An ontology for the conceptualization of an intelligent environment and its operation

Driss Sadoun, Catherine Dubois, Yacine Ghamri-Doudane, B. Grau

An Ontology for the Conceptualization of an Intelligent Environment and its Operation

Driss SADOUNI*†, Catherine Dubois‡§, Yacine Ghamri-Doudane‡¶, Brigitte Grau*‡
*LIMSI/CNRS B.P. 133 91403 Orsay Cedex, France
†University Paris-Sud, 91400 Orsay, France
‡ENSIIE, 1 square de la resistance, 91000 Evry, France
§CNAM-CEDRIC 292 Rue St Martin FR-75141 Paris Cedex 03, France
¶(LIGM)University Paris-Est-Marne-la-Vallee, 75420 Champs sur Marne, France

***Résumé*—Nowadays sensors and actuators are increasingly used in different spaces, creating intelligent environment. This article aims to describe a conceptualization of an intelligent environment and its operation, in order to check its consistency and its conformity. This conceptualization is done through an ontology representing the domain knowledge, whose elements will be instantiated from natural language texts describing the physical configuration of an intelligent environment and a scenario describing the operation desired by the user of the environment. We chose OWL to represent formally our environment augmented with SWRL rules to represent the dynamic aspect of the operation system and SQWRL to query our conceptual model. We show how consistency and conformity are checked thanks to this formalism.**

***Keywords*-Ontology conception ; specifications ; formal verification ; intelligent environment ;**

I. INTRODUCTION

Comfort, energy saving and safety are criteria that nowadays reflect human needs in daily life environments. The profusion and the diversity of sensors, their affordable costs, let imagine different possibilities for an intelligent environment that could adapt itself to person needs. The ambition of our Project « *ENVIE VERTE* »[1] is to enable the piloting of an intelligent environment using textual descriptions in natural language. These descriptions concern on one hand information about the physical description of the intelligent environment (number of sensors, types, locations, interactions, ...) and on another hand, end user needs (do not leave empty rooms lit, detect gas, ...). In order to deploy a system that reflects the needs described by an individual, it is necessary to verify its conformity (the environment is correctly configured) and its logical consistency (no contradiction in its operation). These verifications require dealing with precise and unambiguous specifications of the environment and of the end user requirement. As natural language specifications do not fulfill these requirements, we propose to build an intermediate conceptual representation, enabling a transition towards formal specifications, and allowing checking the logical consistency and the conformity of the intelligent environment. The conceptual representation will therefore be the link, between natural language specifications and formal specifications.

Our present objective is to build a conceptual representation of an intelligent environment that will be instantiated from the description of the environment and end user requirement provided as scenarios. To achieve this, we created a high-level OWL ontology which provides the ability to reason, to do queries and to conclude on the appropriate actions to perform according to a scenario. This conceptual representation will enable to generate consistent and formal specifications which could be further verified using formal methods and tools.

The originality of our approach is the use of the ontology logic formalism to represent and check the consistency and the conformity of specifications, and to realize it in two steps : a static part leads to create individuals for representing a given environment, and a dynamic part leads to create rules for representing end user scenarios.

The paper is organized as follows. First we describe the intelligent environment, its components and the two types of textual descriptions in Section 2. In Section 3 we describe the intelligent environment checking process. In Section 4 we detail our approach to produce a conceptual model representing the natural language (NL) descriptions. In Section 5 we illustrate with an application case how our model will be used to check consistency of NL descriptions. In Section 6 we give a view of related work. And finally we conclude and define some future works.

II. INTELLIGENT ENVIRONMENTS

The intelligent environment consists of a set of communicating objects (sensors, actuators and control devices) which can be seen as a sensor network. These objects influence the operation of equipment (physical processes) located in the environment, under well defined conditions.

1. founded by DIGITEO, project DIM LSC 2010.

We can distinguish a hardware part : the kind of devices, their number, the type of each device, their location ... and a software part which represents the configuration of its operation.

In the following, we briefly describe the general functions of an intelligent environment :

- A sensor detects the occurrence of a phenomenon or measure a quantifiable phenomenon in a bounded area.
- A phenomenon, to be detected or measured by a sensor, must be located in the sensing zone of the sensor and be from a type perceived by the sensor (temperature, motion ...)
- An actuator is fixed on or connected with an equipment of the environment.
- Once a phenomenon (or a set of phenomena) is detected or measured, a control of the collected information is processed and can lead to the activation of one or several actuators trigging a set of actions (turn on, turn off, increase, decrease) on the equipment they are fixed on or connected with.
- An actuator can be activated by a sensor (or a set of sensors), if it is located in its (their) zone of control and is able to analyze the information perceived and transmitted by the sensor(s).

To pilot her intelligent environment, a user will determine, according to the given physical configuration, the general functions that should hold to satisfy her needs.
Figure 1 shows how a user will configure her own system by writing textual descriptions. These descriptions will be analyzed to instantiate automatically elements of an OWL ontology representing the domain knowledge, which will be used to generate formal code. Verification of the resulting model will be processed to detect inconsistencies and missing information and therefore to correct and improve the model by interacting with the user until it is consistent and satisfies her needs. In this article we will focus on the conception of the ontology and the verification it enables to perform.

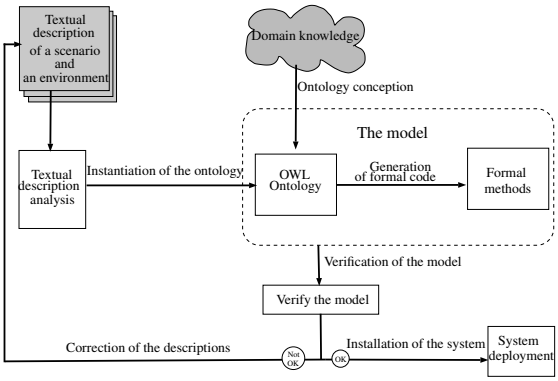

FIGURE 1: Architecture of the configuration process

In this framework textual descriptions are written in English. These descriptions can be separated into two parts :

1) *Description of an intelligent environment* : It describes components of the intelligent environment, (sensors, actuators, physical processes ...), their number, their type, their location and how they can interact. This part will be processed before user requirement and define a static state of the environment. The simple intelligent environment described in the following example is shown Figure 2
   Example : *the green apartment includes a hall, two bedrooms, one bathroom and a living room which includes a kitchen and a dining room. Each room is equipped with motion sensors. Each light bulb is equipped with an actuator.*
2) *User requirements* : These texts describe users needs, i.e. how objects of the intelligent environment have to interact and in which conditions. They allow producing different instantiations of the ontology according to different users scenarios.
   Example : *When a person movement is detected in the living room, illuminate both the kitchen and the dining room.*

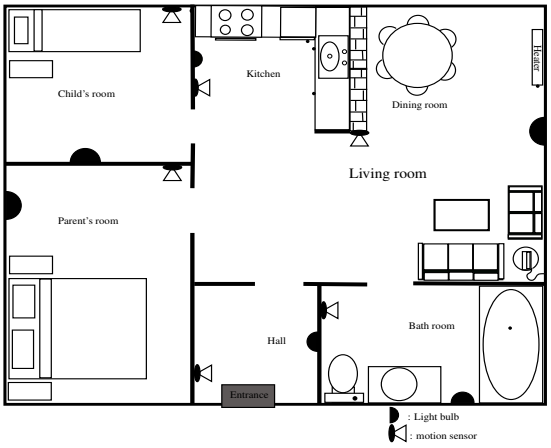

FIGURE 2: The green apartment

III. INTELLIGENT ENVIRONMENT CHECKING

Before deploying the intelligent environment, it is important to check whether its operation is conform to its specifications. Such verifications will be realized by using formal methods, which requires precise and unambiguous specifications. So the conceptual representation of the operation of the intelligent environment must be free of any contradiction and ambiguity. It must also represent all the properties which define the functioning of the modeled environment, to be admissible.

A. Chosen model

We decided to use OWL (Web Language Ontology) for the conception of our model. OWL provides a mean to produce a formal semantic representation, thanks to its logical formalism based on description logics [13] which are logic-based formalisms used for knowledge representation with a high expressivity power [2]. OWL enables to reason on an ontology to check its logical consistency. OWL is extended by a Semantic Web Rule Language SWRL [12] which allows us to represent the dynamic aspect of the intelligent environment operation.
Thanks to the Semantic Query-Enhanced Web Rule Language SQWRL [15], we can query the ontology in order to find anomalies or missing information and thus check the conformity of the model.

B. Necessary checks

Consistency and conformity of the model are checked with respect to two aspects :

- checking physical configuration,
- checking user requirement.

We also will check the state of the intelligent environment for different scenarios and verify general properties as human security, energy saving...

An important part of the verifications will be done using OWL. However, it does not allow non-monotonic reasoning, which limits the verification process. For example it is not possible to represent a dynamic change in the state of an individual. General property verification is also outside the scope of OWL. So we envisage the second part of the verification process using Focal [16] which is an object oriented specification and proof system allowing to incrementally build components and to formally prove their correctness. We will show how we use OWL in this context.

IV. CONCEPTION OF THE ONTOLOGY

An ontology represents a domain knowledge in an understandable way for both human and computer. It is formed by a set of concepts which are organized hierarchically and defined by properties. Several studies have already shown an interest in ontology development of sensor networks [18], [1], intelligent environment [11], or sensor network component operation[19]. In this framework, we want to model the operation of an intelligent environment, taking into account the operation of a sensor network and its interaction with various objects of its environment.

The aim is to identify concepts (classes), terms (individuals) and relations (properties) of the domain, as well as the representation that will be the most appropriate to formally represent the intelligent environment operation.

The conceptual modeling of the intelligent environment is less likely to change than the needs of a user. Thus, the ontology structure (concepts and properties) can be defined and fixed by human study of the domain knowledge of intelligent environments and sensor networks. Contrariwise the ontology instantiation will vary depending on both environment configuration and requirement descriptions. So in the following we will focus on defining concepts and properties.

A. Concepts

From the intelligent environment section, it appears that we need to define concepts that represents *sensor network components*, *locations*, *phenomena* and *physical processes*. In our model we decided to consider that the sensor network components are only *sensors* and *actuators*, since the operation of control devices could be modeled using the logic based formalism of OWL (constraints and inference rules). We also made the distinction of two types of phenomena, those which could appear suddenly (a person movement, gas leak...) that will belong to the class *Event* and those which could be measured (temperature, humidity...) that will belong to the class *Measurable*. It follows that we have to distinguish two concepts of sensors, those which detect an Event, that will belong to the class *Detecting_sensor* and those that measure a Measurable, that will belong to the class *Measuring_sensor*. Figure 3 shows the hierarchical organization of the ontology.

B. Properties

Properties represent interactions of concepts, they relate instances of concepts to some other instances or to a data type (int, string...). See below the properties that model the operating point of view on the network (*between parenthesis appears the name of the related OWL property*) :

- A phenomenon has a type (*Has_type*).
- A sensor has a location (*Located_in*), has a zone of sensing (*Zone_of_sensing*), a zone of control (*Zone_of_control*) and perceives a type of phenomenon (*Perceived_type*).
- A Measuring_sensor measures a measurable phenomenon (*Measure*).
- A Detecting_sensor detects an event (*Detect*).
- An actuator actuates on a physical process (*Actuate_on*) and manages one or several type(s) of phenomena (*Managed_type*)

Figure 3 shows how concepts interact in the ontology. The link *Is_a* describes a taxonomic hierarchy along which concepts are inherited.

The notion of *type* is useful to associate phenomena, sensors and actuators which have to be involved in a same process. Thus we can warrant that a phenomenon will be handled by the proper sensor that will activate the proper actuator.

We distinguish two types of properties : i) the properties which will be created at the instantiation of the ontology, *Located_in*, *Zone_of_sensing* and *Actuate_on* which must be defined in the descriptions and will be used for inference ; ii)

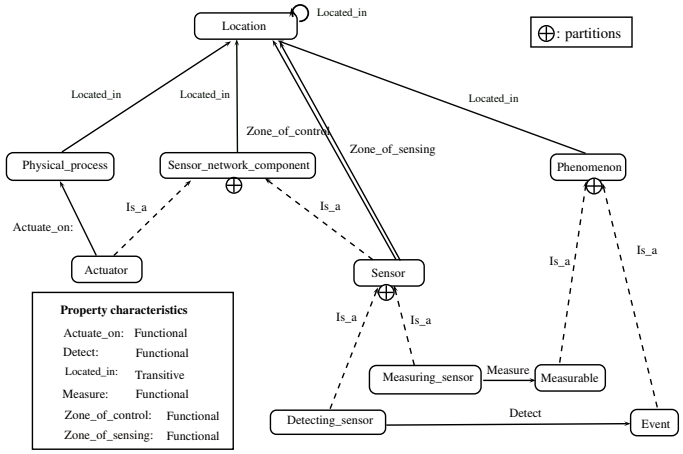

FIGURE 3: The ontology

the properties that will be created by the reasoning process on the ontology, e.g. *Detect* and *Measure* which will be created by inference rules.

C. Individuals

The changing part of the ontology is its instantiation according to a specific environment and a user requirement. Individuals will be extracted automatically from the environment description and a scenario, together with their properties. These properties will be used during the reasoning process, to classify each individual in the class it belongs to.

D. Reasoning on the Ontology

During OWL reasoning, inferences are made, classifying elements of the ontology and creating new assertions while maintaining logical consistency.

1) Classification and Assertion: OWL axioms are used for the hierarchical organization of concepts and for the classification of individuals. Thus, assume C_1, C_2 are concepts and P_1, P_2 are properties. OWL defines two kinds of axioms.

i) *Inclusion axiom* $C_1 \sqsubseteq C_2$ ($P_1 \sqsubseteq P_2$), e.g. $Detecting\_sensor \sqsubseteq Sensor$

Each element of C_1 is an element of C_2, i.e. C_1 is a sub class of C_2

ii) *Equality axiom* $C_1 \equiv C_2$ ($P_1 \equiv P_2$), e.g. $Detecting\_sensor \equiv Sensor \sqcup \exists Detect.Event$

Each element of C_1 is an element of C_2 and vice versa.

To write more expressive conditional rules, we used the Semantic Web Rule Language (SWRL). In our model we distinguish two kinds of SWRL rules.

General rules : Are independent from textual descriptions and represent the generic behavior of the intelligent environment.

Instead of writing a rule for each kind of phenomenon, we modeled two rules in order to deduce that a phenomenon, a sensor and an actuator share the same type.

Rule 1 : If the type of a phenomenon is the same as the type perceived by a sensor, then they share the same type.

$Has\_type(?p,?t), Perceived\_type(?s,?t)$
$\rightarrow Shared\_type(?p,?s)$

Rule 2 : If the type managed by an actuator is the same as the type perceived by a sensor, then they share the same type.

$Managed\_type(?a,?t), Perceived\_type(?s,?t),$
$\rightarrow Shared\_type(?a,?s)$

Rule 3 : A sensor detects an event if the event occurs in its zone of sensing.

$Event(?e), Detecting\_sensor(?s), Shared\_type(?s,?e),$
$Located\_in(?e,?l),\ Zone\_of\_sensing(?s,?l)$
$\rightarrow Detect(?s,?e)$

Rule 4 : A sensor measures a measurable phenomenon if this phenomenon occurs in its zone of sensing.

$Measuring\_sensor(?s), Measurable(?m), Shared\_type(?s,?m),$
$Located\_in(?m,?l), Zone\_of\_sensing(?s,?l)$
$\rightarrow Measure(?s,?m)$

Generated rules : Textual analysis will generate automatically rules to denote users requirement. We differentiate two parts in generated rules, a fix part and a part generated from requirement descriptions. (*This second part is underlined in the rules*)

When a sensor detects an event, the actuator of the same type has to turn on the appliance it actuates on.

$Actuator(?a), Physical\_process(?d), Actuate\_on(?a,?d),$
$Detect(?s,?e), Shared\_type(?s,?a), Located\_in(?a,?l),$
$Zone\_of\_control(?s,?l) \rightarrow \underline{Turn\_on(?a,?d)}$

When a sensor measures a value greater than 30, then the actuator of the same type has to reduce the appliance it actuates on.

$Actuator(?a), Physical\_process(?d), Actuate\_on(?a,?d),$
$Measure(?s,?m), Shared\_type(?s,?a), Located\_in(?a,?l),$
$Zone\_of\_control(?s,?l), Has\_value(?m,?v),$
$\underline{greaterThan(?v,30)} \rightarrow \underline{Reduce(?a,?d)}$

2) Consistency Checking: Classifications and assertions can be made only if they are consistent with the ontology

Let C and C' be classes, P a property, x and y individuals.

- If C' is asserted as a subclass of C, it is necessary to verify that all individuals of the class C' can belong to the class C. So if an individual of C' cannot belong to C then the ontology is not consistent.

- If $C(x)$ is asserted, it is necessary to verify that it is possible for the individual x to belong to the class C. So

if x belongs to C' which is disjoint with C then the ontology is not consistent.

- If $P(x,y)$ is asserted it is necessary to check that x could belong to the domain of the property and y to the range.

3) Conformity Checking: To verify conformity of the intelligent environment, we query our ontology looking for wrong or missing specifications. We do this using SQWRL [15], a SWRL-based language for querying OWL ontologies, providing SQL-like operations to retrieve knowledge from OWL. Thus, we can check whether two opposing actions may occur simultaneously.

V. Application

We will detail a case on which our model is applied in order to show how instantiations and verifications are made. The ontology building is realized under *Protégé* and the consistency and conformity checking are implemented under a java application using *Jena*, a Java framework for building Semantic Web applications.

Physical configuration description : *the green apartment includes a hall, two bedrooms, one bathroom and a living room which includes a kitchen and a dining room. The sensor S_{temp} measures the temperature of the living room. The sensor S_{move} detects movements in the living room. The living is equipped with a light bulb L_{living} on which an actuator AL_{living} is fixed, and the dining room is equipped with a heater H_{dining} on which an actuator AH_{dining} is fixed.*

living room lighting scenario : *When a person movement is detected in the living room, turn on the lights of the living room.*

living room heating scenario : *When a person movement is detected in the living room, and the temperature is below 25 degrees turn on the heater of the dining room.*

A. Assertions and Consistency Checking

Below, assertions that have to be extracted from descriptions, follow by deductions they enable to produce :
Static part
1) - living room which includes a kitchen.
$\rightarrow$ Located_in(dining_room , living_room)
- dining_room is equipped with a light bulb.
$\rightarrow$ Located_in(L_{living}, dining_room)

As *Located_in* is transitive.
$Located\_in(dining\_room, livingroom)$
$\wedge\ Located\_in(L_{living}, living\_room)$
$\Longrightarrow Located\_in(L_{living}, livingroom)$

2) S_{temp} measures the temperature of the living room.
$\rightarrow$ Measure(S_{temp},$temp_{living}$)

As the domain of the property *Measure* is *Measuring_sensor*.
$Measure(S_{temp}, temp_{living}) \Longrightarrow Measuring\_sensor(S_{temp})$

Dynamic part

1) When a person movement is detected in the living room ...
$Event(person\_movement), Sensor(S_{move}),$
$Shared\_type(S_{move}, person\_movement),$
$Located\_in(person\_movement, living\_room),$
$Zone\_of\_sensing(S_{move}, living\_room)$
$\rightarrow Detect(S_{move}, person\_movement)$

Suppose this rule holds for for $Sensor(S_{temp})$, since the domain of *Detect* is *Detecting_sensor*, it will be inferred that S_{temp} is a *Detecting_sensor* and thus an inconsistency will appear since in the conceptualization *Detecting_sensor* and *Measuring_sensor* are disjoint and so the individual S_{temp} cannot belong to both.

2) When a person movement is detected in the living room, and the temperature is below 25 degree turn on the heater of the dining room.

$Actuator(AH_{dining}), Located\_in(AH_{dining}, living\_room),$
$Actuate\_on(AH_{dining}, H_{dining}), Physical\_process(H_{dining}),$
$Detect(S_{move}, person\_movement), Shared\_type(S_{move}, AH_{dining}),$
$Zone\_of\_control(S_{move}, living\_room),$
$Measure(S_{temp}, temperature), Shared\_type(S_{temp}, AH_{dining}),$
$Zone\_of\_control(S_{temp}, living\_room),$
$Has\_value(temperature, ?v), lessThanOrEqual(?v, 24)$
$-> Increase(AH_{dining}, H_{dining})$

The properties Shared_type are deduced by the rule described in 5.4. We can notice that in our model actuators can react to different types of information. The types of this information have to be specified in the environment description. Suppose that there is another rule which holds, asserting $Reduce(AH_{dining}, H_{dining})$. Since in our model properties *Increase* and *Reduce* are disjoint, an inconsistency will be generated which guarantees that an actuator cannot for a given instantiation do opposite actions. The case where another actuator tries to reduce the heater at the same time $Reduce(another\_actuator, H_{dining})$ cannot be prohibited by OWL. We resolve this problem thanks the conformity checking.

B. Conformity checking

By querying the ontology we can check whether the intelligent environment is correctly configured and identify missing specifications.

1) Incoherent scenario: Incoherence could result from the release of two opposite actions.
Opposite properties : This query gives the intersection between sets of reduced and increased physical process. If the intersection is not empty then some asserted properties are opposite and the scenario has to be corrected.
Let *?s1* is the set of reduced physical process, *?s2* is the set of increased physical process and *?s3* is the set of their intersection.

$Reduce(?a1, ?d1) sqwrl : makeSet(?s1, ?d1)$
$Increase(?a2, ?d2) sqwrl : makeSet(?s2, ?d2)$
$sqwrl : intersection(?s3, ?s1, ?s2) sqwrl : size(?n, ?s3)$
$\rightarrow sqwrl : select(?n)$

2) Missing information: Missing information can generate useless devices and prevent to use all the potential of the intelligent environment.

Sensors without Zone of Sensing : *?s1* is the set of all sensors of the ontology, *?s2* is the set of sensors which have a zone of sensing and *?s3* is the set of sensors without a defined zone of sensing.

$Sensor(?s) sqwrl : makeSet(?s1, ?s) Sensor(?s\_zos)$
$Zone\_of\_sensing(?s\_zos, ?l) sqwrl : makeSet(?s2, ?s\_zos)$
$sqwrl : difference(?s3, ?s1, ?s2) sqwrl : size(?n, ?s3)$
$\rightarrow sqwrl : select(?n)$

Actuators without connection on a physical process : ?s1 is the set of all actuators of the ontology, ?s2 is the set of actuators which actuate on a physical process and ?s3 is the set of sensors which have without a physical process to actuate on

$Actuator(?a) sqwrl : makeSet(?s1, ?a) Actuate\_on(?ad, ?d)$
$sqwrl : makeSet(?s2, ?ad) sqwrl : difference(?s3, ?s1, ?s2)$
$sqwrl : size(?n, ?s3) \rightarrow sqwrl : select(?n)$

VI. RELATED WORK

Formal models used for consistency verification or model checking of real-world systems, are generally conceived from natural language documents. Actually most of requirements are written in natural language, which could be well-understood by a domain expert or an ordinary person. So the issue of the transition from informal (natural language texts) to formal (specifications) arises naturally. Several studies have focused on it. [17] aims at applying several formal methods to certify documents of airport standard security regulations, these documents are analyzed by a model engineer in order to produce conceptual (graphical) models in UML applying requirements engineering methods [14]. The resulting models are transformed into formal models which will be verified by formal methods tools [9]. [10] used conceptual graphs as an intermediary representation to formalize the interactions of telecommunication services and generate formal specifications in Z notation.[4] presents how the Two-Level Grammar (TLG), a specification language of requirements [5], is used as an intermediate to transit from natural language into formal specifications in VDM++, an object-oriented version of the Vienna Development Method. These approaches reveal the need of an intermediary representation to go from informal to formal specification.

Our choice of an OWL ontology as conceptual representation is motivated by its semantic which is more expressive than the cited approaches and its logical formalism that allows the system to reason formally on the represented knowledge. Thus the kind of intermediate representation we chose allows us to make verifications at this stage of analysis, using properties of OWL, and to rely on formal methods, that are heavier, only for proving general properties.

The automatic construction of ontology from texts in natural language is a hard task, and a fully automatic approach is not realistic. Identifying the relevant concepts of a domain is semantically too difficult to be done efficiently without a human interaction. So the most common approach consists in a semi-automatic building of ontology. Several methods and tools were developed for ontology conception, [3], [8], [7], [6], however the complexity of our model that does not rely on the definition and the organization of a lot of concepts and relations, did not require their use.

The ontology we propose differs from [18] as their ontology is analogous to a database for querying and searching sensors. Thus all kinds of sensors are modeled according to the perceived types of phenomena. A same remarks can be made for [1] which describes sensor functionality and current state. [19] is closer to our work as it proposes an ontology for actuator discovery and the definition of high-level behaviors. However their modeling requires the description of sub-classes of devices, which entails to define all of them, and may be incomplete, and, in the same way, different types of operations, which are generic properties in our model that are inferred according to the kind of phenomenon perceived.

Our ambition is to allow a non expert user to pilot her own environment, and to recognize erroneous or incomplete specifications in order to give her some help for improving them. Thus, our approach finds its originality with the automatic instantiation of an ontology and the dynamic production of rules from the textual analysis of natural language descriptions, which allows us to do verifications. In our approach, what we need to check is not the conceptual modeling of the ontology but its instantiations, i.e. descriptions of the physical configuration of the environment and of the user requirement.

VII. CONCLUSION AND FUTURE WORK

In this article, a conceptual representation of the operation of an intelligent environment was defined and implemented, using an OWL ontology. The resulting ontology aims at facilitating the transition from natural language descriptions to formal specifications in order to check all along the consistency and the conformity of the conceptual specifications. The model we proposed acts in two steps. The first one, the static part, represents the environment and enables the system to check both its conformity and the network consistency. The second one, the dynamic part, enables to create rules that represent users' scenarios and to verify their consistency. This formalization will allow the system to navigate between texts and formal specifications to correct or improve the textual descriptions.

The conceptual representation which is the bridge between natural language specifications and formal specifications, is the first step in the process of developing easy to use and reliable tools for intelligent environment configuration. Our future work will focus on the automatic analysis of textual descriptions and the development of an interactive process with the user to translate her needs in a formal representation.

RÉFÉRENCES

[1] Avancha, S., Patel, C., Joshi, A. : Ontology-driven adaptive sensor networks. In : Proceedings of the First Annual International Conference on Mobile and Ubiquitous Systems : Networking and Services (MobiQuitous'04. pp. 194–202 (2004)

[2] Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.) : The Description Logic Handbook : Theory, Implementation, and Applications. Cambridge University Press (2003)

[3] Biebow, B., Szulman, S., Paris-nord, U.D. : Terminae : a method and a tool to build a domain ontology (1999)

[4] Bryant, B., Lee, B.S. : Two-level grammar as an object-oriented requirements specification language. In : Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)-Volume 9 - Volume 9 (2002)

[5] Bryant, B.R., Johnson, D., Edupuganty, B. : Formal specification of natural language syntax using two-level grammar. In : Proceedings of the 11th coference on Computational linguistics. pp. 527–532. COLING '86, Association for Computational Linguistics (1986)

[6] Buitelaar, P., Olejnik, D., Sintek, M. : OntoLT : A Protege Plug-In for Ontology Extraction from Text. In : Proceedings of the International Semantic Web Conference (ISWC). Florida, USA (2003)

[7] Buitelaar, P., Olejnik, D., Sintek, M. : A protege plug-in for ontology extraction from text based on linguistic analysis. In : In Proceedings of the 1st European Semantic Web Symposium (ESWS (2004)

[8] Cimiano, P., Vlker, J. : Text2onto - a framework for ontology learning and data-driven change discovery. In : Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB). Lecture Notes in Computer Science, vol. 3513, pp. 227–238. Springer (Jun 2005), http://www.aifb.uni-karlsruhe.de/WBS/jvo/publications/Text2Onto_nldb_2005.pdf

[9] Delahaye, D., tienne, J.F., Vigui Donzeau-Gouge, V.D.G. : Certifying Airport Security Regulations using the Focal Environment. In : Formal Methods (FM). pp. 48–63. Springer (2006)

[10] Fougères, A.J., Trigano, P. : Rédaction de spécifications formelles : élaboration à partir des spécifications écrites en langage naturel. In Cognito - Cahiers Romans de Sciences Cognitives 1(8), 29–36 (1997), http://hal.archives-ouvertes.fr/hal-00569735/PDF/AJF-PhT-Incognito_1997.pdf

[11] Gu, T., Wang, X.H., Pung, H.K., Zhang, D.Q. : An ontology-based context model in intelligent environments. In : In proceedings of communication networkd and distributed systems modeling and simulation conference. pp. 270–275 (2004)

[12] Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosof, B., Dean, M. : Swrl : A semantic web rule language combining owl and ruleml. W3c member submission, World Wide Web Consortium (2004), http://www.w3.org/Submission/SWRL

[13] Horrocks, I., Patel-schneider, P.F., Mcguinness, D.L., Welty, C.A. : Owl : a description logic based ontology language for the semantic web (2007)

[14] Laleau, R., Vignes, S., Ledru, Y., Lemoine, M., Bert, D., ccjuvetu, V., Dubois, C., Peureux, F. : Application of requirements engineering techniques to the analysis of civil aviation security standards. In : SREP'05 International Workshop, In conjunction with 13th IEEE International Requirements Engineering (2005)

[15] O'Connor, M.J., Das, A.K. : Sqwrl : A query language for owl. In : OWL : Experiences and Directions (OWLED), Fifth International Workshop, Chantilly, VA. (2009)

[16] Prevosto, V. : Certified mathematical hierarchies : the focal system. In : Mathematics, Algorithms, Proofs, number 05021 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2005. Internationales Begegnungs- und Forschungszentrum (IBFI), Schloss Dagstuhl (2005)

[17] project, T.E. : http://vasco.imag.fr/EDEMOI/, year = 2003

[18] Russomanno, D.J., K.C.T.O. : Sensor ontologies : from shallow to deep models. Tech. rep. (2005)

[19] Wang, F., Turner, K.J. : An ontology-based actuator discovery and invocation framework in home care systems. In : Proceedings of the 7th International Conference on Smart Homes and Health Telematics : Ambient Assistive Health and Wellness Management in the Heart of the City. pp. 66–73. ICOST '09, Springer-Verlag (2009)