a scenario, concrete values must be assigned to the variables, i.e. a combination of pairs of "variable – value" type which constitutes a contextual information. If one common scenario is launched with two different sets of values of variables in parallel then we have two simultaneously executed realisations of the program. The run notion is used to identify and distinguish such scenarios. The run identifier is used, for example, in interruption of execution of a concrete realisation of the program. Using the run identifier, the inference machine identifies which phases and actions must be interrupted.

To support the universal scenarios, the scenario ontology contains the following classes: Context, Variable, Pair ("variable – value" pair), Object. In addition, for better visualisation of scenarios the scenario ontology contains the following classes: ScenarioScheme, PhaseScheme.

Special visual environment for ontology development is used to plot scenario schemes using graphical primitives. Scenario scheme plotting is done by dragging the visual components that represent phases and decisions from a special palate or library to the scenario's scheme field. Then the components are connected using lines that have various semantic interpretations. Lines that exit the phases indicate the unconditional transition to the next phase or decision. Lines that exit decision blocks indicate transition to one or another variant depending on the decision. Such lines are dotted. The process of visual design of a scenario scheme is presented in Fig. 3.

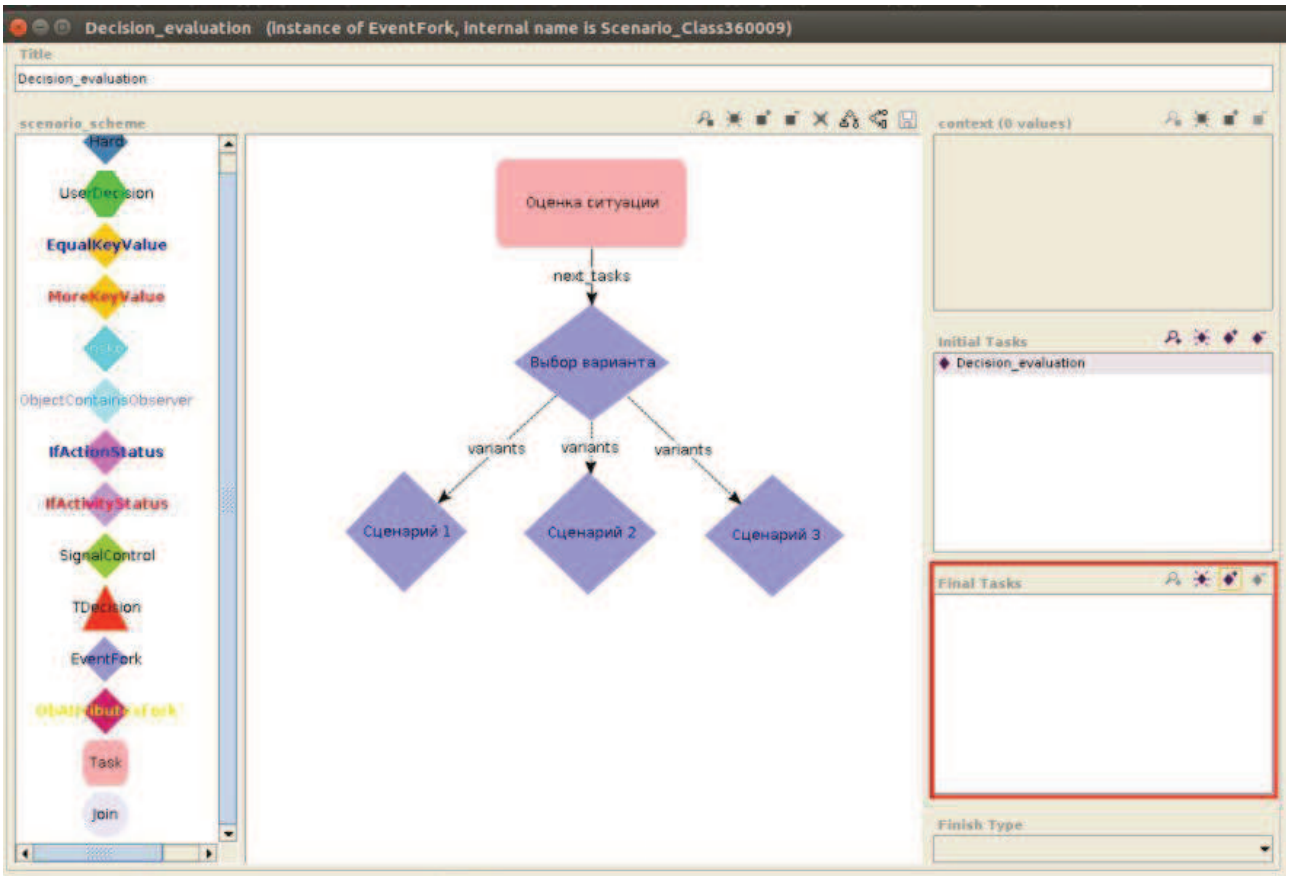

Fig. 3: Visual representation of scenario scheme.

The process of a scenario scheme design ends with the creation of the instance of Scenario and with filling its slots. In particular, an instance of the scenario scheme is entered into the corresponding slot. The slots of beginning and end phases of the scenario are filled as well. Such a scenario is not concrete because its phases are empty, without actions. However, such a scenario can already be executed by the scenario interpretation system. Its progress will be indicated only by messages about fulfilment of one or another phase.

Plotting of phase schemes happens in a similar way to scenario scheme plotting. The difference is that the palate in this case contains visual components that represent elementary actions. It is necessary to create the corresponding subclasses of Action class beforehand. It should be emphasised that there is no need to know and describe the realisation of these actions in advance. It is enough to approximately decide on the necessity of the existence of such notions. Even their names can be easily changed later without interrupting system's operation. For instance, we know that objects must appear from somewhere, thus we feel the necessity of "notion" of "Create" action. Concrete realisation of this notion can be done later. An example of the phase scheme is given in Fig. 4.

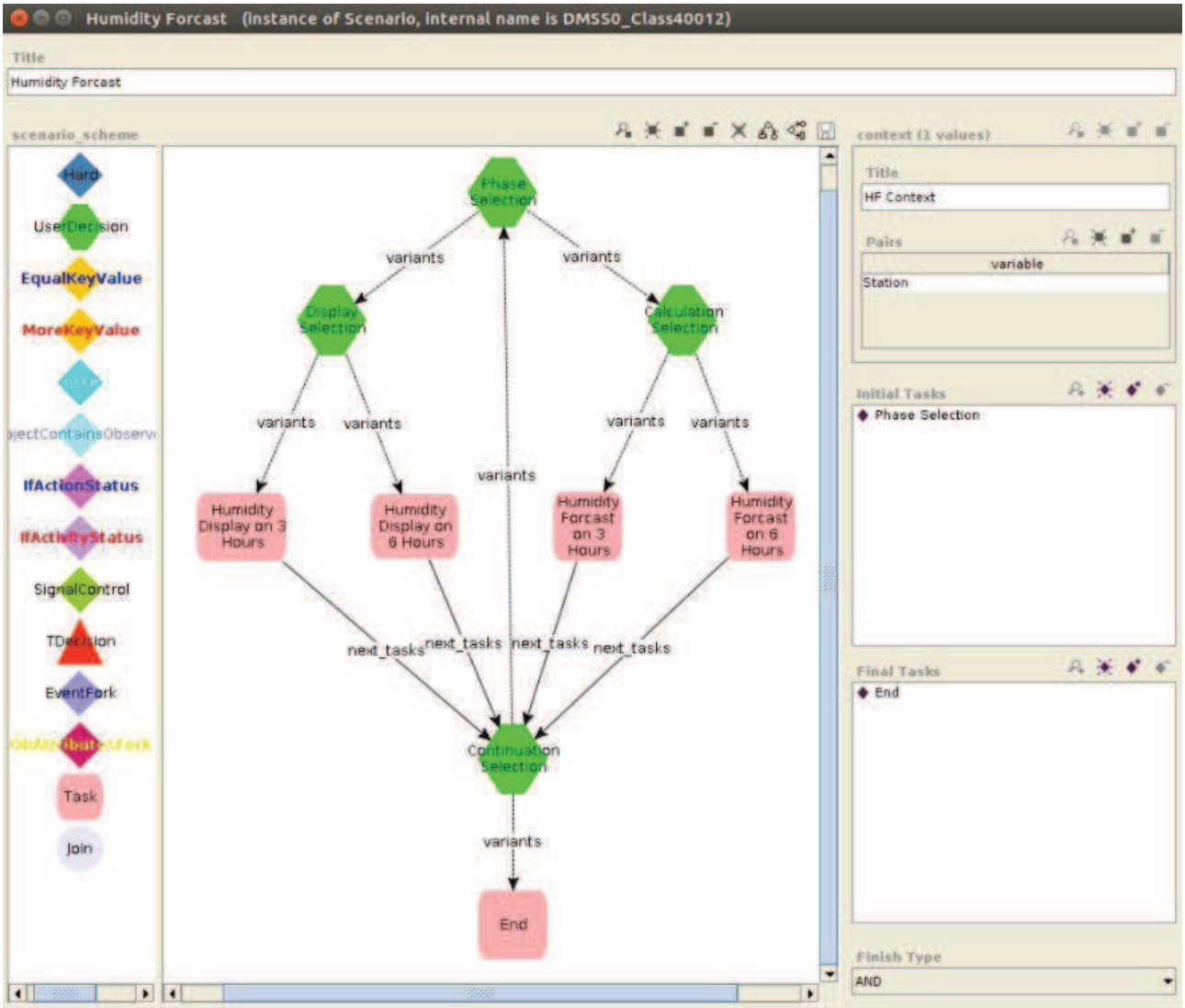

Fig. 4: Example of phase scheme.

The fundamental difference between phase schemes and scenario schemes lies in the fact that the progress of the scenario's execution is continuous only over one branch after reaching the branching point and depends on conditions in the decision block, while the progress of phase's execution is continuous over all the branches of the branching point regardless of any conditions. One should keep in mind this important principle while mentally breaking down the complex process into phases and elemental action.

A phase can have several beginning actions which means that the phase scheme can show several connected graphs like the one in Fig. 4. After initiating the phase execution, all processes that these graphs describe will

ISBN 978-3-9504173-6-4 (CD), 978-3-9504173-7-1 (print)
Editors: M. SCHRENK, V. V. POPOVICH, P. ZEILE, P. ELISEI, C.BEYER, J. RYSER

launch simultaneously and will be executed in parallel. They will later fork in their branching points into more and more parallel processes until terminal actions in all graphs are performed.

The phase is considered terminated when all end actions of this phase have been performed. However, not all terminal actions that are part of the graph scheme can be included in the end actions of a given phase. It means that a phase can end and a transition to another stage will happen before all the processes in this phase are over. Furthermore, the list of end actions of the phase can be empty and it will indicate transition to the next phase right after the start of all processes that are tied to beginning actions of this phase. It is a phase that consists of one action and a phase that consists of several actions with no continuation.

Realisation of decision blocks in scenarios begins with the creation of necessary subclasses of the Decision class for every typical decision-making situation. For every such situation one should write a simple program in CLISP or Jess languages depending on the inference machine used in the scenario system support. The essence of the program lies in the verification of the set of conditions. As a result, the program must return the identifier of a branch over which the scenario must progress. The program is entered into a rule-decision slot of a given situation of a subclass of a Decision class. Initial data for this program will be values of slots of concrete instance of a subclass of a Decision class. During the step of decision block realisation one needs to create these slots (or use the existing ones) and include them into the given subclass. During the execution, the program obtains access to slot values through variables whose names match the names of slots but begin with “?” symbol. In the course of scenario execution, corresponding slots must be filled before entering the given decision block.

6 CONCLUSION

The created ontology, which includes the city environment ontology as a subject domain ontology and a scenario ontology that describes the behaviour of objects in the city environment, can be used as a basis for the creation of a decision-making support system for city environment management. Its distinctive feature is the existence of a scenario ontology that permits to model the most common activities in city management as well as unique ones. This technology is based on pre-designed rules that permit to automatically use contextual information about the city environment.

7 REFERENCES

BREITMAN et al., Breitman K.K., Casanova M.A., Truszkowski W.: Semantic Web: Concepts, Technologies and Applications. Springer-Verlag London Limited 2007.

FORGY et al., Forgy C. Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem. In: J. Artificial Intelligence, no. 19, 1982, pp. 17-37.

GRUBER T.R., Gruber T.R.: Translation Approach to Portable Ontology Specification. Knowledge Acquisition Journal, 1993, vol.5, pp.199-220.

KORPIPÄÄ et al., Korpipää P. and Mäntyjärvi J. An ontology for mobile device sensor-based context awareness. In: Proceedings of CONTEXT, 2003, vol. 2680 of Lecture Notes in Computer Science, 2003, pp.451–458.

POPOVICH et al., Intelligent geographic information systems for monitoring maritime objects. Eds. R.M. Yusupov, V.V. Popovich, St. Petersburg, 2013.

POPOVICH et al., Popovich V.V., Prokaev A.N., Sorokin R.P., Smirnova O.V. About situation recognition on basis of artificial intelligence technology. In: SPIIRAS Proceedings, issue 7, St.Petersburg, 2008.

RAMAPRASAD A., Ramaprasad A., Ortiz A., Syn T. Ontological Review of Smart City Research. In: Twenty-third Americas Conference on Information Systems, AMCIS 2017, Boston, 2017.