VITO GETULI

Ontologies for Knowledge modeling in construction planning

Theory and Application

Ricerche. Architettura, Pianificazione, Paesaggio, Design

Firenze University Press, in collaboration with the Department of Architecture of the University of Florence, promotes and supports the series *Ricerche. Architettura, Pianificazione, Paesaggio, Design*. This initiative aims to offer a contribution to national and international research on the project in all its dimensions, both theoretical and operational. The volumes of the series are evaluated according to renowned best practices at an international level and collect the research results of scholars from the University of Florence and from other national and international institutions. *Ricerche. Architettura, Pianificazione, Paesaggio, Design* fully supports Open Access publishing as an ideal tool to share ideas and knowledge in every research field with an open, collaborative and non-profit approach. Open Access books and book chapters allow the research community to achieve a high research impact as well as rapid dissemination in any editorial form.

ricerche | architettura, pianificazione, paesaggio, design

VITO GETULI

Ontologies for Knowledge modeling in construction planning

Theory and Application

CONTENTS

YOU CAN'T MANAGE KNOWLEDGE, NOBODY CAN. WHAT YOU CAN DO IS TO MANAGE THE ENVIRONMENT IN WHICH KNOWLEDGE CAN BE CREATED, DISCOVERED, CAPTURED, SHARED, DISTILLED, VALIDATED, TRANSFERRED, ADOPTED, ADAPTED, AND APPLIED.

Chris Collison and Geoff Parcell

Nowadays, there is an increasing recognition of the value of effective information and knowledge management (KM) in the construction projects. Infact, Knowledge-based Process modelling is used in many fields and even in construction to support various simulation tasks. In this field, ontology-based semantic modelling is seen as an important means of addressing this problem to construct robust knowledge-based systems. In parallel, the advancement of information technology in the AEC industry makes available in a construction project a richness of design information offered by Building Information Modelling (BIM) IFC-based. The development of an ontological version of the IFC schema has been largely promoted and now the ifcOWL Ontology is available in the sector. But, in the construction planning and scheduling task, BIM has progressively shown limits in terms of semantic representation and efficiency of supporting scheduling processes and systems. Moreover, when we think of a process in any given domain, we generally figure it out as a series of actions that leads to a certain outcome. For a construction project, when it comes the execution phase, the process of site planning and activity scheduling can be assumed as what the planner does in the search for the solution of a complex and faceted problem whose variables are numerous (building design, site characteristics, boundary conditions, technology, materials, labor, etc.) and most of the times, if not unknow, at least highly uncertain. As a matter of facts, the planner deals with this problem (process) on the base of his experience or, in other words, of the knowledge he owns about the problem. Furthermore, in the construction sector the overall project performance is strongly dependent on the site management activity. In this regard, process planning is well-known to play a crucial role and, despite the long-lasting efforts, most of the related issues still need to be fully addressed. In fact, project control is based on a specific project schedule determined considering beforehand numerous constraints such as resource availability, completion deadlines for tasks and budget limitations. Cost increases or delays can easily result from poor estimates, schedules or decisions related to tasks decomposition and choice of construction technologies. The planner, in turn, generally identifies constraints, evaluates interactions and solves the related conflicts on the base of the experience he acquired from previous projects.

This means that having available models and tools able to support construction managers and designers in the task of construction activities planning and scheduling strictly depends to make available a Knowledge Base which maps in a formal way and in machine readable way the construction planning and scheduling domain.

The book deals with the aforementioned issue that are deeply felt both by researchers, tools developers and construction professionals. It is divided in two parts.

Part I is composed by three chapter that stands for a theoretical introduction to approaches and technologies based on Knowledge modelling and management. The basic concepts on "ontologies" are presented to the reader in terms of ontology representation languages, development environments and ontology visualization. After a presentation of ontology theory, a specific chapter is dedicated to present the use of Knowledge Modelling in developing Expert Systems (ESs). The chapter starts with an introduction of ESs with a description of the most important and currently addresses problems that ESs is designed to solve. Then a paragraph is dedicated to provide the reader with an overview ESs for the specific field of construction planning.

Part II is the natural consequence of the part I in which the author presents an application of ontology development for the field of construction planning and scheduling based on the theory of the part I. The application is a research developed by the author himself.

In this part, which represents the core of the book, a Knowledge-Base (KB) which maps the construction planning and scheduling domain by using an ontology-based modelling approach is presented. When building an ontology for knowledge mapping of a particular domain of interest, it is important to reflect on the different variable the domain depends on. In the presented case, the problem of modelling the construction process for construction planning and construction activities scheduling is the result of a complex process involving many decision variables, defined as modelling domains. For this reason, the proposed Knowledge Base does not follow an all-in-one modelling approach but analyzes the individual models by considering singularities of each domains separately. This choice of a multi-ontologies system for modelling the construction process will be justified on the ground that further interrelations can be specified in order to provide a higher flexibility to the knowledge base so that it is open to other extensions in terms of others domains (e.g., risk analysis, health and safety management, paths planning, monitoring systems, etc.).

The KB presented consists of four integrated ontologies and, in order to guarantee the reader an understandable vision of the development process, they are presented by using a unique stepwise framework as following:

- Investigation of the knowledge resources focusing on reviewing already existing ontologies, taxonomies, or other sources within the investigated domain, and on how to reuse them.

- Specification of ontology modelling objectives in order to figure out classes, relations and properties comprised in the ontology by using a list of competency questions such as: Why is the ontology being built? What type of information should be involved in the ontology? A clearly visible structure of the objectives with a graphical representation is provided to the reader for each ontology.

- Definition of the topological structure of the ontology which means the core of the ontologies in terms of classes and class hierarchies defined in detail, relationships between classes, properties and attributes.

- Ontology computation in an ontology editing environment: *Protégé*.

Chapter 4 presents a Construction Scheduling Ontology composed by a reusable and extensible base of concepts and relations for representing the scheduling problem in the domain of construction process. Chapter 5 deals with the presentation of a Construction Workspaces Ontology in order to incorporate the workspace planning domain in the knowledge-base architecture. Chapter 6 presents Construction Product Ontology which is the ontology used to connect the KB to the data structure of Building Information Model according to the IFC-data schema. Then Chapter 7 presents the ontology dedicated to the description of temporal properties of site entities in their evolution across time. It also comprises objects to describe possible relations between time periods in order to define the temporal positions among activities, workspaces and building objects. Finally, Chapter 8 is dedicated to present the architecture of an Ontology-based Expert System called "OnSITEsimu" – developed by the author– which is based on the ontologies presented in the monograph. The aim of this chapter is not to provide details on the ES but to provide the reader with knowledge of how ontologies can support automated reasoning mechanisms.

AEC	Architecture, Engineering and Construction industry
BIM	Building Information Model
BIMs	Buildings Information Models
2D	Two-Dimensional
3D	Three-Dimensional
4D	Four-Dimensional
IFC	Industry Foundation Classes
LoD	Level of Development
LoI	Level of Information
VC	Virtual Construction
VR	Virtual Reality
ES	Expert System
ESs	Expert Systems
KB	Knowledge Base
IE	Inference Engine
UI	User Interface
RE	Rule Engine
OWL	Web Ontology Language
SWRL	Semantic Web Rule Language
JHA	Job Hazard Analysis
GIS	Geographic Information System

A

Activity A process in a project that consumes time and also usually consumes or uses other resources (e.g. people, money, materials and equipment). An activity is the smallest unit of work on a Schedule but, depending upon the hierarchy or level of detail of the schedule, may be divisible into smaller or more detailed activities.

Activity Duration The time calculated or estimated to carry out an activity, generally taking into account a specific level of resources, constraints and methods of working.

Activity- Oriented Scheduling The method of developing a schedule that determines the sequence and timing of activities based on the logical work process only and does not take account of any potential limitations of resources.

B

Bar A line on a bar chart that represents the timing and duration of an activity.

Bar Chart A graphical chart on which activities are represented as bars drawn to a common time scale. Typically, a date scale is drawn across the top of the page and a list of activities down the left hand side of the page. Activity timing and durations are represented by horizontal bars.

Baseline Schedule   A fixed or record schedule against which current or future activity is referenced. Often taken to mean the first or original plan but can be re-set (for instance, following a change to the project scope), at which point the reset schedule becomes the (new) baseline schedule.

BIM   See Building Information Model.

Building Information Model   is a virtual representation of a building, potentially containing all the information required to construct the building, using computers and software. The term generally refers both to the model(s) representing the physical characteristics of the project and to all the information contained in and attached to components of these models. When BIM is used in a sentence, it will depend on the context whether it means building information model or building information modeling. A BIM may include any of or all the 2D, 3D, 4D (time element—scheduling), 5D (cost information) representations of a project.

Building Information Modelling   is the act of creating and/or using a BIM.

C

Calendar   A list of time intervals during which activities can be worked and/or resources used. Typical data includes working days/non working days, start and finish times for shifts. Each activity and/or resource will have a calendar attached to it.

Component   The word component may refer to an element in a 3D model, or it may also indicate an individual part of a BIM, e.g., the mechanical model or the structural model. It will be necessary to derive the specific meaning from the context.

Constructability	This term refers to the analysis of the ability to construct. In the early phases of a project, such analysis can provide valuable input for the practicality of the assembly process of a project. Constructability analysis can take place on various scales, depending on the phase of the project and the level of detail available about the construction process.
Construction Project	This is synonymous with building project, and it refers to the planning, preparation, and construction of a building. Projects typically are performed by individuals who use methods to achieve certain results.

D

Duration	The estimated or actual time required for the completion of an activity, or a group of activities, based upon a particular resource allocation and method of working.
Dimensionality – 2D, 3D, 4D, 5D	The common convention referring to the "geometric dimensions of some physical or abstract system" (Webster's New World College Dictionary), where 2D space is a flat plane; 3D space is three-dimensional space, e.g., length, width, and height; 4D space adds time as a dimension; 5D space will generally refer to cost.

E

Earliest Finish	The earliest time that an activity, or a group of activities, can finish within the constraints, resources and logic of the network.
Earliest Start	The earliest time that an activity, or a group of activities, can start within the constraints, resources and logic of the network.

F

4D Planning and Scheduling   The integration of schedule and graphics to produce a time-based visualization of the development of a project. Predominantly carried out by linking Project Management Software with graphics/drawing software though integrated software is now available.

Field   The field is a term usually referring to the physical construction site when it is used in a discussion of construction topics.

G

Gantt Chart   A Gantt chart is a graphical representation of the duration of activities against the progression of time and is a particular type of Bar Chart though used as a synonym for bar charts in general.

H

Hazard   The potential to cause harm, including ill health and injury; damage to assets, products or the environment; production losses or increased liabilities.

I

Industry Foundation Classes   IFC means industry foundation class; it is a term coined by the International Alliance for Interoperability. The IFC is a standard file format for 3D models that will permit information to be exchanged among all models that can be translated into this file format. It is an attempt to bring about standards for a common language between the various model authoring and analyzing software tools.

Interoperability

Interoperability refers to the ability of different file formats to be integrated with one another and transfer relevant information among one another.

J

Job Safety Analysis (JSA)

A procedure used to review job methods and uncover hazards that may have been overlooked in the layout or design of the equipment, tools, processes or work areas; that may have developed after work started; and that may have resulted from changes in work procedures or personnel (ECI, 2013).

M

Milestone

A zero duration activity used to identify or highlight key points of Events in the project. Milestones are often used to identify the start or completion of sections of the project and therefore useful for Monitoring performance.

Monitoring

The recording, analysing and reporting of project performance and comparing it to the Baseline Schedule.

O

Ontology

In the context of computer and information sciences, an ontology defines a set of representational primitives with which to model a domain of knowledge or discourse. The representational primitives are typically classes (or sets), attributes (or properties), and relationships (or relations among class members). The definitions of the representational primitives include information about their meaning and constraints on their logically consistent application. In the context of database systems, ontology can be viewed as a level

of abstraction of data models, analogous to hierarchical and relational models, but intended for modeling knowledge about individuals, their attributes, and their relationships to other individuals.

Object-Based Model   The use of objects in 3D models renders them more usable and efficient in the BIM process. An object generally represents a physical entity (although nonphysical entities, e.g., events such as inspections, can also be represented by objects) in the project and is able to contain information relevant to the project. Objects are often composed of many parts that would be much more burdensome to the project model if treated as separate parts.

P

Planning   The process of preparing for the commitment of resources in the most effective fashion. It aims to produce a workable schedule that will achieve project goals and serve as a standard against which actual progress can be measured. It defines: (1) What should be done (activities); (2) How should activities be performed (methods); (3) Who should perform each activity and with what means (resources); (4) When activities should be performed (sequence and timing)

Planner   A member of the project team responsible for planning, scheduling and monitoring the progress of the project. Often used as a synonym for Scheduler.

Progress   The measurement of the completeness of an activity or a group of activities or the project as a whole.

Project

A project may be defined as a temporary endeavour undertaken to create a unique product. Projects are performed by people, constrained by limited resources, planned, executed and controlled. A typical construction project includes construction work incorporating planning, design, management or any other works involved until the end of the construction phase – that is it includes construction, alteration, conversion, maintenance, fitting out, commissioning, renovation, repair, upkeep, redecoration, decommissioning, demolition or dismantling. A full definition is available in the Temporary and Mobile Construction Sites Directive (92/57/EEC) and relevant national legislation.

Project Schedule

The term Project Schedule may be interpreted in two ways. (A) The project activities and milestones and durations and planned sequence and timing; (B) The physical document (Bar Chart), for instance, that illustrates and communicates the aforementioned.

Parametric

A parametric object or component is an object (or component) that permits a choice of values for defined parameters. A parameter is a variable value (as in a mathematical equation) that, when it changes, gives a different but related characteristic to the original object. An example is a steel beam in a 3D model that can have the size of the beam as one of its parameters. This means that the specific beam in the model needs to have its size specified, and it will thus reflect its physical size and weight accurately in the model. The chosen values for the parameters generate parametric information. In the case of the steel beam, the size of the beam implies a variety of information that will be determined by its size, e.g., its width, thickness, total weight (resulting from the length), etc.

R

Reschedule

A calculation performed on the tasks and links to ensure that the project is completed in the minimum possible time within the logical and imposed constraints of the plan and any progress that might have been achieved.

Resource

Any goods or services required to complete the work of an activity. For example, labour, materials, plant and money.

Resource-Oriented Scheduling

The method of developing a schedule that determines the sequence and timing of activities based on the logical work process and the availability and limitation of resources. Generally, this involves estimating activity durations based on available resources and the introduction of Resource Links to stimulate the transfer of resources from one activity to another.

S

Schedule

(1) The timetable for a project. Showing how Activities and Milestones are planned to be carried out over a period of time;

(2) The physical document for communicating the Plan, especially timing and sequence.

T

Taxonomy

It is the practice and science of classification. A taxonomy, or taxonomic scheme, is a particular classification. Specifically A Data Taxonomy

is a defined classification of terms, organized hierarchically into any number of levels of category and sub-category as required, and to serve a given purpose.

V

Virtual Construction

Virtual means not real, and this refers to the processes taking place in the computer. Virtual construction is a term used by CIFE and Graphisoft to describe the use of a 3D computer model to simulate not only the design of a structure but also its assembly, the construction process. Virtual construction will likely include the analysis of the construction schedule.

chapter 1

1.1 Basic Concepts

When we think of a process in any given domain, we generally figure it out as a series of actions that leads to a certain outcome. For a construction project, when it comes the execution phase, the process of site planning and activity scheduling can be assumed as what the planner does in the search for the solution of a complex and faceted problem whose variables are numerous (building design, site characteristics, boundary conditions, technology, materials, labor, etc.) and most of the times, if not unknow, at least highly uncertain.

As a matter of facts, the planner deals with this problem (process) on the base of his experience or, in other words, of the knowledge he owns about the problem. As simplistic as this statement could seem, gives us the reason to look more closely to what "knowledge" means and which role does it play for our purpose.

Knowledge is understanding of a subject area (Durkin 1994). It comprises concepts and facts about that subject area, including relations among them and mechanisms for how to combine them to solve problems in that area. Now, this roughly corresponds to what we want to achieve by mean of a computer-aided system capable of simulating what an expert would do for a given problem, acting in a way that humans usually call "intelligent". This is why "artificial intelligence" (AI) is used to indicate the branch of computer science that attempts to approximate the results of human reasoning by organizing and manipulating factual and heuristic knowledge. (Bandwidth Market 2008).

In AI, knowledge is pivotal: it is *stored*, encoded in a suitable format, into computer memory in order to be *retrieved* when it is needed for *reasoning* or, otherwise, to obtain conclusions, inferences and explanations applying problem-solving strategies. Taking a step back, none of these actions could be done before knowledge *acquisition* which consist, namely in gathering, organizing, and structuring knowledge about a topic, a domain, or a problem area, in order to prepare it for putting into the system.

This is where humans decide on how to represent knowledge inside computers. For this purpose, in AI, a number of different representations, or models, have been developed to be

suitable for various types of human knowledge. Leaving their thorough discussion to the related literature, hereunder are discussed the main aspects of the one knowledge representation method on which this work is based: *ontologies*.

Again, many definitions can be used to describe an ontology. To begin with, it comes with a domain. More precisely, it is the study of the *categories* of things that exist or may exist in some domain (Sowa 2000) in order to explain the types of things in that domain. A domain ontology is basically about its terminology (domain vocabulary), all essential concepts in the domain, their classification, their t*axonomy*, their relations (including all important hierarchies and constraints), and domain axioms. A taxonomy (or concept hierarchy) is a hierarchical categorization or classification of entities and terms defined within a domain. A good taxonomy should separate its corresponding entities into mutually exclusive, unambiguous groups and subgroups. The vocabulary and the taxonomy of an ontology together provide a conceptual framework for discussion, analysis, or information retrieval in a domain.

Moreover, the ontology is the fundamental part of the knowledge, and all other knowledge should rely on it and refer to it. In this regard, it can be always associated with its underlying data structure, being both:

• a representation vocabulary, often specialized to some domain or subject matter;
• a body of knowledge describing some particular domain, using a representation vocabulary.

In addition, still at an abstract level, "ontology is a specification of a conceptualization" (Gruber 1993) or, in other words, a simplified view of the world. With *conceptualization* of an area of interest are comprised the concepts, objects, and other entities that are assumed to exist in that area, along with the relationships among them. S*pecification* is then related to a formal and declarative representation.

However, even though an ontology formally represents a certain knowledge domain, implying that it should be a *machine-readable*, yet it is not "active"; it cannot be run as a program. It represents declaratively some knowledge to be used by programs. With this in mind, it goes without saying that the knowledge an ontology captures has not to be subjective; it must be consensual, shared and, to some extent, accepted by a group or a community. Hence an ontology conveys a shared understanding of a domain that is agreed between a number of individuals or agents.

The major purpose of ontologies is therefore not to serve just as as vocabularies and taxonomies; it is *knowledge sharing* and *knowledge reuse* by applications. They don't define just a series of terms; they provide *logical statements* that describe what the terms are, how they

➜

1.1 Musician ontology visualized as a semantic network (from Gasevic et. Al. 2009)

are related to each other, and how they can or cannot be related to each other. They also specify *rules* for combining the terms and their relations to define extensions to the vocabulary.

Before introducing further elements, we will touch base with the above-mentioned concepts by mean of the "musician" example reported in (Gasevic et al. 2009). Let's think of a musician; the essential knowledge about the notion of a musician could be simplified with: *musician*, *instrument*, *albums* (as a product of his/her work), *events* in which he/she participate and even *admirers*, who usually are supposed to attend those events. Broadly speaking, there could be a variety of relations that could be established among these concepts. For the sake of this example we will reduce to just few of those that we consider relevant to represent the knowledge regarding our subject. Namely that: each musician *plays* some instrument and, as main part of its activity he/she could play at concert as well as *record* albums; moreover, admirers *attend* to such events. By mean of these statements we just performed a basic conceptualization of the musician ontology. As a result of this activity, we could in fact informally diagram this ontology as the semantic network shown as follow.

Obviously, the representation suffers from many deficiencies. It is not a formal specification, i.e., it is not expressed in any formal language. Details related to the concepts and relations are missing (e.g. musicians have names, albums have titles, durations and so forth). Likewise, nothing in this semantic network shows explicitly that the musician is the *author* of an album that he/she records (note that recording engineers in music studios can also be said to record albums, but they are usually not the authors). Nevertheless, this first rough passage, though organizing concepts and relations at a very abstract level, could be considered as a valid starting point in the representation of the knowledge, of the "world", related to a musician. Now, established that an ontology is what we need to represent knowledge in such a way that a machine can understand it and to work with it in for some purpose, let's see from where to start in building it.

The effort required is such that *ontological engineering* has been identified as a proper area of expertise which comprises a set of design principles, development processes and activities,

supporting technologies, and systematic methodologies that facilitate ontology development and use throughout its life cycle design, implementation, evaluation, validation, maintenance, deployment, mapping, integration, sharing, and reuse. Knowledge engineering for an intelligent system should always include ontological engineering, which implies using those specific development tools and methodologies.

As one would expect, many methodologies or approaches have been reported in literature. Below are listed the main aspects that can be pointed out, especially regarding their most recent developments:

- most ontology development methodologies that have been proposed focus on *building* ontologies;

- some other methodologies also include methods for *merging, reengineering, maintaining*, and *evolving* ontologies;

- yet other methodologies build on general software development processes and practices and apply them to ontology development;

- there are also methodologies that exploit the idea of r*eusing existing ontological knowledge* in building new ontologies;

- some of the more recently proposed methodologies are based on the idea of using *publicly available community-based knowledge* to simplify and speed-up development of ontologies.

Nonetheless, is important to stress that in deciding what we are going to use the ontology for, and how detailed or general the ontology is going to be, the following three statements can be applied as rules of thumb:

1. *There is no one correct way to model a domain — there are always viable alternatives. The best solution almost always depends on the application that you have in mind and the extensions that you anticipate.*

2. *Ontology development is necessarily an iterative process. During the ontology development we should frequently evaluate and debug it by using it in applications or problem-solving methods or by discussing it with experts in the field, or both. As a result, we will almost certainly need to revise the initial ontology.*

3. *Concepts in the ontology should be close to objects (physical or logical) and relationships in your domain of interest. These are most likely to be nouns (objects) or verbs (relationships) in sentences that describe your domain.*

To clarify what such methodologies consist of, below we provide for example the steps of the ontology-building process proposed in Noy and McGuinness (2001):

- *Determine the domain and scope of the ontology* – This should help create a clear vision of the ontology's coverage, its intended use, the types of questions the information in the ontology should provide answers to, and maintenance guidelines. That is, answer several basic questions:
  - What is the domain that the ontology will cover?
  - For what we are going to use the ontology?
  - For what types of questions, the information in the ontology should provide answers?
  - Who will use and maintain the ontology?

The answers to these questions may change during the ontology-design process, but at any given time they help limit the scope of the model.

- *Consider reusing existing ontologies* – It is almost always worth considering what someone else has done and checking if we can refine and extend existing sources for our particular domain and task. Reusing existing ontologies may be a requirement if our system needs to interact with other applications that have already committed to particular ontologies.
- *Enumerate important terms in the ontology* – This is where building the terminology starts. It is useful to write down a list of all terms we would like either to make statements about or to explain to a user. What are the terms we would like to talk about? What properties do those terms have? What would we like to say about those terms?
- *Define the classes and the class hierarchy* – This step, closely intertwined with the next one, can be performed top-down (identifying the most general concepts and classes first), bottom-up (identifying the most specific ones first), middle-out (starting from some important middle-layer concepts and expanding the hierarchy in both directions), or using a combination of these approaches.
- *Define the properties (slots) of classes* – Describe the internal structure of concepts by explicating their extrinsic properties (e.g., *name*, *duration*, and *use*), intrinsic properties (e.g., *weight*), parts, and relations to other classes and individuals in those classes. In fact, the classes alone will not provide enough information since their mutual relations are defined to represent the internal structure of concepts.
- *Define the facets of the slots* – These are things such as the slot value type, the allowed values (domain and range), the number of values (cardinality), and other features of the values that the slot can take.
- *Create instances* – This includes filling in the slot values for each instance created.

In practice, the conflicts which rise and the number of details which must be taken into account, make anything but simple to implement the outlined steps.

Another more comprehensive methodology is the *Methontology framework* (Fernández-López et al. 1999). It starts from the assumption that the entire ontology lifecycle has to be defined and standardized through the following phases:

- *Specification* – Identification of the ontology's terminology, primary objective, purpose, granularity level, and scope;
- *Conceptualization* – Organizing and structuring in a semiformal way the knowledge acquired during the specification phase, using a set of intermediate representations that both domain experts and ontologists can understand (thus bridging the gap between their mindsets);
- *Implementation* – Using an ontology development environment to formally represent and implement the products of the above two phases, namely concepts, hierarchies, relations, and models.

In addition, Methontology covers processes that run in parallel throughout the ontology life cycle. In particular can be cited: quality assurance, integration, evaluation, maintenance, documentation, and configuration management. Furthermore, it specifies in detail the techniques used in each activity, the products that each activity outputs, and how they have to be evaluated.

Also, knowledge acquisition is recognized as a crucial activity in the framework that influences the specification and conceptualization phases and count for the long shoulder-to-shoulder work with domain experts. It comprises the use of various knowledge acquisition techniques to create a preliminary version of the ontology specification, as well as all of the intermediate representations resulting from the conceptualization phase.

The basic concepts provided above should have put us in the condition to figure out what an ontology could be, for which purposes could be useful and which could be the first steps in its development. Next paragraphs will complete our overview on ontologies touching base with three fundamental aspects related to their: *representation languages*; *development environments* and *visualization*.

1.2 0ntology representation languages

When we know how to design and build an ontology, there is the need to think about deploying it. But what does it mean? In other words, the ontology itself has to be represented in a way that is suitable for the computer and system applications intended to work with it. This paragraph looks at the various machine-readable standards, from XML to RDFS, leading up to OWL that is the language used for the proposed ontology, presented in Chapter 3.

There are several ontology representation languages. Some of them were developed at the beginning of the 1990s within the AI research branch. Others were presented in the late 1990s and later, resulting from the AI community and the World Wide Web Consortium (W3C). The early ontology representation languages belong to the pre-XML era, whereas the later ones are XML-based. Most of the later ones were developed to support ontology representation on the Semantic Web, and hence they are also called "Semantic Web languages." Other common names for them are "Web-based ontology languages" and "ontology markup languages".

In the following list some of the best-known examples of ontology representation languages are provided along with specific references to their more detailed description:

- KIF (Genesereth and Fikes 1992), based on first-order logic;
- Ontolingua (Gruber 1992), built on top of KIF but including frame-based representation;
- Loom (MacGregor 1991), based on description logics.
- Among the widely used Web-based ontology languages can be found:
- SHOE (Luke and Heflin 2000), built as an extension of HTML;
- XOL (Karp et al. 1999), developed by the AI center of SRI International as an XML-ization of a small subset of primitives from the OKBC protocol called OKBC-Lite;
- RDF (Manola and Miller 2004), developed by the W3C as a semantic network-based language to describe Web resources;
- RDF Schema (Brickley and Guha 2004), also developed by the W3C, is an extension of RDF with frame-based primitives; the combination of both RDF and RDF Schema is known as RDF(S);
- OIL (Fensel et al. 2001), which is based on description logics and includes frame-based representation primitives;
- DAML+OIL (Horrocks and van Harmelen 2002) is the latest release of the earlier DAML (DARPA Agent Markup Language), created as the result of a joint effort of DAML and OIL developers to combine the expressiveness of the two languages;
- OWL, or Web Ontology Language (Smith et al. 2004), developed under the auspices of the W3C and evolved from DAML+OIL; OWL is currently the most popular ontology representation language.

The reason why during the years a number of ontology representation languages have been developed are clearly expressed in the following sentence by (Decker et. Al 2000): «Ideally, we would like a universal shared knowledge-representation language to support the Semantic Web, but for a variety of pragmatic and technological reasons, this is unachievable in practice. Instead, we will have to live with a multitude of metadata representations».

In this regard, there are several implications. For example, a system-application may already use another ontology, developed in a language other than that which the developers have chosen for building a new ontology. Many studies demonstrate that merging different ontologies, developed by using different ontological languages can require a lot of effort - two languages coming from different branches of the ontology-language research communities may not be compatible, and may require a strong manual adaptation in order to provide a satisfying interoperability.

However, developers may be constrained by the fact that development tools may support only a few languages. The situation is even worse for new applications where for design considerations is chosen a more appropriate language different from the one used by previous applications which though need to be consulted as well. On the other hand, an appropriate ontology representation language may facilitate the integration of an ontology into a new application. Moreover, a newer ontological representation language may just be more expressive than the other languages of choice, and translators to and from those other languages may already exist as well. We should note that the ontology community has already developed a number of such translators, although many of them suffer from partial loss of knowledge in the translation process.

1.3 Ontology development environments

Building ontologies is a complex and time-consuming activity, especially when developers have to implement them in an ontological language, without any supports in terms of software applications. To ease this activity, in the mid-1990s the first ontology building environments were created. They provided an interface to support developers in the main activities of the ontology development process, such as conceptualization, implementation, consistency checking, and documentation. In the last few years, a number of ontology tools have been created with different purposes.

In (Gomez-Pèrez 2002) a list of the main groups of those tools distinguished by function is presented:

- *Ontology development tools*. This group includes tools and integrated suites that can be used to build a new ontology from scratch. In addition to the common editing and browsing functions, these tools usually support ontology documentation, ontology export and import to/from different formats and ontology languages, ontology graphical editing, ontology library management, etc.
- *Ontology evaluation tools*. They are used to evaluate the content of ontologies and their related technologies. Ontology content evaluation addresses the issues related to the

integration and use of ontologies and ontology-based technologies in other information systems.

- *Ontology merge and alignment tools.* These tools are used to solve the problem of merging and aligning different ontologies in the same domain.
- *Ontology-based annotation tools.* With these tools users can insert instances of concepts and relations in ontologies and maintain (semi)automatically ontology-based markups in Web pages. Most of these tools have appeared recently, in the context of the Semantic Web.
- *Ontology querying tools and inference engines.* These allow querying ontologies easily and performing inferences with them. Normally, they are strongly related to the language used to implement ontologies.
- *Ontology learning tools.* They can derive ontologies (semi)automatically from natural language texts, as well as semi-structured sources and databases, by means of machine learning and natural language analysis techniques.

Some suites integrate tools from several of the aforementioned groups while some other tools provide just a limited set of functions. In this chapter, we focus only on the first four groups of ontology tools, comprehensively referring to them as **ontology development tools**.

Ontology tools' technology has improved enormously since the creation of the first environments. Considering the evolution of ontology development tools since they appeared in the mid-1990s, we can distinguish two groups:

1. Tools whose knowledge model maps directly to an ontology language. These tools are developed for a specific language.
2. Integrated suites that have an extensible architecture and whose knowledge model is usually independent of an ontology language. These tools provide the developers with a core set of ontological services and can be extended with other modules to provide more functions. All those are called *ontology development environments*.

Both of those groups could be intended as graphical ontology editors that help the developers to organize the conceptual structure of the ontology. Graphical ontology development environments integrate an ontology editor with other tools and usually support multiple ontology representation languages. They cover the entire ontology development process and the subsequent use of the ontology.

The tools we are going to present below are among the most adopted and useful and stand in an advanced development stage, since their corporate users usually belong to R&D departments of companies and universities.

Protégé   (Noy et al. 2000) It is developed by Stanford Medical Informatics (SMI) group at Stanford University. The first *Protégé* tool was created in 1987; its main aim was to simplify the knowledge acquisition process for expert systems.The core of *Protégé* is its ontology editor, which can be extended with plug-ins that add more functions to the environment. Most of these plug-ins are available in the *Protégé Plug-in Library*, where contributions from many different research groups can be found. We can identify three groups of plug-ins that can be developed for *Protégé*:

> **(1) Tab plug-ins**. These are the most common types in *Protégé* and provide functions that are not covered by the standard distribution of the ontology editor. To perform their task, tab plug-ins extend the ontology editor with an additional tab so that users can access its functions from it. The following functions are covered by some of the plug-ins available: ontology graphical visualization (Jambalaya tab and OntoViz tab), ontology merge and versioning (PROMPT tab), management of large on-line knowledge sources (UMLS and WordNet tabs), OKBC ontology access (OKBC tab), constraint building and execution (PAL tab), and inference engines using Jess (Friedman-Hill, 2003), Prolog, FLogic, FaCT, and Algernon14 (Jess, Prolog, FLORA, OIL, and Algernon tabs respectively).

> **(2) Slot widgets.** These are used to display and edit slot values without the default display and edit facilities. There are also slot widgets for displaying images, video and audio, and for managing dates, for measurement units, for swapping values between slots, etc.

> **(3) Backends.** These enable users to export and import ontologies in different formats: RDF Schema, XML, XML Schema, etc. In terms of ontology editor, it browses and edits the ontology's class taxonomy using a tree structure, defines global slots, attaches slots to classes, creates class instances, etc. After ontologies have been developed, they can be exported and imported with some of the backends provided in the standard release or as plug-ins: RDF(S), XML, XML Schema, and XMI.

OntoLingua   (Farquhar et al., 1997) It was created in the mid-1990s by the Knowledge Systems Laboratory (KSL) at Stanford University. The main objective of this system was to ease the collaborative building of ontologies in the

Ontolingua language and to provide a repository of ontologies. It also supports a World Wide Web (WWW) interface and translation into different formats. OntoLingua is an ontology library and server that can be accessed with a traditional Web browser. By assembling and extending the ontologies obtained from the library and tools in OntoLingua, authorization can be provided.

WebOnto (Domingue, 1998) It was developed by the Knowledge Media Institute (KMi) at the Open University (United Kingdom). It was presented in 1997 as a Web application to browse and develop collaboratively OCML ontologies. It is freely available to edit and browse ontologies though users must request from the WebOnto developers a user account in order to have editing capabilities. The main features of WebOnto are the management of ontologies using a graphical interface, the automatic generation of instance-editing forms from class definitions, the inspection of elements taking into account the inheritance of properties, and consistency checking and support for collaborative work using broadcasting and receiving and making annotations.

OntoSaurus (Swartout et al., 1997) It was developed at the same time as the Ontolingua Server by the Information Sciences Institute (ISI) at the University of Southern California. consists of two modules: the ontology server, which uses the LOOM KR system and provides concept classification and instance matching functions; and the Web ontology editor and browser, which generates dynamically HTML pages (including images and other text documentations) from the LOOM ontologies stored in the ontology server.

OntoEdit (Sure et al., 2002a) It is an ontology engineering environment initially created by the Institute AIFB at the University of Karlsruhe and currently commercialized by Ontoprise GmbH. Like *Protégé*, its ontology editor comprises the core modules of the application and can be extended with plug-ins. In addition, the knowledge model is based on frames. It can represent concepts and their attributes, binary relations between concepts, formal axioms, and instances. With the OntoEdit's ontology editor, the ontology concept taxonomy can be browsed and edited and relations, instances, etc., can also be edited by means of a tree structure, as well as in *Protégé*.

KAON (Maedche et al., 2003) It is a tool suite developed by the Institutes AIFB and FZI at the University of Karlsruhe. In terms of architecture, the KAON tool suite is being developed with a flexible architecture. It is organized in three

layers: (a) backends for ontology storage, (b) middleware for the services offered by the tool suite, and (c) client applications that access the ontology middleware and provide end-user applications. The KAON tool suite is completely implemented in Java and can be extended with plug-ins that provide to developer a flexible tool. The ontology editor is called OI-Modeler and is composed by a graphical user interface for managing the ontology.

1.4 Ontology visualization

When an ontology is developed, its visualization is very useful and helps working with the ontology itself. Different visualizations for ontologies have been presented in the last couple of years. While some of them are implemented as standalone applications, most visualizations are provided as plugins for ontology editors like *Protégé*. With reference to (Lohmann, Negru, Haag, & Ertl, 2014) in this paragraph an overview of the main approaches for ontology visualization is provided.

A lot of approaches use *graphs* as the method to visualize ontologies since they seem to be the natural way to represent the structure of the concepts and relationships in a domain of knowledge. The graphs are, very often, rendered in force-directed or sometimes in hierarchical layouts, resulting in a very comprehensive visualization. However, only few visualizations methods show complete ontologies, while most focus just on certain aspects. *OWLViz, OntoTrack and KCViz* depict only the class hierarchy of ontologies. Likewise, *GLOW* represents the class hierarchy but uses a radial tree layout and hierarchical edge bundles to display additional property relations.

There are few applications able to provide the developer with a more comprehensive graph visualizations that represent all key elements of ontologies and not only the class hierarchy and their relationships. For instance, *TGViz* and *NavigOWL* use very simple graph visualizations where all nodes and links look the same except for their color. This is different in *GrOWL* and *SOVA*, which define more elaborated notations using different symbols, colors, and node shapes.

In addition, some applications have been presented able to depict a 3D graph visualization of the ontologies. In this regard, we can cite OntoSphere, or tools that use hyperbolic trees to visualize ontologies, such as *OntoRama* and *Ontobroker*.

Differently from all those applications that use common node-link diagrams to represent ontologies; there are various applications that apply other diagram typologies to visualize ontologies.

For example, *Jambalaya* and *OWLVisMod* use tree maps to depict the class hierarchy of ontologies. Jambalaya also provides a nested graph visualization *called SHriMP* that allows to split up the class hierarchy into different views. *CropCircles* is a related visualization technique that visualizes the class hierarchy of ontologies with the goal to support the identification of "undermodeled" ontology parts. All these approaches visualize once again mainly the class hierarchy, without considering other property relations.

Cluster Maps use a visualization technique that is based on nested circles and has also been successfully applied to ontologies. Instead of showing the class hierarchy, it visualizes individuals grouped by the classes they are instances of. Each individual is represented by a small circle that is shown inside a larger circle representing the class. Similar techniques are used in *VisCover* and *OOBIAN Insight* that additionally provide several interactive filtering capabilities. While offering appealing visualizations, these tools show only a selection of classes along with their instances but do not provide complete visualizations of ontologies.

A powerful type of diagram related to OWL and often used to visualize ontologies is the class diagram of the Unified Modeling Language (UML). A common issue related to this kind of visualization is the need for the understanding of UML class diagrams which is generally familiar to most users with an IT background but not for those coming from other domains, who could find difficult to interpret UML diagrams correctly.

Since any OWL ontology can be represented as RDF graph, it can also be visualized using the common RDF notation. In this context, a very powerful visualization tool is VOWL which provides an intuitive visualization that is also understandable to users less familiar with ontologies. It uses a graph visualization and is also available as a plug-in for *Protégé*. A detailed description of this tool is later provided since it has been used to visualize the ontologies developed and presented in this monography.

Having a look to the previous content, they can be synthetized the following characteristics related to the visualization techniques:

- many visualization applications utilize diagram for ontology visualization (graph visualization, treemap, UML), they are in 2D or 3D and focus on specific aspects of ontologies such as the ontology three.
- other works implement a stepwise approach of ontology exploration, where only a root class is shown at the beginning and the user has to navigate through the visualization.

chapter 2

2.1 Expert System methodologies

The success of any Expert System (ES) relies mainly on the ability to formalize and represent the knowledge within a discipline. Often this knowledge consists in a collection of subjective, incomplete, ill-defined, and informal information. Indeed, a side benefit of expert system development is the formal organization of information that was previously unexpressed. ESs are codified as a branch of applied Artificial Intelligence (AI) and their basic concept is to simulate the action of an expert to solve a specific problem by using computer aided technologies. As before mentioned, expert systems are used to address a number of problem-solving activities. In many cases, it has been demonstrated that if a tool is fine to solve a problem related to an activity (e.g., design), the same tool is inadequate to solve a problem for a different activity (e.g., planning). To address this situation, the AI research branch, including researchers and software developers and vendors, turned their attention to developing the so called "domain-specific tools". A domain-specific tool is defined as a tool designed to be used to develop an expert system for a particular problem-solving activity. In the table below the common typologies of problems addressed by expert system developers are listed, while their detailed description is later reported in order to provide the reader of this monography with a complete overview. The proposed list is adapted from (Hayes-Roth et al., 1983).

➡️
Table 1
List of problems solved by Expert Systems

Problem to solve	Problem description
Design	Configuring objects with constraints
Simulation	Modeling system components and their interaction
Planning	Selecting and sequencing activities according to a set of constraints to achieve a predefined goal
Scheduling	Assigning times, costs and resources to the set of activities in a plan
Selection	Selecting the best choice from a number of possibilities which respond to a number of predefined objectives
Prediction	Identifying solution to solve system malfunctions
Interpretation	Inferring likely consequences of given situations
Instruction	Diagnosing, debugging, and repairing student behaviour
Diagnosis	Inferring system malfunctions from observables
Monitoring	Comparing observations to expectations
Control	Governing system behaviour to meet specifications

The following section describe some of the most important and currently addressed problems that an expert system is designed to solve according to the previous list, referring to real cases of ESs already developed.

- **Design**
  DESIGNER (MacCallum, 1982) assists in general design processes. A key characteristic in the design of engineering systems is complexity: a designer's task to specify the characteristics of a system, given a set of required functional objectives to be achieved in a given environment. The system works with basic concepts in the design process and applies them to a generic task in order to produce an acceptable design. The system was developed at the University of Strathclyde, Great Britain.

  GOES (Langley et al., 1995) Graphics-Oriented Expert Shell, is a shell built inside a standard CAD environment. It is intended to associate logical engineering design procedures with graphical representations. It is particularly useful for managing and documenting large-scale control and automation design activities. GOES operates as an intelligent CASE tool, and eases the task of identifying, assembling, and parametrizing function block subroutines of distributed control systems. It was developed at CMS-CAD Inc., Montreal, Que., Canada.

- **Simulation**
  ORBIS (Evans and Sanders, 1994) Object-oriented Rule Base Interactive System, is an expert system simulation tool designed to be used in a variety of simulation environments; for example: stand-alone interactive simulations, batch runs for collecting Monte Carlo statistics, and real-time, man-in-the-loop simulations. The ES tool contains objects, data, algorithms, and rule sets specific to the simulation which serve to generate the desired behavior of the simulation. An important feature of the ORBIS simulation development software is that an expert system, implemented as rule sets, controls simulated objects.

- **Planning**
  GHOST (Navinchandra et al., 1988) is a general-purpose planning system in the area of construction. It reasons about an object's attributes and relationships between objects to define project activities. GHOST starts with a high-level set of tasks and

refines them into subnetworks of more detailed tasks. The system combines object-oriented programming with rulesets, and employs a blackboard structure.

KBLPS (Knowledge-Based Logistics Planning Shell) assists in the design of an expert system for planning allocation and transportation resource applications. It is composed by planning algorithms for over-constrained distribution problems, and a decision-centered graphical user interface. It provides planners with a "what-if" ability to see the impact on changed information on the expected results. It can cycle through scenarios, breaking resource allocations until the most feasible recommendation is developed.

• **Instruction** A software application was developed by (Reinhardt and Schewe, 1995) at Wurzburg University, Germany, for building intelligent tutoring systems to train medical students in diagnosing patient cases. It uses case data and problem-solving knowledge to handle classification problems. The principal idea of the system is to present case data and to control the student's actions by comparing them to the underlying expert system.

• **Interpretation** **SSI** (Arai et al., 1992), developed at Osaka University, Osaka Japan, is a shell for the development of signal interpretation expert systems. The shell is a product of the design of two expert systems for speech signal processing, where the analysis of the two systems revealed common functions and modules applicable to a wide range of signal interpretation problems.

• **Monitoring** **PREMON** (Predictive Monitoring system), developed by (Doyle et al., 1987) at NASA Ames Research Center, uses a device to perform real-time monitoring. PREMON has been developewd to perform three interacting activities: (1) causal simulation to generate predictions about the behavior of a physical system; (2) sensor planning to assess the importance of a device's behavior and allocate sensor resources appropriately; and (3) sensor interpretation to verify expected sensor values against actual sensor readings and raise alarms when necessary.

In order to provide the reader with an overview of ESs, a categorization of expert system typologies is presented:

1. **Rule-based systems**

   A rule-based ES contains information obtained from a human expert, and represents that information in the form of rules, such as IF–THEN. Rules are used to operate on data to inference and reach appropriate conclusion. These inferences consist essentially in a computer program that provides a methodology for reasoning about information in the rule base. They work on a database.

2. **Knowledge-based systems**

   They are also defined 'human-centered'. They are attempts to understand and initiate human knowledge in computer system. Many of such applications exist in the field of medical treatment.

3. **Neural networks**

   It is a model that emulates a biological neural network. This concept is used to implement software simulations for the massively parallel processes that involve processing elements interconnected in network architecture.

4. **Fuzzy expert systems**

   Fuzzy ESs use the method of fuzzy logic, which deals with uncertainties. This model, which uses the mathematical theory of fuzzy sets, simulates the process of normal human reasoning by allowing the computer to behave less precisely and logically than conventional computers. This approach is adopted to adhere more closely to decision-making processes since they are not always black and white, true or false; but they often involve gray areas.

5. **Case-based expert systems**

   Their basic idea is to adapt solutions viable for previous problems and use them to solve new problems. In such systems, descriptions of past experience of human specialists, depicted as cases, are stored in a database for their later retrieval when the user encounters a new case with similar parameters. Therefore, the system searches for stored cases with similar problem characteristics, finds the closest fit, and applies the solutions of the old case to the current one.

6. **Ontology-based expert systems**

   They are used to develop a systematic analysis of knowledge within a domain of interest. Therefore, by using ontology-based modelling they discretize the domain knowledge and formally describe a given problem. Then, by using reasoning mechanisms (rule-based) they operate on such knowledge extracting a solution. This is possible transferring the knowledge-base in a machine-readable language.

Many of the methods mentioned are product oriented, in the sense that the ES's activity is dedicated mainly to the knowledge base, and to some extent to the inference engine and user interface.

2.2 Expert Systems for construction planning

In the construction sector the overall project performance is strongly dependent on the project management activity. In this regard, process planning is well-known to play a crucial role and, despite the long-lasting efforts, most of the related issues still need to be fully addressed. In fact, project control and monitoring is based on a specific project plan determined considering beforehand numerous constraints such as resource availability, completion deadlines for tasks and budget limitations. Cost increases or delays can easily result from poor estimates, schedules or decisions related to tasks decomposition and choice of construction technologies. The planner, in turn, generally identifies constraints, evaluates interactions and solves the related conflicts on the base of the experience he acquired from previous projects. In this context, few process planning aids exists, and the available tools are rather analysis tool of existing plans than tools for plan formation. The main reason of the inadequacy of planning tools in addressing this multi-variable problem (resources, tasks, technologies, budget, schedule, etc) is due to the programming methodology used in developing the tools themselves. Algorithmic or procedural programming implemented in commercial construction and project management solutions cannot, in fact, conveniently represent, formalize or leverage acquired expertise. Moreover, they don't provide for an integrated process planning environment, vanishing the full potential of other computer-aided systems such as CAD for design or CAM for manufacturing. Integrated tools for comprehensive process planning could allow since the design stage considerations related to manufacturability or constructability, fostering the achievement of better products.

Therefore, even nowadays, despite the huge steps moved forward in terms of available technologies and computational capabilities, is still possible to refer to the issues raised decades ago from (Zozaya-Gorostiza, 1989), to briefly summarize the main motivations for developing computer tools for process planning in construction:

• process planning is crucial in design and project management;
• process planning is a difficult, knowledge-intensive, challenging process;
• there are virtually no integrated computer tools that assist during the complete process planning and project management cycle; and
• there are similarities in process planning across different domains.

In addition to the mentioned motivations and recalling the elements provided in the previous paragraph, also the reasons presented by (Mohan, 1990) could still be considered relevant in explaining why such automated systems for construction planning cannot rely on algorithmic or procedural programming, hence needing to be developed as Expert Systems. In fact, unlike for manufacturing, ready algorithmic solution cannot address day-to-day construction problems namely because:

- construction is nonrepetitive in nature, being each project different in design, layout, materials, technologies;
- uncertainties of construction environments in respect of labor and equipment productivity, market forces, resource and materials availability, regulatory agencies' influences, weather conditions;
- in many construction situations, there is not enough time to make a detailed analysis of all the influencing factor so that decisions are often made on the spot in order not to interrupt the construction process, distancing from the plan;
- many construction decisions like safety management, bid evaluation and decision to bid, risk management are qualitative and subjective in nature, needing heuristic approaches;
- on a construction project, all the necessary information on a subject is almost never completely known and the decision have to be taken on the basis of incomplete information.

Now, for the stressed motivations, ESs application to process planning and scheduling is certainly not a brand-new topic in the AEC industry, though not fully explored. Nonetheless, before discussing the existing cases of ESs developed for this purpose in the construction field, we provide below a general overview on their functioning, based on the discussion and classification adopted in (Zozaya-Gorostiza, 1989).

Process planning can be described at the core as a problem that involves identifying a set of actions that will achieve a specific goal. In order to address practical process planning and scheduling applications with automated systems, considerable domain-specific knowledge must be taken into account. In this regard three approaches could be considered: (1) classical AI plan formulation systems; (2) optimization models for scheduling; and (3) knowledge-based aids to planning tasks.

1. **AI plan formulation systems**

   When dealing with AI-aided plan formulation systems a handy classification can be set on two criteria: *type of plans* produced and *system's architecture*. According to the first criterion, planners can be divided into *linear*, producing sequences of ordered actions,

and *nonlinear*, producing networks of partially ordered actions. The second criterion instead distinguishs between different level of systems structure complexity, from *simple* architecture to ones in which actions are organized in layers or regions of a *blackboard*. In this regard, the features and limitations of the various planners' typologies can be listed as follow:

• *Linear planners:* the solution search space can be represented as a network of nodes, where each node represent a possible complete state in the problem and nodes are connected by *actions* or operators that transform one state into another. Given the operators, the initial and the desired state, linear planners are conceived to perform a heuristic search of this solution space in order to find a sequence of operators that will transform the initial situation into the desired situation. However, since the number of operators applicable to any given model might be quite large, the state tree they need to search for the final path, or in other words for the plan or sequence of operators that leads from the initial state to the desired goal, is in effect undesirably large and computationally intractable even for relatively simple problems. For this reason, several enhancements of the early formulation have been proposed. Among those STRIPS (STanford Research Institute Problem Solver) was designed to generate linear plans by reasoning at any point in the planning process about the differences between the desired and the current state in order to choose the better following operator. Furthermore, its early version was later evolved into ABSTRIPS with the introduction of a superior approach that considered the distinction of different level of abstraction for the problem world, for which searching subsequent solutions (plans), increasing the system's efficiency.

• *Nonlinear planners:* these systems assume the modification of the concept of a plan from a linear sequence of actions into a partial order of steps whose sequence of execution doesn't need to be necessarily fully specified. Such planning systems uses what is called a *procedural net* where, basically, each node (action) has an associated list of *preconditions* that must be true to execute the action and a list of *effects* that are added to or deleted from the world model when the action is executed. Among systems based on this functioning was developed NOAH (Network Of Action Hierarchies) that carries out the planning process first creating a small procedural net of actions at an abstract level and then repeatedly expanding it until the plan has been defined to the most detailed level in terms of *primitives* (simple operations). Another nonlinear planner that developed the first attempted systems is DEVISER that was the first to consider time constraint in the generation of plans, specifying when sets of goals should be satisfied and how long their conditions should be preserved. To represent this constraint, nodes of the network are

distinguished between *activities*, that can be either "actions", "events" or "inferences" and *scheduled events* whose occurrence is fixed in time regardless of the structure of the plan. The significant enhancement achieved with this approach is related to the fact that during the planning process all conditions of each activity are satisfied throughout the duration of the activity, being the system capable of backtracking when the earlier-start-time of an activity is greater than its latest-start-time. Nonetheless, though generating feasible plans, it does not comprise mechanisms for relaxing constraints if each of them cannot be satisfied.

- *Meta-planners:* these systems are conceived with a different approach and, rather than with the successive expansion of plan *actions*, are concerned with the order of execution of the different *planning operations*. Planning operators and data structures are separated into different layers so that operators of a given layer control the execution of operators of the layer immediately below. Operators of the upper layers are distinct and more general than operators in the lower levels, and communication between consecutive layers is carried out through a message interface. Meta-planners generate plans by dividing the planning problem into subproblems and analyzing the interactions among the subproblems.

- *Blackboard planners:* their functioning is based on the same principles of blackboard problem-solving frameworks for multi-task planning problems. In particular, multi-task planning involves selecting and ordering several plan tasks from a list of desired tasks, being the problem in answering "which of its potential actions should an AI system perform at each point int the problem-solving process?"

  In addition to this brief discussion of the main features of the classical AI planning systems is worthwhile to notice that they are capable of identifying plans to achieve goals only in highly restricted planning domains.

2. Optimization methods for planning and scheduling

Once here, it would be simplistic to reduce the planners' tasks to the generation of a network of activities to be executed in respect of their precedence and mutual constraints. Additional question must be answered before a plan is suitable to pass to the execution phase, such:

- *what* type of resources (labor, equipment and materials) should be assigned to the activities?
- *how* much of these resources should be used by each activity?
- *when* should each activity start?

The set of the related decisions made by the planner are reflected in a *project schedule* that serves to control and monitor the execution of the project. In this regard, numerous algorithmic procedures can aid in selecting the optimal answers to the previous questions. Since the introduction of the early scheduling models such as the *Critical Path Method* (CPM) and the *Project Evaluation and Review Technique* (PERT), more sophisticated models have been proposed for the computation of project schedules and can be divided into two categories, based on problem assumptions and purposes:

- *deterministic*, which assume activity durations are fixed or expressed as a function of the cost related to their completion, are usually applied to compute schedules that minimize total completion time or total budget;
- *probabilistic*, which assume activity durations are stochastic variables with probability distributions, are usually adopted to estimate total completion time or to simulate the execution of a project.

Basically, deterministic scheduling models can be considered as descendants of the CPM with the consideration of uncertainties, while probabilistic ones are extensions of the PERT model. To touch base with the main principles behind project schedule optimization, and so providing a beneficial background to the topics discussed later in this monography, a brief review of the main characteristics of deterministic scheduling models, in consideration of the assumptions related to network topology, resource consumption profiles, activity durations and costs, is reported as follow:

- *Models with fixed activity durations*: assume known activity duration, unlimited resources, and fixed network topology.
- *Models for time-cost trade-offs*: assume activity durations expressed as function of activity costs and fixed network topology; allow for constraints on the total completion time or on the total budget; don't impose limits on the resource consumption profiles.
- *Models with resource considerations*: used to schedule projects with limited resources or when peak resource requirements must be reduced; assume fixed network topology.
- *Models for combined planning and scheduling*: unify within the same framework both planning and scheduling process; they mark an essential difference from the others, allowing the change of the network topology.

Is important to notice that despite the distinction, the cited typologies are not independent, with models that can be considered as extensions or specializations of others.

3. Knowledge-based systems for construction and manufacturing planning

Construction and manufacturing planning processes present at different levels of detail and abstraction very close issues and can be considered as the same topic for the purposes of this review. Below the main features of construction and manufacturing planning and scheduling process are presented, grouped according to five interdependent elements or steps, along with some of the models and tools developed to aid the planner in achieving the desired goals:

- *Recognition of design features and elements:* this involves the definition and classification of relevant elements or features in the final design. A model of the design problem must contain a taxonomy to permit subsequent analysis of the design features and elements.

- *Definition of work tasks and precedence relationships:* the importance of defining work task and precedence relationships in the planning process is well-known in construction and is based on the generation of appropriate work breakdown structures (WBS). In this regard, we report here two main approaches.

  The first uses a hierarchical decomposition of project tasks along three perspectives: physical (major end items, systems and components); *organizational* (related to work tasks and to the part of the organization responsible for the task); *resource* (groups work tasks with respect to the type of resources they use). *Work packages* result then by grouping simple operations on the base of the project's decomposition using these perspectives and represent meaningful elements of the project for scheduling and monitoring purposes.

  The second approach (Baracco) aims to integrate duration and cost estimation in the model comprising *design* and *site* information. Elements-of-work related to each design element are identified with a unique code by the planner as activity performed to construct that specific design element. Project activities are created aggregating elements-of-work and can be used to form a network and a cost estimate based on the amount of work or quantity take-offs. The planner determines precedence among them.

  Furthermore, other systems have been studied regarding the problem of the automatic creation of the project activity networks. For instance, GHOST (Generator of Hierarchical schedules for cOnSTruction), is capable of finding precedencies among a set of given activities by using a set of critics. First it creates a network where all activities are performed in parallel; then applying critics to find physical precedencies among activities, leveraging knowledge about construction and physical

relationships, is able to order them in subsequent level of detail since a satisfying solution is obtained. Despite generating too much redundancy during hierarchical inheritance and though not free from various other limitations, GHOST paved the way for the further development of other expert systems for construction planning (PIPPA, OAR-PLAN, LIFT-2, LIFT-3, etc).

- *Choice of technologies, processes and resource:* For the selection of the technology to perform tasks, possible packages of labor and equipment available must be considered. Technology package or crew are usually chosen upon knowledge retrievable from estimating books, manuals or, more generally, from direct planner's experience. Then, the number of machines or crews of the selected kind to be assigned to the activities must be determined. In this regard the planner usually can rely on two approaches: first, he knows approximatively from previous projects the duration of the activity and adjust the number of the crews allocated to achieve this duration, not considering the interactions between the activities; alternatively, with what is known also as *assembly line balancing*, could be carried out a simulation of the manner in which different crews and machine types could be used to perform activities, trying to achieve a continuous use of the crews. Most of the developed aids for technology selection are process simulation models as for instance, CYCLONE (CYCLic Operations NEtwork). These tools allow for the fast evaluation of alternative technologies or methods, though the user has to manually define the operations network and its topology cannot be changed during the simulation. Therefore, they are considered more of *analysis* rather than synthesis aid for the planner.

- *Estimation of activity durations and costs:* Despite estimation of activity durations can be considered as one the most important tasks in the planning process, it is still carried out usually on the base of average productivities found in literature or in company records with eventual adjustments made from the planners on the base of their own experience. In this regard, expert systems have been developed to capture and leverage planners' expertise (e.g. MASON, for masonry construction activities duration estimation), using hierarchical, rule-based processes which starts from maximum crew productivity and considers various modifications, downtimes, specific characteristics of the site or of the job, to estimate an accurate duration of the activities.

- *Preparation, evaluation and maintenance of project schedules:* In light of what has been discussed, two major phases can be identified for process planning. The first preconstruction or premanufacturing phase involves some of the abovementioned planning activities, such as activity duration estimation, technology choice, and eventually leads to the preparation of a work schedule. However, when entering the later process phase,

other computer systems should be used to assist the planner in maintaining and evaluating project schedules, in order to avoid that schedule updating is reduced to a trivial archival record-keeping process, carried out by lower-level scheduling engineers, rather than a replanning process fundamental in the support of the decision-making during project execution. In this regard, to provide an example of an expert system developed for this purpose we can refer to PLATFORM. Given a project schedule and recording the actual duration of completed activities, the system identifies two kind of factors influencing activities completion compared to their initial estimated duration: positively (*knights*, shorter time) or negatively (*villain*, longer time). Having included in the model schedule for each activity an optimistic and pessimistic duration, knights and villains are applied by the systems to dynamically reschedule the uncompleted activities in the network, setting their duration equal to their optimistic/pessimistic outcome on the base of the evidence collected. It's worthwhile to mention that the user is, however, always left with the option to override the system's recommendation after the updating process.

Coming to the conclusion of this paragraph, whose prior intention is to provide the reader with an handy overview rather than a thorough review on the development of expert systems for process planning, we want to remind the fact that a plan formulation system basically focus on identifying sequences or networks of activities to meet certain goals, and in doing so merges methods and knowledge from three main areas, as discussed above: *plan formulation*, *project scheduling* and *process planning*.

Specifically, in the *field of construction* few expert systems exist and most of them were set out from 1987 to 2005.

Among them and in the construction-related literature and textbooks some research subjects related to planning and scheduling can be found (Mohan, 1990). Main subjects include coding activities, sequencing activities, representing schedules and leveling resources. A number of automatic construction planners were developed based on artificial intelligence techniques, precisely. They define activities and their sequential relationships; some also estimate their durations. Those well-know are the following:

- *GHOST* (Navinchandra et al., 1988),
- *Construction Planex* (Zozaya-Gorostiza, 1989),
- *ConsPlans* (Kano, 1990),
- *SIPEC* (Kartam and Levitt, 1990),

- *BUILDER* (Cherneff et al., 1991),
- *Know-Plan* (Morad 1991),
- *CASCH* (Echeverry, 1991),
- *HISCHED* (Shaked, 1995),
- *Case-Plan* (Dzeng and Tommelein, 1997),
- *FASTRAK-APT* (Lee et al., 1998)
- *CBRidge Planner* (Tah et al., 1999).

2.3 Ontology-based modelling in AEC Industry

Modeling plays a significant role in representing the domain of construction process. In the construction industry, process modeling is used mainly to support simulation. Elsewhere, ontologies can provide a powerful modelling approach. As defined by Gruber (1995), «ontology is a formal representation of an abstracted view of a domain that describes the objects, concepts and relationships between them that holds in that domain for a stated purpose or concisely an explicit and formal specification of a conceptualization».

Nowadays, ontology-based modelling is central to many applications as largely explained in Motta (2000), such as for medical and biological systems, information management and integration systems, electronic commerce and web services and ontologies themselves are used within the realm of artificial intelligence to capture knowledge, and create a model of the knowledge base. It has emerged that in recent years the development of domain ontologies in the AEC Industry has been identified as pivotal point to develop knowledge management and integrated workflows (Zhou et al., 2016). In this regard, an overview is proposed below.

Lima (2005) implemented the e-COGNOS platform testing the benefits of using semantic systems for adequate search and indexing capabilities. Secondly, the work they presented allows a systematic approach for formally documenting and updating organizational knowledge. Lastly, their work enhances the customization of functions in knowledge management systems. The e-COGNOS platform presented the first comprehensive ontology-based portal for knowledge management in the construction domain.

Another example is the ontology DOCK 1.0. It aims to develop a conceptual structure of key terms in the construction domain and their relationships and behavior. Besides representing concepts, DOCK 1.0 emphasizes the importance of context when representing construction knowledge. In addition, modality was inserted to allow users of DOCK 1.0 to generate a range of types of a particular concept (El-Diraby, 2013).

Akinci et al. (2010) envisioned that semantic CAD/GIS web services can provide a way to address the lack of interoperability between CAD and GIS platform.

Scheduling Ontology Studies	Object	Construction Field
Scheduling Task Rajpathak et al. (2000)	Scheduling Cost Control	
OZONE Smith et al. (1997)	Logistic Scheduling	
Kasis-Sophina Hori et al. (1995)	Generic scheduling	
CommonKADS Gobin and Subramanian (2009)	Scheduling	
COMIREM Smith, et al. (2005)		
Job Assignment Ontology Rajpathak (2001)	Scheduling	
Industry Foundation Classes (IFC) BuildingSmart (2004)		•
Mephisto Lambert and Nowak (2009)		
OnSITEsimu *Proposed in this book*		•

Benevolenskiy et al. (2012) developed a distributed multi-model-based Management In-formation system for simulation and decision-making on a construction project. The ma-jor challenge of the system was the management of the information and model logistics as well as the interdependencies among the application models. In order to support the retrieval of information from different project phases, domains and organizations and their combination in coherent multi-models, an ontology framework was developed even though the same ontology was not structured with a computational language in or-der to customize the process.

A domain ontology for construction concepts in urban infrastructure products was de-veloped by Diraby (2011). Wang and Boukamp (2011) presented a framework aiming to improve access to a company's JHA knowledge by using ontologies for structuring knowl-edge about activities, job steps, and hazards.

Zhong et al. (2012) developed an ontology-based semantic modeling approach of reg-ulation constraints based on proposed CQIE ontology and construction process. They

Other fields	Generic	Specific	Integration with other ontologies	Toolkit Integration
•	•		Time	•
• Transportation logistics				•
• manufactory				
•	•			
• Crisis-action logistics planning	•			
•		•		•
		•	Building Structure	
• Military and national security domains		•		
		•	Time Space Building	•

concluded that the proposed regulation-based automated construction quality compliance checking as a parallel activity to construction planning and execution can improve efficiency, reduce errors, and save human resources.

Recently, Zhang et al. (2015) investigated a new approach to organize, store and re-use construction safety knowledge. A construction safety ontology is proposed to formalize the safety management knowledge. It consists of three main domain ontology models, including Construction Product Model, Construction Process Model, and Construction Safety Model. The interaction between safety ontology and Building Information Modeling (BIM) is also explored.

Finally, in order to understand how other fields, which have high-level scheduling approaches, addressed the problem of scheduling activities and resources, the most important reviewed studies are grouped and reported in Tab. 2.

Among those, OZONE could certainly provide an excellent framework of concepts for our research in terms of scheduling entities (Smith and Becher, 1997). The OZONE ontology

represents a synthesis of extensive prior work in developing constraint-based scheduling models for a range of applications in manifesting, space and transportation logistics.

2.4 Construction workspaces management in AEC Industry

One of the most important factors able to affect the efficient and safe delivery of construction projects is the site workspaces availability (Dawood, 2005). Workers, equipment, temporary facilities have different space requirements and they compete with each other for space usage throughout the entire life of a project (Cai, 2014). Furthermore, workspaces interact with each other dynamically, their locations and volumes change in three dimensions and across time according to a specific schedule information. In this context, traditional construction scheduling techniques such as Gantt charts and network diagrams are inadequate for managing site workspaces, mainly, due to their lack of spatial representation (Chau, 2004), (Dawood, 2006), (Dawood, 2009), (Moon et al., 2014). For the aforementioned reasons, incorporating workspace consideration from the spatial and temporal perspective in construction planning plays a pivotal role. Several expressions have been used to describe this process to involve workspace management including 'spatial modelling', 'execution space analysis', 'schedule-workspace management', 'workspace planning' and 'time-space analysis'. According to Kassem (2015), the definition *Construction Workspace Management* includes three elements: (a) generation and allocation of workspaces, (b) detection of congestion and spatial temporal conflicts and (c) resolution of identified conflicts.

Since these components are considered in the research, a deeper investigation has been carried out.

a. Most of early studies, between 1992 and 2006, focused on **generation of workspaces** and **detection of conflicts**.

Thabet (1994) proposed a methodology in which workspace are marked up in an CAD environment and then are broken down into work blocks and linked with related activities. By using a 'space capacity factor' congestion is detected, and rule-based strategy is used to solve the congestion. Sanvido et al. (1995) presented a scheduling model that incorporates workspace constraints in the scheduling of repetitive work in multistorey buildings. Their model proposed a method to define and quantify several workspace parameters such as physical space demand and surrounding space demand. Once again Riley and Sanvido (1997) presented a space planning method in the case of a multistorey building. The space conflict was reviewed empirically on

2D plan view by defining space type and identifying an execution location order for each considered space. Guo (2002) analyzed workspace overlapping introducing the values: Interference Duration Percentage (IDP) and Interference Space Percentage (ISP) in 2D floor plan.

Akinci and Fischer (2000) presented a methodology for the generic generation and allocation of workspaces to activities that considers the construction methods in use. The methodology includes an ontology for capturing spatial requirements that is implemented in ad-hoc 4D CAD environment (i.e., 4D SpaceGen) for the automatic generation of workspaces.

The so-called Geometry-based Process Model (GPM) was developed by Akbas (2004); it models the transformations in construction processes as sequences of crews acting on geometric work locations. It uses a simple process description: work locations are processed by crews.

Dawood et al. (2005) applied entity-based 4D CAD technology for detecting workspace congestion to help identify potential safety hazards on-site using critical space-time analysis (CSA) in 4D visualization. The proposed CSA associates certain visual features for workspace planning with the workspace competition. The PECASO (Patterns Execution and Critical Analysis of Site-space Organization) prototype was developed to comprise and evaluate the outcome of the CSA.

Song and Chua (2005) established methodologies for a system which models temporal 3D space, using COmponent State Network CEntric Model (COSCEM) aiming to present a platform for integrating project information including product, process, space and intermediate function.

Thomas et al. (2006) considered a real case study to analyze the effects of documentation of workspace conflicts and labor productivity in order to minimize the workspace congestion in the case of a multistorey building project.

Moon et al. (2009) proposed an integrated approach in which workspaces are generated using the bounding volume of each individual objects and then are linked with schedule activities in a 3D CAD Environment.

Winch and North (2006) suggested a Critical Space Method (CSA) in order to solve the construction space scheduling problem and developed AreaMan and SpaceMan for the CSA system that supports managers to analyze the spatial overloading between work execution spaces.

Chavada et al. (2012) developed approaches and a prototype for visually analyzing congestion of activity execution workspaces (AEWs) with the severity of congestions (CgS) by

calculating the conflicting volume between workspaces in nD CCIR (Dawood and Mallasi, 2006), (Mallasi, 2006).

The so-called method 'Spatial Network' in which workspace requirements are considered only at a relatively high level of detail for resources such as laborers and building objects was proposed by Bargstädt and Elmahdi (2010).

Cai and Su (2014) presented a lifecycle approach to the modelling and planning of construction workspaces taking into account the evolution pattern of activities' space requirements. The aim was reached using an object-oriented structure of workspace with both geometric and temporal attributes. This research was implemented in an ad-hoc workspace modelling and planning environment, independent from commercial scheduling and design platforms.

Jongeling et al. (2008) considered the distance between the different types of work as the main relevant factor which affects a safe and a productive work execution, by manually extracting 4D spatial content from 4D CAD models.

Recently Kassem (2015) created an Industry Foundation Class (IFC) compliant 4D tool for workspace management. The methodology and the tool provide a holistic solution to the approach of workspace management through the allocation of workspace to site activities, the detection of congestion, special and temporal conflicts and their resolution within a 4D environment in an interactive real-time manner, aided with analytic data from a centralized database.

Finally, an interesting prototype was proposed by Zhang et al. (2015). It consists in a BIM-enabled approach for activity-level construction site planning dedicated to improving construction safety and reducing site congestion. The method allows the automated workspace visualization and the visual checking of workspace conflicts on a commercial BIM platform, using an ontology approach previously formalized in (Zhang et al., II, 2015) and integrating Global Positioning System (GPS) data loggers attached to the hardhats of work crew.

The review has shown that, with reference to the generation of site workspaces in 2D/3D modelling environments, when dealing with BIM-environments -both IFC-compliant and non-compliant- mark-ups are used for the generation of workspaces. Moreover, design automation in spatial modelling is still missing, and current studies on the topological interactions among workspaces and building objects as well as integrated methodologies for on-site workspaces planning are currently insufficient.

b. On **conflicts detection among workspaces** the reviewed approaches can be grouped in three research branches, namely related to:

1. detection of physical conflicts between the site workspaces;
2. detection of schedule conflicts or, in other words, the detection of a temporal overlap between tasks that is mainly considered by the models which use traditional representation of the construction process (i.e., Gantt Chart and Network Diagrams);
3. site congestions identification (Zhang, 2015) that considers the ratio between the volume of resources occupying a workspace and the volume of the workspace which is available on site for a given activity or a set of activities. Often defined as 'scheduling overlapping ratio' (H. Moon et al., 2015).

c. For **workspaces conflict resolution**, different methods have been reviewed. They can be schematized as follow:

(a) mathematical algorithms;
(b) artificial intelligence methods (i.e., genetic algorithms, fuzzy logics and ant column models);
(c) rule-base heuristic approaches supported by databases.

Nonetheless, many of these approaches present too complex mathematical models, tending to lose connection with the real dynamics of construction sites, and can directly manage only a small part of a given building project while being extremely demanding in terms of understanding of the adopted mathematical model to be practically implemented by project managers. For these reasons, they are mostly limited to research purposes, going unused by practitioners.

Despite it could be considered a more closed field, advancements in *Site Layout Planning* contribute to increase safety and productivity of construction projects as well. This field has interested many researchers in the past three decades. Several models have been developed with the common objective to determine the optimum location of site objects in the available space on site, while considering their mutual workflow interactions. Even though generally sharing the same objective, these models used different approaches to define and address the problem. Leaving aside the first generation of site layout models, which ignored the progressive changes that occur within the site environment, the second generation of site layout models considered the importance of incorporating the time factor. For this reason, these models are reckoned as *'Dynamic' Site Layout Planning*.

Briefly, they model site facilities setups, relocations and demolitions across the

construction stages and explore the design constraints for introducing site facilities inside a construction site, changing the site facility and material demand positions and taking into account the actual use of site space over time, constraints on available locations within the site and the cost of site facility relocations (Zouein and Tommelein, 1999). They were developed on mathematical models for the optimization of the site layout, avoiding potential spatial conflicts and minimizing the relocations cost.

In this regard, Elbeltadi et al. (2004) applied a genetic algorithm to solve the dynamic layout problem with safety zones but did not consider the facility relocations. Moreover, a 4D CAD integrated site planning system that integrated schedules, 3D models, resources and site spaces for dynamic construction site planning was developed by Ma et al. (2005). Differently, Su et al. (2012) presented a geographic information system for the dynamic material site layout planning of building renovation projects that did not consider material flow.

Application

chapter 3

The objective of the following paragraphs is to present a complete Knowledge-Base (KB) based on ontologies, developed by the author, to cover the domain of construction planning and scheduling having as input a Building Information Model IFC-compliant (details in *Chapter 6*).

In order to address this challenge several different approaches are adopted by researchers (Atkinson et al., 2007). In this regard, the comparison with models and data-base has been considered and the reasons that justify the choice of using ontologies as methodology for knowledge modelling are listed below:

a. *Opportunity to consider and reuse existing ontologies.* Considering, for instance, the need for the representation of the notion of time in different domains, they might be included the concepts of time intervals, points in time, measures of time as well as in the construction scheduling problem where this issue results addressed from other investigated research perspectives. Therefore, reusing existing ontologies stand as an integration opportunity, assuming that the system could interact with other application domains.

b. *Ontologies support consistency checking and reasoning which are among the objects of the proposed approach* (Models do not, databases do not). One of the roles assigned to ontologies in systems engineering is to realize "intelligent databases" that can offer various kinds of reasoning services on data at runtime. Instead of the 'data integrity check' used in databases, the 'consistency check' can be performed using ontologies and automated reasoning can be performed based on rulesets.

c. *Ontology represents knowledge in an intuitive way in the form of classes and properties* (Databases do not). This is an important reason considering that users (i.e., project managers) have to interact with them.

d. *It is much easier to present the complex structure of the construction process by using ontologies than using a relational database.* In fact, the objects of the proposed ontology-based expert system are to have a flexible body which can be easily adapted. For example, if we consider the diversity of construction processes, it could be that the proposed ontology

doesn't take into account some concepts or relationships or, even more, the reasoning mechanisms based on different scenarios (e.g., spatial-temporal optimization of workspaces) and that it can't meet the requirements expressed by the user. With the definition of new entities, the ontology can adapt to address new scenarios, meanwhile, in a database it is most likely that the whole table structure had to be reviewed.

Now, it is worthwhile to recall that in order to make the Knowledge Base *machine-interpretable*, a number of languages exists to support the creation of ontologies (see *Chapter 1*). The most common languages for that purpose are: *KIF*[1], *F-Logic*[2], *RDF*(S) and *OWL*. All of them, though with different expressive power, have a well-defined syntax which makes them processable by computers.

It has been chosen **OWL**, the *Web Ontology Language*, to compute the ontologies. Other than being a standard from the World Wide Web Consortium (W3C) and the most

---

[1] The *Knowledge Interchange Format* (KIF) is a formal approach used for knowledge exchange among computer programs that are different in nature. The semantics of KIF are based on the correlation between the terms and sentences of the language and the conceptualization of the world in terms of objects, functions and relations. KIF uses declarative semantics for representing the meaning of expressions using first order predicate calculus and reasoning rules. This is a very early approach and lacks in its inability to transmit declarative information between large systems which is the aim of an ontology structured for the construction activity simulation.

[2] *F-Logic* is another approach, where well-defined semantics of logics are integrated with frame-based languages to provide formal semantics and resolution-based proof procedures. This was developed particularly as a database logic language comprising the object-oriented features such as object identity, complex objects, inheritance, methods, and rules.

widespread ontology language (Baader et al., 2003), the reasons of its choice are twofold, as reported below:

e. BIM systems and models are equipped with a standardized interface for data exchange which is the IFC (Industry Foundation Classes) standard (OpenBIM, 2016)[3]. Some pilot schemes in academic research have tried to make IFC available as an OWL ontology to allow the usage of semantic web technologies as explained in Drogemulle and Schevers (2005) and Beetz (2009). Thanks to these research efforts, it is only a short while since the ifcOWL ontology, which is precisely meant to be used to allow extensions towards other structured data sets, is available. This would mean that a practical data-exchange between a given BIM and our model could be established.

f. The possibility that the Knowledge Base relies on the ontology which underpins a BIM, would accomplish higher robustness in the future to any software applications whose functioning is based on the presented KB. In this way, it would also be possible to provide our modelling domain (classes, relationships and properties) with a logical and ontological link with BIM ontology (ifcOWL).

Based on these assumptions, in the next paragraph we present and specify the ontological structure of the Knowledge Base.

3.1 Ontological structure of the Knowledge Base

In this section the assumption on which the author built the Construction Process Ontology are described. The presented ontology consists in a formal description of concepts (**OWL classes**) referred to the task of construction planning and scheduling. Each concept, within the ontology, is described by using various relationships with other concepts or attributes (**OWL properties**) and restrictions on properties (**OWL restrictions**). The properties define precisely the requirements for membership of the class. Such an ontological framework together with a set of instances (**OWL individuals**) that specify the ontology application to a case study, forms a **Knowledge-Base**.

More precisely 'OWL properties' are *binary relations* on classes (see '*requires*' in Figure 3.2) of mainly two types:

- **Object-properties.** They are relations between two classes or individuals.
- **Datatype-properties.** They link an individual to a *Datatype-value* (e.g., real number, decimal number, string, Boolean value, time instance, etc.) In other words, they are used to relate an individual to a concrete data value.

---

[3] BuildingSMART maintains a framework for software companies to collaborate in supporting open standards for BIM.

An object property linking the class
`Construction Method` to the individual
`Installation Space`

A datatype property linking the individual
`Installation Space` to the data literal '5',
which has a type of an xsd:integer

An Example of a transitive property: `requires`

Moreover, OWL allows the meaning of properties to be enriched through the use of
property characteristics[4] (i.e., functional **FU**, inverse **IN**, transitive **TR**, symmetric **SY**,
asymmetric reflexive **AS**, irreflexive **IR**).

It is evident that classes are the cornerstone of the ontology. For example, a class of '*safety_
spaces*' could represent all the workspaces with the same end-use likely to be in a construc-
tion site. Specific spaces on the application domain will be instances of this class. Classes
may be organised into a s*uperclass-subclass* hierarchy with Sub-classes that specialise ('are
subsumed by') their Super-classes. In this regard, the class of '*workspaces*' could be divid-
ed into hazard and non-hazard spaces or into paths, warehouses, material staging areas,

---

[4]Listed below the main typologies of object-property (relations) and their specification:
• If a property is functional, for a given individual, there can be at most one individual that is related to the indi-
vidual via the property.
• If a property is inverse functional then it means that the inverse property is functional. For a given individual,
there can be at most one individual related to that individual via the property.
• If a property is transitive, and the property relates individual a to individual b, and also individual b to individ-
ual c, then we can infer that individual a is related to individual c via the property.
• If a property is symmetric, and the property relates individual a to individual b then individual b is also related
to individual a via the property.
• If a property is asymmetric, and the property relates individual a to individual b then individual b cannot be re-
lated to individual a via the property.
• A property is said to be reflexive when the property must relate individual a to itself.
• If a property is irreflexive, it can be described as a property that relates an individual a to individual b, where in-
dividual a and individual b are not the same.

3.3 Selection of visual notions to represent ontologies in relation to scripts developed in OWL language (Lohmann, 2014)

laydown area and so forth. Now one fact turns out to be evident: the ontology strictly depends to the objectives.

In order to help and assist in the development, exploration and verification of the ontology, its visualization is crucial. Although several visualizations for ontologies the Visual Notation for OWL Ontologies –VOWL- has been chosen in this monography to represent the proposed ontologies (Lohmann et al., 2014). A sample of the used graphical primitives and color scheme is shown in Figure 3.3.

Regarding the development of an ontology, there is not one standardized methodology and several possible approaches can be adopted.

Therefore, in the following Figure 3.4 is schematically presented a stepwise framework for developing the ontologies herein presented. The framework is composed by five consequential steps, all necessary to define a part of the ontologies structure.

1. *Investigation of the knowledge resources.* This step focuses on reviewing already existing ontologies, taxonomies, or other sources within the construction domain, and on how to reuse them. In other words, both ontological and non-ontological resources have been investigated. A clear example of a non-ontological knowledge source consists in the information that can be collected from direct construction site observation, that proved to be essential in respect of scheduling and workspace ontologies.

2. *Specification of objectives.* In order to figure out which and how many classes, relations and properties are comprised in the ontology, the modeling objectives has been obtained in this step by answering a list of competency questions such as the following: Why is the

3.4 Research tasks in the ontology development

	Investigation of the knowledge resources	Specification of objectives	Overall framework	Topological Relationships	Ontology Specification and editing in *Protégé*
Main Steps					
Main Deliverables	Reviewing already existing ontologies, non ontological resources and construction site observation	Specification of objectives	Main selected concepts (classes) and their formal explanation	Definition of: > classes > class hierarchy > relationships > properties	> Specification of data values for reasoning > consistency checking

ontology being built? What type of information should be involved in the ontology? In order to give a clearly visible structure of the objectives, a graphical representation is proposed for each sub-ontology.

3. *Definition of the overall framework of the ontology.* In this step a list of the main selected concepts (classes) and their formal explanation are presented.

4. *Definition of the topological relations and integration with other domain.* The core of the ontologies is here presented. Classes and class hierarchies are defined in detail, relationships between classes are established and properties and attributes are identified according to the objectives.

5. *Ontology specifications and computation in the ontology editing environment.* The presented ontologies have been modelled by using *Protégé* (Horridge, 2011). In order to design a correct and not redundant ontology, the consistency of the ontology has been checked using an automated tool.

3.2 Modelling domains

When building an ontology for knowledge mapping of a particular domain of interest, it is important to reflect on the different variable the domain depends on. In the presented case, the problem of modelling the construction process for construction workspaces planning and construction activities scheduling is the result of a complex process involving many decision variables, defined as *modelling domains*[5]. As a first step for developing

---

[5] (Heijst et al., 1997) distinguish different types of ontologies, e.g. domain ontologies, which express conceptualizations for particular domains; and generic ontologies, whose concepts are considered to be common to many domains.

3.5
Conceptualization of modelling domains which compose the Knowledge Base

the Knowledge Base, it is necessary to define the different variables at play from which planning and scheduling approaches will be dependent.

These domains shall be drawn from the aims of the system applications that will base on such a KB. In our case the aim may be summarized as follow: «the knowledge base should support system and software applications that, based on an Information Model (IFC-based), may work in the branch of construction planning and scheduling».

That is why the proposed Knowledge Base does not follow an *all-in-one modelling approach* but analyze the individual models by considering singularities of each domains separately. This choice of a *multi-ontologies* system for modelling the construction process is justified on the ground that further interrelations can be specified in order to provide a higher flexibility to the knowledge base so that it is open to other extensions in terms of others domains (e.g., risk analysis, health and safety management, paths planning, monitoring systems, etc.).

Overall, the designed Knowledge Base is composed of four modelling domains. They are coded by using the following four sub-ontologies, interrelated as schematized in Figure 3.5.

1. *Construction **Scheduling** Ontology*: this ontology contains all those elements for representing the scheduling problem and constraints. It provides a structured foundation for analyzing the information requirements of a construction schedule which should depend on availability and typology of resources, space-temporal constraints, allocation of workspaces and so forth (details in *Chapter 4*).

2. *Construction **Space** Ontology*: it contains the site workspaces representation and the property set able to activate the reasoning mechanisms and the built-in algorithm to allocate workspaces themselves. In fact, workspaces need to be represented with their basic geometrical and capacity properties and need to be linked to the building objects (details in *Chapter 5*).

3. *Construction **Time** Ontology:* it is the ontology dedicated to the description of temporal properties of site entities in their evolution across time. It also comprises objects to describe possible relations between time periods in order to define the temporal positions among activities, workspaces and building objects. It plays a pivotal role in developing rule-based reasoning mechanisms for minimizing overlapping activities in terms of workspaces. It works by mean of a connection with a Calendar to the Knowledge base (details in *Chapter* 7).

4. *Construction **Product** Ontology:* this ontology represents the domain of Building Information Models (BIMs) and describes the functional, geometrical and topological information of the building objects –products- that the Knowledge Base needs to get. Based on IFC-schema and more specifically on the ifcOWL ontology, a new sub-ontology has been specified to represent all those ontological objects that a knowledge base, working on the construction planning domain, should include. (details in *Chapter* 6).

The high-level structure of the ontological framework is depicted in Figure 3.6.

3.6 Graphical representation of the ontological framework of the Knowledge Base capturing the integration of four sub-ontologies and their specification by means of 'individuals'

chapter 4

4.1 Specification of modelling objectives

Central object of the scheduling ontology is to define a reusable and extensible base of concepts and relations for describing and representing scheduling problems in the domain of construction process. The modelling objectives are specified by means of a list, later explained in a diagrammatic form in Figure 4.1:

1. include scheduler capabilities and experience in the scheduling domain;
2. automate the scheduling process of the structural construction sequence which are user-independent;
3. provide a constraint-based framework of the scheduling architecture which encapsulates reusable concepts and intelligent relations for configuring and customizing the scheduling solution:
4. maximize productivity by minimizing idle time and by putting as many activities as possible in parallel if their workspaces do not overlap.

4.2 Overall framework of the Scheduling Ontology

In the proposed scheduling framework, the ontology can be formally represented as a mapping from a *twelve-dimensional* space. Those input parameters provide the necessary components to specify the scheduling task, and are connected by using binary relations with specific 'property characteristics' (Figure 4.2).

1. **Construction Method**, (CM) = {cm1,...., cmn}. This class is an abstract entity that drives the ontology and describes the construction work execution. The construction schedule, linked to a given Building Information Model, should have as many Construction Methods as the number of Object types.
2. **Work Description**, (WD) = {wd1...., wdn}. It describes with generic terms the construction execution referred to a given Construction Method, its spaces and resources on site.
3. **Demand**, (De) = {de1,...., den}. This class contains both construction procedures and safety rules that are formally and graphically simulated by using the 'workspace ontology'.

4. **Construction Product,** (CP) = {cp1,...., cpn}. This class comprises all the building objects that are primarily part of the construction of the building itself. Its categorization comes directly from the IFC-schema as later described in Chapter 6. Hence, the ifcBuildingObjects, contained in a given BIM, are converted in individuals that are referred to as instances of the class 'Construction Product'.

5. **Condition**, (Cn) = {co1,...., con}. This abstract entity describes conditions that must be achieved at the beginning (pre-condition) or ending (post-condition) of a Construction Method. A Condition can be expressed in terms of activities or milestone in a time period.

6. **Resource**, (Re) = {re1,...., ren}. To define a Construction Method, it is necessary to choose specific Resources with a specific proposed set. Semantics and properties of

those resources vary according to the type of Resource and define their available capacity across time. A number of resources have been proposed which should cover those required by a construction process.

7. **Constraint,** (Cs) = {co1,...., con}. In getting system applications to work on the solution to a given construction scheduling problem, constraints determination and satisfaction is essential. Generally, a constraint restricts the set of values that can be assigned to a given variable according to Smith et al. (2005). Our scheduling domain provides the means to model three types of constraints that restrict the assignment of Start and End-Times and the physical allocation in site of Resources and Workspaces related to each construction activity:

   a. **Resource-depended.** It designates the condition under which a Resource (e.g., scaffolding, labor crew, etcetera) can be assigned to a given construction activity or restrict the physical capabilities of resources to handle more activities simultaneously;

   b. **Time-depended.** It defines the possible relations between objects within the construction process (e.g., before, meets, overlaps, during, equals, etc.) and their time periods;

   c. **Space-depended.** It consists in a family of three sub-constraints which are strictly connected to the workspace simulation (e.g., equipment space, labor crew space, hazard space, etc.) and all those constraints which can be automatically extracted by the IFC Building Structure (e.g., if workspaces of two activities overlap, they can't run simultaneously in construction site).

   Moreover, a further classification of constraints has been introduced:

   d. **User-set.** They derive directly from the user who can directly add specifications and constraints depending on his experience;

   e. **System-set.** They are automatically generated by the ontological structure due to properties assigned to the relationships;

   f. **Building-set.** They derive directly from the Information Model the applications that will work, for example, by using transformation rules (e.g., a beam should be constructed after connected columns);

   g. **Simulation-set.** They derive from the 'workspaces conflicts checking process'.

8. **Phase,** (Ph) = {ph1,...., phn}. A group of strongly related construction processes defines a Phase which ends with a Milestone.

9. **Process,** (Pr) = {pr1,...., prn}. A process represents the most abstract class that groups various activities.

10. **Activity,** (Ac) = {ac1,...., acn}. In the proposed architecture, a schedule is represented as a network of Activities that will produce a number of Construction Products by using

Workspaces. To schedule an Activity, it is necessary to choose Resources that produce the time intervals to assign to each activity depending on their capacity level.

11. **Milestone,** (Mi) = {mi1,...., min}. A Milestone represents a Phase finalization and is connected to a given Time Instant.

4.3 Topological structure

After having identified classes, in this paragraph is depicted the framework model of the scheduling ontology in terms of the classes, properties and relation diagrams. For readability sake and to enhance the understanding of the discussion, different font and text colors have been used to highlight ontological objects and to distinguish their belonging to the different ontologies that compose the Knowledge Base (Figure 4.2).

Core class of the present ontology is ConstructionMethod, being other classes dependent on it. In fact, by using the relationships and properties listed below each construction method is described in terms of required resources, activities and workspaces. All these classes are inextricably linked in an intelligent framework.

Since a ConstrucitonMethod *produces* or *consumes* a number of construction products, the class ConstructionProduct contains a list of individuals which represent the building elements (e.g. columns, beams, slabs, walls, etc) and their information requirements, and provide the main interface for connecting the scheduling problem to a given Information model IFC-based. For this reason, it follows the structure of the IFC schema and mainly includes sub-types of *IfcBuildingElement*.

Moreover, in order to carry out a certain Construction Method within a given construction site, some Condition is implied to exist before (*precondition*) or after (*postcondition*). Also, a Construction Method *isDescribedBy* a WorkDescription which specifies with generic terms its execution, the allocation of spaces and the required resources. A Work-Description, in turn, can be defined as a Procedure or a SafetyRule depending on a set of principles or conditions specified in the class Demand. This means, for example, that if the user links two workspaces to a construction method the system automatically classifies this relation depending on the workspace: a procedure if it's a safety space, a safety

⊕
4.2 Textual notations used in the body of the text in order to distinguish the ontology objects

(Class) ·····*relation*·····▷ (Class)	(Class) ·····*relation*·····▷ (Class)	(Class) ·····*relation*·····▷ (Class)
SPACE Ontology	**SCHEDULING** Ontology	**TIME** Ontology

4.3 Class hierarchy in the construction scheduling ontology:
resources types on the left side and constraints types on the right side

rule if it's hazard space. Each procedure or safety rule contained in a Demand *requires* a number of Resource.

The class Resource is also central to the definition of our scheduling ontology. It represents an entity which is assigned to a Construction Method for its execution. Each Resource can *handle* one or more activities simultaneously and is provided by a specific property set. These properties are all those which effect its availability and utilization in function of its specific Capacity (e.g., *hasCapacityLevel*). Making efficient use of Resources in supporting activities becomes then the key-task managed by the rule-engine in solving the scheduling problem. A resources class hierarchy has been proposed which models each sub-class in terms of its dynamically changing amount of *CapacityLevel*. The class hierarchy is explained in Figure 4.3 and the main class restrictions are depicted in Figure 4.4.

Going on, an Activity *isFollowedBy* an interrelated set of sub-activities. To define a schedule, each Activity is related to the MicroLevelWorkspaces required to be performed in site. A Process is modelled as an abstract entity which *isComposedOf* a number of Activities. More Processes make up a Phase which ends with a Milestone and requires MacroLevelWorkspaces

➡
4.4 OWL restrictions to specify the scheduling ontology as regards to the resources' types

➡
4.5 OWL classes restrictions as regard the scheduling ontology

➡
4.6 Visualization of the ontological graph representing classes and relations of the Construction Scheduling Ontology (Snippet from the Ontology Modeling Environment *Protégé*)

to being performed within the construction site. These relations must follow a number of restrictions as presented in Figure 4.3 and Figure 4.4. Finally, a Milestone involves one or more ConstructionProduct.

In Figure 4.6 is visualized the conversion in OWL language of the described ontological model.

4.4 Specification of the entities

In the table below specifications of classes, relationships and their properties are given to the reader.

In the figure below is printed the visualization of the ontology computation with all the afore-mentioned properties as extracted from the ontology editor *Protégé* by using the VOWL vi-sualization functionalities.

below and next pages
Table 3
Specification of entities, relations and properties in the scheduling ontology

1. OWL Class	CONSTRUCTION METHOD		
Entity definition	Abstract entity which describes the Construction Work by means of consumed and produced Work Products and specifies the input goals that drive the ontology. The **Construction Products** are represented by one or more Building Elements [ISO 16739]		
Datatype Properties:	hasName:	type: ***name*** assignment	
Object **Properties**:		**Range and Properties**	
	produces:	**Domain**: ConstructionMethod **Range**: ConstructionProduct	**FU**
	consumes:	**Domain**: ConstructionMethod **Range**: ConstructionProduct	**FU**
	hasPrecondition:	**Domain**: ConstructionMethod **Range**: Condition	
	hasPostcondition:	**Domain**: ConstructionMethod **Range**: Condition	**TR-AS**
	isDefinedBy:	**Domain**: Construction Method **Range**: WorkDescription	

2. OWL Class	**WORK DESCRIPTION**		
Entity Definition	Abstract entity which describes the construction execution, allocation of spaces and required resources for each activity by using generic terms in relation to the construction methods that they are going to use. A generic description is used for the **Work Description** in order to better fulfil the research objective to merge the construction user experience into the Ontology.		
Datatype Properties:	hasDescription:	type: *string* assignment	
		Properties	
Object Properties:	*isDescribedBy:*	**Domain**: Construction Method **Range**: WorkDescription	**FU**
	defines:	**Domain**: WorkDescription **Range**: Demand	**TR**

3. OWL Class	**DEMAND**		
Entity Definition	Abstract entities which describe the specific way to perform the description of a **Construction Method** that is, in turn, modelled and expressed in terms of the sub-classes **Procedures** and **Safety Rules**. The group of outstanding methods at any point determine the scheduling problem to be solved.		
Datatype Properties:	hasPriorityLevel: *The relative importance of the Demand, provides a basis for establishing a partial ordering over the entire set of demands and their procedure.*	type: *real number* assignment	
		Properties	
Object Property:	*imposes:*	**Domain**: Demand **Range**: Constraint	**FU**
	requires:	**Domain**: Demand **Range**: Resource	**FU**

4. OWL Class	CONSTRUCTION PRODUCT		
Entity Definition	It is a building object. This class comprises all elements that are relevant parts of the construction of a building. Construction products are all physically existent and tangible things [ISO 6707-1]. A Product is realized through the execution of some set of Activities. A Construction Method is assigned to each product.		
Datatype Properties:	For its datatype properties see the Product Ontology.		
	Range and Properties		
Object Property:	*imposes*:	**Class**: Constraint	TR
	isConsumedBy:	**Class**: Construction Method	TR
	isProducedBy:	**Class**: Construction Method	TR

5. OWL Class	CONDITION	
Entity definition:	Situation that must be achieved at the beginning (pre-condition) or ending (post-condition) of a **Construction Method**	
Datatype Properties:	hasName:	type: ***name*** assignment
	hasObjective:	type: *string* assignment
	Properties	
Object Properties:	*precondition:*	**Domain**: Construction Method **Range**: Condion
	postcondition:	**Domain**: Construction Method **Range**: Condion

6. OWL Class	**RESOURCE**	
Entity Definition	It is a central entity in the definition of the scheduling problem because it supports or enables the execution of a **Construction Method** and its related Activities. Resources are considered as finite supply entities whose availability constraints "when" and "how" **Activities** are executed	

	hasName:	type: ***name*** assignment
	hasCapacityLevel	type: ***Real Number*** assignment
	Each Resource is defined as a limited capacity entity which can be modelled as an integer that indicate how many activities it can handle in parallel at any given time. The Capacity is a quantity of some unit measure (number of workers, volume, surface, weight, etc.) which is available to perform activities over time.	
	hasUniformCapacity:	type: ***Boolean*** assignment
Datatype Properties:	*It considers Capacity as a scalar quantity and imposes that (Constraint) the sum of the Capacity used by all supported Activities the Capacity of the considered Resource.*	
	hasHeterogeneousCapacity:	type: ***Boolean*** assignment
	The slot considers Capacity as the sum of more than one Uniform Capacity	
	hasMultidimensionalCapacity:	type: ***Boolean*** assignment
	It considers Capacity in terms of more than two quantities and imposes that (Constraint) for each different unit measure the sum of the Capacity utilized by all the Activities the Capacity of the considered Resource.	
	hasSpeed:	type: ***Real Number*** assignment
	It considers how fast the Resource takes to perform the Activities.	

		Properties	
	isRequiredBy:	**Domain**: Resource **Range**: Demand **Inverse**: Requires	**F U / TR**
Object Properties:	imposes:	**Domain**: Resource **Range**: Constraint	**FU**
	handles:	**Domain**: Resource **Range**: Activity	**FU**

6.1 Sub-Class	**Spatial Resource**

Entity Definition: A Spatial Resource is a sub-class of a Resource. This type of resource is time dependent in the sense that its availability for executing the assigned **Construction Method** depends on the available time period of a particular resource and this resource occupies a **Micro-level space** which is a space linked to the execution of a given building object.

DatatypeProperties:

occupies	**Domain**: Spatial Resource **Range**: Macro Workspace

6.2 Sub-Class	**Non-Spatial Resource**

Entity Definition: It is a resource which is time independent. Time does not play an important role in the allocation of **Construction Methods** on the non-spatial resources. This resource does not occupy neither Macro-level or Micro-level workspace.

6.3 Sub-Class	**Available Capacity Resource**

Entity Definition: It is a resource whose availability depends on the residual amount of Capacity. The class contains the following two subclasses.

6.3.1 Sub-Class	**Reusable Resource**

Entity Definition: An Available-Capacity Resource whose capacity becomes available for reuse after an Activity to which it has been allocated finishes. Its main property is the Setup-Duration which specifies how long it takes to configure the considered Resource for another Activity.

Properties

SetupDuration:	type: time duration assignment
hasSetupDuration:	**Domain**: Reusable Resource **Range**: Temporal Entity

6.3.2 Sub-Class	**Standard Resource**

Entity Definition: It is an Available-Capacity Resource whose *capacity*, once allocated to an activity does not become available again. Hence, in this case, the Activity consumes the Resource.

6.4 Sub-Class	Atomic Resource

Entity Definition: It is the smallest resource which is not further divisible and can only support one Activity at a time. This class distinguishes two subclasses listed below.

6.4.1 Sub-Class	Unit-Capacity Resource

Entity Definition: This Resource can only be used by one **Activity** during any given Time-Interval

6.4.2 Sub-Class	Batch-Capacity Resource

Entity Definition: This Resource can support more than one **Activity** if three different conditions are satisfied:
- it has enough Capacity depending to the supported Activities;
- the Activities require the same Resource configuration and are temporally synchronized.

6.5 Sub-Class	Aggregate Resource

Entity Definition: An Aggregate Resource can contain a number of smaller Atomic Resource. Its capacity depends on the capacity of its sub-resources and it can be independently allocated to multiple activities in different Time Interval

Object Properties: *containsSubResource:*   **Domain**: Aggregate Resource
Range: Resource

6.5.1 Sub-Class	Homogeneous Resource

Entity Definition: Is a subclass of an Aggregate Resource composed of more than one sub-resource of the same type.

6.5.2 Sub-Class	Heterogeneous Resource

Entity Definition: Is a subclass of an Aggregate Resource composed of Resources of different type and capacity.

7. OWL Class	CONSTRAINT
Entity Definition	A constraint restricts the set of values that can be assigned to a relation between two time intervals in function of different variables according to the scheduling problem. A number of Constraints has been identified which are modelled as sub-classes.

Datatype Properties:	hasApplicability: *It is a value assignment which should be **Hard** or **Soft***	type: **Boolean** assignment

Object Properties:	*isImposedByResource:*	**Properties** **Domain**: Constraint **Range**: Resource **Inverse of:** Resource_Imposes **Subclass of:** Is_Imposed_By
	isImposedByDemand:	**Domain**: Constraint **Range**: Demand **Inverse of**: Demand_Imposes **Subclass of**: IsImposedBy
	isImposedByWorkspace:	**Domain**: Constraint **Range**: Workspace **Inverse of:** Workspace_Imposes **Subclass of:** Is_Imposed_By
	isImposedByConstructionProduct:	**Domain**: Constraint **Range**: Workspace **Inverse of:** Construction_Product_Imposes **Subclass of**: Is_Imposed_By

7.1 Sub-Class	Resource Depended
	Entity Definition: This class of constraints imposes conditions according to whether a *capacity*, assigned to a Resource is compatible to perform a given Activity.

7.2 Sub-Class	Time Depended
	Entity Definition: A Time-depended constraint restricts the values of temporal decision variables. In this section three types have been included:

7.2.1 Sub-Class	Absolute-Time Constraint
	Entity Definition: An Absolute-Time-Constraint identifies a lower or upper bound on the value of a Time Point which is anchored on a calendar. This constraint borrows Time Points from the following classes: *Method, Milestone* and *Schedule Availability Period.*

7.2.2 Sub-Class	Interval Relation Constraint
	Entity Definition: An Interval-Relation-Constraint specifies the relation between two different Time Intervals according to the Time Ontology.

7.2.3 Sub-Class	Duration Constraint
	Entity Definition: A duration-constraint restricts the temporal separation between the Start-Point and End-point of an interval

7.3 Sub-Class	**Space-depended Constraint**

Entity Definition: These are constraints which depend on the workspaces allocation within construction site and impose a relation between the time interval which contain those spaces.

7.3.1 Sub-Class	**Building Constraint**

Entity Definition: A Building-Constraint imposes a time interval (see *Time Ontology*) between two time intervals referred to the execution of the related construction product. It represents an execution constraint which reflects the structural connection between two Building Objects.

This information comes directly from the BIM data storage, in particular by the classes IfcStructuralConnection and precisely *IfcRelConnectsStructuralMember*For example, if a column has a structural connection with a beam, this means that the beam cannot be built before the column. Hence, the structural connection is rendered by the rule-engine in a Time Depended Constraint which synchronizes the occurrence of the two related Activities Time Intervals (I_1) and (I_2) with an Interval-Relation *Before*.

7.3.2 Sub-Class	**Spatial Constraint**

Entity Definition: This constraint maps the spatial relations which occur between two workspaces. For example, this could be the case of a space that must be located on a fixed position in relation to another a spatial constraint that has been added by the user.

7.3.3 Sub-Class	**Spatial-Conflict Constraint**

Entity Definition: A Spatial-Conflict-Constraint define a physical constraint which impacts the assignment of an Interval-Relation-Constraint to Resources and Activities. A physical constraint traduces a conflict detection (as codified in the '*clash report*') between two workspaces in the ontology.

hasName:	type: ***string*** assignment
hasDistance:	type: ***real number*** assignment
hasConstrainedSpace:	**Domain**: Spatial-Conflict Constraint **Range**: Workspace

8. OWL Class	PHASE		
Entity definition:	A group of strongly-related Construction Methods defined in a given order. A Phase ends with a Milestone.		
Datatype Properties:	hasName:	type: ***name*** assignment	
Object Properties:	*groups:*	**Properties** **Domain**: Phase **Range**: Process	**FU**

9. OWL Class	PROCESS	
Entity definition:	A process represents the most abstract class that groups various activities. If a building component is composed of more than one construction products, the Process is referred to the building components and the Activities to construction products. A sub-process further specifies a Process.	
Datatype Properties:	hasName:	type: ***name*** assignment
Object Properties:	*isComposedOf:*	**Properties** **Domain**: Process **Range**: Activity

10. OWL Class	ACTIVITY	
	Entity definition: The Activity is an abstract entity that represents the construction operation related to the installation of a Building Object. An activity requires Micro-Level Workspaces. It is the lower level of a Construction Method that could be, in turn, formed by an interrelated set of activities. A 1 to 1 ratio should be established between Activity and Construction Product.	
Datatype Properties:	hasName:	type: ***name*** assignment
Object Properties:	*requires:*	**Properties** **Domain**: Activity **Range**: Workspace
	hasTemporalEntity:	**Domain**: Activity **Range**: Temporal Entity

11. OWL Class	MILESTONE	
	Entity definition: A meaningful event. A Milestone involves one or more Work Product. A Milestone represents for instance a Phase finalization.	
Datatype Properties:	hasName:	type: ***real number*** assignment
		Properties
Object Properties:	*imposes:*	**Domain**: Milestone **Range**: Constraint
	endWith:	**Domain**: Phase **Range**: Milestone

In the figure below is printed the visualization of the ontology computation with all the aforementioned properties as extracted from the ontology editor *Protégé* by using the VOWL visualization functionalities.

4.7 Scheduling Ontology (Snippet from the Ontology Modeling Environment *Protégé*)

4.8 Scheduling ontology edited in Protégé and visualized with VOWL notations (Lohmann, 2014) in a force-directed layout. The graphical representation of OWL entities is made of visual elements. Blue circles represent classes and sub-classes; blue rectangles represent property labels of relations, the ones with no border represent datatypes (Snippet from *Protégé*)

chapter 5

A construction site is a dynamic environment and the workspaces, supporting the execution of each construction activity, are one of the most relevant resource that affects efficiency and productivity of the construction project (Kassem et al., 2015). As a matter of fact, in order to handle their simulation, is of prior importance to dynamically manage over time workspaces requirements in terms of geometries, locations and interactions with all those other spaces that describe the life-cycle spatial evolution associated with the construction activities.

For these reasons, once again, the use of an ontology is crucial to incorporate workspace planning from the spatial and temporal perspectives in the Knowledge Base architecture. The next paragraph describes the in-depth study carried out to develop the proposed Construction *Space Ontology*.

5.1 Specification of modelling objectives

The most challenging work of the Space-Ontology has been to define appropriate default attributes to define workspaces in order to support and manage the following aspects:

i.   **Workspace physical entity.** Regards the definition of a set of default Spatial Data (e.g., dimensions, orientations and positions) to be associated to workspaces in order to support a planning process in software or system applications.

ii. **Workspace structure.** Indicates how workspaces are organized within a construction site, defining a Spatial Data Structure which is the base for describing the spatial relationship between entities based on their geometry locations.

iii. **Workspace modelling.** This aspect considers the need to model workspaces efficiently by using and application, avoiding a manual modelling process that would mean an extensive input work by the user.

iv. **Workspace analysis and management.** Such aspect aims to foster the development of a Knowledge-base that includes an ontology to analyze workspaces allocation.

The aforementioned aspects are graphically specified in the following Figure 5.1. Capture and compute, by using an ontological structure, the concepts in the domains from (i) to (iv) along with their mutual relationships within the core of the ontology, is described in the next paragraphs.

5.2 Overall framework of the Space Ontology

With regard to classifications and descriptions of construction spaces, many research efforts have been made. Each of them proposes a different categorization which reflects the relationship between research objectives and workspace management.

Assuming the need for the further development of an authoritative categorization about the construction workspaces, this research extends the one proposed by Akinci and Fischer (2000) which, in turn, is an extension of 'Components, Actions and Resources (CAR)' proposed by Darwiche (1988) and Jagbeck (1994).

Hence, in order to capture the space evolution patterns, in the proposed space-ontology the construction workspaces that can involve physical changes on the construction site are represented as the combination of three categories:

1. *Macro-level spaces*: this category represents the large-scale spaces located across sites, consisting in layout areas that are not occupied or required by an individual activity but rather by a number of activities which define a 'Phase' within the 'scheduling ontology' previously presented.

2. *Micro-level spaces*: they represent the spaces required by each stated construction method for the installation of an 'objects type'. Therefore, they are spaces located within proximity to the building objects they are referred to.

But unlike Akinci and Fischer (2000), in this category are not included the spaces occupied by the building components to be installed. This is because the proposed space-ontology is a part of an Expert System which is integrated with BIMs where the considered spaces would be redundant with all those that are already included in the given BIM and which have to be processed.

3. *Bound spaces*: this workspace category, not included in Akinci and Fischer (2000), represents the site boundaries and objects that stand on site before the commencement of construction and hence have a known location on site.

In addition, to consider every workspace as a dynamic entity, instead of a static object, located in a continuously changing construction site setting, the space ontology considers the following concepts:

- **Physical Information:** this property set drives the process of workspace generation in terms of 3D shapes and all those other physical properties such as weight in case of a storage area.

- **Topological interaction** with other workspaces included in the same construction method; for example, if a labor crew space requires a protected space on its right side, the ontology should capture and manage this relation;

- **Topological interaction with building components** to be installed; for example, if a workspace has to be located in a fixed position in relation to the building components to be installed or, on the contrary, it doesn't have a fixed preference;
- **Reversible properties:** these properties are linked with the shape theory of Cordier and Portie (1989) and include some general properties that a workspace should have and, therefore, that the ontology should capture in order to manage the space allocation in site. Considering for example the *containability* property in the case of a crane's volume, the rule-engine should not manage it as constraint for the schedule generation because that workspace is allowed to contain other spaces and a conflict eventually highlighted by the 'conflicts checking process' should be overlooked.

All the space typologies are described using formal descriptions (vocabulary) that state precisely the requirements for membership of the class.

> **EXAMPLE**
>
> For instance, the class **StorageArea** would contain all the individuals that are storage areas in our domain of interest which might be **ElectricalMaterial_StorageArea** or **InstrumentationMaterial_StorageArea**. Classes are organized into a superclass-subclass hierarchy. Subclasses specialize ('are subsumed by') their superclasses. Another example could regard the classes **MicroLevelSpace** and **LaborCrewSpace**. A Labor Crew Space is defined as subclass of **MicroLevelSpace**, meaning that 'All labor crew spaces are micro-level spaces' and 'All individuals (user-setted) of the class **Labor-CrewSpace**, for every given construction method, will be automatically members of the class **MicroLevelSpace**'.

5.3 Topological structure

Here, the framework models of the 'space ontology' is depicted in terms of classes, properties and relations, by mean of the same text notation introduced in the previous Figure 5.2. As before mentioned, core class of the space ontology is ConstructionMethod. It is the first user interface regarding the ontology compilation and comprise all those data that shall be provided for the workspace planning in the prosed system architecture. It means that the user adds individuals to all those classes, described below, the ontology is made of. Each construction method *isDefinedBy* a WorkDescription. A WorkDescription *defines* a Demand in terms of Procedure and SafetyRule. A Demand *requires* some Resources. Different classes of Resources are defined according to the Scheduling Ontology but regarding the matter in question, care should be taken on the class Spatial-Resource. A Spatial Resource shall occupy a **Micro-Level-Workspace**. To manage this relation a *cardinality restriction*[1], which specifies the *exact* number of relationships that an individual must participate in, is added (Figure 5.4). Instead a *Minimum Cardinality Restriction* is used to ensure that the user will link at least a resource to each construction method.

---

[1] In OWL language, we can describe the class of individuals that have at least, at most or exactly a specified number of relationships with other individuals or datatype values. The restrictions that describe these classes are known as Cardinality Restrictions. For a given property P, a Minimum Cardinality Restriction specifies the minimum number of P relationships that an individual must participate in. A Maximum Cardinality Restriction specifies the maximum number of P relationships that an individual can participate in. A Cardinality Restriction specifies the exact number of P relationships that an individual must participate in.

EXAMPLE

By using the aforementioned *restrictions* if the user, for example, adds an individual **ColumnInstallation_LaborCrew** that is a member of the class **LaborCrew-Space** and in parallel he doesn't add a **Spatial-Resource** the Consistency Reasoner will suggest an inconsistency. Otherwise, if the user adds an individual of the class Procedure which could be, for example, **ColumnInstallation_Procedure**, without linking a Resource, the reasoner suggests again the inconsistency due to the *Minimum Cardinality Restriction* (min 1) which drives the relationship between classes of Procedures and Resources. In this way, the ontology ensures that on one hand the given resource is graphically simulated and that on the other hand the resource is computed by using the Knowledge base included within the Scheduling Ontology.

Going on in the description, a Resource might *handle* one or more Activities which, in turn, *require* at least one Micro Level Workspace. Adding a number of individuals, the user may choose how many Workspaces handle an activity, with their attached properties. In such ontology framework, restrictions are used to specify that a Procedure requires at least one Resource, and that a SpatialResource occupies exactly one WorkSpace and so forth as shown in the figure above.

Different types of workspaces are included in the model by using superclass-subclass relationships. Hence, the class WorkSpace contains three subclasses: Macro-WorkSpaces, Micro-WorkSpaces and Bound-Spaces which represents physical entities in terms of site boundaries and objects that reside on site before the commencement of activities. Five subclasses support a more explicit description of 'Micro Level' space subtypes:

5.4 Workspaces entities organized in the space ontology in a class hierarchy

LaborCrew-Space represents the space required by a labor crew during the execution of a Construction Method. Protected-Spaces are required to protect the construction product for a given time interval. Equipment-Spaces identified spaces occupied by the equipment during the execution. In many cases both labor and equipment might *generate* a Hazard-Space which is considered different from a Safety-Space that represents a safety distance between two workspaces to prevent safety hazards such as collision between two spaces or a tolerance space from objects falling from height or moving in site. The complete class hierarchy consist of the classes given in the following figure.

Although the construction of the abovementioned class hierarchy may have seemed rather intuitive so far, in OWL *subclass* means *necessary implication*[2].

Each class of Workspaces contains four groups of Datatype Properties which support their geometrical and non-geometrical representations. Datatype properties link an individual (which in this case means a workspace entity user-entered, e.g. **ScaffoldingSpace**) to a *Datatype value*. In other words, they describe relationships between an individual and data values. The proposed classification is set out in Figure 5.5.

---

[2] The fundamental taxonomic constructor for classes is rdfs:subClassOf. It relates a more specific class to a more general class. If X is a subclass of Y, then every individual of X is also an individual of Y. The rdfs:subClassOf relation is transitive which means, if X is a subclass of Y and Y a subclass of Z then X is a subclass of Z.

➡
5.5 Diagram of the property set for representing a workspace within the Space Ontology

➡
5.6 Graph visualization representing classes and relations in the Space Ontology (Snippet from the Ontology Modeling Environment *Protégé*)

5.4 Specification of the entities

In the following tabs specifications about classes, relationships and properties are provided.

Tab. 4
Specification of entities, relations and properties in the space ontology

1. OWL Class	**WORKSPACE**
Entity definition	A workspace represents a physical entity within the construction site. It can be used for various purposes according to the proposed sub-classes. The list of properties and motivations for their computation within the system are listed in the tabs.

Datatype Properties:	hasID: type: *name* assignment

hasID: type: ***name*** assignment

hasCapacityDimension: type: ***real number*** assignment

hasDimension: type: ***real number*** assignment
Indicates the dimension of the footprint area of a rectangular prism that acceptably approximates the workspace.

hasSmallestUnit   type: ***real number*** assignment
If the workspace is divisible, this property indicates the dimension of its smallest units. For example, some construction objects such as material are in packages, boxes or other units. When being located on site, they can be located in separated units if occur. The dimension of the smallest unit is needed to decide the size of split location. Object such as equipment are rigid in size and this aspect is reflected in the flexibility property

hasHeight: type: ***real number*** assignment
Includes the highest point of the space

hasCentroid (X-Y axis):   type: ***real number*** assignment
Indicates coordinates of geometric centroid of footprint

hasLocation (X-Y axis):   type: ***real number*** assignment
Indicates coordinates of workspace location after that the site simulation is carried out and its optimal allocation is defined

hasMobility: type: ***boolean*** assignment (YES/NO)
Indicate if the object is mobile or stationary

hasMovability: type: ***boolean*** assignment (YES/NO)
Indicates if it is acceptable to change the location of object during the project

hasContainability: type: ***boolean*** assignment (YES/NO)
Indicates if the object can be used later to contain another object

hasFexibility: type: ***string*** assignment
Indicated the flexibility of object's shape (flexible/sizable/rigid)

hasOrientation: type: ***real number*** assignment
The angle by which the object is rotated when located on site referred to its reference construction product

hasPotenzialHazard: type: ***real number*** assignment
A value which represents a rough quantification of safety hazards related to the activity whose workspaces are simulated

		Properties	
Object Properties:	imposes:	**Domain**: Workspace **Range**: Constraint	**FU**
	hasTopologicalInteraction:	**Domain**: Workspace Range: Workspace	**TR**
	Define the interconnectivity level among two workspaces using a set of standardized values.		
	hasInteractionValue:	**Domain**: Workspace **Range**: Workspace	
	The Interaction Value defines the proximity level which occurs between two workspace in a range scale from 0 to 10.		
	isRequiredBy:	Domain: MicroSpace Range: Activity	**FU-TR**
	isRequiredBy:	Domain: MacroSpace Range: Phase	**FU-TR**

Follow the list of subclasses that describe all kinds of workspaces considered by the proposed ontolog*ical model.*

1.1 Sub-Class	**Macro WorkSpace**

Entity Definition: spaces located across sites in terms of site-layout requirements. Phases require Macro-Level workspaces, Activities require Micro-Level Workspaces.

1.1.1 Sub-Class	**StorageArea**

Entity Definition: The area required to keep material or tools for the time between they are delivered to site to the their use.

1.1.2 Sub-Class	**GenericArea**

Entity Definition: Whatever area which is required for the site layout organization to ensure Phase progress.

1.2 Sub-Class	**Micro WorkSpace**

Entity Definition: workspaces required by an activity which are located within the proximity of the components (construction products) being installed.

1.2.1 Sub-Class	**LaborCrew Space**

Entity Definition: represents the space required by the labor crew installing the construction product

generates:	**Domain**: LaborCrew Space **Range**: Hazard Space

1.2.2 Sub-Class	Equipment Space

Entity Definition: represents the space required by the equipment supporting either the Construction Product or the labor crews.

generates:	**Domain**: LaborCrew Space **Range**: Hazard Space	**TR**

1.2.3 Sub-Class	Hazard Space

Entity Definition: represents a hazard space generated by a Labor Crew space or Equipment space

1.2.4 Sub-Class	Protected Space

Entity Definition: Represents the space required to protect the construction product for a given time interval.

1.2.5 Sub-Class	Safety Space

Entity Definition: Represents a tolerance (safety distance) between two workspaces to prevent safety hazards such as collision between two spaces or a tolerance space from objects falling from height.

1.3 Sub-Class	Bound Space

Entity Definition: A bound space is a physical entity which represent the site boundary and objects that reside on site before the commencement of construction and hence have a known location on site. For examples Bound Spaces could comprise trees, existing buildings, marked areas on site such as unavailable, unsafe areas, life lines. They occupy space on site and their space is deduced from the total site land. Their Topological Interaction with other workspaces is disjoint and their Interaction Value with other workspaces is 0, the smallest.

In the figure below is printed the visualization of the ontology computation with all the aforementioned properties as extracted from the ontology editor *Protégé* by using the VOWL visualization functionalities.

5.8 Space ontology edited in Protégé and visualized with VOWL notations (Lohmann, 2014) in a force-directed layout. The graphical representation of OWL entities is made of visual elements. Blue circles represent classes and sub-classes; blue rectangles represent property labels of relations, the ones with no border represent datatypes, and green rectangles represents the property label of data-types (Snippet from *Protégé*)

chapter 6

A Building Information Model (BIM) is provided with a standardized interface for data exchange which is nowadays widely accepted in AEC Industry: the so-called IFC (Industry Foundation Classes) standard. The IFC serves as a basis for the exchange of building model data where building components are expressed in terms of objects with attributes. The proposed approach aims to make use of this information in order to link the knowledge base to the IFC-data schema.

To achieve this goal different methods have been proposed in literature with the common approach to extract and reuse building data (Dhillon et al., 2014). Unlike in these methods, the presented approach tries to merge entities and relations required for the space scheduling purposes as included in the IFC. This would be desirable in order to define an ontology-based common data structure that is at once completely integrated with the other previously presented modeling domains (scheduling and space ontologies). This is achieved by using the *ifcOWL ontology* (Buildingsmart, 2014) which is precisely meant to allow extensions towards other structured data sets made available using semantic web technologies such as the one used in our system (OWL language).

Since the i*fcOWL ontology* is quite complex because of the huge number of classes and properties it contains (Figure 6-1), an in-depth study has been carried out in order to filter only the required entities for achieving our construction planning purposes.

This approach produced a reduced ontology, here called 'Construction Product Ontology', for the description of the building structure in the proposed system together with the required building objects information. In the next paragraph is presented the sub-ontology after a brief exploration of the IFC structure.

6.1 IFC-based Building Model exploration

Industry Foundation Classes (IFC) represents an object-oriented format which provides a universal base for data exchange in building lifecycle. It has been developed by the

➔

6.1 Ontology metrics of IFC-schema visualized within the ontology editing environment (Snippet from *Protégé*)

International Alliance for Interoperability (IAI) on the basis of EXPRESS language as a part of the STEP standard [ISO 103030] for the product data exchange. The schema of IFC is quite complex. Concerning the scope of this work, in order to represent just an IFC building model, it is essential to show only the objects needed to formulate the construction planning domain.

The IFC building model is represented with a hierarchical spatial structure to organize a building project which is comprised in a so-called IfcProject. First element within the structure is included in the class of IfcSite which defines the area of land on which the project construction will be completed. A building (IfcBuilding) in IFC may have one or multiple stories (IfcBuildingStorey). Each building storey may have zero or multiple storeys. Each building storey may have assigned zero or more spaces with certain functions (IfcSpace) related to it -i.e., a building structure which has only one wall is a building with zero spaces-. For example, rooms in IFC are represented by the IfcSpaces class with a predefined PropertySet. Building elements and opening elements are represented as subtypes of spatial structure elements (IfcSpatialStructureElement). Each building element (IfcBuildingElement) has zero or more opening elements (IfcOpeningElement) i.e., a wall without any door or window has zero openings, whereas each opening element (like door, window) is attached to only one building element. IfcSpatialStructureElement links between building elements and upper structure of building (project, site, building, storey and space) as it defines spatial structure of a building and its parts.

Each IfcProduct, that is an abstract representation of any object that relates to a geometric or spatial context, is located in IfcGrid which is a planar design grid defined in 3D space used as an aid in locating structural and design elements. The position of the grid (ObjectPlacement) is defined by a 3D coordinate system. The relative placement of a product in relation to the placement of another product or the absolute placement of a product within the geometric representation context of the project is defined by the class IfcLocalPlacement. The particular geometric representation of a product is defined by IfcShapeRepresentation which includes several *RepresentationIdentifier*. It is derived from refers to geographic locations (IfcLocalPlacement) of building elements and their geometries. Geometric representation in IFC is built on solid geometries.

6.2 Topological structure

The entities and properties that have been selected are able to define a link between a given application, that may work on the construction planning domain by using the proposed Knowledge, and a BIM IFC-based, and are listed and graphically depicted below (Figure 6.4). Moreover, the main topological relations with the other sub-ontologies are specified.

a. IfcBuildingElement. This abstract class, that works as super-type, comprises all elements that are primarily part of the construction of a building, i.e., its structural and space separating system. Sub-types are IfcBeam, IfcColumn, IfcCurtainWall, and so forth. The classes included in the knowledge base of the presented system architecture are graphically represented in (Figure 7-3). They are defined as sub-classes of the master-class ConstructionProduct. In this way, for each of them a different ConstructionMethod will be defined and, in turn, will consist of a number of individuals grouped within classes (i.e., Labor-Crew-Space, Protected-Space, Equipment-Space, etc.) and properties, as described in the Construction Space Ontology. By doing so, each building objects composing the given BIM, will be allocated to a specific construction method via the transitive relation *isProducedBy*.

b. Then, within the IFC-based data structure, each element is specified by using a number of capabilities, mainly through *property sets* (e.g., Material, classification, documentation, boundary, coverings, etc.). For our scheduling purpose, in order to provide the system with information able to support the rule-engine in the automatic generation of the structural construction sequence, the objectified relationship IfcRelConnectsElements has been selected. It handles the connectivity between elements with a 1 to 1 relationship. The connectivity may be related to the shape representation of the connected entities by providing a connection geometry or a point connection with attributes with assigned values on X, Y

Ontology visualization within the editing environment

Objects types extracted from IFC

Objects property extracted from IFC

and Z-axis. If such a relation exists between two building objects, making a comparison between the height attribute on Z-axis, a rule-engine -which works by using such ontologies- could establish a time relation *IntervalBefore* between the items with a higher Z-value and the other one, automating the generation of a structural sequence.

c. Furthermore, for spatial planning purposes, a reduced BIM, which includes space availability in site -building objects and workspaces- should be generated in order to:

- simplify the representation of the geometries of building elements and
- find the optimal workspaces allocation in respect to the location of the building elements to which they are related.

Therefore, every IfcBuildingElement will be represented in the proposed space ontology as a bounding box, which shows the maximum extent of the body within the coordinated system established by the attributes of the IfcLocalPlacement class. The bounding box representation is the simplest geometric representation available. It is defined by a Corner, as three-dimensional Cartesian point, along with three oriented length measures defining the X, Y and Z values of the box as depicted in point (2) and (5) of Figure 6.3.

(1) The IfcGrid defines a placement coordinate system using the ObjectPlacement.

(2) The IfcBoundingBox is defined by the lower left corner (3) and the upper right corner (XDim, YDim, ZDim measured within the parent co-ordinate system)

(3) The entity *IfcRelConnectsStructuralMember* has properties describing the connection (e.g., SupportedLenght)

(5) Selected entities and properties from the IFC data structure to support the proposed model

6.3 Selected entities and properties in the ifcOWL ontology to link scheduling and space ontology to a BIM Model IFC-based

6.3 Introduction of BIM data in the KB

In specifying the Knowledge-Base, two sources of information are considered: building data from a given BIM according to aforementioned specifications, and user's experience. Their integration in the knowledge base by means of ontologies –OWL individuals-, has been designed as later specified and graphically represented in the figure below (Figure 6.4).

1. The configuration process starts having a Building Information Model to be processed, whatever the *Level of Development* (LOD) is made.

2. The given BIM, with parsing IFC schema defined in EXPRESS format (ISO 10303-11:20041)[1], is converted in OWL format (later called *ifcOWL**). Doing so, the expert system is provided with BIM in an ontological structure (classes, relations, properties, individuals).

3. The ifcOWL* is merged with our 'construction scheduling ontology' so that only the needed ontological components are captured (i.e., geometry information of building objects with their bounding boxes and local placements, structural connections among objects). This process is carried out by using a functionality of the editing environment *Protégé*. In this way OWL-individuals are generated using BIM project information.

4. The user, at this point, can add individuals to the ontology structure in terms of construction methods (see *Scheduling Ontology* in Chapter 4) and their workspaces with their related property sets (see *Space Ontology* in Chapter 5). It should be noted that, at the beginning of the configuration process, the 'construction process ontology' works as a *neutral model* which contains no specific object but only abstract entities.

5. At this point, the reasoner updates the ontology, the Knowledge base is uniquely populated with specific individuals and is ready to support reasoning mechanisms.

---

[1] ISO 10303 specifies a language by which aspects of product data can be defined. The language is called EXPRESS. ISO 10303-11:2004 also specifies a graphical representation for a subset of the constructs in the EXPRESS language. This graphical representation is called EXPRESS-G. EXPRESS is a data specification language as defined in ISO 10303-1. It consists of language elements that allow an unambiguous data definition and specification of constraints on the data defined.

6.4. Designed information-flow implemented to introduce BIM data (as filtered according to the Product ontology) and user experience within the Knowledge-Base

chapter 7

The representation of temporal relationships and temporal properties in the proposed model is fundamental considering that a schedule describes the construction process across time. In this regard, we are concerned with the exploration of temporal relationships between ontological entities. Therefore, we will adopt a discrete, linearly-ordered time domain, and we will focus on absolute time. The specific time relations we are interested in are those of Allen's interval algebra (Allen and Ferguson, 1997). Time dimension is incorporated into the model by associating time intervals to relationships between entities. Finally, the time ontology presented in (Cox and Little, 2016) have been chosen after reviewing the most used by other authors and customized for our purposes. In the following paragraph its structure is presented.

7.1 Topological temporal entities

Four key points have driven the modelling of the time variable in the knowledge-base with the aim to manage the time progression of entities in site and the translation of spatial constraints (e.g., workspaces conflict, etc.) into temporal constraints (time interval between conflicted entities) to be imposed into the schedule generation:

i. Temporal Relations between entities included in the other sub-ontologies;

ii. Representation of time positions;

iii. Duration of construction intervals;

iv. Temporal Reference System of the Expert System.

By doing so, unlike traditional Gantt Chart and network diagrams, temporal relationships with properties can be established between two entities and, moreover, each entity can be linked to more than one entity at the same time and with different relationships types.

Therefore, with reference to (i), the TIME ONTOLOGY is based on binary relations on intervals in order to represent the temporal information on which a schedule may be structured and on which the problem of automatic reasoning can rely. This is carried out in the ontology by using the class TemporalEntity which has object-properties able to assign temporal

instants to the individuals (e.g., building objects, workspaces, etc.) for the definition of their beginning and end (i.e. *hasBeginning* and *hasEnd*). This class defines the time properties of each Activity which, in turn, is related with other classes, i.e., Resources and Workspaces via the relationships *handles* and *requires* respectively. Such a network allows the ontology to automatically assign the same temporal interval of an Activity to the Resources and Workspaces which handle the Activity itself when a time interval has been assigned to only one of them. This is achieved assigning the transitive property to the object-properties *hasBeginning* and hasEnd.

Interval and Instant are two subclasses of TemporalEntity. This specification allows the system to mark a distinction between a Milestone and an Activity in the Schedule, in fact: a Milestone is an Instant -that can be seen as an interval with zero length- while an Activity is an Interval having an actual duration. The object-property *isA* is used to define these logical connections.

ProperInterval is a subclass of Interval and is used to define possible binary connections between two intervals (i.e. *intervalMeets*, *intervalOverlaps*, *intervalBefore*, *intervalDuring* etc.), and therefore between to Activities and their related classes, (e.g., resources and workspaces). In this respect, the interested relations are those of (Allen and Ferguson, 1997). Such classes are the most important for the construction schedule generation because they provide the rule-engine (see the following chapter) with the new relations that have to be established between all those entities included in the 'conflicts checking process'.

With reference to (ii), three classes describe the temporal position within a reference system and they all have an object property hasTRS to indicate the Temporal Reference System (TRS). TimePosition has properties to describe the position using both a number (i.e. a temporal coordinate), or a nominal value.

With reference to (iii), the duration of an interval can have different descriptions: class Duration describe the duration as a number. GeneralDurationDescription has different properties to specify a duration (e.g. hours, days, months, etc.) and DurationDescription fixes the temporal reference system used in the proposed ES to the Gregorian calendar. Specifications of classes with their interaction domains and range as computerized by using OWL language in *Protégé* are described below in Tab.5 and graphically depicted in Figure 7.1.

7.2 Specification of entities in the Time Ontology

Below, the structure of the Time Ontology (Cox and Little, 2016) which drives the *OnSI-TEsimu* in defining (1) *temporal relations*, (2) *temporal reference systems*, (2) *time position* with time unit and (4) *interval duration* is presented.

⬇

below and next page
Table 5
Specification of entities, relations and properties in the time ontology

T1. OWL Class	TEMPORAL ENTITY	
Entity definition:	This Class define a temporal interval which is assigned to each Activity.	
		Properties
	before:	**Domain**: Temporal Entity **Range**: Temporal Entity
	If a temporal entity T1 is before another temporal entity T2, then the end of T1 is before the beginning of T2	
	after:	**Domain**: Temporal Entity **Range**: Temporal Entity
Object Properties:	*If a temporal entity T1 is after another temporal entity T2, then the beginning of T1 is after the end of T2*	
	hasBeginning:	**Domain**: Temporal Entity **Range**: Instant
	hasEnd:	**Domain**: Temporal Entity **Range**: Instant
	hasDuration:	**Domain**: Temporal Entity **Range**: Duration
	This object property fixs the duration of a Temporal Entity expressed as a nominal value	

T2. OWL Class	INTERVAL	
Entity definition:	This class define a Temporal Entity with an extent or duration	
		Properties
Object Properties:	inside:	**Domain**: Interval **Range**: Instant

T3. OWL Class	INTERVAL RELATION	
Entity definition:	A Temporal Entity with a duration defined by using the class Interval which is supported by the seven interval relations listed below in object properties.	

		Properties
Object Properties:	intervalBefore:	**Domain and Range**: ProperInterval **Inverse Property**: *After*
	intervalMeets:	**Domain and Range**: ProperInterval **Inverse Property**: *MetBy*
	intervalOverlaps:	**Domain and Range**: ProperInterval **Inverse Property**: *OverlappedBy*
	intervalStarts:	**Domain and Range**: ProperInterval **Inverse Property**: *StartedBy*
	intervalDuring:	**Domain and Range**: ProperInterval **Inverse Property**: *Contains*
	intervalFinishes:	**Domain and Range**: ProperInterval **Inverse Property**: *FinishedBy*
	intervalEquals:	**Domain and Range**: ProperInterval **Inverse Property**: *Equals*

T4. OWL Sub-class	**DATE TIME INTERVAL**	
Entity definition:	This is a subclass of ProperInterval, defined using the multi-element DateTimeDescription	
Datatype Properties:	xsdDateTime:	**Domain**: DateTimeInterval **Range**: xsd:DateTime
Object Properties:		**Properties**
	hasDateTimeDuration:	**Domain**: DateTimeInterval **Range**: GeneralDateTimeDescription

T5. OWL Sub-class	**INSTANT**	
Entity definition:	This is a subclass of TemporalEntity which define a TemporalEntity with zero duration	
Datatype Properties:	inXsdDateTime: *Position of an instant, expressed using xsd:DateTime*	**Domain**: Instant **Range**: xsd:DateTime
Object Properties:	inTimePosition:	**Properties** **Domain:** Instant **Range:** TimePosition
	inDateTime:	**Domain:** Instant **Range:** GeneralDateTimeDescription

7.1 Time ontology edited in *Protégé* and visualized in a force-directed layout by means of made of visual elements. Blue circles represent the class hierarchy; blue rectangles represent relations, the green ones represent the property label of data-types, yellow rectangles represent the data assignment (Snippet from *Protégé*)

chapter 8

From the theory and application sections previously presented, it has emerged that construction planning and scheduling methodologies have known obstacles in their practical application mainly because of two facts: (a) the lack of interlinking between automated models, human planning, and digital building models (b) the lack of a standardized and codified knowledge Base able to support digital and automated construction planning processes. Moreover, the complexity to hold together factors at play in simulating and scheduling construction site activities and the complexity to grinding out a detailed construction process simulation often overwhelm solving capabilities of human planners even if they have deep knowledge which could provide decisive assistance to a hypothetical integrated system architecture.

However, although the total automation is being mooted in most of domains, the aforementioned components should work in synergy, bringing problem-solving strength to the table joining forces to give expression to a unique system architecture. This is complicated by the facts that human planners do not reason about construction process plans at the level of workspaces as well as their temporal-space allocations because of the off-putting amount of time to model workspaces geometries and simulate all of them across time as well as their interaction. Even if in the proposed ontologies has been clearly emerged the need to represent workspaces as a key concept.

In the light of these facts, this chapter represents the conclusion of the book in which it will be presented how ontologies could be able to support planning and scheduling processes. This is carried out presenting an Expert System Architecture supported by the presented ontologies developed by the author. Due to the object of this book only the architecture of the expert system is presented in order to clarify the importance to have available a computerized knowledge base to use to design reasoning mechanisms.

8.1 OnSITEsimu Expert System

OnSITEsimu is an *expert system* proposed in literature by the author which uses the ontologies presented in the previous chapters to works. In this book, which is strictly dedicated to

provide the reader with theory and applications of knowledge modelling and in particular of ontologies modelling, the architecture of such an expert system is presented in order to clarify how ontologies could be able to support automated reasoning processes.
OnSITEsimu has been designed for computerizing a novel approach to the construction spatial scheduling (BIM-compliant) and to act as a human expert to solve the complex problem to identify the earliest construction sequence considering the temporal space allocation of resources in site. It could be considered as an intelligent workspaces planner and activities scheduler of the construction process. Its main characteristics can be expressed in terms of blocks it is composed of:

1. **A knowledge base**

   The knowledge Base (KB) contains the domain-specific knowledge required to solve the problem. According to the goals, we consider the domain of building construction process focusing on site items, allocation and optimization of workspaces required to execute activities on site and their mutual relations as well as with the building objects. Once, such a knowledge -construction site entities- is organized so that it will be ready for use for the knowledge representation. It involves the preparation of a knowledge map, by using a specific *computational language*, and its own encoding in the KB. For this purpose, it has been used *ontologies* for modelling and the OWL language (Web Ontology Language) for computerizing –produce a script- such a KB and make it machine-readable. Just a KB alone is not of much use if there are no facilities for manipulating the knowledge to deduce something from the KB itself. This is carried out by the *Inference Engine*.

2. **An inference engine**

   The Inference Engine (IE) carries out the reasoning whereby the expert system reaches a solution. The system is designed to automate the generation of the *shortest schedule* given a number of input data (e.g., information on building objects, construction methods, resources and workspaces dimensions, etc.). It utilizes a set of spatial heuristics and predefined rules to schedule the activities in a chronological manner.

   To do this, the KB and information provided by both an user and a given Building Information Model have combined in order to infer new knowledge (i.e., the shortest possible total duration) by using a pre-designed rule-set.

3. **A user interface**

   Generally, the user interface is the block where the user interacts with the expert system. In the proposed model, two main user interface have been identified:

⬆
8.1 OnSITEsimu: Expert System architecture

- the *ontology-editing environment* that works as repository of the KB where the user adds individuals within the conceptual knowledge the system is equipped with. Those individuals describe construction methods he has in mind to use for each type of building object the given BIM is composed of;
- a *Virtual Construction* (VC) *application* able to simulate dynamic construction scenarios in a 4D BIM-based environment. This is crucial to visualize the result of the reasoning mechanisms (e.g., workspaces allocation and dimension as well as the activities scheduling), in terms of building production sequence with related workspaces availability.

When dealing with an expert system the first step is to define the development issues in terms of goals, role, and problem-solving approach. In order to guide the reader in understanding how to develop an ontology-based expert system the development issues of OnSITEsimu are listed below. The main ***goals*** of the model are:

- To develop a unified ontological model; that means to capture the main items, also called entities, in construction site, as well as their attributes and interrelationships. It is an attempt to propose a taxonomy for construction site concepts in order to wrap the existing product model in construction, the so-called Industry Foundation Classes (IFC) which doesn't contain that structure by now;

- In addition, the following categories should be considered extensively –properties- to support a fuller semantic representation of construction site activities: (a) *Building Products*, (b) *Resources and workspaces*, (c) *Time Relations* between entities, (d) *Scheduling Constraints*;
- Implement such a Knowledge-Based structure in a computer interpretable language in order to attach automated reasoning mechanisms.

Up to this point, the system thus might add pieces to a more extensive building project control in which constructability, site conditions management and productivity gain are the major objectives. By continuing:

- Introduce a theoretical spatial scheduling algorithm to find the temporal-space allocation of workspaces and define a new taxonomy to codify workspace typologies and their mutual relations;
- Reach *logical* and *technological interoperability* with a given Building Information Model (BIM) to be processed, according to the international standards of IFC;
- Automate the extraction of the shortest construction sequence.

Therefore, the scope of the listed goals can be considered to provide flexibility to construction process simulation by using a predefined structure without running manual modeling to represent and operate on site conditions.

The *role* of the model, as depicted in the following figure, is that the ES can be seen as an intelligent blackboard that allows to reason on a solution of a construction sequence providing both (a) visualization functionalities in terms of 4D BIM simulation, model-enrichment with site workspaces, flexibility and easily updating and (b) artificial intelligence mechanisms.

In defining the ***problem-solving approach***, the following points have been considered:

- the modeling approach of the construction process because of its structure across time. This is carried out dividing the problem into identifiable variables at play which have drawn up the modeling domains. These domains are: (a) Building-Model, (b) Workspaces-Model, (c) Scheduling-Model and (d) Time-Model. These ontologies, presented in the previous chapters will be filled in with specifications from the application domain (the given BIM) and it will compose the Knowledge Base ready to feed the Rule-Engine;
- the combination of such a knowledge base with a workspace planning algorithm and an activity scheduling algorithm. In this sense the system should combine the actions of an automated planner which guides the system when it comes to finding a solution of the optimal workspace allocation for all those construction methods the given BIM

8.2 Graphical representation of the general
architecture of the system and its functionality

requires and an automated solver which guides the scheduling strategy by using a rule-engine, again based on ontological reasoning.

8.2 Operational framework and model computerization

In the following paragraph, an overview which describes -step-by-step- the proper functioning of the Expert System is presented. It integrates a number of operation modules that encompass the identification of problem domain, the analysis of knowledge content as well as the development of reasoning processes aiming to automate production of the shortest schedule as graphically depicted in the figure below.

1. **Knowledge Base:**

   As it stated in the previous paragraph, all the necessary input data to launch the expert system are stored in a predefined Knowledge-Base in the form of ontologies by using classes, relationships and properties; all those traduced in *Web Ontology Language (OWL)* by using an ontology editor called *Protégé* which allows the system to define, operate and export the KB in the same format (*.OWL). Within the KB, it is important to draw a distinction between the *Construction Process Ontology* which contains generic entities -called neutral model- and 'individuals' which specify the entities for the given case study on

8.3 Operational modules for the proper functioning of the proposed system

which the system will need to operate – called *application domain* –. Their integration composes the Knowledge-Base which works as a structured knowledge repository for the system. The reason of this specification is that the knowledge representation is uncoupled from the user who simply fills it in.

a. Neutral Model

The Neutral Model provides the knowledge sources in the form of ontologies. It is generic, such as a *conceptual data model* which contains all the possible items a construction process simulation should consider such as type of site resources, workspaces, scheduling constraints, building elements, time relationships between entities and so forth. For each site entity, a predefined *property set* has been designed. Entities, relations and properties are not randomly assigned but they come from ontologies even because they are necessary to activate a Rule-Engine, which will be able to modify the KB on the inside by means of predefined rules. A badly structured knowledge-base would not allow the Rule-Engine to operate for generating the construction schedule. The result is a *construction process model* that includes nodes (entities) and intelligent relationships in the form of an ontology. In Figure 3-7 just a preview of some entities, as incorporated within the system, by means of the ontology editing environment, are shown in order to provide an intelligible model explanation.

b. Application Domain

The conceptual data model, represented by the ontologies, is transformed into the corresponding *physical data model*, called Application-Domain, with information originating from a specific building project, in the form of a Building Information

8.4 Ontology entities used in the model preparation to structure the knowledge base (Snippet from *Protégé*)

Model, on which the system will start to operate. Here, the user plays a pivotal role because he provides properties of the construction methods, he intends to use specifying entities within the ontology. The transformation takes place by using two clusters of information for building project control and workspaces control:

• **Products information for the** *Building Information Model*

Having the IFC ontology[1] at its disposal from the BIM application with which the digital Building Model was produced, the ES generates OWL-instances based on IFC-instances by using a IFC-to-OWL conversion process as codified by *Ghent University's 'IDLab'* in (IDLab 2013).

By doing so the *Product Ontology*, as specifically codified in the ontology of the chapter 6 for the proposed system, is filled in with building objects information, specifically selected, such as 'Local Placement' on X, Y and Z-axis of a constrained reference system (relative to the building grid axes), 'Bounding Box Geometry' that surrounds the shape of the building object itself (XDIM, YDIM, ZDIM) and 'Structural Connection' which is the appointed entity by the IFC-schema for representing structural supports

---

[1] Using an object-based inheritance hierarchy, IFC defines three abstract concepts as well as OWL Ontologies: object definitions, relationships, and property sets, whose specialized sub-classes are used to define a given BIM model.

or connecting elements (nodes) of a given Building Product with others. This step is crucial to highlight building objects types (e.g., IfcBeam, IfcColumn, IfcWall, IfcRoof, IfcRamp, IfcWindow, IfcSlab, IfcDoor, etc.) and to produce a simplified building model with a bounding box representation of the building objects as well as to automatically generate the structural schedule of the given BIM by using the Rule-Engine that will operate on such information.

• Construction methods information for the *workspaces control*

The Space Ontology (Chapter 5), as codified in the same language (*.OWL), represents the conceptual data model which captures the workspaces properties (e.g., Dimensions, Orientation, Movability Level, etc.), their mutual relations (e.g., Topological Interaction, Interaction Value, etc.) and relations with the building objects for each Construction Method. All those properties have to be specified by the user. In the ontology framework, the number of construction methods equal the number of Object Types, thanks to a one-to-one association ratio. This data structure supports an automated reasoning mechanism via a built-in algorithm, able to find the optimal workspaces configuration pattern within the site environment. Once the Knowledge Base -Neutral model plus Application Domain- is completed, it works as a structured data source for the reasoning mechanisms as described below.

Visualization of the given BIM to be processed

Object type:
Roofing Beam

Object type:
Precast column

Assieme: Assieme Telaio strutturale :
Building_Product_TRUSS

Object type:
Precast Beam

Given BIM to be processed
by using the proposed expert system

Object type:
Foundation plinth

Bounding Box Model (OnSITEsimu)

Built-in algorithm uploading within the BIM-based
modelling environment for sculpting geometries

From a detailed Building
Element to its 3D shape
which corresponds to the
outer shell

Modelli generici : Trave reticolare L1 :
Building_Product_TRUSS

8.6 Bounding Box model automatically generated by using a built-in algorithm in Dynamo

2. **'Bounding boxes model' generation:**

a. Workspaces Model

Once the user defines the workspaces properties of each resource assigned to a specific construction method according to the *Workspace-Ontology*, the geometries are automatically generated at one time by means of a specific built-in algorithm, specifically developed by using a visual programming environment -BIM-compliant- called *Dynamo*®. Therefore, the workspaces can be visualized in a BIM-based modelling environment *Autodesk Revit*, the same application the given BIM was produced, but only afterwards their optimal layout location will be determined.

b. Simplified building model

Just like the previous one and using the *Dynamo*'s script, a simplified building model, later called '*reducedBIM*' which utilizes the bounding boxes that surround the shape of 3D elements for each building objects is generated.

3. **Reasoning Mechanisms:**

Once things get this far, the expert system requires three different automated reasoning mechanisms:

• one able to find the *optimal workspace configuration pattern* for each construction method, based on constrains defined by the user and working on information include within the workspace ontology;

• one able to check *workspaces conflicts* once that all required workspaces geometries for each building objects are sculpted via the built-in algorithm;

• one able to produce a *construction schedule* on the base of a predefined rule-set, included in the Rule-Engine, that interacts with the knowledge-base by modifying relationships among the *individuals*.

An overview of each reasoning mechanism as well as their computerization are given below.

1. Workspaces planning process

Once the workspaces properties for each construction method are defined by the user (e.g., Dimensions, Interaction values, etc.), it remains to find their optimal layout allocation (constrain-based) with reference to the building objects. This is carried out by using a configurational analysis based on *Space Syntax Methodology* (Hillier 2007) which is a calculation technique in environment that enables parametric manipulation of geometries – *Grasshopper* –. The workspaces configuration pattern is generated in the form of a planar graph by using a bubble diagram, which

WORKSPACE	1-CONSTRUCTION METHOD	2-AMOUNT	3-Has Height - m	4-Has Length - m	5-Has Width - m	6-Has Area - m²	7-Has Capacity Dimensions - m	8-Has Level	9-Has Smallest unit X - m	10-Has Smallest unit Y - m	11-Has Smallest unit Z - m	12-Has Mobility	13-Has Movability	14-Has Containability	15-Has Flexibility	16-Has Orientation
Precast column	column installation	1	7,1	0,4	0,4	0,16	1,6	-	-	-	-	NO	NO	NO	rigid	0°
Hoisting machine	column installation	1	26	7	8,3	58,1	30,6	-	-	-	-	YES	YES	NO	rigid	0°
Truck	column installation	1	5	19,9	4,7	93,53	49,2	-	-	-	-	YES	YES	NO	rigid	0°
Labor crew	column installation	1	2	2,6	2,6	6,76	10,4	-	0,6	0,6	2	NO	YES	NO	flexible	0°
Lifting area	column installation	1	2	2	2	4	8	-	-	-	-	NO	YES	NO	sizable	0°

(A) Information from the construction workspace ontology

(B) Workspace graph representation

1 - Lifting Area   4 – Labor crew
2 – Truck   5 – Hoisting machine
3 – Precast column

4 m² 58,1 m² 79,43 m² 6,76 m² 0,16 m²

Hoisting Machine   Truck   Column
Labor crew   Lifting area

(C) Circles representing workspaces on the graph

Topological interaction

(D) Workspaces connectivity graph

Depth ► 2
Depth ► 1
Precast column
Depth ► 0

(E) Space syntax analysis, re-distribution of workspaces on depth levels from the column point of view

Y
Relative
Reference System
Object-oriented
(X_1,Y_1)
$(0,0)$
(X_4,Y_4)
(X_5,Y_5)
X
(X_2,Y_2)

(F) From *Grasshopper*: Generation of the workspaces allocation pattern - planar graph

Building Object ▼
Column

(G) Workspaces geometry representation in their optimal allocation as automatically sculpted by using coordinated from (F)

8.7 Graphical outputs of the workspaces planning process to define the spatial configuration pattern of each construction methods

is deducted by Nourian et al. (2013) algorithm and especially customized and integrated for our model. Such a built-in algorithm works on *Grasshopper*, a parametric modelling tool for CAD Environment. In this way, the expert system extracts the coordinates of workspaces allocations on the X-axis and Y-axis at the same height (Z-axis) of their connected building object but, this time, by using a relative reference system, centered on the reference object.

The figure below shows a preview of the graphical results of the workspaces planning process for a specific construction method –the column installation- included in the case study:

- on the left side, the workspaces geometries together with the building object are automatically sculpted and randomly located in a BIM-modelling environment,
- then their optimal allocation is generated and finally
- the workspaces a reallocated by using the new coordinates as suggested by the algorithm itself.

2. **Automated workspace conflict checking process and 4D simulation**

At this point, the system is carrying a building model -stored in the form of ontologies and visualized within a BIM application- which includes the building objects and all the needed workspaces in their optimal layout allocation as defined by the previous planning process. In view of the axiom that two activities cannot run concurrently if their workspaces clash, the system requires a *workspace conflict detection process*. This process is carried out by temporarily getting out of the ontologies, by means of a time-based clash test supported by the scheduled structural sequence as it was generated by the Rule-Engine. This choice is judged to optimize the number of detected conflicts since, for example, the workspace that handles the activity of a column installation will certainly conflict with the workspace required by the activity of the beam installation structurally linked with the same column. If been checked, it would be an inconsistent conflict for the scheduling purpose.

Therefore, after completing the schedule overlapping identification, the physical conflict verification of the workspaces starts to review the conflicts. The activities that do not have overlapping schedules are excluded from the checking process because the two activities do not have the workspaces concurrently. The physical conflict amongst the workspaces is determined by an *adjacency distance* that calculates the shortest distance on the external specific surface between two workspaces. Once the workspace conflict verification process is completed a structured *clash report* is extracted in the form of strings with the (**.txt*) extension which is once again

⬆
8.8 Graphical output of the workspaces conflict detection process

imported within the ontology in the form of workspaces properties in order to provide the system with all those required data to activate again the Rule-Engine able to solve, at this time, the conflicts by establishing a new time relations between all the conflicting activities by using a pairwise comparison. The Report-Tab includes the following data for each clash result: Item ID, Clash Point, Start Date, End Date, Distance, Grid Location.

Such a conflict checking process together with the checking rules has been computerized in *Navisworks®* (4D BIM-based simulation environment) and integrated within the system by establishing a coherent information flow.

Figure 8.8 shows an extrapolation of detected conflicts, considering only four columns, from the case study during their installation phase.

3. **Construction Schedule Generation**

The Knowledge-Base combined with a Rule-Engine compose the core of the expert system. This latter is used to apply, on the KB, a set of predefined *reasoning rules* that work as artificial intelligence for the system in order to generate the structural sequence first and then the earliest construction completion sequence, also called 'shortest schedule' of the given BIM.

The Rule Engine has been implemented in the same ontology-based editing environment *Protégé®* due to the fact that, doing so, rules can automatically modify relations and properties within the knowledge base. Rules provide the description of how to solve the scheduling problem by using an IF-THEN structure that relates given information – *facts*

8.9 Structural sequence visualized in 4D BIM-based simulation environment by using data provided by the knowledge base after operating the rule-engine

– in the IF part to some actions in the THEN part. The rules are written by using the *Semantic Web Rule Language* (SWRL) for the reason that it is well integrated with the *OWL Language* used to make the knowledge base machine-readable.

From a scheduling point of view two main characteristics of the model must be clarify:

- The model considers *fixed activity duration* without 'float' also know in scheduling as 'slack'. In further detail, we do not assume that the goal is to appropriately allocate the available resources, but the system explores the solution of resource allocation in which activity durations only depend on the type and number of resources allocated to the activities.

- It is a *resource allocation model*, that schedule activities when there are constraints on the total amount of available resources. For example, if two column installation activities can run together because the system did not check out conflicts between their workspaces, but both require the crane availability which is however considered a Unit-Capacity-Resource able to support one activity at a time, the Rule-Engine schedules the activities by establishing an 'IntervalBefore' time-relation in order to execute them in two consecutive time period. An IF-THEN rule within the Rule-Engine manages this scheduling purpose.

The Rule-Engine acts in two different time periods, according to the scheduling algorithm, but consequential in order to generate two different construction schedules as appointed below.

i. Structural Construction Sequence

On the basis of data provided in the IFC-schema, and in particular considering the information included within the entity IfcRelConnectsStructuralMember, which defines all needed properties describing the connection between structural members (Buildingsmart, 2014), the system operates, by means of the Rule-Engine, by establishing new time relations between building objects defining the structural sequence (e.g., building items included in the Objects Family IfcBeam will be related with IfcColumn via IntervalBefore time-relation, as well as IfcWall with IfcWindow and IfcDoor, etc.). Once the knowledge base has been updated by means of new time relations according to the time-ontology, the schedule is visualized in a 4D BIM simulation environment that convert the scheduling data in visual data by means of a 4D simulation, as depicted in the figure below.

ii. Earliest Construction Completion Sequence

Having results of the 'structural schedule' as previously defined in point (i), the system proceeds by expanding such a schedule with data regarding the detected workspaces conflicts -included in the 'clash report'-, primarily because the conflict checking process is carried out on the structural sequence enriched with required workspaces. More specifically, in the same way as the previous schedule, each conflict is converted in a new temporal relationship among all those entities (Building Objects, Resources, Workspaces, etc.) somehow linked with the conflicted workspaces. Once again, this is carried out by using an IF-THEN rule implemented in the rule-based reasoning-machine. This rule, combined with the others (e.g., resource levelling), updates the ontology with new relationships or properties which compose the final schedule. It represents the *Earliest Construction Completion Sequence*, the goal of the proposed expert system. Therefore, to recap, the expert system architecture is graphically represented in the figure below. The system architecture based on the ontologies clarifies the importance to have a predefined knowledge-base able to support automated reasoning mechanism.

➡

8.10 Operational Framework of the proposed Expert System from the point of view of the ontology which drives the system to operate by means of a knowledge-base and a rule-engine

REFERENCES

Akbas, R. 2004. *Geometry-Based Modeling and Simulation of Construction Processes*. STAN-FORD UNIVERSITY, CIFE Technical Report.

Akinci, B., and J. Kunz, R. Levitt M. Fischer. 2000. "Representing Work Spaces Generically in Construction Method Models." CIFE Working paper (Stanford Univeristy) DOI: 10.1061/(AS CE)0733-9364(2002)128:4(296).

Akinci, B., H. Karimi, A. Pradhan, C. Wu, and G. Fichtl. 2010. "CAD and GIS interoperability through semantic web services." J*ournal of Information Technology in Construction* 13: 39–55. http://www.itcon.org/2008/3.

Akinci, B., M. Fischer, and J. Kunz. 2000c. "Automated Generation of Work Spaces Required." Edited by Stanford Univeristy. *CIFE Working Paper*.

Allen, J., and G. Ferguson. 1997. *Actions and events in interval temporal logic*. ed., Kluwer O. Stock. http://dx.doi.org/10.1007/978-0-585-28322-7_7.

Atkinson, C., M. Gutheil, and K. Kiko. 2007. "On the Relationship of Ontologies and Models." Edited by Chair of Software Technology. ISBN: 978-3-88579-190-4.

Azhar, S., M. Hein, and B. Sketo. 2008. "Building Information Modeling (BIM): benefits, risks and challenger." *Proceedings of the 44th ASC Annual Conference*. Alabama. 627-634.

Baader, F., D. Calvanese, D. Mcguiness, D. Nardi, and P., F. Patel-Schneider. 2003. *Description Logic Handbook: Theory, Implementations, and Applications*. Cambridge: Cambridge University Press.

Bandwidth Market (2008), Bandwidth Market, Glossary http://www.bandwidthmarket.com/glossary/A7.html.

Bargstädt, H., and A. Elmahdi. 2010. "Automatic generation of workspace requirements using qualitative and quantitative description." *Proceedings of the 10th International Conference on Construction Applications of Virtual Reality*. Japan: K. Makanae, N. Yabuki, K. Kashiyama. ISBN: 978-4-9905537-0-8 C3000. http://convr2010.com/proceedings_contents

Beetz, J., J. Van Leeuwen, and B. De Vries. 2009. "IfcOWL: A case of transforming EXPRESS schemas into ontologies." *Journal Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 23 (01): 89-101.DOI: 10.1017/S0890060409000122.

Benevolenskiy, A., K. Roos, P. Katranuschkov, and J. Scherer R. 2012. "Construction processes configuration using process patterns." *Advanced Engineering Informatics* (Elsiever) (26): 727-736.DOI: 10.1016/j.aei.2012.04.003.

Bordoli, A., and D. Baldwin. 2014. *A Handbook for Construction Planning and Scheduling*. West Sussex: John Wiley & Sons. ISBN: 978-0-470-67032-3.

Brickley D, Guha RV (2003) RDF Vocabulary Description Language 1.0: RDF Schema. W3C Working Draft. http://www.w3.org/TR/PR-rdf-schema.

Broekstra J, Klein M, Decker S, Fensel D, van Harmelen F, Horrocks I (2001) Enabling Knowledge Representation on the Web by Extending RDF Schema. In: Shen VY, Saito N (eds) 10th International World Wide Web Conference (WWW10). Hong Kong, pp 467–478. DOI: 10.1145/371920.372105.

Buildingsmart. 2014. *Industry Foundation Classes Release 4* (IFC4). http://www.buildingsmart-tech.org/ifc/IFC4/final/html/link/structural-connectivity.htm.

Cai, X., Su, H. 2014. "Life Cycle Approach to Construction Workspace Modeling and Planning." *Journal of Construnction Engineering and Management* (ASCE) March. DOI: 10.1061/(ASCE)CO.1943-7862.0000855

Chau, K., W., M. Anson, and J., P. Zhang. 2004. "Four-dimensional visualization of construction scheduling and site utilization." *Journal of Construction Engineering and Management* (ASCE) 130: 598–606. DOI: 10.1061/(ASCE)0733-9364(2004)130:4(598).

Chavada, R., N. Dawood, and M. Kassem. 2012. "Construction workspace management: the development and application of a novel nD planning approach and tool." *Journal of Information Technology Construction (ITcon)* 12: 213–236. https://www.itcon.org/paper/2012/13.

Cheng, T., and J. Teizer. 2013. "Real-time resource location data collection and visualization technology for construction safety and activity monitoring applications." *Automation in Construction* 34: 3–15. DOI: 10.1016/j.autcon.2012.10.017.

Cherneff, J., M., R. Logcher, and D. Sriram. 1991. "Integrating CAD with construction-schedule generation." *Journal of Computing in Civil Engineering* (ASCE) 5 (1): 68-84. DOI: 10.1061/(ASCE)0887-3801(1991)5:1(64).

CIOB. 2011. *Guide to Good Practice in the Management of Time in Complex Projects*. Oxford: Wiley-Blackwell. ISBN: 978-1-444-32961-2.

Connor, O., and D. Martin. 2009. *Importing Data into Protégé-OWL*. Stanford Center for Biomedical Informatics Research. http://protege.stanford.edu/conference/2009/slides/ImportingDataProtegeConference2009.pdf.

Cordier, J., M., and T. Porte. 1989. *Shape Theory: Categorical Methods of Approximation*. New York: Dover publications. ISBN: 9780486466231.

Cox, S., and C. Little. 2016. "Time Ontology in OWL." http://www.w3.org/2006/time.

Darwiche, A., R. Levitt, and B. Hayes-Roth. 1988. "OARPLAN: Generating Project Plans by Reasoning about Objects, Actions and Resources." *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 2 (3): 169-181. DOI: 10.1017/S0890060400000639.

Dawood, N., and S. Sikka. 2009. "Development of 4D based performance indicators in construction industry." *Engineering Construction and Architectural Management* (Wiley) 16: 438–458. DOI: 10.1108/09699980910988357.

Dawood, N., and Z. Mallasi. 2006. "Construction workspace planning: assignment and analysis utilizing 4D visualization technology,." *Computer-Aided Civil and Infrastructure Engineering* (Elseiver) 21 (7): 498–513. DOI: 10.1111/j.1467-8667.2006.00454.x.

Dawood, N., D. Scott, E. Sriprasert, and Z. Mallasi. 2005. "The virtual construction site (VIRCON) tools: An industrial evaluation." *Journal of Information Technology in Construction (Special Issue: From 3D to nD modelling)* 10: 43-54. http://www.itcon.org/2005/5.

Dhillon, R., K., M. Jethwa, and H., S. Rai. 2014. "Extracting Building Data from BIM with IFC." International Journal on Recent Trends in Engineering and Technology 11 (1). DOI: 01.IJRTET.11.1.1500.

Domingue J (1998) Tadzebao and WebOnto: Discussing, Browsing, and Editing Ontologies on the Web. In: Gaines BR, Musen MA (eds) 11th International Workshop on Knowledge Acquisition, Modeling and Management (KAW'98). Banff, Canada, KM4:1–20. http://ksi.cpsc.ucalgary.ca/KAW/KAW98/domingue/.

Drogemulle, H., and R. Schevers. 2005. "Converting the Industry Foundation Classes to the Web Ontology Language." 2005 First International Conference on Semantics, Knowledge and Grid, Beijing (China), 27-29 Nov. 2005. DOI: 10.1109/SKG.2005.59.

Dudáš, M., O. Zamazal, and V. Svátek. 2014. "Roadmapping and nav- igating in the ontology visualization landscape." Edited by Springer. *Knowledge Engineering and Knowledge Management* (K. Janowicz, S. Schlobach, P. Lambrix, and E. Hyvönen) 8876: 137–152. DOI: 10.1007/978-3-319-13704-9_11.

Durkin, J. (1994), *Expert Systems: Design and Development*, Macmillan, New York. ISBN: 0-02-330970-9.

Dzeng, R., J., and I., E. Tommelein. 1997. "Boiler erection scheduling using product models and case-based reasoning,." *Journal of Construction Engineering and Management* (ASCE) 123 (3). DOI: 10.1061/(ASCE)0733-9364(1997)123:3(338).

Echeverry, D. 1991. "Factors for generating initial construction schedules." Department of Civil Engineering, University of Illinois, PhD thesis, Urbana-Champaign. https://apps.dtic.mil/dtic/tr/fulltext/u2/a243662.pdf.

Elbeltadi, E., T. Hegazy, and A. Eldosouky. 2004. "Dynamic layout of construction temporary faciltties considering safety." J*ournal of Construction Engineering and Management* 130 (4): 534-541. DOI: 10.1061/(ASCE)0733-9364(2004)130:4(534).

El-Diraby. 2013. "Domain ontology for construction knowledge." *Journal of Construction Engineering and Management* (ASCE) 768–784. DOI: 10.1061/(ASCE)CO.1943-7862.0000646.

El-Diraby, T., and h. Osman. 2011. "Automation in Construction." *A domain ontology for construction concepts in urban infrastructure products* 20 (8): 1120–1132. DOI: 10.1016/j.autcon.2011.04.014.

Ernstrom, B., and et al. 2006. *The contractors' Guide to BIM*. Associated General Contractors of America. https://www.engr.psu.edu/ae/thesis/portfolios/2008/tjs288/Research/AGC_GuideToBIM.pdf

Farquhar A, Fikes R, Rice J (1997) The Ontolingua Server: A Tool for Collaborative Ontology Construction. International Journal of Human Computer Studies 46(6):707–727. DOI: 10.1006/ijhc.1996.0121.

Fensel D, van Harmelen F, Horrocks I, McGuinness DL, Patel-Schneider PF (2001) OIL: An ontology infrastructure for the Semantic Web. IEEE Intelligent Systems & their applications 16(2):38–44. DOI: 10.1109/5254.920598.

Ferguson, J., Allen. 1997. "Actions and Event in interval temporal logic." http://dx.doi.org/10.1007/978-0-585-28322-7_7.

Fortineau, V., T. Paviot, L. Louis-Sidney, and S. Lamouri. 2012. "SWRL as a rule language for ontology-based models in power plant design." DOI: 10.1007/978-3-642-35758-9_53.

Friedman, E., H. 2003. *Rule-Based Systems in Java – JESS IN ACTION*. Greenwich: Manning Publications. ISBN: 9781930110892.

Fu, B., N., F. Noy, and M., A. Storey. 2013. "Indented tree or graph? a usability study of ontology visualization techniques in the context of class mapping evaluation." Edited by Springer. *The Semantic Web* (H. Alani, L. Kagal, A. Fokoue, P. Groth, C. Biemann, J. Parreira, L. Aroyo, N. Noy, C. Welty, and K. Janowicz) 8218: 117–134. DOI: 10.1007/978-3-642-41335-3_8.

Gambatese, J., A., M. Behm, and J., W. Hinze. 2005. "Vviability of designing for construction worker safety." *Journal of Costruction Engineering and Management* (ASCE) 131 (9): 1029-1036. DOI: 10.1061/(ASCE)0733-9364(2005)131:9(1029).

Gasevic, D., Djuric, D., Devedzic, V., (2009). Model Driven Engineering and Ontology Development. Springer, ISBN: 978-3-642-00281-6; DOI: 10.1007/978-3-642-00282-3.

Genesereth MR, Fikes RE (1992) *Knowledge Interchange Format. Version 3.0. Reference Manual*. Technical Report Logic-92-1. Computer Science Department. Stanford University, California. http://logic.stanford.edu/publications/genesereth/kif.pdf.

Gómez-Pérez, A. & Corcho, O. (2002), "Ontology languages for the Semantic Web," IEEE Intelligent Systems, Vol. 17, no. 1, pp. 54–60. DOI: 10.1109/5254.988453.

Grosso, W., E., H. Eriksson, R., W. Fergerson, J., H. Gennari, S., W. Tu, and M., A. Musen. 1999. "Knowledge modeling at the millennium: The design and evolution of Protege-2000." Technical Report, Stanford University, Institute for Medical Informatics, Standford.

Gruber, T., R., (1992) Ontolingua: A Mechanism to Support Portable Ontologies. Technical report KSL-91-66, Knowledge Systems Laboratory, Stanford University, Stanford, California.

Gruber, T., R., (1993b) *Toward principles for the design of ontologies used for knowledge sharing*. In: Guarino N, Poli R (eds) International Workshop on Formal Ontology in Conceptual Analysis and Knowledge Representation. Padova, Italy. (Formal Ontology in Conceptual Analysis and Knowledge Representation) Kluwer Academic Publishers, Deventer, The Netherlands. https://tomgruber.org/writing/onto-design.pdf.

Gruber, T., R. 1995. "Toward principles for the design of ontologies used for knowledge sharing?" Edited by Elseiver. *International Journal of Human-Computer Studies* 43 (5): 907–928. DOI: 10.1006/ijhc.1995.1081.

Gu, N., and K. London. 2010. "Understanding and facilitating BIM adoption in the AEC industry." *Automation in Construction* 19: 988–999. DOI: 10.1016/j.autcon.2010.09.002.

Guo, H., H. Li, G. Chan, and M. Skitmore. 2012. "Using game technologies to improve the safety of construction plant operations." *Accident Analysis and Prevention* 48: 204–213. DOI: 10.1016/j.aap.2011.06.002.

Guo, S., J. 2002. "Identification and resolution of work space conflicts in building construction." *Journal of Construction Engineering and Management* (ASCE) 128: 287–295. DOI: 10.1061/(ASCE)0733-9364(2002)128:4(287).

Guo, S., S., and C., W. Chan. 2011. "A comparison and analysis of some ontology visualization tools." Edited by SEKE. *Proceedings of the 23rd International Conference on Software Engineering & Knowledge Engineering*. Knowledge Systems Institute Graduate School. 357–362. ISBN: 9781510841611.

Halpin, H. & Shepard, H. (2006), Evolving Ontologies from Folksonomies:Tagging as a Complex System [Online] http://www.ibiblio. org/hhalpin/homepage/notes/taggingcss.html

Hanson, B., and J. Hillier. 1984. *The Social logica of space*. Cambrige University Press. DOI: 10.1017/CBO9780511597237.

Hegazy, T., M., and E. Elbeltagi. 1999. "EvoSite: Evolution-based models for site layout planning." J*ournal of Computing in Civil Engineering* (ASCE) 13 (3): 198–206. DOI: 10.1061/(ASCE)0887-3801(1999)13:3(198).

Heijst, G., A., Th. Schreiber, and B., J. Wielingua. 1997. "Using Explicit Ontolgies in KBS Development." *International Journal of Human-Computer Studies* (Academic Press) 46: 183-292. DOI: 10.1006/ijhc.1996.0090.

Hillier, B. 2007. *Space is the Machine*. London: Cambridge University Press. ISBN 978-0-9556224-0-3.

Hori, M., Y. Nakamura, H. Satoh, K. Maruyama, T. Hama, S. Honda, T. Takenaka, and F. Sekine. 1995. "Knowledge-level analysis for elicting composable scheduling knowledge." *Artificial Intelligence in engineering* (Elsiver Science Limited) 9: 253-264. DOI: 10.1016/0954-1810(95)00004-3.

Horridge, M. 2011. "A Practical Guide To Building OWL Ontologies Using Protégé 4 and CO-ODE Tools Edition 1.3." Guide, The University Of Manchester. https://gmakris.files.wordpress.com/2017/06/protegeowltutorialp4_v1_3.pdf

Horrocks, I. 2004. *SWRL: A Semantic Web Rule Language Combining OWL and RuleML.* https://www.w3.org/Submission/SWRL/.

Horrocks I, Fensel D, Harmelen F, Decker S, Erdmann M, Klein M (2000) OIL in a Nutshell. In: Dieng R, Corby O (eds) 12th International Conference in Knowledge Engineering and Knowledge Management (EKAW'00). Juan-Les-Pins, France. (Lecture Notes in Artificial Intelligence LNAI 1937) Springer- Verlag, Berlin, Germany, pp 1–16. DOI: 10.1007/3-540-39967-4_1.

IDLab, Ghent Univeristy's. 2013. IFC-to-RDF converter. IDLab. https://github.com/IDLabResearch/IFC-to-RDF-converter/wiki/IFC-to-RDF.

Jagbeck, A. 1994. "MDA Planner: Interactive Planning Tool Using Product Models and Construction Methods." *Journal of Computing in Civil Engineering* 8 (4): 536-554. DOI: 10.1061/(ASCE)0887-3801(1994)8:4(536).

Kalinowski, K. 2012. "Multistage decision making process of multicriteria production scheduling." *Journal of Machine Engineering* 12 (3). ISSN (online): 2391-8071. http://yadda.icm.edu.pl/baztech/element/bwmeta1.element.baztech-3b007c94-10cc-4d71-bd24-135074647948

Kano, N. 1990. "A knowledge-based system for construction planning and scheduling: a prototype system based on the down-from the-top methodology." *The 7th International Symposium on Automation and Robotics in Construction*. Bristol. 303-311. DOI: 10.22260/ISARC1990/0039.

Karp PD, Chaudhri V, Thomere J (1999) *XOL: An XML-Based Ontology Exchange Language*. Version 0.3. Technical Report. http://www.ai.sri.com/~pkarp/xol/xol.html.

Kartam, N., A., and R., E. Levitt. 1990. "Intelligent planning of construction projects." *Journal of Computing in Civil Engineering* (ASCE) 4 (2): 155-176. DOI: 10.1061/(ASCE)0887-3801(1990)4:2(155).

Kassem, M., N. Dawood, and R. Chavada. 2015. "Construction workspace management within an Industry Foundation Class-Compliant 4D tool." *Automation in Construction* 52: 42-58. DOI: 10.1016/j.autcon.2015.02.008.

Katifori, A., C. Halatsis, G. Lepouras, A. Vassilakis, and E. Giannopoulou. 2007. "Ontology visualization methods – a survey." *ACM Computer Survey* 39 (4). DOI: 10.1145/1287620.1287621.

Katifori, A., E. Torou, C. Halatsis, G. Lepouras, and Vassilakis C. 2006. "A comparative study of four ontology visualization techniques in Protégé: Experiment setup and preliminary results." *International Conference on Information Visualisation*. IEEE. 417–423. DOI: 10.1109/IV.2006.3.

Kim Jongeling, J., M. Fischer, C. Mourgues, and T. Olofsson. 2008. "Quantitative analysis of workflow, Temporary Structure Usage, and Productivity Using 4D Models." *Automation in Construction* 17: 780-791. DOI: 10.1016/j.autcon.2008.02.006.

Kogut P, Cranefield S, Hart L, Dutra M, Baclawski K, Kokar M, Smith J (2002) UML for Ontology Development. The Knowledge Engineering Review 17(1):61–64. DOI: 10.1017/S0269888902000358.

Konig, M., and A. Marx. 2013. "Modeling and simulating spatial requiremts of construction activities." *Winter simulation conference 2013*. R. Pasupathy et al. DOI: 10.1109/WSC.2013.6721694.

Krishnamoorthy, C., S., and S. Rajeev. 1996. *Artificial Intelligence and Expert Systems for Engineers*. CRC Press LLC. ISBN: 978-0-8493-9125-5.

Lambert, D., and C. Nowak. 2008. "The Mephisto Conceptual Framework" *Command, Control, Communications and Intelligence Division DSTO Defence Science and Technology Organization*. Australia. https://apps.dtic.mil/sti/pdfs/ADA515394.pdf.

Lanzenberger, M., J. Sampson, and M. Rester. 2010. "Visualization in ontology tools." *Proceedings of the International Conference on Complex, Intelligent and Software Intensive Systems*. IEEE. 705–711. DOI: 10.1109/CISIS.2009.178.

Lee, K., J., H., W. Kim, J., K. Lee, and T., H. Kim. 1998. *Case and constraint-based project planning for apartment construction*. Vol. 19. 1 vols. AI Magazine. DOI: 10.1609/aimag.v19i1.1350.

Liao, S.H. 2005. "Expert System methodologies and applications - a decade review from 1995 to 2004." *Expert Systems with Applications* (Elsiever) (28): 93-103. DOI: 10.1016/j.eswa.2004.08.003.

Lima, C., T. El-Diraby, and J. Stephens. 2005. "Ontology-based optimi- sation of knowledge management in e-construction." *Journal of Information Technology in Construction* 10: 305–327. http://www.itcon.org/2005/21.

Little, S., and C. Cox. 2016. *Time Ontolgy in OWL*. http://www.w3.org/TR/2016/WD-owl-time-20160712/.

Lohmann, S., S. Negru, F. Haag, and E. Ertl. 2014. "Visualizing Ontologies with VOWL." *Semantic Web* (IOS Press). DOI: 10.3233/SW-150200.

Lopez, M., F., A., G. Pérez, and N. Juristo. 1997. "Methontology: From ontological art towards ontological engineering." *Ontological Engineering Spring Symp. Series*. American Asso- ciation for Artificial Intelligence. 33-40. www.aaai.org/Papers/Symposia/Spring/1997/SS-97-06/SS97-06-005.pdf.

Luke S, Heflin JD (2000) SHOE 1.01. Proposed Specification. Technical Report. Parallel Understanding Systems Group. Department of Computer Science. University of Maryland. http://www.cs.umd.edu/projects/plus/SHOE/

Ma, Z., Q. Shen, and J. Zhang. 2005. "Application of 4D for dynamic site layout planning and management of construction projects." *Automation in Construction* 14: 369-381. DOI: 10.1016/j.autcon.2004.08.011.

MacGregor R (1991) Inside the LOOM clasifier. SIGART bulletin 2(3):70–76

Maedche A, Motik B, Stojanovic L, Studer R, Volz R (2003) Ontologies for Enterprise Knowledge Management. IEEE Intelligent Systems 18(2):26–33. DOI: 10.1109/MIS.2003.1193654.

Mallasi, Z. 2006. "Dynamic quantification and analysis of the construction workspace congestion utilising 4D visualisation." *Automation in Construction* (Elseiver) (15): 640–655. DOI: 10.1016/j.autcon.2005.08.005.

Manola F, Miller E (2003) RDF Primer. W3C Working Draft. http://www.w3.org/TR/rdf-primer/

McGuinness, N., F. 2001. "Ontology Development 101: A Guide to Creating Your Fisrt Ontology." Standford Univerisity, Standford. https://protege.stanford.edu/publications/ontology_development/ontology101.pdf.

Mikulakova, E., M. Konig, E. Tauscher, and Beucke. K. 2010. "Knowledge-based schedule geenration and evaluation." *Advanced engineering Informatics* 24: 389-403. DOI: 10.1016/j.aei.2010.06.010.

Mohan, S. 1990. "Expert systems applications in construction management and engineering." *Journal of Construction Engineering and Management* (ASCE) 116 (1): 87-99. DOI: 10.1061/(ASCE)0733-9364(1990)116:1(87).

Moon, H., H. Dawood, and L. Kang. 2014. "Development of workspace conflict visualization system using 4D object of work schedule." *Advanced Engineering Informatics* (Elseiver) 28: 50–65. DOI: 10.1016/j.aei.2013.12.001.

Moon, H., H. Kim, C. Kim, and L. Kang. 2014. "Development of a schedule workspace interference management system simultaneously considering the overlap level of parallel schedules and workspaces." *Automation in Construction* 39: 93–105. DOI: 10.1016/j.autcon.2013.06.001.

Moon, H., H. Kim, V. R. Kamat, and Kang L. 2015. "BIM-based Construction Scheduling Method Using Optimization Theory for Reducing Activity Overlaps." *Journal of Computing in Civil Engineering* (ASCE) 29 (3). DOI: 10.1061/(ASCE)CP.1943-5487.0000342.

Moon, H., L. Kang, N. Dawood, and S. Ji. 2009. "Configuration method of health and safety rule for improving productivity in construction space by multi-dimension CAD system." *Proceedings of ICCEM/ICCPM*. Korea. 27-30.

Morad, A., A., and Y., J. Beliveau. 1991. "Knowledge-based planning system." *Journal of Construction Engineering and Management* (ASCE) 117 (1): 1-12. DOI: 10.1061/(ASCE)0733-9364(1991)117:1(1).

Motawa, I., and A., Almarshad. 2012. "A Knowledge-based BIM system for building maitenance." *Automation in Construction* (Elseiver) 29: 173-182. DOI: 10.1016/j.autcon.2012.09.008.

Motta, E. 2000. "The knowledge modelling paradigm in knowledge engineering." *Handbook of Software Engineering and Knowledge Engineering* (World Scientific Publishing) 1: 1-29. DOI: 10.1142/9789812389718_0024.

Mun, D., and K. Ramani. 2011. "Knowledge-based part similarity measurement utilizing ontology and multi-criteria decision making technique." *Advanced Engineering Informatics* 119-130. DOI: 10.1016/j.aei.2010.07.003.

Navinchandra, D., D. Sriram, and R. Logcher. 1988. "GHOST: project network generator." *Journal of Computing in Civil Engineering* (ASCE) 2 (3): 239-254. DOI: 10.1061/(ASCE)0887-3801(1988)2:3(239).

Nourian, P., S. Rezvani, and S. Sariyildiz. 2013. "Designing with Space Syntax A configurative approach to architectural layout, proposing a computational methodology." *eCAADe 31*. 357-365. ISBN: 978-94-91207-04-4.

Noy, N., R., W. Fergerson, and M., A. Musen. 2000. "The knowledge model of Protege-2000: Combining interoperability and flexibility." *Proceedings of the 12th international conference on knowledge engineering and knowledge management*. France. DOI: 10.1007/3-540-39967-4_2.

Noy, N.F. & McGuinness, D.L. (2001), Ontology Development 101: A Guide to Creating Your First Ontology, Knowledge Systems Laboratory, Stanford University, CA.

OpenBIM, BuildingSMART. 2016. *ifc-overview*. http://www.buildingsmart-tech.org/specifications/ifc-overview.

Rajpathak, D. 2001. "The Task Ontology Component of the Scheduling Library." *Knowledge Media Institute* (The open Univeristy). Pilot Study.

Rajpathak, D., E. Motta, and R. Roy. 2000. "A Generic Task Ontolgy for Scheduling Applications." http://projects.kmi.open.ac.uk/akt/publication-pdf/generictaskontology.pdf.

Riley, D., R., and V. Sanvido. 1997. "Space planning method for multistory building construction." *Journal of Construction Engineering and Management* (ASCE) 171–180. DOI: 10.1061/(ASCE)0733-9364(1997)123:2(171).

Sacks, R., O. Rozenfeld, and Y. Rosenfeld. 2009. "Spatial and Temporal Exposure to Safety Hazards in Construction." *Journal of Cosntruciton Engineering and Management* (ASCE) 135 (8): 726-736. DOI: 10.1061/(ASCE)0733-9364(2009)135:8(726).

Said, H., and G. Lucko. 2016. "Float Types in Construction Spatial Scheduling." *Journal of Construction Engineering and Management* (ASCE) 142 (12): 1-12. DOI: 10.1061/(ASCE)CO.1943-7862.0001202.

Sanvido, D. R. Riley and V. E. 1995. "Patterns of Construction-Space Use in Multistory Buildings." *J. Constr. Engine. Manage*. 123 (4): 464-473. DOI: 10.1061/(ASCE)0733-9364(1995)121:4(464).

Sayed, K., A. Turner, B. Hillier, S. Iida, and A. Penn. 2014. Space syntax methodologies. Edited by Bartlett School of Engineering Architecture. London.

Shaked, O., and A. Warszawski. 1995. "Knowledge based system for construction planning of high-rise buildings." *Journal of Construction Engineering and Management* (ASCE) 110 (2): 172-182. DOI: 10.1061/(ASCE)0733-9364(1995)121:2(172).

Smith, S., F., and M., A. Becker. 1997. "An Ontolgy of Constructing Scheduling Systems." *AAAI Symposium on Ontological Engineering*. Stanford: AAAI Press. https://www.aaai.org/Papers/Symposia/Spring/1997/SS-97-06/SS97-06-016.pdf.

Smith, S., F., G. Cortellessa, D., W. Hildum, and D., W. Ohler. 2005. "Using a scheduling domanin ontolgy to compute user-oriented explanations." *Planning, Scheduling, and Constraint Satisfaction: From theory to Practise*.

Song, Y., and D. Chua. 2005. "Detection of spatio-temporal conflicts on a temporal 3D space system." *Advanced Engineering Software* (36): 814–826. DOI: 10.1016/j.advengsoft.2005.03.025.

Sowa JF (1999) *Knowledge Representation: Logical, Philosophical, and Computational Foundations.* Brooks Cole Publishing Co., Pacific Grove, California ISBN: 0-534-94965-7.

Steuer, J. 1993. "Defining virtual reality: dimensions determining telepresence." *Journal of Communication* 42: 73–93. DOI: 10.1111/j.1460-2466.1992.tb00812.x.

Su, X., R. Andohar, J. Pan, and A. Kandil. 2012. "GIS-based dynamic construction site material layout evaluation for building renovation projects." *Automation in Construciton* 27: 40-49. DOI: 10.1016/j.autcon.2012.04.007.

Sure Y, Erdmann M, Angele J, Staab S, Studer R, Wenke D (2002a) OntoEdit:Collaborative Ontology Engineering for the Semantic Web. In: Horrocks I, Hendler JA (eds) First International Semantic Web Conference (ISWC'02). Sardinia, Italy. (Lecture Notes in Computer Science LNCS 2342) Springer-Verlag, Berlin, Germany, pp 221–235. DOI: 10.1007/3-540-48005-6_18.

Swartout B, Ramesh P, Knight K, Russ T (1997) Toward Distributed Use of Large-Scale Ontologies. In: Farquhar A, Gruninger M, Gómez-Pérez A, Uschold M, van der Vet P (eds) AAAI'97 Spring Symposium on Ontological Engineering. Stanford University, California, pp 138–148. https://www.aaai.org/Papers/Symposia/Spring/1997/SS-97-06/SS97-06-018.pdf.

Tah, J., H., M., V. Carr, and R. Howes. 1999. "Information modeling for case-based construction planning of highway bridge proj- ects." *Advances in Engineering Software* (Elseiver) 30 (7): 495-509. DOI: 10.1016/S0965-9978(98)00128-8.

Thabet, W., Y., and Y., J. Beliveau. 1994. "Modeling work space to schedule repetitive floors in mul- tistory buildings." *Journal of Construction Engineering and Management* 120 (1): 96–116. DOI: 10.1061/(ASCE)0733-9364(1994)120:1(96).

Thomas, H., R., D., R. Riley, and S., K. Sinha. 2006. "Fundamental principles for avoiding congested work areas - a case study." P*ractice Periodical on Structural Design and Construction* (11): 197–205. DOI: 10.1061/(ASCE)1084-0680(2006)11:4(197).

Wang, H., and F. Boukamp. 2011. "Ontology-based representation and reasoning framework for supporting job hazard analysis." *Journal of Computing in Civil Engineering* (ASCE) 25 (6): 442–456. DOI: 10.1061/(ASCE)CP.1943-5487.0000125.

Wikipedia. 2017. Industry Foundation Classes. https://en.wikipedia.org/wiki/Industry_Foundation_Classes.

Winch, G., M., and S. North. 2006. "Critical space analysis." *Journal of Construcition Engineering and Management* 132: 473–481. DOI: 10.1061/(ASCE)0733-9364(2006)132:5(473).

Zhang, S., F. Boukamp, and J. Teizer. 2015. "Ontology-based semantic modeling of construction safety knowledge: towards automated safety planning for job hazard analysis (JHA)." *Automation in Construction* (Elseiver) 52: 29–41. DOI: 10.1016/j.autcon.2015.02.005.

Zhang, S., J. Teizer, N. Pradhananga, and C., E. Eastman. 2015. "Workforce location tracking to model, visualize and analyze workspace requirements in building information models for construction safety planning." *Automation in Construction* 60: 74-86. DOI: 10.1016/j.autcon.2015.09.009.

Zhong, B., L. Ding, H. Luo, Y. Zhou, Y. Hu, and H. Hu. 2012. "Ontology-based semantic modeling of regulation constraint for automated construction quality compliance checking." *Automation in Construction* (Elsevier) (28): 58–70. DOI: 10.1016/j.autcon.2012.06.006.

Zhou, Z., Y., M. Goh, and L. Shen. 2016. "Overview and Analysis of Ontology Studies Supporting Development of the Construction Industry." *Journal of Computing in Civil Engineering* (ASCE). DOI: 10.1061/(ASCE)CP.1943-5487.0000594.

Zouein, P., P., and I., D. Tommelein. 1999. "Dynamic layout planning using a hybrid incremental solution method." *Journal of Construction Engineering and Management* 125 (6): 400-408. DOI: 10.1061/(ASCE)0733-9364(1999)125:6(400).

Zozaya-Gorostiza, C., C. Hendrickson, and D., R. Rehak. 1989. *Knowledge-Based Process Planning for Construction and Manufacturing*. Academic Press. ISBN: 0127819002.

ricerche | architettura, pianificazione, paesaggio, design

Published Books
1. Alessandro Brodini, *Lo Iuav ai Tolentini: Carlo Scarpa e gli altri. Storia e documenti*, 2020
2. Letizia Dipasquale, *Understanding Chefchaouen. Traditional knowledge for a sustainable habitat*, 2020

Nowadays, there is an increasing recognition of the value of knowledge management in the construction projects and ontology-based semantic modelling is seen as an important means of addressing this problem, even if a knowledge-base which maps the construction planning and scheduling domains, in a formal and machine-readable way, is still missing. Addressing this issue, the book is divided in two parts. Part I, *theory*, is a theoretical introduction of on ontologies concepts and expert systems. Part II, *application*, presents a research of ontologies development for semantic modelling of construction scheduling, workspace, product and time domains. The last chapter presents the architecture of an ontology-based expert system, to show how ontologies can support automated planning mechanisms.

Vito Getuli holds his doctorate in Civil and Environmental Engineering from University of Florence and University of Braunschweig (Germany). He is research fellow and adjunct professor at Department of Architecture (UniFI) with expertise in Construction Management, BIM, Artificial Intelligence, VR/AR.