Ontology for a Panoptes Building: Exploiting Contextual Information and a Smart Camera Network

Editor(s): Álvaro Sicilia, Universitat Ramon Llull, Spain; Pieter Pauwels, Universiteit Gent, Belgium; Leandro Madrazo, Universitat Ramon Llull, Spain; María Poveda Villalón, Universidad Politécnica de Madrid, Spain; Jérôme Euzenat, INRIA & Université Grenoble Alpes, France
Solicited review(s): Florian Vandecasteele, Universiteit Gent, Belgium; Two anonymous reviewers

Roberto Marroquin [*], Julien Dubois and Christophe Nicolle
Laboratoire Le2i - FRE 2005, CNRS & Arts et Métiers, Université Bourgogne Franche-Comté, Dijon, France
E-mail: {roberto-enrique.marroquin-cortez, julien.dubois, cnicolle}@u-bourgogne.fr

Abstract. The contextual information in the built environment is highly heterogeneous, it goes from static information (*e.g.,* information about the building structure) to dynamic information (*e.g.,* user's space-time information, sensors detections and events that occurred). This paper proposes to semantically fuse the building's contextual information with extracted data from a smart camera network by using ontologies and semantic web technologies. The developed ontology allows interoperability between the different contextual data and enables, without human interaction, real-time event detections and system reconfiguration to be performed. The use of semantic knowledge in multi-camera monitoring systems guarantees the protection of the user's privacy by not sending nor saving any image, just extracting the knowledge from them. This paper presents a new approach to develop an "all-seeing" smart building, where the global system is the first step to attempt to provide Artificial Intelligence (AI) to a building.

Keywords: BIM, IFC, smart camera network, context-aware system, semantic interoperability, semantic gap, building AI

1. Introduction

In Greek mythology, Argus Panoptes was a giant with a hundred eyes. It was impossible to deceive his vigilance, for only some of his eyes slept while the rest were awake [36]. Argus was the servant of Hera. At his death, Hera rewarded the giant's fidelity by placing his eyes on the feathers of the peacock, her sacred animal. "To have the eyes of Argus" is a popular expression which means to be lucid and vigilant.

The term Panoptes means "all-seeing", which within the built environment is a quest in terms of access control, flow control and activity control. In that context, a Panoptes building would characterize a smart building equipped with a network of cameras which could, in real-time, combine the different information seen and deduce the triggering of actions.

In classical multi-camera based systems (MCBS) there is a monitor room with a central processing server where all the information is collected and analyzed in real-time by one or more human operators.

However, as the size of the network increases, it becomes more difficult (or even impossible) for the human operators to monitor all the video streams at the same time and to identify events. Furthermore, having a large amount of information makes it infeasible to create a relation between past and current actions.

Based on our experience, some issues and limitations of MCBS deployed in built environments have been identified, such as:

[*]Corresponding author. E-mail: roberto-enrique.marroquin-cortez@u-bourgogne.fr.

– Selecting and filtering relevant information from the large amount of generated data.
– Dealing with missing information and non-detectable information. For example, a person may become *not-visible* to a camera due to their position or an obstruction.
– Reconfiguration of the cameras according to their context. The camera should adjust its configuration to ease the task of identifying a specific object or action. For example, the cameras should be able to adjust their parameters (*e.g.,* aperture and shutter speed) according to the light in the environment.
– Integration of data from different nodes and different domains. A MCBS should be able to link the contextual information with the information coming from the different camera nodes to identify events and take decisions.
– Privacy protection. There are many privacy laws that restrict the monitoring of people, therefore, a MCBS should be able to extract the useful information from an image/video while protecting the privacy of the individuals.

Many efforts have been devoted to deal with the aforesaid limitations of the MCBS. The most prominent one is to rely on smart cameras (SCs) to perform visual tasks semi-autonomously (with minimal human interaction).

Smart cameras are specialized cameras that contain not only the image sensor but also a processing unit and some communication interfaces. In a few words, SCs are self-contained vision systems [32,58]. The use of SCs in the built environment has become a growing trend due to the rich contextual data provided.

In the built environment, context is an essential factor since it provides information about the current status of users, places, objects, sensors and events. We assume that a smart building is a context-aware system because it extracts, interprets and uses the contextual information to automatically adapt its functionality according to the contextual changes.

A Panoptes building is a type of smart building that uses only SC sensors and its main task is to monitor the different activities that occur in the built environment; in contrast to the smart building which uses different types of sensors and which mainly focuses on managing/monitoring the energy consumption.

The creation of a Panoptes building is a complicated task due to the integration of data coming from different domains around the knowledge of the building.

Many works have been done using semantic web standards such as Resource Description Framework (RDF) and Web Ontology Language (OWL) to represent contextual data [20]. On those systems, the ontology plays a crucial role in enabling the processing and sharing of information and knowledge, *i.e.,* the use of an ontology allows interoperability between different domains.

This paper presents an ontology for a Panoptes building that re-purposes and integrates information from different domains composing the built context. The proposed ontology is the kernel of the WiseNET (Wise NETwork) system, which is a context-aware system whose main function is to perform reasoning about heterogeneous sources of information [30]. Explicitly, the WiseNET system enhances the information of a smart camera network (SCN) with contextual information to allow autonomously real-time event/anomalies detection and system reconfiguration.

The main contribution of this paper is the semantics-based system, which allow us to overcome the MCBS limitations and some computer vision problems, especially the privacy protection which nowadays is an important factor to consider. This is achieved by the semantic fusion of Industry Foundations Classes (IFC) data with sensor information and other domain information in the Panoptes context.

The rest of the paper is organized as follows. Section 2 introduces the terminology used in the paper as well as summarizes related works. Section 3 gives an overview of the WiseNET system. Section 4 describes the development process of the WiseNET ontology and its links to existing ontologies. Section 5 presents the ontology population from the IFC data. Section 6 presents the ontology population from the SCs which consists of a *static population*, performed during the system configuration, and a *dynamic population* which is performed each time the SCs detect a person. Sections 7 presents some evaluations and use cases and finally, Section 8 presents the conclusions and prospectives.

2. Background and related work

Nowadays, MCBS have become a part of our daily life. They can be found in cities, commercial centers, supermarkets, offices, and even in houses.

The advances in image sensor technology allow us to have SCs, which are low-cost and low-power systems that capture high-level description of a scene and analyze it in real-time [58]. These SCs can extract necessary/pertinent information from different images/video by employing different image processing algorithms such as face detection [55], person detection [10], people tracking [18], fall detection [44], object detection [15], etc.

Smart camera networks have been used in the built environment for a long time. The main applications focus on the following problematics:

- **Study of space-use:** space-use is a research that aims at analyzing the relation between built spaces and their use. An example of this research was presented by Tomé *et al.* which established correlations between the occupancy/movement patterns and the space properties by using computer vision and Radio Frequency Identification (RFID) in a university environment [52].
- **Monitoring of elderly people:** this application focus on health care and ambient assisted living. Some notable works in this field are: Nawaz *et al.* which used SCs to monitor the activities of elderly people while protecting their privacy [34]; and Crispim-Junior *et al.* which developed a multi-camera based framework to recognize the activities of elderly people during a clinical protocol in a hospital [25].
- **Security, monitoring and surveillance:** these are the most well-known applications in the built environment; they consists in using the visual information to monitor the activities of the building users. Yu *et al.* developed a robust image processing algorithm to perform multi-camera multi-object tracking in a built environment [59]; other examples in this field can be found in the survey of Winkler *et al.* [57].

Most of the previous applications use a SCN deployed in a built environment to obtain and analyze different types of information. Therefore, they might be considered as Panoptes building applications.

The main function of a Panoptes building is to combine the different information obtained by the SCN and to deduce the triggering of actions/events in real-time. In this context, a Panoptes building application should understand the static building information as well as perceive (accurately) the dynamic and evolving data, *i.e.,* it should be aware of its context.

The creation of a context-aware system in the built environment is a complex task; it requires information from different domains such as environment data, sensing devices, spatio-temporal facts and details about the different events that may occur.

For example, the required event information could be a set of concepts and relations concerning the different events that may occur in a built environment, their location, the time they occurred, the agents involved, the relation to other events and their consequences. In the case of the sensor information, the required data could be the description of the different sensing devices, the processes implemented on them and their results. Regarding the environment, the required data could be the building topology and the different elements contained in the spaces.

The built environment data can be obtained using the Building Information Modeling (BIM) of a building. BIM becomes a general term designing the set of numerical data, objects and processes appended during the life-cycle of a building [14]. From the design, construction and facility management steps, the BIM allows practitioners and managers to exchange data in a uniform way using the IFC standard [54].

The IFC gives the base of description of all elements making the building, both semantic and graphic [24]. This allows to aggregate all heterogeneous software dedicated to the built environment in an interoperable manner.

In the domain of interoperability three levels are described: technical, organizational and semantics [23]. The IFC aims the technical interoperability level [11]. The organizational level is in charge of the practitioners according to the law of each country and the rules of each enterprise. The semantics level aims to clearly specify the meaning of each element making the BIM.

An important work was made to bring the IFC EXPRESS schema into the semantic web world using OWL as the schema modeling language [39]. The result is the `ifcowl` ontology whose main objective is to convert the IFC concepts and instances data into equivalent RDF data. The conversion procedure from EXPRESS to OWL can be found in [39,40].

According to Studer, an ontology is a formal, explicit specification of a shared conceptualization [47]. In other words, an ontology is a set of concepts and relations used to describe and represent an area of concern. Currently, the most common language for representing ontologies is OWL-2, which is the recommendation of the World Wide Web Consortium (W3C) [56].

Other recommended technologies/languages in the semantic web domain are: RDF, used for representing information in the form of a graph composed of triples

[9]; RDF Schema (RDFS), which provides a vocabulary for creating a hierarchy of classes and properties [3]; SPARQL[1], used to query RDF data [51]; and the Semantic Web Rule Language (SWRL[2]), which is used for extending the OWL model with rule axioms [22].

One important application of ontology is *semantic fusion*, which consists in integrating and organizing data and knowledge coming from multiple heterogeneous sources and to unify them into a consistent representation. Some important works in this domain are:

- Hong *et al.* presented some context-aware systems where the ontology plays a central role for enabling interoperability between devices and agents which are not designed to work together [20].
- Dibley *et al.* developed an ontology framework that combines a sensor ontology with a building ontology and other supporting ontologies [12].
- SanMiguel *et al.* used an ontology for combining image processing algorithms with knowledge about objects and events [43].
- Chaochaisit *et al.* presented a semantic connection between sensor specification, localization methods and contextual information [6].
- Town presented an ontology that fuses multiple computer vision stages with context information for image retrieval and event detections [53].
- Suchan and Bhatt developed a framework to reasoning about human activities using commonsense knowledge founded in qualitative spatio-temporal relations and human-object interactions [48].

Based on the state of the art, the information extracted by a SCN could be converted to semantic data. Moreover, a specially designed ontology would enable to deal and merge multi-sources of heterogeneous data (*i.e.,* from different cameras, other types of sensors, *a priori* data from the environment). Hence, we propose the creation of a semantic based system which is a context-aware system that uses an ontology to combine the information extracted by a SCN with logic rules and knowledge of what the camera observes, building information and events that may occur.

[1]SPARQL is a recursive acronym for SPARQL Protocol and RDF Query Language.

[2]Currently (February, 2017) SWRL is not a W3C recommendation yet.

3. WiseNET system

We developed a new framework called WiseNET, which is a semantics-based system that fuses heterogeneous sources of data such as data coming from sensors and the different contextual information. Due to the application of the paper (Panoptes building), we focus on a specific type of sensor: SCs; however, the system is defined to include other type of sensors such as temperature, humidity, depth sensor, etc.

The main goal of WiseNET is to improve classical computer vision and deep learning systems by considering the contextual information of the environment and by performing real-time reasoning. As a result, WiseNET may overcome some limitations of computer vision (*e.g.,* false detections and missed detections), some drawbacks of deep learning (*e.g.,* the need of a large amount of training and testing data) and limitations of MCBS (presented in Section 1) while allowing real-time event/anomalies detection and system reconfiguration.

Figure 1 illustrates an overview of the WiseNET architecture. The system is articulated in three sections: the smart camera network, the monitor unit and the central unit.

3.1. Smart camera network

The SCN is a set of smart cameras distributed in an environment. The main functions of the SCN, in the WiseNET system, is to extract low-level features from a scene (such as person detection as shown in Figure 1), to convert the extracted data into knowledge and to send it to the central unit. More information regarding the type of smart cameras used can be found in [30].

3.2. Monitor unit

The monitor unit's main function is the visualization of the static and dynamic information; this unit automatically retrieves information and presents it in a graphical manner, for example an occupancy map (as shown in Figure 1) or a heat map (as shown in Figure 10). Also, the monitor unit implements some queries to answer questions such as: how many people are present in a room? what is the location of a person? and many others (see Section 4.1).

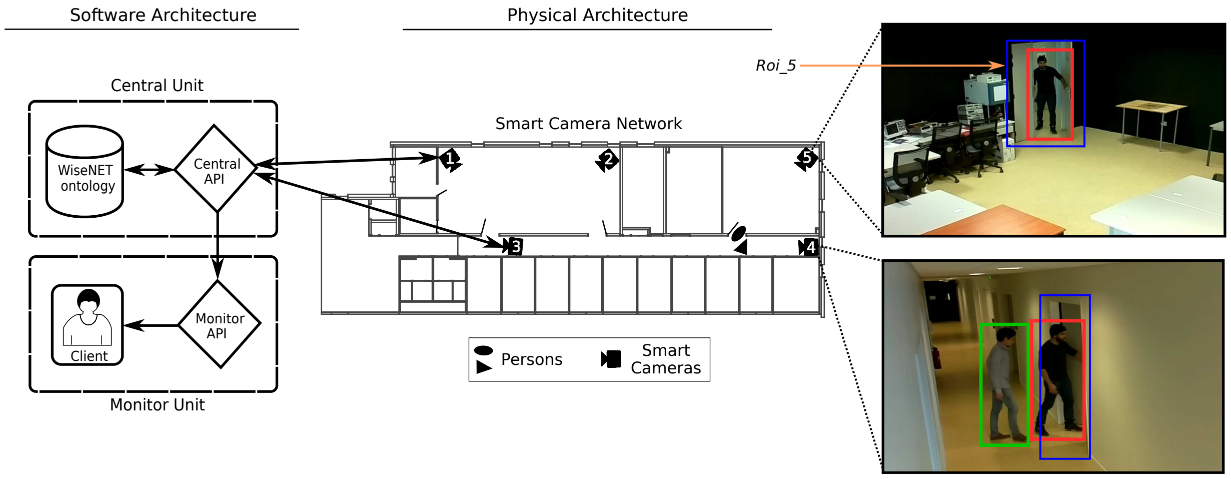


Fig. 1. WiseNET system overview: the smart camera network is in charge of extracting pertinent information from a scene, to convert the data into knowledge and to send it to the central API, which afterwards populates the WiseNET ontology with that knowledge. The central API is also responsible for reconfiguring the image processing algorithms in the smart cameras and for transferring data to the monitor unit from which a client/user may visualize the history of activities. Notice that all the smart cameras are connected to the central API, however to keep the image cleaner, only the connection of two smart cameras are shown. The color code of the bounding boxes in the camera images (right side) are: blue represent regions of interest (*i.e.,* `Roi_5`), in this case a door, green represent a detection and red represent a detections around a region of interest.

3.3. Central unit

The central unit is composed of two elements the central API and the WiseNET ontology:

- **The central API:** is in charge of the management of the ontology, for example: capturing the knowledge coming from the SCN and inserting it into the ontology, retrieving inferred knowledge from the ontology, transferring data to the monitor unit, sending new configurations to the smart cameras, and other services.
- **The WiseNET ontology:** is responsible for enabling interoperability between the incoming knowledge streams and the contextual data (*e.g.,* environment information, previous knowledge and sensor information) in order to deduce new knowledge and detect events/anomalies based on the history of activities.

Our system differs from other computer vision systems mainly by four factors. Firstly, rather than improving algorithms [59] or processing signals [44] it aims to give a meaning to the information extracted by using computer vision techniques [53]. Secondly, no images are sent, the SCs only send the extracted knowledge. Thirdly, the WiseNET system combines context information with the camera information to overcome missed detections or non-detectable information (*e.g.,* people outside a camera's field of view). Finally, the system uses an ontology to fuse the different kinds of information presented in a Panoptes building, such as: information of the environment, time, SCs, events, detectable objects, etc.

The focus of this paper is the description of the central unit. In sections 4 and 5, the WiseNET ontology development process will be shown as well as its population with static data. In section 6, the central API will be presented, specifically the process of dynamically inserting data into the ontology.

4. Formal modeling

The WiseNET ontology is the kernel of the WiseNET system. It is defined in OWL-2 and it incorporates a vast corpus of concepts in the domain of a Panoptes building. The ontology provides a vocabulary for combining, analysing and re-purposing the information coming from the SCN deployed in a built environment. The main function of the WiseNET ontology is to perform real-time event/anomalies detection and initiate system reconfiguration.

4.1. Ontology development

The ontology development process followed was the Noy and McGuinness methodology [35]. This methodology consists of seven steps which are: determine the scope of the ontology, consider reuse, enumerate classes, define classes and properties, define constrains and create instances (ontology population shown on Sections 5 and 6).

4.1.1. Scope of the ontology

The ontology scope was determined by thinking about the kind of knowledge that should be covered by the ontology and its use, *i.e.,* its domain.

Table 1 presents some competency questions that helped to determine the focus of the ontology. These questions should be answered by the ontology eventually providing the different kinds of knowledge that should be contained in the WiseNET ontology. Roughly, it is knowledge about the environment, events, people, sensors and time.

4.1.2. Links to existing ontologies

When developing a new ontology it is recommended to reuse existing ontologies as much as possible. Thus, one can focus on defining the specific knowledge of the application. The reuse of external ontologies not only saves time but also gives the advantage of using mature and proved ontological resources that have been validated by their applications and (some) by the W3C.

The WiseNET ontology reuses resources from many different ontologies (see Table 2). However, there are six key ontologies that cover most of the required concepts of the different domains, these are:

- The `DUL` ontology, which provides a set of concepts used for interoperability between different ontologies [16]. This ontology gives the necessary properties to combine spatial information with different types of data.
- The `event` ontology, which deals with the notion of events and their different properties such as location, time, agents, factors and products [42]. This ontology provides most of the vocabulary required for describing activities and events that may happen.
- The `ifcowl` ontology, which is a semantic representation of the IFC schema (standard for representing building and construction data) [39,40]. Most of the environment concepts required (such as the structure of the building, its topology and the different elements contained in a space) can be obtained from the IFC. The IFC data used during the experimentations was defined using the IFC2x3 specification, therefore we focused on this version of the `ifcowl`. However, the system should be able to cope with newer versions of the IFC.

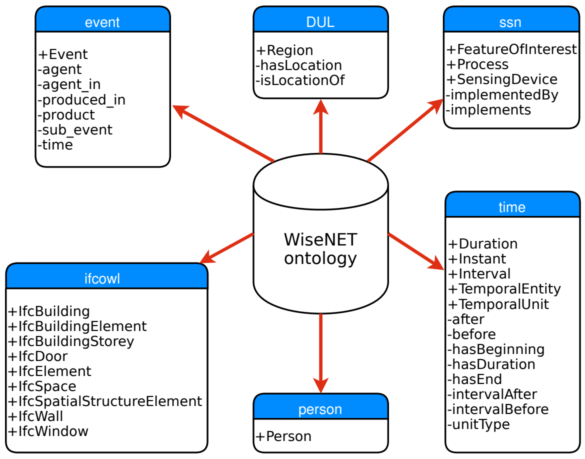


Fig. 2. Primary classes and properties reused by the WiseNET ontology. Classes are marked with (+) and properties with (-). The namespaces and brief description of the ontologies can be found in Table 2.

- The `person` ontology, which provides the minimum set of classes and properties for describing a natural person [41].
- The `ssn` ontology, which describes sensors, observations, sensing processes, measurement capabilities and related concepts [7]. This ontology provides the required vocabulary to describe the sensors used in our system.
- The `time` ontology, which provides concepts for describing the temporal properties of resources [8]. This ontology provides all the required concepts about instants, intervals, their duration and their topological relations.

Figure 2 shows the primary classes and properties reused by the WiseNET ontology. The external ontologies were not imported, from most of them only a small portion was reused. Not importing the external ontologies results in ease of ontology maintenance and performance improvement, which is a critical factor since our goal is to obtain real-time reasoning.

4.1.3. Classes and properties of WiseNET

Many of the competency questions involve more than one type of knowledge. Hence, the WiseNET ontology should be able to collect and combine the information from the different domains. In that context, it is necessary to define new concepts (classes and properties) that allow us to complete the information from the different domains, to describe attributes of instances

Table 1

Selected competency questions used for developing the WiseNET ontology

Competency questions	
How many people are/were in a space?	What is the most visited space in the building?
Where is a person located?	What and where are the nearest sensors of sensor X?
What is the position of all the people?	Which building elements does a camera observes?
Where was a person in the last T minutes?	Which is the most used door in the building?
How many rooms does a storey has?	Is somebody in a restricted area?
Is a space empty/occupied?	At what time does a person entered/left a space?
Is there a person not visible (occluded)?	How long does a person stayed in a space?
What are all the spaces connected to space X?	Where were all the people at time T?
Which types of sensors are in the building?	When and where does an event occurred?
What is the position of all the sensors?	Which person was involved in an event?
From which door a person entered/left a space?	Which events happened in the last T minutes?
Where does a person stayed the longest time?	When and where does person X met person X2?
At what time there are more people in a space?	How many people entered/left a space in a day?

X, X1 are id's and T is a variable.

Table 2

Full list of prefixes and namespaces used in WiseNET ontology and in this document

Prefix	Namespaces	Description
`DUL`	http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#	DOLCE+DnS ultralite ontology
`event`	http://purl.org/NET/c4dm/event.owl#	The event ontology
`ifcowl`	http://ifcowl.openbimstandards.org/IFC2X3_TC1#	The IFC2X3 ontology
`owl`	http://www.w3.org/2002/07/owl#	The OWL 2 schema vocabulary
`person`	http://www.w3.org/ns/person#	ISA Person core vocabulary
`rdf`	http://www.w3.org/1999/02/22-rdf-syntax-ns#	The RDF concepts vocabulary
`rdfs`	http://www.w3.org/2000/01/rdf-schema#	The RDF schema vocabulary
`ssn`	http://purl.oclc.org/NET/ssnx/ssn#	Semantic sensor network ontology
`time`	http://www.w3.org/2006/time#	OWL-Time ontology
`wisenet`	http://wisenet.checksem.fr/#	The WiseNET ontology
`wni`	http://wisenet.checksem.fr/inst#	WiseNET instances
`xml`	http://www.w3.org/XML/1998/namespace	XML specification
`xsd`	http://www.w3.org/2001/XMLSchema#	XML schema definition

according to our needs and, more importantly, to relate (*i.e.,* link) the different domains [3].

After having the complete terminology of the ontology, some constraints and characteristics of the class expressions and the property axioms need to be defined. Axioms are a set of formulas taken to be true and which every assignment of values must satisfy. Those constraints and characteristics determine the expressiveness and decidability of the ontology, and their definition depends on the description logic used.

4.2. Ontology decidability

Description logics (DLs) are a family of formalism used for representing knowledge [1]. The most notable applications for the DLs is to provide the logical formalism for ontologies languages such as OWL. OWL-2, the current W3C ontology language recommendation, is based on the expressive description logic $\mathcal{SROIQ}(\mathcal{D})$ [13].

$\mathcal{SROIQ}(\mathcal{D})$ provides high expressive power with high computational cost of reasoning. Hence, to meet

[3]The complete set of classes and properties of the WiseNET ontology can be found at `http://wisenet.checksem.fr/#/ontology`

Listing 1: SWRL rule for `spaceConnectedTo`.

```
Space(?x), Space(?y),
Door(?d), spaceContains(?x,?d),
spaceContains(?y,?d) -> spaceConnectedTo(?x,?y)
```

Listing 2: SWRL rule for `hasNearbySensor`.

```
Space(?x), Space(?y),
SmartCamera(?s1), SmartCamera(?s2),
spaceConnectedTo(?x,?y), hasLocation(?s1,?x),
hasLocation(?s2,?y) -> hasNearbySensor(?s1,?s2)
```

a more suitable compromise between the expressive power and the computational cost, the WiseNET ontology was defined using the $\mathcal{SHOIQ}(\mathcal{D})$ language [27]. A definition of the $\mathcal{SHOIQ}(\mathcal{D})$ constructors and some examples referencing the WiseNET ontology can be found in the Table 3.

Horrocks and Sattler presented a tableau decision procedure for $\mathcal{SHOIQ}(\mathcal{D})$ that solves the ontology consistency problem and allows the use of reasoning services, thus demonstrating the decidability of $\mathcal{SHOIQ}(\mathcal{D})$ [21].

However, knowledge representation formalisms of the semantic web (such as DLs) have expressive limitations, for example composition of complex classes from classes and properties. Those limitations can be overcome by rule-based knowledge, specifically by using SWRL (Semantic Web Rule Language) rules [22]. SWRL rules are represented as implication of an antecedent (Body) and a consequent (Head):

$$b_1, b_2, ..., b_n \rightarrow h \;,$$
$$\text{where } b_1, b_2, ..., b_n : \text{Body and } h : \text{Head.}$$

Reasoning becomes undecidable for the combination of OWL + SWRL, therefore the expressivity of SWRL needs to be reduced in order to assure decidability. Although, many procedures exists to guarantee decidability of SWRL, the DL-safe rules were adapted [33]. This procedure consists in restricting the number of possible variables assignments, *i.e.,* restricting the application of rules only to known OWL individuals (named individuals).

Examples of DL-safe rules implemented in the WiseNET ontology are presented in Listing 1 and Listing 2. The first one states that if there are two spaces 'x' and 'y', and both contain the door 'd', then those spaces are connected to each other. The second one states that if there are two spaces 'x' and 'y', and two SCs 's1' and 's2', and 'x' is connected to 'y', and 's1' is located in 'x' and 's2' is located in 'y', then those SCs are nearby each other.

Notice that the property `spaceConnectedTo` could be also obtained from queries but it is quite complex, therefore it was decided to formulate a rule to define it.

Once the ontology is formally defined and implemented in a triplestore (a database for the storage of triples), the last step of the ontology development is the insertion of instances (population). The next two sections will present the population from an IFC file and the SC information respectively.

5. Ontology population from IFC

After inserting the WiseNET ontology in the system, the *a priori* information about the built environment needs to be populated. The required information can be extracted from the IFC file of the environment. This population is performed only once, at the initialization of the system, and is thus considered as a *static population*. Notice that if there is a reconfiguration of the building, the static population should be re-performed using the new IFC file.

The I3M (Institut Marey et Maison de la Métallurgie) building located in Dijon (France), is used as an example for this section. The I3M building has three storeys, from which we will focus on the third storey where a SCN has been deployed.

An IFC2x3 file, describing all the elements composing the I3M building, was obtained from the company in charge of the construction of this building and it was generated using the Revit CAD software[4].

Only a small portion of the IFC file is needed in the WiseNET system. Therefore, to improve the ontology performance (and reduce its complexity), only the required information is extracted from the IFC file and populated in the ontology.

Figure 3 shows, in the form of a graph, the main classes and properties required to be extracted from an IFC file and populated on the ontology. The extracted/populated data consists of information about the building, building storeys, spaces, elements contained

[4]http://www.autodesk.com/products/revit-family/overview

Table 3

Definition of $\mathcal{SHOIQ}$ $(\mathcal{D})$ constructors

$\mathcal{SHOIQ}$ $(\mathcal{D})$ constructor	Definition	Examples in WiseNET ontology using Turtle syntax [9]
$\mathcal{S}$	Transitivity of roles	`:isPartOf rdf:type owl:TransitiveProperty.`
$\mathcal{H}$	Role hierarchies	`:aggregates rdfs:SubPropertyOf :environmentProperties.`
$\mathcal{O}$	Nominals	`_:x rdf:type owl:AllDifferent;` `owl:distictMembers(:PersonDetector,:FaceDetector).`
$\mathcal{I}$	Inverse role	`ssn:implements owl:inverse ssn:implementedBy.`
$\mathcal{Q}$	Qualified number restrictions	`_:x rdf:type owl:Restriction;` `owl:onProperty :spaceContains;` `owl:minQualifiedCardinality "1"^^xsd:integer;` `owl:onClass :SpaceConnector.`
$(\mathcal{D})$	Datatypes	`:xywh rdf:type rdfs:Datatype.`

`_:x` represents a blank node (anonymous individual).

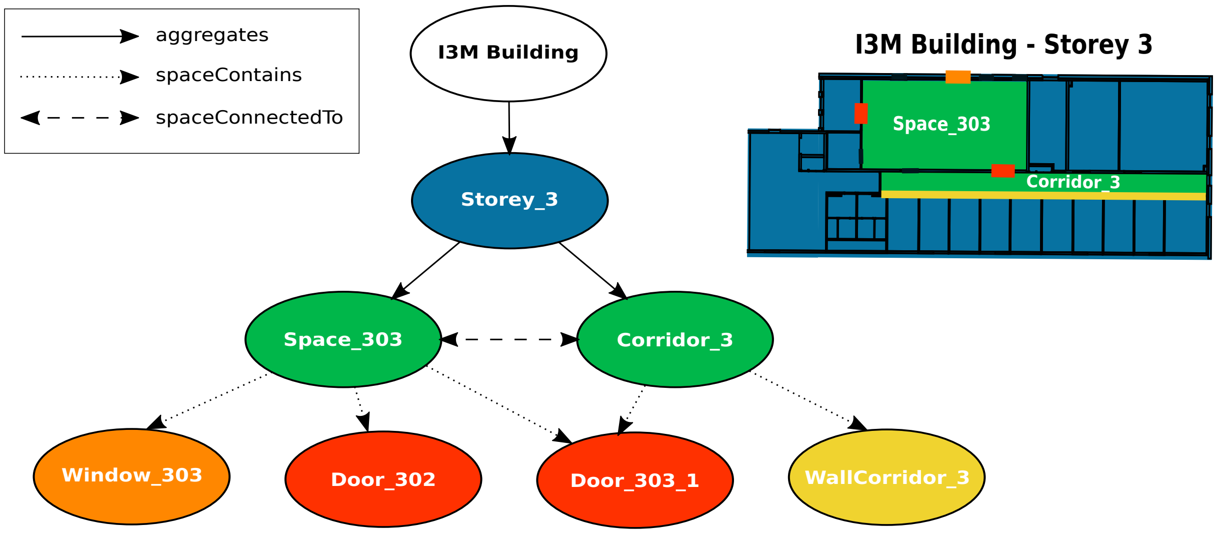


Fig. 3. A fragment of the graph showing the main classes and properties required to be extracted from the IFC file. On the right side a 2D view of the third storey of the I3M building is shown. The graph represents a small selection of spaces and elements present on the third floor. The colors of the graph nodes have two functions, first to make a correlation between the graph and the 2D view, second, to denote different IFC classes: the `Storey_3` belongs to the class `IfcBuildingStorey`; the `Space_303` and the `Corridor_3` belongs to the class `IfcSpace`; the `Door_302`, the `Window_303` and the `Wall_Corridor_3`, belongs to the classes `IfcDoor`, `IfcWindow` and `IfcWall` respectively.

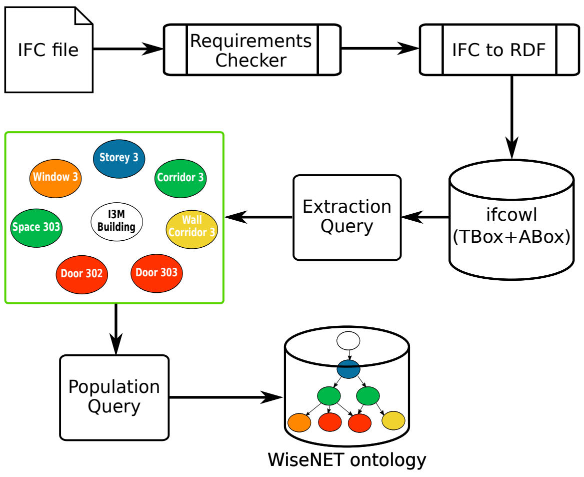


Fig. 4. Extraction and population framework.

in those spaces and the relation between the different concepts (*e.g.,* the building topology).

A framework was developed for extracting and populating the required IFC data into the WiseNET ontology (see Figure 4). The framework employs semantic web technologies and it consists mainly of four processes: a requirements checker of the IFC file, a conversion of the IFC into RDF, the extraction of the pertinent instances and finally, the population of the extracted instances and their relating properties into the WiseNET ontology.

The framework starts with an IFC file of a built environment, in this case the I3M building. To make use of the proposed framework the IFC file must contain instances of the following entities: `IfcBuilding`, `IfcSpace`, `IfcDoor`, `IfcWall`, `IfcWindow`, `IfcRelSpaceBoundary`, `IfcRelAggregates` and `IfcBuildingStorey`. The *requirements checker* process is in charge of verifying the instantiation of those classes.

Afterwards, the IFC file is converted to RDF by using the *IFC-to-RDF converter* by Pauwels and Oraskari [37]. The result of the conversion is the `ifcowl` ontology which consists of a semantic representation of the IFC2x3 schema (TBox) with the instances of the I3M building (ABox).

5.1. Extraction query

The ABox of the `ifcowl` is queried using SPARQL to extract the pertinent instances. The Listing 3 shows the SPARQL code used for the extraction, where:

- Line 4 obtains the building instance by using its class.
- Line 7 acquires the array of building storeys that decompose the building; and line 8 obtains the storeys inside that array.
- The same is done for the spaces that decompose the storeys on lines 11-12.
- Lines 15-16 obtain the elements that are contained in a space.
- Lines 19-22 filter out the undesired elements just leaving the doors, windows and walls.
- Line 25 gets the classes of all the elements and line 26 filters out the binding to `owl:Thing` (binding that is deduced if the query is executed with reasoning).

The result of the query is the *extracted table*, where its columns corresponds to the variables used with the SELECT operator (line 1).

In `ifcowl`, the relations `IfcRelAggregates` and `IfcRelContainedInSpatialStructure` are defined as intermediate classes instances. However, the use of intermediate instances for defining relations are often unneeded and their presence in the RDF graph raises its complexity unnecessarily [38]. Therefore, to simplify the extracted graph and make it more intuitive, those relations were omitted. Moreover, during the population query, direct relations between the different concepts are created by using the WiseNET properties `aggregates` and `spaceContains`.

5.2. Population query

The population query consists in creating the relations between the extracted instances. To accomplish this, the *extracted table* is processed row by row.

For exemplification, let us assume that the first row of the *extracted table* has the following values:

```
?building = inst:Building_I3M,
?storey = inst:Storey_3,
?space = inst:Space_303,
?element = inst:Door303_1,
?elementType = ifcowl:IfcDoor,
```

where `inst` is a prefix related to the IFC to RDF conversion and is used for defining the ABox in the `ifcowl` ontology.

Now, those instances are the input of the population query shown in the Listing 4, where:

- Line 1 inserts the defined triples into the WiseNET ontology.
- Lines 3-5 relate the extracted instances of the `ifcowl` with the WiseNET properties.

Listing 3: SPARQL query for extracting instances from the `ifcowl` ontology. Lines that start with # are comments.

```
SELECT ?building ?storey ?space ?element ?elementType
WHERE {
   # Get building
   ?building rdf:type ifcowl:IfcBuilding.

   # Get building storeys
   ?storey_array ifcowl:relatingObject_IfcRelDecomposes ?building;
                 ifcowl:relatedObjects_IfcRelDecomposes ?storey.

   # Get spaces: room, corridors, hall, etc
   ?space_array ifcowl:relatingObject_IfcRelDecomposes ?storey;
                ifcowl:relatedObjects_IfcRelDecomposes ?space.

   # Get elements: doors, windows, walls, floor, furnitures, etc
   ?element_array ifcowl:relatingSpace_IfcRelSpaceBoundary ?space;
                  ifcowl:relatedBuildingElement_IfcRelSpaceBoundary ?element.

   # Filter elements to just keep doors, walls and windows
   {?element rdf:type ifcowl:IfcDoor}
   UNION {?element rdf:type ifcowl:IfcWall}
   UNION {?element rdf:type ifcowl:IfcWallStandardCase}
   UNION {?element rdf:type ifcowl:IfcWindow}.

   # Get the classes of elements
   ?element rdf:type ?elementType.
   FILTER (?elementType != owl:Thing)
}
```

Listing 4: SPARQL query for inserting relations on the WiseNET ontology. Prefixes `in:` and `wnt:` refer to `inst:` and `wisenet:` respectively. Lines that start with # are comments.

```
1  INSERT DATA {
2   # Creating relations
3   in:Building_I3M wnt:aggregates in:Storey_3.
4   in:Storey_3 wnt:aggregates in:Space_303.
5   in:Space_303 wnt:spaceContains in:Door_303_1.
6
7   # Inserting the types
8   in:Building_I3M rdf:type ifcowl:IfcBuilding.
9   in:Storey_3 rdf:type ifcowl:IfcBuildingStorey.
10  in:Space_303 rdf:type ifcowl:IfcSpace.
11  in:Door_303_1 rdf:type ifcowl:IfcDoor.
12 }
```

– Line 8-11 states the class of each instance.

This process needs to be repeated for all the rows of the *extracted table*, which is achieved by using an external loop.

To summarize, in order to populate the WiseNET ontology with the *a priori* environment knowledge, the relevant information from the IFC file needs to be extracted, then shared and inserted into the WiseNET ontology using queries, rules and linked data techniques such as Uniform Resource Identifiers (URIs) and RDF.

Linked data technology connects data from one data-source to other data-sources, in our case, linked data allows the WiseNET ontology to obtain extra information from the `ifcowl` if required (*e.g.,* the dimensions of a door, its material and the dimensions of a wall).

5.3. IFC extension

The IFC data could be enhanced to allow the deduction of security restrictions and the design of rules regarding the space usage. Particularly, we propose to extend the IFC by adding extra information concerning:

– **Space functionality**: spaces have many usages, such as: corridor, office, co-working room, lobby,

kitchen and infirmary room. The space functionality can be added in the WiseNET ontology as a subclass of the `IfcSpace` class. By considering the space functionality it is possible to design specific rules according to each function. For example, knowledge that people should not loiter in a particular corridor could be inserted into the ontology in form of a rule. Another example could be the knowledge that in the infirmary room it is normal to have people laying down.
– **Space alarm property**: spaces could have different types of alarm, such as: fire alarm, siren and light alarm. We consider that a smart building should know the information about which spaces have alarms, the type and what triggers them. All those types of alarms and their triggers can be added in the WiseNET ontology by using the properties `spaceHasAlarm` and `hasReaction`, respectively.
– **Security system property**: a door might have different types of security systems, such as: keylock system, card reader, keypad and biometric systems. This information could be used to know the level of restriction of a space. The door security systems can be added in the WiseNET ontology by using the property `hasSecuritySystem`.

The IFC extension is performed during the *System Configuration* (see Section 6.1).

6. Ontology population from smart cameras

The built information has already been added to the WiseNET ontology, the next step is to populate the information about the sensors.

A complete SCN has been installed in the third storey of the I3M building[5]. The SCs are based on the Raspberry Pi 3 system [30].

There are two types of information that need to be populated concerning the SCs. Firstly, the SC setup information, which consists in describing the SCs and their relation to the built environment. Secondly, the information about detections which occurs each time the SCs perform a detection.

The first one is inserted once, during the system configuration, therefore it is considered as a *static population*. The second one will be inserted each time there is a detection, therefore it is considered as a *dynamic population*.

6.1. System Configuration

Figure 5 presents the *System Configuration* interface used for adding and setting up SCs in the system and to extend the IFC information. The *System Configuration* helps specifically to perform the following tasks:

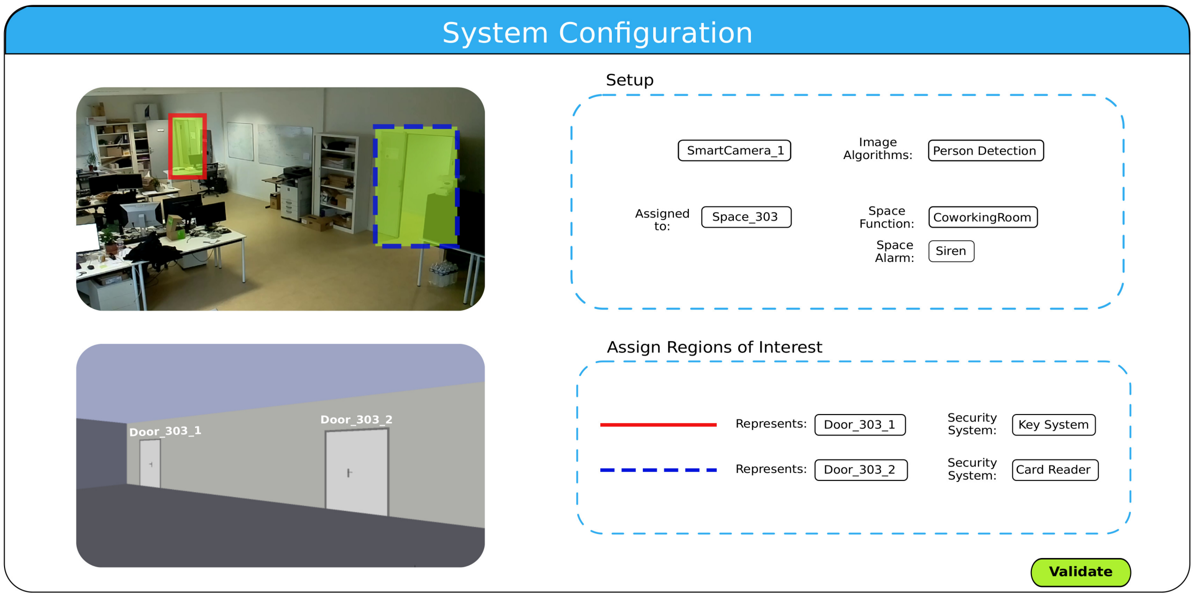
– **Assigning a SC to a space** The system automatically proposes a set of spaces to attach the SCs to. Those spaces are obtained by querying the `ifcowl` ABox.
– **Giving a semantic meaning to image regions** There is a semantic gap between the visual information of an image and its physical representation. Consequently, the interface allows the camera image to be labelled manually, by drawing regions of interest (ROIs) and assigning their representation in the built environment.
The system automatically proposes a set of elements for representation (*e.g.,* doors and windows) according to the selected space. This information is obtained by querying the `ifcowl` ABox.
– **Assigning image processing** An SC could implement several image processing algorithms. Based on our application, the following algorithms were implemented: motion detection [26, 61], face detection [55], fall detection [44] and person detection [10], with person detection being the default algorithm[6].
– **Extend IFC** The IFC could be enhanced by adding functional facts to the spaces, information about the presence and the type of alarm and information about the security systems of a door.

The *System Configuration* software is connected to a SPARQL endpoint that inserts the information set by the user.

This step can be seen as a soft camera calibration that requires only the knowledge of the location of the cameras in the building. This differs from many MCBS, that require overlapping between the fields of views of the cameras and their orientation, leading to a time-consuming and skill-dependent calibration process [46].

[5]A demonstration of the deployed network can be found at `http://wisenet.checksem.fr/#/demo`

[6]The details of the image processing algorithms are outside the scope of this paper and they will be presented in a future work.



Setup

SmartCamera_1
Image Algorithms: Person Detection
Assigned to: Space_303
Space Function: CoworkingRoom
Space Alarm: Siren

Assign Regions of Interest

🟥 Represents: Door_303_1 Security System: Key System
🟦 Represents: Door_303_2 Security System: Card Reader

Fig. 5. Graphic User Interface (GUI) of the System Configuration software. In this example the `SmartCamera_1` is being configured to perform `Person Detection` algorithm and is being assigned to the `Space_303`. Additionally, the `Space_303` is being defined as a `CoworkingRoom` with an alarm of type `Siren`. Furthermore, the blue region of interest (located in the top-left image) is assigned to represent the `Door_303_2` which has as security system *Card Reader*, similar for the red region of interest. The 3D view (bottom-left) was obtained from the IFC file. The system makes automatic suggestions according to the spaces and their elements.

An important contribution of using an ontology during the SC setup is the automatic suggestion of pertinent elements according to the space. This is achieved by using the *static population* of the building information.

6.2. *Smart camera dynamic population*

The dynamic population consists of two phases, first the knowledge extraction performed by the SCs and then the knowledge processing performed by the central API.

6.2.1. *Knowledge extraction*

The main functions of the SCs are: to detect pertinent information using different image processing algorithms, to extract the knowledge from it and to send it to the central API.

For our Panoptes building application the main detectable objects are people. After detecting some pertinent information, the SC describes what it observes by using the vocabulary defined in the WiseNET ontology, *i.e.,* extract the knowledge of the scene. This addition of semantic meaning to what the camera observes is a problem known as `semantic gap` [53].

Finally, an SC-message containing the scene description is created using the JSON syntax (a lightweight format to serialize structure data[7]), where all the fields correspond to terms defined in the WiseNET ontology. This SC-message is then sent to the central API by using web services.

For instance, consider the scene observed by the `SmartCamera_4` where two people are being detected (bottom-right image in the Figure 1). The `SmartCamera_4` extracts the knowledge of the scene and sends the SC-message shown in Listing 5, where each field correspond to:

- `SmartCamera`: SC instance that sends the message.
- `inXSDDateTime`: synchronized timestamp, shared by all the SCs in the network. This was obtained by implementing a NTP (Network Time Protocol) server [31,45].
- `detections`: array of detections observed in a time instance.
- `ImageAlgorithm`: computer vision algorithm used to perform the detections.
- `RegionOfInterest`: states if the detection was made around an ROI, in this case, one de-

[7]JSON (JavaScript Object Notation) https://www.w3schools.com/js/js_json_syntax.asp

Listing 5: Example of a message sent by the smart camera to the central API. Notice that the message is composed of two detections.

```
{
  "SmartCamera": "SmartCamera_4",
  "inXSDDateTime": "2017-04-17T10:18:45.922Z",
  "detections": [
    {
        "ImageAlgorithm": "PersonDetection",
        "RegionOfInterest": "null",
        "xywh": [82,22,24,52],
        "visualDescriptors": [0.0,0.1,...,0.7]
    },
    {
      "ImageAlgorithm": "PersonDetection",
      "RegionOfInterest": "Roi_5",
      "xywh": [107,20,30,50],
      "visualDescriptors": [0.2,0.4,...,0.12]
    }
  ]
}
```

tection (red bounding box in Figure 1) was made around a door (`Roi_5`).

- `xywh`: coordinates of the detection box (green and red bounding box in Figure 1). These coordinates will be used in the monitor API for projecting the detections in the building map.
- `visualDescriptors`: array of visual features used for describing a detection. This array was obtained by firstly removing some noise in the detections and then a normalized 2D HSV histogram was computed [49][8]. The color space HSV (Hue-Saturation-Value) was chosen instead of standard RGB (Red-Green-Blue), due to its fast computation time and mainly due to its robustness to lighting changes [49,60]

It is important to remark that the choice of visual descriptors may vary according to the application and resources, therefore it is possible to use different color spaces, physical characteristics (*e.g.,* height, head size and shoulder width) [4], or robust feature descriptors such as Histogram of Oriented Gradients (HOG) [10], Scale-Invariant Feature Transform (SIFT) [29] or Speeded-Up Robust Features (SURF)[2].

[8]The details of the process to obtain the array of visual descriptors is beyond the scope of this paper and will be presented in a future work.

6.2.2. Knowledge processing

Once the central API receives the SC-message, it creates and inserts the `Detection` and `PersonInSpace` semantic network shown in Figure 6.

A `Detection` is a type of event that occurs in a specific point in time/space. A `PersonInSpace` event (`PIS-E`) is a container of `Detections` relating a specific `Person` with a specific `Space` during a period of time. While the `Person` is in the `Space` the `PIS-E` is active and `Detections` can be attached to it. When the `Person` leaves the `Space` the `PIS-E` is closed and no more detections can be attached to it. Two or more `PIS-Es` are related if they involve the same `Person` instance.

To achieve the semantic network creation-insertion, the central API performs the process presented in Figure 7. The process starts by receiving a SC-message, then the `Space` where the camera is located is obtained by using the property `DUL:hasLocation`. Then, the `Time-Instant` and the `BoundingBox` instances are created. Afterwards, the central API checks if the detection belongs to an existing `Person` in the current `Space` (point A in Figure 7). This is done by:

1. Getting all `Person` instances that are in the detection's location and that have an active `PIS-E`. This is achieved by executing the query shown in Listing 6, where:
   - Line 4 gets the space where the SC is located.
   - Lines 7-10 get all the active `PIS-E` in that space and the person involved.
   - Line 13 gets the visual descriptor array of the person.
2. Comparing the visual descriptors of the selected `Person` instances with the detection's visual descriptor (the one contained in the SC-message). The Chebyshev distance was chosen to measure the similarity between arrays.

The Chebyshev distance is defined as

$$d_\infty(x,y) = \max_{1 \leq i \leq n} |x_i - y_i| \,,$$

where x and y are the arrays of visual descriptors and n is the number of features in each array. From its definition it can be stated that the distance between two arrays is the greatest difference along any dimension [5].

If the Chebyshev distance between the detection's visual descriptor and the visual descriptor of an exist-

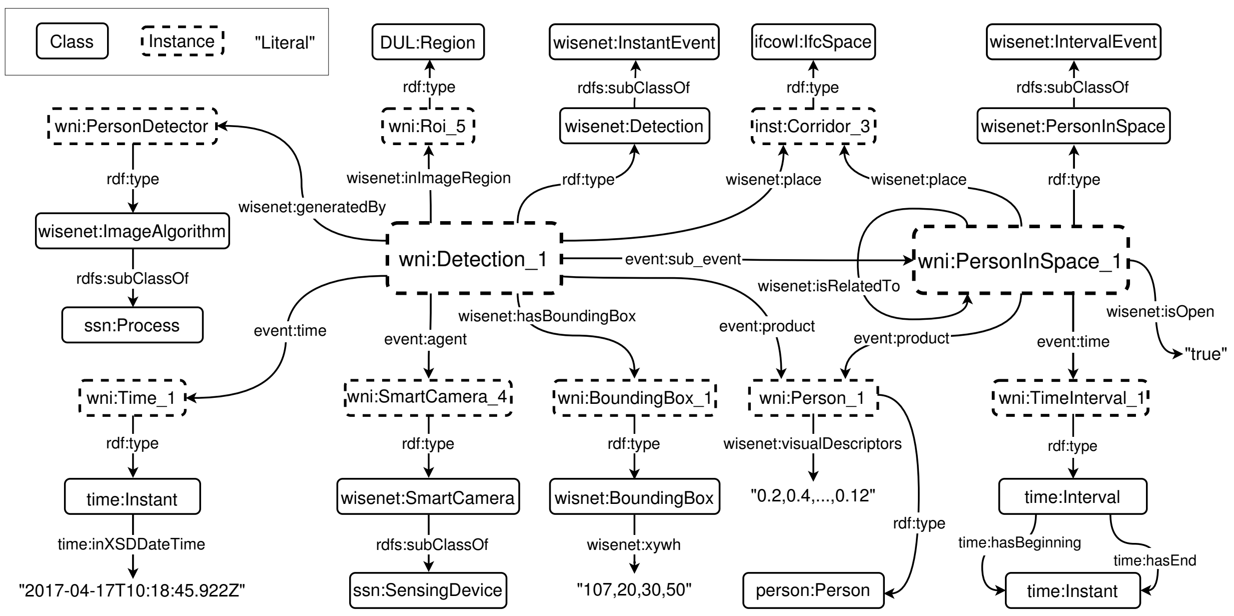


Fig. 6. Semantic network of *Detection* and *PersonInSpace* events, using as example the instances in Listing 5.

Listing 6: Query to get `Person` instances that have an active `PIS-E` in a specific space. Using as example instances of Listing 5. Lines that start with # are comments.

```
SELECT ?person ?descriptors ?event
WHERE {
 # Get Space
 wni:SmartCamera_4 DUL:hasLocation ?space.

 # Only consider active PIS
    ?event: rdf:type wisenet:PersonInSpace;
    ?event event:product ?person;
           wisenet:place ?space;
           wisenet:isOpen "true"^^xsd:boolean.

 # Get Persons's descriptors
 ?person wisenet:visualDescriptors ?descriptors.
}
```

ing `Person` is lower than an empirical threshold, then the detection corresponds to a `Person` instance located in the current `Space` (the branch yes of point A in Figure 7. This will be referred as "the A.yes branch"). By consequence, the `Person`'s visual descriptor and `BoundingBox` are updated with the detection's visual descriptor and `xywh` coordinates, respectively. Also, the end-time of the corresponding `PIS-E` is updated with the detection's time.

If no matching `Person` is found in the current `Space`, then a new `PIS-E` will be created (the A.no branch in Figure 7). Before its creation, the central API checks if the detection belongs to a `Person` located in a neighbor space (point B in Figure 7). This is done by executing the query shown in Listing 7, where:

- Lines 4-8 get the active events in the neighbor spaces.
- Lines 11-13 check that the last detection of the `PIS-E` occurred around the same time as the new detection. The time threshold is 5 seconds before the detection timestamp (2017-04-17T10:18:45.922Z).
- Lines 16-18 check that the last detection of the `PIS-E` occurred around the same ROI.

After executing the query in Listing 7, the visual descriptors of the selected `Person` candidates are compared with the detection's visual descriptor using the Chebysev distance.

If a corresponding `Person` instance is found, then it means that the person is passing from one space to another (the B.yes branch in Figure 7). Hence, the new `PIS-E` will involve the same `Person` and it will be related to the `Person`'s old `PIS-E`. Also, the `Person`'s visual descriptor, `BoundingBox` and location will be updated. If no matching `Person` is found in the neighbor spaces, then a new `Person` instance is created (the B.no branch in Figure 7).

Listing 7: Query to get related PIS-E, using as example instances of Listing 5. Lines that start with # are comments.

```
SELECT ?person ?event ?descriptors
WHERE {
          # Get PIS in neighbour spaces
          inst:Corridor_3 wisenet:spaceConnectedTo ?s.
          ?event rdf:type wisenet:PersonInSpace;
                 wisenet:place ?s;
                 wisenet:isOpen "true"^^xsd:boolean.
          FILTER (?s != inst:Corridor_3)

          # Check if the last detection occurred in the last 5 seconds
          ?event event:time/time:hasEnd ?timeInstEnd.
          ?timeInstEnd time:inXSDDateTime ?timeEnd.
          FILTER (?timeEnd > "2017-04-17T10:18:40.922Z"^^xsd:dateTime)

          # Check if the last detections occurred around the same ROI
          wni:Roi_5 wisenet:represents ?roi_element.
          ?detection wisenet:isSubEventOf ?event;
                     wisenet:inImageRegion/wisenet:represents ?roi_element.

          # Get the PIS-E's person and its descriptors
          ?event event:product ?person.
          ?person wisenet:visualDescriptors ?descriptors.
}
```

The workflow continues by creating and inserting the new PIS-E and Detection instances with all the properties shown in Figure 6.

The process is then repeated for each detection in the SC-message.

Finally, the central API performs a cleaning process which consists of:

- **Closing some PIS-Es:** if a Person belongs to two PIS-Es and one of them has not being populated for certain time, then the oldest PIS-E is closed. This is done by setting the property wisenet:isOpen to "false".
- **Detect noisy instances:** if a PIS-E has a low number of detections and it has not being populated for a certain time, then the PIS-E is closed and its related Person and Detection are consider as 'noise' by setting the property wisenet:isNoise to "true".

Notice that the central API executes by demand, it does not run in the background, *i.e.*, that it only runs each time a SC-message is received or when a service request is received (*e.g.*, by the monitor API).

7. Evaluation, use case and discussion

This paper focused on creating an ontology model to fuse and re-purpose the different types of information required by a Panoptes building. Once the model is assembled it needs to be evaluated to verify if it satisfies its intent.

Currently, the smart camera network (SCN) has been deployed and the algorithms of motion detection, face detection, fall detection and person detection were implemented on it. This section presents an evaluation of the WiseNET ontology, some use cases of the complete system and a discussion on how WiseNET may overcome some issues of classical computer vision.

7.1. WiseNET ontology evaluation

According to Hitzler *et al.* the accuracy criteria is a central requirement for ontology evaluation [19]. This criteria consists in verifying if the ontology accurately captures the aspects of the modeled domain for which it has been designed for.

The development of the WiseNET ontology was based on some competency questions (see Table 1), therefore the evaluation consists in showing that those questions can be answered by the ontology. Listing

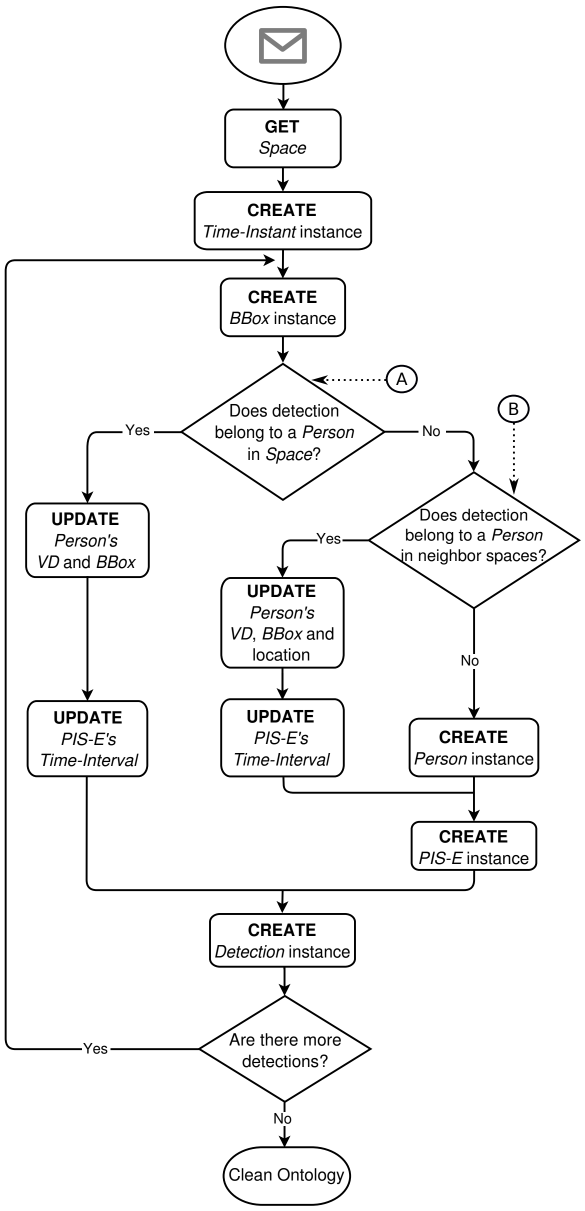


Fig. 7. Dynamic population process performed by the central API. The process starts with the reception of a message from a smart camera, and finishes with the insertion of pertinent knowledge. The GET, UPDATE and CREATE processes involve SPARQL queries using SELECT, DELETE and INSERT commands. *PIS-E* stands for `PersonInSpace` event, *Bbox* for `BoundingBox` and *VD* for visual descriptor. The points A and B are used as reference in the text.

8 and 9 present the queries to answer some competency questions. These questions were selected because they involve aspects which are important in a Panoptes building.

– **Question 1:** involves time-space knowledge of a person. Notice that a person may entered a space more than once.

  ∗ Lines 5-8 get the `PIS-E`s relating the specific person with the specific space.
  ∗ Line 9 gets the starting times of the `PIS-E`.

– **Question 2:** involves knowledge of the SCs and the IFC elements that it monitors.

  ∗ Lines 16 get the ROIs that the SC observes .
  ∗ Line 17 gets the type of elements that those ROIs represent.
  ∗ Line 18 removes the binding.

– **Question 3:** is about counting the `PIS-E` in a specific time-space.

  ∗ Lines 25-27 get the `PIS-E`s related to the specific space.
  ∗ Lines 28-29 get the starting and end time of the `PIS-E`s.
  ∗ Lines 30-31 considers only the `PIS-E`s that started before the specific time and finish after it.

– **Question 4:** involves the time duration a person is in a space.

  ∗ Lines 38-41 get the `PIS-E`s relating the specific person with the specific space.
  ∗ Line 42-43 get the duration and its units of the `PIS-E`s.

– **Question 5:** is about counting the number of people that have been in each space until now. This can be considered as an *accumulated heat map* because it is in a period of time, not at a precise time.

  ∗ Lines 50-51 get all the `PIS-E`s and its spaces.
  ∗ Line 53 groups the results by spaces.
  ∗ Line 48 counts the number of `PIS-E`s in each space.

Notice that the *space more visited* could be easily obtained by adding the line `ORDER BY desc(?numberOfPeople) LIMIT 1`, after Line 53. This orders the spaces in a descending order, according to their number of people, and provides only the first one, the one with the most people.

– **Question 6:** is about counting which door has been used the most. When a person goes through a door from one space to another, many detections are performed but only one PIS-E is created, therefore, the door usage is determined by counting the number of PIS-E that have started around it.

  ∗ Line 59 gets all the PIS-E.
  ∗ Lines 60-61 get the detections attached to the PIS-E and their starting time instant.
  ∗ Line 62 gets the detections that have as time the starting time of the PIS-E, *i.e.*, the detections that started the PIS-E.
  ∗ Lines 63-64 filters the detections in order to keep those that were made around a door.
  ∗ Line 66 groups the results by doors.
  ∗ Line 67 orders the doors in a descending order, according to their number of detections, and provides only the first one, the most used one.

7.2. *Use case*

Consider the test scenario depicted in Figure 8, where two people are walking in the I3M building. The SCN extracts the pertinent information from the scene, and then sends the SC-messages to the central API. The central API processes those messages and generates the new knowledge that will be inserted into the ontology by following the workflow shown in Figure 7.

Table 4 shows the dynamic population performed by the central API at each timestamps of Figure 8. It is important to remark the following:

– For each detection performed by the SCs, a Detection instance is created and attached to a Person instance and to a PIS-E instance (timestamps 1-9).
– The Person instances are only created once, at the beginning of the test scenario (timestamp 1). This is due to the person re-identification process and the relation between events.
– The first time a person is detected in a new space a PIS-E instance is created (timestamps 1, 3, 6 and 8).
– If a detection is made around an ROI, then the new PIS-E may be related to an existing PIS-E (timestamps 3, 6 and 8), meaning that they involve the same Person instance.

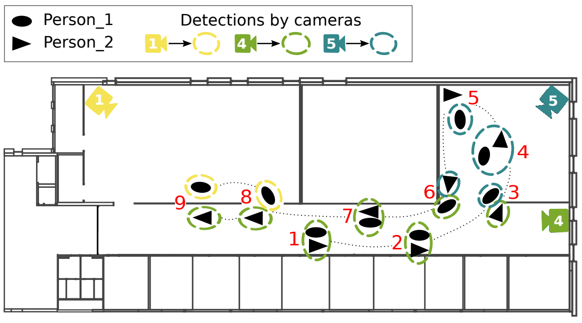


Fig. 8. WiseNET system people tracking. Two people are walking in a built environment while being detected by the SCN. For conciseness we only considered 3 SCs and 9 timestamps (numbers in red). Notice that at timestamp 5, Person_2 is not being detected. The dynamic population at each timestamp can be found in Table 4. This scenario is a fragment of a video set which can be found at http://wisenet.checksem.fr/#/demo.

– The timestamps shown are continuous, meaning that between all the timestamps there exists many others. For example between timestamps 3 and 4 there is a detection (at timestamp 3.5 for example) that explains the creation of PIS-E_4, which was omitted for conciseness.
– At timestamp 5 the Person_2 is not being detected however, notice that at timestamp 6 no new Person was created, this is because the system deduced that the person is still in the space. More details on this will be presented in Section 7.3.

During the test scenario, the central unit was deployed on a Intel(R) Xeon E5-2430@2.20GHz CPU with 4GB RAM and 40GB HDD. The time performance of the dynamic population was evaluated by verifying the time required to process different number of detections, as shown in Figure 9. The graph was obtained by sending to the central API a single SC-message composed of multiple different detections. From the graph, it can be observed that the time to deal with 15 detections (< 1s) is still compatible with real-time analysis.

Regarding the privacy protection, it is important to remark that the SCN used in the WiseNET system does not send or save any images, thereby protecting the privacy of the individuals. However, one exception could be made if the ontology infers that an illegal act is occurring, in which case, the central API can use that inferred knowledge to send a message to the SC telling it to start recording and to save the images locally (to have them as proof).

From the dynamic population many use cases could be imagined. For example, Figure 10 shows a heat map

Listing 8: Queries to answer selected competency questions. Lines that start with # are comments.

```
###########
# QUESTION 1: A what times does 'Person_1' enter the 'Space_303'?
SELECT ?enteringTimes
WHERE {
        ?x rdf:type wisenet:PersonInSpace;
           wisenet:place inst:Space_303;
           event:product wni:Person_1;
           event:time ?timeInterval.
        ?timeInterval time:hasBeginning/time:inXSDDateTime ?enteringTimes.
}

###########
# QUESTION 2: Which building elements does the 'SmartCamera_1' observes?
SELECT DISTINCT ?elementsType
WHERE {
        wni:SmartCamera_1 wisenet:hasRegionOfInterest ?y.
        ?y wisenet:represents/rdf:type ?elementsType.
        FILTER (?elementsType != owl:NamedIndividual)
 }

###########
# QUESTION 3: How many people were in the 'Corridor_3' at time '2017-04-17T10:18:45.922Z'?
SELECT COUNT (?x) AS ?numberOfPeople
WHERE {
        ?x rdf:type wisenet:PersonInSpace;
           wisenet:place inst:Corridor_3;
           event:time ?timeInterval.
        ?timeInterval time:hasBeginning/time:inXSDDateTime ?startTime.
        ?timeInterval time:hasEnd/time:inXSDDateTime ?endTime.
        FILTER (?startTime < "2017-04-17T10:18:45.922Z"^^xsd:dateTime &&
                ?endTime > "2017-04-17T10:18:45.922Z"^^xsd:dateTime)
}

###########
# QUESTION 4: How long does the 'Person_1' stayed in the 'Corridor_3'?
SELECT ?timeDuration ?temporalUnit
WHERE {
        ?x rdf:type wisenet:PersonInSpace;
           wisenet:place inst:Corridor_3;
           event:product wni:Person_1;
           event:time ?timeInterval.
        ?timeInterval time:hasduration/time:numericDuration ?timeDuration.
        ?timeInterval time:hasduration/time:unitType ?temporalUnit.
}

###########
# QUESTION 5: How many people have been in each space (accumulated heat map)?
SELECT ?space (COUNT (?x) AS ?numberOfPeople)
WHERE{
        ?x rdf:type wisenet:PersonInSpace;
           wisenet:place ?space.
}
GROUP BY ?space
```

Listing 9: Queries to answer selected competency questions (continued from Listing 8). Lines that start with # are comments.

```
###########
# QUESTION 6: Which is the more used door?
SELECT ?door (COUNT (?detections) AS ?usage)
WHERE{
        ?events rdf:type wisenet:PersonInSpace;
              wisenet:hasSubEvent ?detections.
              event:time/time:hasBeginning ?timeInstant.
        ?detections event:time ?timeInstant;
                  wisenet:inImageRegion ?door.
        ?door rdf:type ifcowl:IfcDoor
}
GROUP BY ?door
ORDER BY DESC(?numberOfPeople) LIMIT 1
```

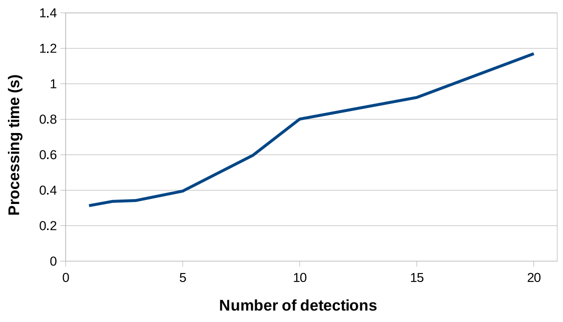


Fig. 9. Graph showing the average time required by the central API to process different number of detections.

representing the number of people in each space at timestamps 2 and 3 from Figure 8. The heat map was obtained by using the query to answer the *Question 5* in Listing 8. At timestamp 3 the system deduces that a person passed from one space to another by using a specific door (shown by the arrow in Figure 10). This deduction was obtained by knowing that the spaces involved in the `PIS-Es` are connected by a specific door.

More use cases could be developed such as:

- **Path guidance:** automatically control if a building user takes the correct path and reaches his final destination.
- **People counting:** count people in building/spaces and control if the number of people is appropriate according to the space limitations.
- **Fire evacuation:** in the case of a fire or smoke detection, the system could guide the people to different exits while avoiding the places where the fire/smoke is located.

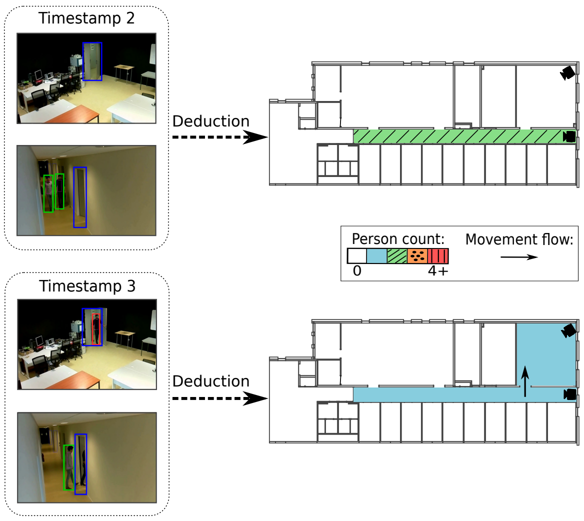


Fig. 10. Heat map representing the number of people. The timestamps 2 and 3 were taken from Figure 8. The right side shows what the SCs observe and the left side shows what the WiseNET system deduces from it. In the camera images, the blue bounding box represents an ROI, the green bounding box represents a detection and the red one represents a detection around an ROI. The complete heat map can be found at `http://wisenet.checksem.fr/#/heatmap`.

- **Smart light control system:** react according to the person's presence and not to the person's motion, which is an important problem with most of automatic light control systems.
- **Building maintenance:** check the usage of the building elements/spaces in order to schedule their maintenance procedures.

Table 4

Dynamic population performed during the people tracking scenario shown in Figure 8. The processing time denotes the time required by the central API to perform the dynamic population.

Timestamp	Dynamic population	Processing time (s)
1	`Person_1` created `Detection_1` created attached to: `Person_1` `PIS-E_1` `PIS-E_1` created attached to: `Person_1`	0.351
	`Person_2` created `Detection_2` created attached to: `Person_2` `PIS-E_2` `PIS-E_2` created attached to: `Person_1`	
2	`Detection_3` created attached to: `Person_1` `PIS-E_1`	0.323
	`Detection_4` created attached to: `Person_2` `PIS-E_2`	
3	`Detection_5` created around: `Roi_5` attached to: `Person_1` `PIS-E_3` `PIS-E_3` created related to: `PIS-E_1`	0.347
	`Detection_6` created attached to: `Person_2` `PIS-E_2`	
4	`Detection_7` created attached to: `Person_1` `PIS-E_3`	0.339
	`Detection_8` created attached to: `Person_2` `PIS-E_4`	
5	`Detection_9` created attached to: `Person_1` `PIS-E_3`	0.287
6	`Detection_10` created around: `Roi_5` attached to: `Person_1` `PIS-E_5` `PIS-E_5` created related to: `PIS-E_3`	0.349
	`Detection_11` created around: `Roi_5` attached to: `Person_2` `PIS-E_4`	
7	`Detection_12` created attached to: `Person_1` `PIS-E_5`	0.334
	`Detection_13` created attached to: `Person_2` `PIS-E_6`	
8	`Detection_14` created around: `Roi_2` attached to: `Person_1` `PIS-E_7` `PIS-E_7` created related to: `PIS-E_5`	0.346
	`Detection_15` created attached to: `Person_2` `PIS-E_4`	
9	`Detection_16` created attached to: `Person_1` `PIS-E_7`	0.342
	`Detection_17` created attached to: `Person_2` `PIS-E_4`	

7.3. Overcoming computer vision limitations

A semantics-based system such as WiseNET appears promising to overcome some of the computer vision limitations, in particular:

- **Missed detections:** this occurs when the system does not detect (misses) pertinent information. The WiseNET system could overcome this problem by using its global knowledge in order to advice a specific SC node to focus its detection resources in certain ROI because a person should/will appear there.
- **False detections:** a false detection occurs when the system detects false information. The WiseNET system overcomes this problem by checking if a `PIS-E` is only populated during a small period of time. If it is, then the `PIS-E` and its related `Person` and `Detection` are consider as a false detection, and they can be removed from the system.

Another way to avoid false detections could be the addition of logical rules, such as, if a person is detected for the first time inside a room without passing through a door this could be considered as a false detection. Moreover, some semantics of the image could also be exploited as the position of the floor and the dimensions of the people.

– **Training and testing data:** some of the biggest drawbacks of deep-learning technologies are the long training time and the big amount of data set for training/testing the models [28]. In contrast, the WiseNET system does not require any training/testing data. Moreover, a great advantage of semantics-based systems is the flexibility to add rules according to the precise knowledge of an expert, unlike deep-learning in which the decision-making process is a black box.
– **Occlusions:** this occurs when pertinent information is outside the field of view (FOV) of a camera. We propose to illustrate this by using an example where WiseNET efficiently deduces that a person is still in a space even if he is not being detected by the SC (see Figure 11). This is achieved by using contextual information and a rule stating that if the last detection of a person is not around an ROI, then the person is still in the space.

Those and other advantages of WiseNET will be deeply studied in future works, specifically a quantitative evaluation will be done comparing the system's performance using other computer vision techniques such as deep learning.

8. Conclusion and prospectives

In this work we tried to develop an "all-seeing" smart building, which we have called Panoptes building. With that motivation, the WiseNET ontology was developed. Its main goal is to fuse the different built environment contextual information with that coming from the SCN and other domains, to allow real-time event detections and system reconfiguration. The purpose of the developed ontology is to create a kernel of a Panoptes building system, rather than working towards publishing another generic ontology. The ontology development procedure was performed using different semantic technologies, and it consisted of: defining a set of questions that the ontology should answer (competency questions); reusing different domain ontologies (`DUL`, `event`, `ifcowl`, `person` and `ssn`); creating a set of classes and properties to connect the different domain ontologies and to complete the application knowledge; defining a set of constrains and extending the expressiveness by using logic rules; and finally, populating the ontology with static information (built environment and system setup) and dynamic information (SC detections).

The WiseNET system is a semantics-based real-time reasoning system, that fuses different sources of data and is expected to overcome limitations of MCBS. The WiseNET system selects relevant information from the video streams and adds contextual information to overcome problems of missing information due to false/missed detections. Additionally, it relates events that occurred at different times, without human interaction. It also protects the user's privacy by not sending nor saving any image, just extracting the knowledge from them. It may as well, reconfigure the SCN according to the inferred knowledge. To summarize, the WiseNET system enables interoperability of information from different domains such as the built environment, event information and information coming from the SCN.

A future goal of the WiseNET system is to offer services to building users according to information coming from a network of heterogeneous sensors deployed on the built environment and contextual information. This is a highly complex task due to the large scope of the building system, that goes from the static physical structure of the built environment to the internal environment in terms of the dynamic building users and the way how they interact with the building facilities.

The future works will focus on completing the externalization of the ontology knowledge using the bidirectionality between the SCN and the central unit, which is a novelty in the semantic web domain. Specifically, the mechanism of using the ontology to interact with external devices by checking the inferred knowledge and using it to reconfigure the SCN by: triggering a specific action (*e.g.,* recording and triggering an alarm), changing the image processing algorithm or by saying to the SC to focus on a specific ROI. Moreover, the dependency on the central API leaves open the question of automatically output the ontology knowledge, especially the inferred knowledge.

Immediate next steps involve adjusting the WiseNET with the updated version of the `ssn` ontology [17] and study the adaptation to BOT (Building Topology Ontology) [50], which is a simplified version of the `ifcowl` ontology similar to the one obtained after the extraction and population framework.

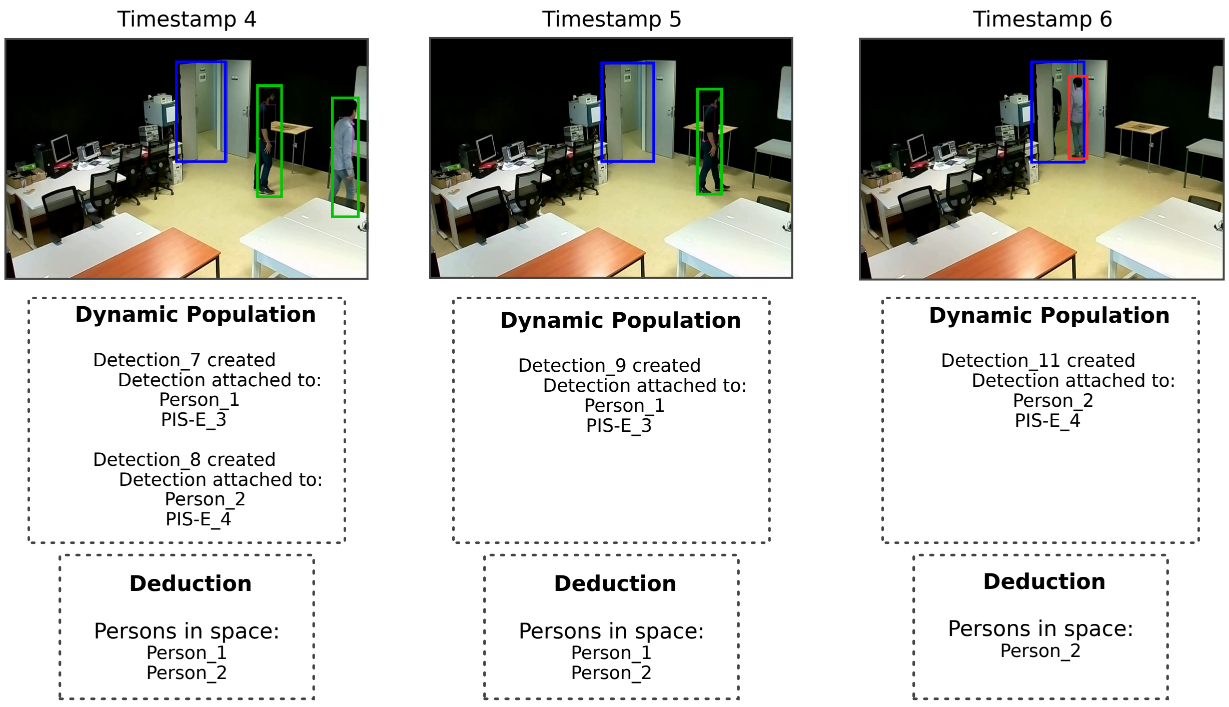


Fig. 11. Example of occlusion. The timestamps used were taken from Figure 8. Even if one of the people disappeared at timestamp 5, the WiseNET system deduces (deduction box) that he is still inside the space because his last detection was not made around the ROI. In the camera images, the blue bounding box represents an ROI, the green bounding box represents a detection and the red one represents a detection around an ROI.

9. Acknowledgement

The authors would like to thank the Conseil Régional de Bourgogne Franche Comté and the french government for their funding.

We would also like to thank Ali Douiyek and Arnaud Rolet for the technical assistance and management of the technical team. As well as D.Barquilla, T.Coulon, Q.Enjalbert, A.Goncalves, L.Lamarque, E.Grillet, G.Kasperek, M.Le Goff, E.Menassol and K.Naudin for their help in the development of the website and the 3D visualization tools of the building.

References

[1] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003. ISBN 0-521-78176-0.

[2] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc J. Van Gool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008. https://doi.org/10.1016/j.cviu.2007.09.014.

[3] Dan Brickley and Ramanathan V Guha, editors. *RDF Schema 1.1*. W3C Recommendation, 25 February 2014. URL https://www.w3.org/TR/rdf-schema/.

[4] Andres Burbano, Samir Bouaziz, and Marius Vasiliu. 3d-sensing distributed embedded system for people tracking and counting. In Hamid R. Arabnia, Leonidas Deligiannidis, and Quoc-Nam Tran, editors, *Proceedings of 2015 International Conference on Computational Science and Computational Intelligence*, pages 470–475. IEEE, 2015. https://doi.org/10.1109/CSCI.2015.76.

[5] Sung-Hyuk Cha. Comprehensive survey on distance/similarity measures between probability density functions. *International Journal of Mathematical Models and Methods in Applied Sciences*, 1(4):300–307, 2007. URL http://www.naun.org/main/NAUN/ijmmas/mmmas-49.pdf.

[6] Wirawit Chaochaisit, Masahiro Bessho, Noboru Koshizuka, and Ken Sakamura. Human localization sensor ontology: Enabling OWL 2 DL-based search for user's location-aware sensors in the IoT. In *Tenth IEEE International Conference on Semantic Computing, ICSC 2016, Laguna Hills, CA, USA, February 4-6, 2016*, pages 107–111. IEEE Computer Society, 2016. https://doi.org/10.1109/ICSC.2016.31. URL https://doi.org/10.1109/ICSC.2016.31.

[7] Michael Compton, Payam M. Barnaghi, Luis Bermudez, Raul Garcia-Castro, Óscar Corcho, Simon J. D. Cox, John Graybeal, Manfred Hauswirth, Cory A. Henson, Arthur Herzog, Vincent A. Huang, Krzysztof Janowicz, W. David Kelsey, Danh Le Phuoc, Laurent Lefort, Myriam Leggieri, Holger Neuhaus, An-

driy Nikolov, Kevin R. Page, Alexandre Passant, Amit P. Sheth, and Kerry Taylor. The SSN ontology of the W3C semantic sensor network incubator group. *Journal of Web Semantics*, 17: 25–32, 2012. https://doi.org/10.1016/j.websem.2012.05.003.

[8] Simon Cox and Chris Little, editors. *Time ontology in OWL*. W3C Recommendation, 19 October 2017. URL https://www.w3.org/TR/owl-time/.

[9] Richard Cyganiak, David Wood, and Markus Lanthaler, editors. *RDF 1.1 Concepts and Abstract Syntax*. W3C Recommendation, 25 February 2014. URL https://www.w3.org/TR/rdf11-concepts/.

[10] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA*, pages 886–893. IEEE Computer Society, 2005. https://doi.org/10.1109/CVPR.2005.177.

[11] Tarcisio Mendes de Farias, Ana Roxin, and Christophe Nicolle. SWRL rule-selection methodology for ontology interoperability. *Data & Knowledge Engineering*, 105:53–72, 2016. https://doi.org/10.1016/j.datak.2015.09.001.

[12] Michael Dibley, Haijiang Li, Yacine Rezgui, and John Miles. An ontology framework for intelligent sensor-based building monitoring. *Automation in Construction*, 28:1–14, 2012. https://doi.org/10.1016/j.autcon.2012.05.018.

[13] Patrick Doherty, John Mylopoulos, and Christopher A. Welty, editors. *The Even More Irresistible SROIQ*, 2006. AAAI Press. URL http://www.aaai.org/Library/KR/2006/kr06-009.php.

[14] Chuck Eastman, Paul Teicholz, Rafael Sacks, and Kathleen Liston. *BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors,*. John Wiley & Sons, 2nd edition, 2011.

[15] Pedro F. Felzenszwalb, Ross B. Girshick, David A. McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010. https://doi.org/10.1109/TPAMI.2009.167.

[16] Aldo Gangemi. Ontology:DOLCE+DnS ultralite homepage, 2010. URL http://ontologydesignpatterns.org/wiki/Ontology:DOLCE+DnS_Ultralite. Last update on 4 March 2010.

[17] Armin Haller, Krzysztof Janowicz, Simon Cox, Danh Le Phuoc, Kerry Taylor, and Maxime Lefrançois, editors. *Semantic Sensor Network Ontology*. W3C Recommendation, 19 October 2017. URL https://www.w3.org/TR/vocab-ssn/.

[18] Arun Hampapur, Lisa Brown, Jonathan Connell, Ahmet Ekin, Norman Haas, Max Lu, Hans Merkl, and Sharath Pankanti. Smart video surveillance: Exploring the concept of multiscale spatiotemporal tracking. *IEEE Signal Processing Magazine*, 22 (2):38–51, 2005. https://doi.org/10.1109/MSP.2005.1406476.

[19] Pascal Hitzler, Markus Krötzsch, and Sebastian Rudolph. *Foundations of Semantic Web Technologies*. Chapman and Hall/CRC Press, 2010. URL http://www.semantic-web-book.org/.

[20] Jongyi Hong, Euiho Suh, and Sung-Jin Kim. Context-aware systems: A literature review and classification. *Expert Systems with Applications*, 36(4):8509–8522, 2009. https://doi.org/10.1016/j.eswa.2008.10.071.

[21] Ian Horrocks and Ulrike Sattler. A tableau decision procedure for *SHOIQ*. *Journal Automated Reasoning*, 39(3):249–276, 2007. https://doi.org/10.1007/s10817-007-9079-9.

[22] Ian Horrocks, Peter F. Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosof, and Mike Dean. *SWRL: A Semantic Web Rule Language Combining OWL and RuleML*. W3C Member submission, 21 May 2004. URL https://www.w3.org/Submission/SWRL/.

[23] International Organization for Standardization. ISO 19439:2006 enterprise integration – framework for enterprise modelling, 2006. URL https://www.iso.org/standard/33833.html.

[24] International Organization for Standardization. *ISO 16739:2013 Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries*. ISO, 2013. URL https://www.iso.org/standard/51622.html.

[25] Carlos Fernando Crispim Junior, Vincent Buso, Konstantinos Avgerinakis, Georgios Meditskos, Alexia Briassouli, Jenny Benois-Pineau, Ioannis Kompatsiaris, and François Brémond. Semantic event fusion of different visual modality concepts for activity recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(8):1598–1611, 2016. https://doi.org/10.1109/TPAMI.2016.2537323.

[26] Pakorn KaewTraKulPong and Richard Bowden. An improved adaptive background mixture model for real-time tracking with shadow detection. In Paolo Remagnino, Graeme A. Jones, Nikos Paragios, and Carlo S. Regazzoni, editors, *Video-Based Surveillance Systems*, chapter 11, pages 135–144. Springer, 2002. https://doi.org/10.1007/978-1-4615-0913-4_11.

[27] Yevgeny Kazakov. RIQ and SROIQ are harder than SHOIQ. In Gerhard Brewka and Jérôme Lang, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Eleventh International Conference, KR 2008, Sydney, Australia, September 16-19, 2008*, pages 274–284. AAAI Press, 2008. URL http://www.aaai.org/Library/KR/2008/kr08-027.php.

[28] Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015. https://doi.org/10.1038/nature14539.

[29] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. https://doi.org/10.1023/B:VISI.0000029664.99615.94.

[30] Roberto Marroquin, Julien Dubois, and Christophe Nicolle. WiseNET - smart camera network interacting with a semantic model: PhD forum. In *Proceedings of the 10th International Conference on Distributed Smart Camera, Paris, France, September 12-15, 2016*, pages 224–225. ACM, 2016. https://doi.org/10.1145/2967413.2974036.

[31] David Mills, Jim Martin, Jack Burbank, and William Kasch. *Network Time Protocol Version 4: Protocol and Algorithms Specification*. RFC 5905, 2010. https://doi.org/10.17487/RFC5905.

[32] Romuald Mosqueron, Julien Dubois, Marco Mattavelli, and David Mauvilet. Smart camera based on embedded HW/SW coprocessor. *EURASIP Journal on Embedded Systems*, 2008, 2008. https://doi.org/10.1155/2008/597872.

[33] Boris Motik, Ulrike Sattler, and Rudi Studer. Query answering for OWL-DL with rules. *Journal of Web Semantics*, 3(1):41–60, 2005. https://doi.org/10.1016/j.websem.2005.05.001.

[34] Tahir Nawaz, Bernhard Rinner, and James M. Ferryman. User-centric, embedded vision-based human monitoring: A concept and a healthcare use case. In *Proceedings of the 10th International Conference on Distributed Smart Camera, Paris,*

France, September 12-15, 2016, pages 25–30. ACM, 2016. https://doi.org/10.1145/2967413.2967422.

[35] Natalya F Noy and Deborah L McGuinness. Ontology development 101: A guide to creating your first ontology. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, Stanford University, 2001. URL https://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html.

[36] Ovid. *Metamorphoses*, chapter i.569-747. Oxford University Press, 1986. Translation by A. D. Melville.

[37] Pieter Pauwels and Jyrki Oraskari. IFC-to-RDF-converter, 2016. URL https://github.com/mmlab/IFC-to-RDF-converter. Last update on 30 April 2016.

[38] Pieter Pauwels and Ana Roxin. SimpleBIM: From full ifcOWL graphs to simplified building graphs. In S. Christodoulou and R. Scherer, editors, *Proceedings of the 11th European Conference on Product and Process Modelling (ECPPM 2014), Limassol, Cyprus, 7-9 September 2016: eWork and eBusiness in Architecture, Engineering and Construction*, pages 11–18. CRC Press, 2016.

[39] Pieter Pauwels and Walter Terkaj. EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology. *Automation in Construction*, 63:100–133, 2016. https://doi.org/10.1016/j.autcon.2015.12.003.

[40] Pieter Pauwels, Thomas Krijnen, Walter Terkaj, and Jakob Beetz. Enhancing the ifcOWL ontology with an alternative representation for geometric data. *Automation in Construction*, 80:77–94, 2017. https://doi.org/10.1016/j.autcon.2017.03.001.

[41] ISA Programme. ISA programme person core vocabulary, 2013. URL https://www.w3.org/ns/person/. Last update on 18 November 2013.

[42] Yves Raimond and Samer Abdallah. The event ontology, 25th October 2007. URL http://motools.sf.net/event/event.html.

[43] Juan Carlos San Miguel, José María Martínez Sanchez, and Alvaro García-Martín. An ontology for event detection and its application in surveillance video. In Stefano Tubaro and Jean-Luc Dugelay, editors, *Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2009, 2-4 September 2009, Genova, Italy*, pages 220–225. IEEE Computer Society, 2009. https://doi.org/10.1109/AVSS.2009.28.

[44] Benaoumeur Senouci, Imen Charfi, Barthélémy Heyrman, Julien Dubois, and Johel Mitéran. Fast prototyping of a SoC-based smart-camera: a real-time fall detection case study. *Journal of Real-Time Image Processing*, 12(4):649–662, 2016. https://doi.org/10.1007/s11554-014-0456-4.

[45] Fikret Sivrikaya and Bülent Yener. Time synchronization in sensor networks: a survey. *IEEE Network*, 18(4):45–50, 2004. https://doi.org/10.1109/MNET.2004.1316761.

[46] Stanislava Soro and Wendi Rabiner Heinzelman. A survey of visual sensor networks. *Advances in Multimedia*, 2009: 640386:1–640386:21, 2009. https://doi.org/10.1155/2009/640386.

[47] Rudi Studer, V. Richard Benjamins, and Dieter Fensel. Knowledge engineering: Principles and methods. *Data & knowledge engineering*, 25(1-2):161–197, 1998. https://doi.org/10.1016/S0169-023X(97)00056-6.

[48] Jakob Suchan and Mehul Bhatt. Deep semantic abstractions of everyday human activities - on commonsense representations of human interactions. In Aníbal Ollero, Alberto Sanfeliu, Luis Montano, Nuno Lau, and Carlos Cardeira, editors, *ROBOT 2017: Third Iberian Robotics Conference - Volume 1, Seville, Spain, November 22-24, 2017*, volume 693 of *Advances in Intelligent Systems and Computing*, pages 477–488. Springer, 2017. https://doi.org/10.1007/978-3-319-70833-1_39.

[49] Shamik Sural, Gang Qian, and Sakti Pramanik. Segmentation and histogram generation using the HSV color space for image retrieval. In *Proceedings of the 2002 International Conference on Image Processing, ICIP 2002, Rochester, New York, USA, September 22-25, 2002*, pages 589–592. IEEE, 2002. https://doi.org/10.1109/ICIP.2002.1040019.

[50] Walter Terkaj, Georg Ferdinand Schneider, and Pieter Pauwels. Reusing domain ontologies in linked building data: The case of building automation and control. In Stefano Borgo, Oliver Kutz, Frank Loebe, Fabian Neuhaus, Kemo Adrian, Mihailo Antovic, Valerio Basile, Martin Boeker, Diego Calvanese, Tommaso Caselli, Giorgio Colombo, Roberto Confalonieri, Laura Daniele, Jérôme Euzenat, Antony Galton, Dagmar Gromann, Maria M. Hedblom, Heinrich Herre, Inge Hinterwaldner, Andrea Janes, Ludger Jansen, Kris Krois, Antonio Lieto, Claudio Masolo, Rafael Peñaloza, Daniele Porello, Daniele P. Radicioni, Emilio M. Sanfilippo, Daniel Schober, Rossella Stufano, and Amanda Vizedom, editors, *Proceedings of the Joint Ontology Workshops 2017 Episode 3: The Tyrolean Autumn of Ontology, Bozen-Bolzano, Italy, September 21-23, 2017.*, volume 2050 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2017. URL http://ceur-ws.org/Vol-2050/FOMI_paper_4.pdf.

[51] The W3C SPARQL Working Group, editor. *SPARQL 1.1 Overview*. W3C Recommendation, 21 March 2013. URL https://www.w3.org/TR/sparql11-overview/.

[52] Ana Tomé, Martijn Kuipers, Tiago Pinheiro, Mário Nunes, and Teresa Heitor. Space–use analysis through computer vision. *Automation in Construction*, 57:80–97, 2015. https://doi.org/10.1016/j.autcon.2015.04.013.

[53] Christopher Town. Ontological inference for image and video analysis. *Machine Vision and Applications*, 17(2):94–115, 2006. https://doi.org/10.1007/s00138-006-0017-3.

[54] Renaud Vanlande, Christophe Nicolle, and Christophe Cruz. IFC and building lifecycle management. *Automation in construction*, 18(1):70–78, 2008. https://doi.org/10.1016/j.autcon.2008.05.001.

[55] Paul A. Viola and Michael J. Jones. Rapid object detection using a boosted cascade of simple features. In *2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001), with CD-ROM, 8-14 December 2001, Kauai, HI, USA*, pages 511–518. IEEE Computer Society, 2001. https://doi.org/10.1109/CVPR.2001.990517.

[56] W3C OWL Working Group, editor. *OWL 2 Web Ontology Language Document Overview (Second Edition)*. W3C Recommendation, 11 December 2012. URL http://www.w3.org/TR/owl2-overview/.

[57] Thomas Winkler and Bernhard Rinner. Security and privacy protection in visual sensor networks: A survey. *ACM Computing Surveys*, 47(1):2:1–2:42, 2014. https://doi.org/10.1145/2545883.

[58] Wayne H. Wolf, I. Burak Özer, and Tiehan Lv. Smart cameras as embedded systems. *IEEE Computer*, 35(9):48–53, 2002. https://doi.org/10.1109/MC.2002.1033027.

[59] Shoou-I Yu, Yi Yang, and Alexander G. Hauptmann. Harry Potter's Marauder's Map: Localizing and tracking multiple persons-of-interest by nonnegative discretization. In *2013 IEEE Conference on Computer Vision and Pattern Recogni-*

tion, Portland, OR, USA, June 23-28, 2013, pages 3714–3720. IEEE Computer Society, 2013. https://doi.org/10.1109/CVPR.2013.476.

[60] Jun Yue, Zhenbo Li, Lu Liu, and Zetian Fu. Content-based image retrieval using color and texture fused features. *Mathematical and Computer Modelling*, 54(3-4):1121–1127, 2011. https://doi.org/10.1016/j.mcm.2010.11.044.

[61] Zoran Zivkovic and Ferdinand van der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters*, 27(7):773–780, 2006. https://doi.org/10.1016/j.patrec.2005.11.005.