A formal ontology for industrial maintenance

Mohamed Hedi Karray, Brigitte Chebel-Morello, Noureddine Zerhouni

A formal ontology for industrial maintenance

Authors:

1- Dr. Mohamed Hedi KARRAY, LGP-ENIT, University of Toulouse
2- Dr. Brigitte CHEBEL-MORELLO, Automatic Control and Micro-Mechatronic Systems Department, FEMTO-ST institute
3- Pr. Noureddine ZERHOUNI, Automatic Control and Micro-Mechatronic Systems Department, FEMTO-ST institute

Corresponding author:

Mohamed Hedi KARRAY

*e-Mail : mkarray@enit.fr

Abstract

The rapid advancement of information and communication technologies has resulted in a variety of maintenance support systems and tools covering all sub-domains of maintenance. Most of these systems are based on different models that are sometimes redundant or incoherent and always heterogeneous. This problem has lead to the development of maintenance platforms integrating all of these support systems. The main problem confronted by these integration platforms is to provide semantic interoperability between different applications within the same environment. In this aim, we have developed an ontology for the field of industrial maintenance, adopting the METHONTOLOGY approach to manage the life cycle development of this ontology, that we have called IMAMO (Industrial MAintenance Management Ontology). This ontology can be used not only to ensure semantic interoperability but also to generate new knowledge that supports decision making in the maintenance process. This paper provides and discusses some tests so as to evaluate the ontology and to show how it can ensure semantic interoperability and generate new knowledge within the platform.

Key words: ontology, maintenance systems integration, semantic interoperability

1 Introduction

Industrial maintenance is a fundamental function in the business process and thus requires the development of computer systems (Liyanage & Kumar, 2003). Such developments have become possible through information technology as well as advances in automatic control and optimization.

Maintenance is a complex process comprised of object selection, sensor installation, data acquisition, data analysis, decision making, maintenance operation planning, reporting to operators, management of stocks and other phases. Thus, the complexity of industrial systems, comprised of over 10,000 devices and various software systems renders maintenance tasks difficult (Bangemann, et al., 2006).

Maintenance covers all domains of a business, from the plant and the equipment to be maintained, to organization according to different strategies (preventive maintenance, predictive maintenance, corrective maintenance), to managing operators and material (handling, hoisting) and spare parts, to computer-assisted diagnostic systems, to documentation management, etc. Various maintenance support systems and tools have evolved and have become essential for maintenance process management. They cover all types of domain such as CMMS (Computerized Maintenance Management Software), diagnostic support systems, prognosis systems, resource management systems like ERP (Enterprise Resource Planning) and other systems. All these systems are currently based on different models that are usually complementary, but sometimes redundant, sometimes incoherent and always heterogeneous.

Under this volatile diversity, the maintenance actors who are the eventual users of these systems need to have "the right information in the right format in order for the right people to do the right things at the right time" (Lee, Liao, Lapira, Ni, & Li, 2009). It has therefore become necessary to integrate all maintenance support systems into a global platform for maintenance management.

Thus, in order to address these problems, computer systems relating to maintenance have been developed and are in use today (ENIGMA[1], CASIP[2], ICAS-AME[3], Remote Data Sentinel[4], IMS/DB™ [5], WSDF (Hung, Chen, Ho, & Cheng, 2003), PROTEUS (Bangemann, et al., 2006), TELMA[6], etc.). These platforms come either from industry or from academia. Muller et al. (Muller, Marquez, & Iung, 2008) classify them as: proprietary platforms (i.e. ICAS), platforms developed within projects (i.e. PROTEUS) or platforms for research and education (i.e. TELMA). In fact, many projects have been undertaken in order to provide fully integrated and intelligent platforms. Several surveys found in the literature summarize these different works (Muller, Marquez, & Iung, 2008) (Jardine, Daming, & Banjevic, 2006) (Levrat, Iung, & Crespo-Marquez, 2008) (Campos, 2009).

[1] http://www.enigma.com/

[2] http://www.predict.fr/

[3] http://www.esrgtech.com/Documentation/DODSymposium_GreatIdeas.pdf

[4] http://dynamite.vtt.fi/conference_pres/telma_wceam_el_bi.pdf

[5] http://www.imscenter.net/

[6] http://dynamite.vtt.fi/conference_pres/telma_wceam_el_bi.pdf

The principal problem confronted by these integration platforms is to provide the means for different applications to move from coexistence to interoperability and cooperation within the same environment.

Most of the existing platforms use Web services (Bangemann, et al., 2006) (Muller, Marquez, & Iung, 2008) to guarantee interoperability between the various integrated applications, despite the fact that setting up adaptors between these Web services and standardizing the exchanged data is a very complicated task.

While research and technologies have successfully addressed many syntactic-level interoperability problems, they do not often address the semantics of data. The diversity of information content and formats is a salient factor in nearly all distributed systems, and the major challenge is to make diverse information systems interoperate at the semantic level while retaining their differences (March, Hevner, & Ram, 2000), which is the case in most maintenance platforms.

Alone, the integration of applications is not sufficient to furnish maintenance actors with the right information, exploitable at the right time. Maintenance platforms must also reinforce the exploitation of maintenance knowledge by developing the standardization of information and knowledge in terms of understanding, interpretation and sharing, thus improving semantic interoperability. Ontology engineering appears to be the best way to respond to these problems since ontologies have well-defined terminologies whose semantics are unambiguous (Guarino, Formal Ontology and Information Systems, 1998) due to their formal and explicit representation of a common understanding of domain concepts and their relationships.

Indeed, according to (Mizogouchi & Bourdeau, 2004) an ontology provides: 1) a basic conceptual structure from which it is possible to develop systems based on knowledge that is shared and reused, and 2) interoperability between information sources and knowledge.

In this paper we investigate the development of domain ontology. This maintenance ontology will be exploited so as to encourage the sharing and reuse of knowledge, as well as to show the integration of semantic interoperability solutions into the maintenance platform. In another paper, we have proposed a semantic mediator (Karray, Chebel-Morello, & Zerhouni, 2010) based on a domain ontology covering all aspects of maintenance and we are currently working towards its validation. Those studies, however, are not within the scope of this paper.

For Section 2 we selected a methodology of ontology development (METHONTOLOGY) (Fernández-López, Gómez-Pérez, & Juristo, 1997), a tool and an appropriate language in order to construct a domain ontology for industrial maintenance that we called IMAMO (Industrial MAintenance Management Ontology). The METHONTOLOGY approach ensures both better management of the ontology life cycle and a progressive development process during which a set of activities to be performed has been identified. They are: *plan*, *specify*, *acquire knowledge*, *conceptualize*, *formalize*, *integrate*, *implement*, *evaluate*, *document*, and *maintain*.

Section 3 is devoted to the IMAMO development process. To perform the activities of *acquire knowledge*, *conceptualize* and *integrate*, we refer to different projects undertaken along the same lines as MIMOSA[7] and PROTEUS[8], and projects implemented in different areas related to industrial maintenance such as PROMISE[9]

[7] http://www.mimosa.org/

[8] http://www.proteus-iteaproject.com/

and SMAC[10]. Concerning the activity *formalization*, a UML ontological model has been built in collaboration with maintenance experts. The UML model is then encoded in ALCQHI, a description logic variant adopting the approach proposed in (Berardi, Calvanese, & De Giacomo, 2005). We then render the ontology operational in the activity *implementation* by transforming the UML ontological model into PowerLoom (Chalupsky, MacGregor, & Russ, 2010), a logic-based representation language for ontology presentation.

In the *evaluation* activity, we adopt a business-oriented approach based on actual cases of use in order to evaluate the ontology at the application level so as to improve the maintenance process. Some examples from the Java API of PowerLoom are provided and discussed. In addition, some metrics (Tatir & Budak Arpinar, 2007) are used to characterize the ontological model.

In Section 4 some perspectives for future investigation are discussed and we conclude in Section 5.

2 Adopted Approach: Construction methodology, tools and languages

When a new ontology is built, several basic questions arise related to the methodologies, tools and languages to be used in its development process (Corcho, Fernandez-Lopez, & Gomez-Perez, Methodologies, tools and languages for building ontologies: Where is their meeting point?, 2003). In this section these topics are summarized and an overview of the issues is provided so as to present the approach we adopted in the construction of our ontology.

2.1 Ontology development methodologies

Ontology development methodologies support the creation of ontologies. Thus, Fernandez et al. in (Fernández-López, Gómez-Pérez, & Juristo, 1997) affirm that the ontology development process refers to the activities necessary to build ontologies. Several methodologies such as Tove, METHONTOLOGY, On-To-Knowledge, AFM, OntoClean, DILIGENT, NeOn, etc. have been developed (Mizoguchi, 2004) (Corcho, Fernandez-Lopez, & Gomez-Perez, Methodologies, tools and languages for building ontologies: Where is their meeting point?, 2003) (Fernandez-Lopez & Corcho, 2004).

The AFM (Activity-First Method) methodology is dedicated to the development of task ontologies (Mizoguchi, 2004). It starts the building process after determining the source document from which the ontology will be extracted.

Ushold and King's methodology (Uschold & King, 1995) is useful in the early phase of development of an informal ontology.

TOVE (The Toronto Virtual Enterprise) is the most formal among the existing ones in that it first enumerates the questions to be answered by the resulting ontology and expresses them in a formal language so as to use them for verification of the ontology (Fox, 1992). Its competency question strategy is popular and usable in any methodology.

[9]PROMISE : http://www.promise.no/

[10]SMAC : http://www.femto-st.fr/en/Research-departments/AS2M/Research-groups/COSMI/SyMI/The-SMAC-project.php

On-To-Knowledge[11] works well, especially for knowledge management applications (Mizoguchi, 2004).

METHONTOLOGY is based on a set of activities leading to the construction of an ontology (Fernández-López, Gómez-Pérez, & Juristo, 1997). This set is based on the key activities identified by the software development process and methodologies used in knowledge engineering. This methodology includes the identification of an ontology's development process, a life cycle based on evolving prototypes and techniques used to carry out each activity in the management, development and support activities.

OntoClean is oriented towards validation of taxonomies (Guarino & Welty, 2002) (Welty & Andersen, 2005). It is based on very general ontological notions drawn from philosophy to characterize the relevant aspects of the intended meaning of the properties of ontology components. These aspects are represented by formal meta-properties which impose several constraints on the taxonomic structure of an ontology in order to assess and validate the choices made.

DILIGENT (DIstributed, Loosely controlled and evolvInG Engineering of oNTologies) is intended to assist domain experts in a distributed environment and to enable ontologies to evolve (Pinto, Tempich, & Staab, 2004). This methodology focuses on collaborative ontological engineering. The center of this methodology is an argumentation system that facilitates discussion of the logic design changes that are introduced in different phases of the ontology lifecycle.

NeOn (Suárez-Figueroa, 2010) was created within the NeOn project[12] in order to build ontology networks. Specifically, in terms of NeOn dimensions, the methodology includes the benefits of collaboration provided by DILIGENT. In addition, NeOn takes into consideration the proposal made by METHONTOLOGY and On-To-Knowledge in the use of competency question issues for the business specification of the ontology. With regard to ontology reuses, NeOn considers the list of activities proposed by METHONTOLOGY as a starting point, and it offers guidelines for improvement and expansion. For the construction of these ontology networks, NeOn offers nine scenarios relative to the adopted method of construction.

NeOn is a combination of methods that can be considered as the evolution and expansion of METHONTOLOGY, considering activities in greater detail and including collaboration and context.

Thus, according to (Mizoguchi, 2004), when developing a large-scale ontology, METHONTOLOGY and On-To-Knowledge are very useful. Corcho et al. in (Corcho, Fernandez-Lopez, & Gomez-Perez, Methodologies, tools and languages for building ontologies: Where is their meeting point?, 2003) therefore concluded that METHONTOLOGY was the most mature approach.

In our case, we have not developed an ontology network nor a task ontology, no more than a taxonomy ontology, but a single ontology for the field of maintenance. Consequently, we have adopted METHONTOLOGY to develop the domain maintenance ontology.

2.2 METHONTOLOGY: the adopted methodology

[11] http://www.ontoknowledge.org

[12] http://www.neon-project.org/

Gómez-Pérez et al. in (Fernández-López, Gómez-Pérez, & Juristo, 1997) and (Gómez-Pérez, 1996) assert that the ontology development process refers to the activities needed in order to build ontologies. In this context, METHONTOLOGY has been proposed as a structured method for the building of ontologies. Its aim is to cover the overall life cycle of ontologies and it includes a set of activities to be performed during the ontology development process comprised of: *schedule*, *control*, *quality assurance*, *specification*, *conceptualization*, *formalization*, *implementation*, *maintenance*, *knowledge acquisition*, *integration*, *evaluation*, *documentation*, and *configuration management*.

As shown in Fig.1, METHONTOLOGY breaks these activities down into3 levels: management activities (*schedule*, *control*, and *quality assurance*), development activities (*specification*, *conceptualization*, *formalization*, *implementation* and *maintenance*) and support activities (*knowledge acquisition*, *integration*, *evaluation*, *documentation*, and *configuration management*).

Concerning development activities, the specification activity states why the ontology is being built, what its intended uses are and who the end-users are. The conceptualization activity in METHONTOLOGY organizes and converts an informally perceived view of a domain into a semi-formal specification using a set of intermediate representations (IRs) based on tabular and graph notations that can be understood by domain experts and ontology developers. The conceptualization activity results in the conceptual model of the ontology.

The formalization activity transforms the conceptual model into a formal or semi-computable model. The implementation activity builds computable models in an ontology language. The maintenance activity updates and corrects the ontology if needed.

2.3 Ontology development tools

Ontology development tools are environments intended to support the ontology development process and the subsequent ontology usage. Apart from the common edition and browsing functionality, these tools usually include ontology documentation, ontology exportation and importation from different formats, graphical views of the ontologies built, ontology libraries, attached inference engines, etc. (OntoWeb Consortium, 2002).

Tools for building ontologies have increased exponentially in recent years. As examples of these tools we find OntoEdit[13], Protégé2000, Protégé 3.4.5 and Protégé 4.1 supporting respectively OWL.1.0 and OWL.2., PowerLoom API and PowrLOOM GUI, TopBraid Composer[14], NeOn Toolkit[15], etc. (Corcho, Fernandez-Lopez, & Gomez-Perez, 2003) (Mizoguchi, 2004).

In fact, when a new ontology is going to be built, several basic questions arise concerning the tools to be used: Which tool(s) give support to the ontology development process? How are the ontologies stored (in databases, XML or ASCII files)? Does the tool have an inference engine? Do tools have translators for different ontology languages? What is the quality of the translations? How can applications interoperate with ontology servers? Do tools have forward and backward translators to/from different ontology implementation languages?

[13] http://www.semtalk.com/

[14] http://www.topquadrant.com/

[15] http://neon-toolkit.org/

How can applications interoperate with ontology tools? How can the developed ontologies be used in real applications? Do tools enable querying information about an ontology? etc. (OntoWeb Consortium, 2002).

2.4 Ontology languages

Diverse languages are proposed for ontology implementation (Corcho, Fernandez-Lopez, & Gomez-Perez, 2003) (Mizoguchi, 2004). They can be classified in two categories according to their chronological order of appearance, before and after the boom of the web.

At the beginning of the 1990s, a set of Artificial Intelligence-based ontology implementation languages was created. Basically, the Knowledge Representation paradigm underlying such ontology languages was based on first-order logic (e.g. KIF), on frames combined with first-order logic (e.g. Ontolingua, OCML and FLogic), or on DL (e.g Loom).

The rise of the Internet then led to the creation of ontology languages that exploited the characteristics of the Web. These languages, such as RDF(S) and OWL are called web-based ontology languages.

However, some differences exist within the primitives available in each language for the representation of concept taxonomies. For example, Ontolingua[16], LOOM[17] and OWL[18] are the most expressive, since they allow the creation of exhaustive and disjointed subclass partitions of a concept [35]. In addition, functions can be defined easily in this set of languages. Rules, though, can only be defined in LOOM and OWL 2.0, and procedures can only be defined in Ontolingua and LOOM (although they cannot be executed). In LOOM and OWL the inference engine also performs automatic concept classifications (Corcho, Fernandez-Lopez, & Gomez-Perez, 2003).

PowerLoom[19], the successor to Loom, provides a language and environment for constructing intelligent, knowledge-based applications. It uses a fully expressive, logic-based representation language (a variant of KIF) and a natural deduction inference engine that combines forward and backward chaining to derive what logically follows from the facts and rules asserted in the knowledge base. While PowerLoom is not a description logic, it does have a description classifier which uses technology derived from the Loom classifier to classify descriptions expressed in full first order predicate calculus. PowerLoom uses modules as a structuring device for knowledge bases, and ultra-lightweight worlds to support hypothetical reasoning.

2.5 The adopted language:

Consequently, to implement IMAMO we chose PowerLoom (Chalupsky, MacGregor, & Russ, 2010), seeing that it offers reasoning facilities for concepts and individuals. Its distinguishing feature in relation to other DL systems is the incorporation of an expressive query language for retrieving individuals, and its support for rule-based programming. The expressivity of PowerLoom provides a good scalability to large ontologies and

[16] http://www.ksl.stanford.edu/software/ontolingua/

[17] http://www.isi.edu/isd/LOOM/

[18] http://www.w3.org/TR/owl-features/

[19] http://www.isi.edu/isd/LOOM/PowerLoom/

knowledge bases. It allows different reasoning mechanisms such as logical deduction, hypothetical reasoning, equality reasoning, arithmetic and reasoning with inequalities. This allows us to check the consistency of the model and the rules of our ontology, and also to design improvements straightforwardly.

In addition, PowerLoom has a static and dynamic query optimizer that is similar to optimizers used in database systems. The dynamic optimizer operates for each conjunctive sub-goal based on actual bindings. Given this mechanism it is possible to run PowerLoom queries that return hundreds of thousands of solutions. PowerLoom also has a powerful relational database interface that allows it to utilize the power of databases for handling large assertion bases. Maintenance systems can take advantage of these cited performances of PowerLoom to ensure semantic interoperability and to provide new services responding to the needs of maintenance actors. They are expected to benefit from the capacities of the PowerLoom reasoning engine.

3 The IMAMO development process

The creation of a domain ontology for industrial maintenance was instantiated within the scope of the SMAC (Semantic MAintenance and lifeCycle) project. Financed by the Intereg 4 program undertaken by France and Switzerland, the project was launched in collaboration between academic (the University of Franche-Comté, the Femto-ST Institute, Ecole Polytechnique Fédérale de Lausanne) and industrial (em@systec, Tornos, GMCH) groups from both countries. The aim of this project is to provide a semantic interoperable platform of industrial maintenance ensuring knowledge capitalization and reuse in order to track and develop equipment lifecycle. An initial ontology called SMAC-Model was developed, but it was more oriented towards product lifecycle. We thus launched the creation of IMAMO (Industrial MAintenance Management Ontolgy) to compensate for this lack and to use this ontology in the maintenance platform that we are developing.

Consequently, in this section we present the IMAMO development process (*specification*, *conceptualization*, *formalization*, *implementation* and *maintenance*), in which we followed the METHONOTOLOGY philosophy, joining the use of support and development activities. We have therefore exploited the *knowledge acquisition* activity alongside the development activity *specification*. During the *conceptualization* activity, we worked on the basis of the support activities *knowledge acquisition* and *integration*. The support activity *evaluation* needs to be repeated for the different development activities such as *conceptualization*, *formalization* and *implementation*, however, due to the structure of this paper, this activity is presented in a separate subsection containing the various evaluation types related to these three activities.

3.1 Specification

The goal of the specification phase is to produce either an informal, semi-formal or formal ontology specification document written in natural language, using respectively a set of intermediate representations or competency questions (Grüninger & Fox, 1994).

Given that ontology creation is not a small task, this requires not only skills in information technology but also in the conceptualized domain (Frankovic & Budinska, 2006), hence the importance of the *knowledge acquisition* activity in editing the specification document.

3.1.1Knowledge acquisition

To acquire knowledge about the field of maintenance, we refer to standards, research projects and experts in industrial maintenance. Concerning standards, we adhere to the AFNOR[20] norms and MIMOSA[21] standards. The PROTEUS[22] and PROMISE[23] projects also serve as a basis. Finally, we adopt the business expertise of various maintenance experts, managers and operators from different companies such as Cegelec SA[24] France & Germany, Tornos[25] (Switzerland), Peugeot[26] (Belfort, France) and em@systec[27] (France). Various research studies such as those of Retour et al. (Retour, Bouche, & Plauchu, 1990), Kaffel (Kaffel, 2001) and Rasovska et al. (Rasovska I. , 2006) are also taken into account.

AFNOR defines maintenance as "the combination of all technical, administrative and managerial actions during the life cycle of an item intended to retain it in, or restore it to, a state in which it can perform the required function". In accordance with this definition, Retour et al. (Retour, Bouche, & Plauchu, 1990) present the maintenance function as a set of activities grouped into two subsets: activities with technical predominance and activities with management predominance.

In the same context, Rasovska et al. (Rasovska, Morello-Chebel, & Zerhouni, 2007) divide the maintenance process into four fundamental technical and business fields, identified in the maintenance domain: (i) equipment analysis, which consists of functional analysis and failure analysis, (ii) fault diagnosis and expertise, which aim to help the operator during his intervention to diagnose the problem and to establish a prognosis so as to anticipate the breakdown and to solve it (iii) resource management, which deals with resource planning for all maintenance interventions, and (iv) maintenance strategy management, which represents the decision support concept for maintenance managers.

Consequently, the concepts which should be identified must cover all these fields and activities. To facilitate our identification of concepts, we identified all those concepts related to each layer presented above. Identification of the main concepts of each layer is based on the models of MIMOSA-CRIS (Kahn, 2003), the PROTEUS project (Rasovska, Chebel–Morello, & Zerhouni, 2008), the PROMISE SOM (PROMISE, 2008) and SMAC projects (Matsokis, Karray, Morello-Chebel, & Kiritsis, 2010).

3.1.2 Specification document

[20] http://www.afnor.org/

[21] http://www.mimosa.org/

[22] http://www.proteus-iteaproject.com/

[23] http://www.promise.no/

[24] http://www.cegelec.fr/

[25] http://www.tornos.com/

[26] http://concessions.peugeot.fr/belfort

[27] http://www.emasystec.com/

After the acquisition phase in the previous activity, we produced a first excerpt from a specification document for IMAMO, presented in Table 1. The key fields in this table are the specification of the domain, the purpose of ontology building, the formalization level of this ontology and its scope.

3.2 Conceptualization

In this activity, the domain knowledge is structured according to a conceptual model that describes the problem and its solution in terms of the domain vocabulary identified in the ontology specification activity. In this phase Gómez-Pérez et al. recommend a set of intermediate representations for conceptualization such as a concept classification tree, a data dictionary, a table of rules and others (Gómez-Pérez, 1996). A complete Glossary of Terms (GT) must first be constructed, including concepts, instances, verbs and properties. This activity is mainly based on the support activities of knowledge acquisition and integration.

3.2.1 Support activities

While ontologies are built to be reused, their reuse is one of the important issues in their construction. According to Pinto et al. there are two different reuse processes (Pinto, Tempich, & Staab, 2004) merge and integration. Both of these reuse processes are included in the overall process of ontology building. Merge is defined as the process of building an ontology for one subject reusing two or more different ontologies from that same subject (Pinto, Tempich, & Staab, 2004). In an integration process, on the other hand, the source ontologies are aggregated, combined or assembled, to form the resulting ontology, possibly after reused ontologies have undergone changes such as extension, specialization or adaptation. It should be noted that we have adopted both processes by reusing and integrating several concepts from other ontologies or models.

The lack of a formal ontology in the industrial maintenance domain must also be noted. Despite industrial maintenance being different by definition from software maintenance, there are some junctions between the two. Thus, when we launched our search for already existing industrial maintenance ontologies for potential merge and/or integration into IMAMO, we also took into account existing software maintenance ontologies.

3.2.1.1 Knowledge acquisition

Several teams have attempted to build an ontology for maintenance. In the software maintenance field, Kitchenham et al. (Kitchenham, et al., 1999) developed a preliminary ontology to identify a number of factors that influence maintenance (SMO: software maintenance ontology). Ruiz et al. (Ruiz, Vizcaino, Piattini, & García, 2004) developed a semi-formal ontology in which the main concepts of products, staff, activities, processes, workflow and actions are described. This ontology, besides representing static aspects, also represents dynamic issues related to the management of software maintenance projects. Some concepts from these ontologies have been reused in IMAMO, though we have not followed the same concept nomination in all cases. The mapping between these concepts, however, has been done.

Concerning ontologies and models of an industrial scope, MIMOSA, as mentioned above in the knowledge acquisition phase, is the first initiative to unify data elements to be exchanged for special equipment such as condition monitoring tools or dedicated assets by establishing MIMOSA-CRIS (common relational information system), a relational database model of maintenance information (Kahn, 2003). Matsokis and Kiritsis (Matsokis

& Kiritsis, 2009) propose an ontology-based approach for the management of product lifecycle, such as the extension of the ontology proposed in the PROMISE project (PROMISE, 2008) which provides a semantic object model (SOM) for product data and knowledge management.

To create IMAMO we started from models developed in the PROTEUS project (Rasovska, Chebel–Morello, & Zerhouni, 2008), to publish a first version of a maintenance ontology (Karray, Chebel-Morello, & Zerhouni, 2009). This ontology was composed of 62 concepts and 70 relations integrating the main concepts used in PROTEUS. Then, as a part of the SMAC project, we mapped the previously mentioned ontology with the PROMISE model. As result, Matsokitis and Karray (Matsokis, Karray, Morello-Chebel, & Kiritsis, 2010) proposed a more evolved version of this ontology by orienting it towards the maintenance field while integrating some concepts related to the maintenance area included in the MOF (middle of life) phase of PLM (product lifecycle management). This ontology, called the SMAC-model, is formalized by UML and implemented with OWL-DL via Protégé.

3.2.1.2 Reuse and integration

We then returned to the field of maintenance, integrating concepts from the SMAC model in relation to the lifecycle of equipment so as to take into account (1) the beginning of the life concerning the design phase, (2) the middle of life phase by tracking all the events and states of the equipment's health, and (3) the end of life, via the calculation of indicators supporting the decision for re-use and disassembly. Since, as was mentioned above, MIMOSA-CRIS is considered to be the reference of the domain, when creating IMAMO we also took into consideration the classes used in this model.

Based on their various data dictionaries, we manually mapped MIMOSA-CRIS, SMAC-Model, PROMISE and SMO in order to reuse some concepts while creating IMAMO. Some labels of reused concepts were changed, but the alignment between these ontologies is achieved by the addition of equivalence or subsumption rules to these concepts. Concept names are changed to obtain more expressive terms (to eliminate ambiguities) or to choose terms in concordance with the different existing models. An example of some reused concepts integrated into IMAMO is presented in Table 2 based on our different manual mappings between different models and ontologies.

However, further integration at the instance level is possible (exploitation of concepts from other ontologies to be used as instances of IMAMO's concepts).For example, the functional ontology proposed by Kitamura and Mizoguchi (Kitamura & Mizoguchi, 1998) can be integrated as instances of the concepts "function" and "subfunction".

Thus, the fault ontology proposed by Kitamura and Mizoguchi (Kitamura & Mizoguchi, An Ontological Analysis of Fault Process and Category of Faults, 1999) can be integrated also as instances of concepts related to diagnosis and problem solving in IMAMO.

3.2.2 Glossary of terms and data dictionary

Hence, we began the conceptualization of IMAMO by building the glossary of terms. The concepts are first classified in the glossary respecting the four fields identified by Rasovska et al. and mentioned above. We then

refined this list of concepts by deepening the first classification. The second classification goes further than the first by breaking each layer down into sub-layers. We note that some concepts are shared between different layers or sub-layers. This is done purposely in order to count all the concepts in each layer so as to obtain a clearer vision and a more precise identification. Next we edited the data dictionary based on the European norm NF EN 13306:2001 published by AFNOR. Due to limits of space we will not show these steps in three separate tables, but will present only the data dictionary with the conceptual model for a better understanding of the different views of the ontology (see section 3.2.5).

We note, however, that IMAMO will be a generic ontology; different details can be neglected and left to users (by *user* we mean ontologists who will exploit the ontology) according to their needs. In this case, users may adapt, evolve and maintain the ontology.

In our case we will focus only on concepts and relations; tasks and maintenance activities will be instances of concepts referring to activity. Relations between concepts will not be presented in a table or in a data dictionary, but will be presented as associations with cardinalities within the conceptual model of the ontology.

3.2.3 Concept classification trees

After defining the data dictionary, we edit the concept classification trees. We notice that the domain is very broad; nevertheless, the ontology that we develop will not contain a lot of trees. This is due to the aim of obtaining a rich ontology with different types of relations and not a hierarchical ontology like taxonomy. *Is-a*, *is-component-of*, *has* and other verbs are the relations supported by the ontology. Fig. 2 summarizes some of the concept classification trees in IMAMO (i.e. *is-a* relations).

3.2.4 Editing rules

As mentioned above, this investigation will mainly focus on concepts and relations as well as rules for the consistency of concepts.

The following rule presents the constraints of the composite relationship between differing sets of physical equipment. In fact, “physical equipment” is composed of physical equipment which has Component as exploitation mode. Also, “physical equipment” cannot be composed by itself.

$$\text{ComposedBy} \equiv \text{PhysicalEquipment}\,?\text{X} \sqcap ((\text{PhysicalEquipment}\,?\text{Y}) \sqcap \mathit{Has\_Exploitation\_mode}(\text{PhysicalEquipment}\,?\text{Y}\ \mathit{component}))$$

Other specific rules and formulas will be left to the evolution tasks and the choice of users (ontologists who will exploit the ontology). Here we show only some examples of rules that can be edited. We edit these rules through the description logic ALCQHI. The rules can then be translated and edited by the implementation language if it allows rule definition. Rules enrich the ontology and allow greater semantic reasoning as well as understandability.

For example, because of the following defined rule, the identification of critical components is possible without defining a new concept called “critical component”. The rule describes “critical component” as any physical equipment having a functional-degree property value greater than or equal to five.

$$\text{CriticalComponent} \equiv \text{PhysicalEquipment} \sqcap (\leq \mathit{FUNCTIONNAL-DEGREE}\ \ 5)$$

3.2.5 Conceptual model

The Unified Modeling Language (UML) may be a good candidate for representing ontologies and knowledge (Cranefield S. , 2001). Cranefield and Purvis in (Cranefield & Purvis, 2000) noted that UML models have some common features regarded as characteristic of the declarative knowledge representation paradigm (Bézivin, 2000). Knowledge expressed via UML is easily accessible for human comprehension. In a UML model, knowledge can be changed easily due to the modular nature of object-oriented modeling. Also, new knowledge can be derived from UML models by reasoning their contents (Cranefield S. , 2001). From this point of view, UML can be regarded as an appropriate candidate for knowledge representation. Cranefield focuses on the benefits of using UML as an ontology language, Bézivin stresses that UML addresses the concept of representation and more specifically the ontology definition presented in (Charlet, Bachimont, & Troncy, 2004). In this study we adopted the UML class diagram to formalize IMAMO. This choice is supported by the graphical expressivity and the semantic power of UML recommended in the various investigations mentioned above. This expressivity facilitates the exchange between domain experts and human understanding of the ontology. Also, the ontology of the domain, although formalized independently of the reasoning methods, has a structure which depends on how acquired knowledge will be used for reasoning since experts deliver knowledge adapted to their reasoning. Reasoning methods will be considered in the implementation phase.

For the best understandability and in order to obtain a more readable representation, especially on the relationship level, the class diagram has been separated into eight views, according to the layer classification used in the phase of glossary identification. We note that these views do not present sub-ontologies or packages. These views are:

- the structural view, which presents the equipment composition and is related to the equipment analysis layer. The conceptualization of this view is presented in Fig. 3. Table 3 is its appropriate data dictionary;
- the functional and dysfunctional view, which characterizes different functionalities of the equipment and its components, as well as the fault diagnosis and expertise field. Fig. 4 shows the conceptualization of this view and Table 4 presents its related data dictionary;
- the event view which presents the triggering events launched after failures and/or degradation, related to the fault diagnosis and expertise field. Fig. 5 shows the appropriate conceptualization of this view and Table 5 presents it related data dictionary;
- the informational view, which presents various resources (documents, human, software, tools, indicators, etc.), related to equipment and maintenance tasks as well as to maintenance strategy and processes, and also related to the resource management and maintenance strategy management layers. The conceptualization of this view is presented in Fig. 6. Table 6 is its appropriate data dictionary;
- the interventional view presenting concepts related to the intervention process. The conceptualization of this view is presented in Fig. 7. Table 7 is its appropriate data dictionary;
- the strategy view, which presents the managerial aspects of maintenance strategy and contracts. Fig. 8 shows the appropriate conceptualization of this view and Table 8 presents it related data dictionary;

- the process view, which presents all technical, administrative and managerial processes. The conceptualization of this view is presented in Fig. 9 and its appropriate data dictionary is presented inTable9;
- the middle-of-life view, which presents concepts allowing the equipment's life cycle to be tracked. Please see Fig. 10 and Table 10, presenting respectively the conceptualization and the data dictionary of this view.

3.3 Formalization

To transform the conceptual model into a formal model, it must be formalized using frame-oriented or description logic representation systems.

With the UML model we cannot know whether the ontology design correctly models the knowledge. However, the expressiveness of the UML constructs may lead to implicit consequences that can go undetected by the designer in complex diagrams and cause various forms of inconsistencies or redundancies in the ontological model (Berardi, Calvanese, & De Giacomo, 2005). Hence, it would be highly desirable to automatically detect relevant formal properties of the UML ontology model such as the above-mentioned inconsistencies and redundancies. To render the model operational, it must be formalized (Van Der Straeten, Simmonds, Mens, & Jonckers, 2003).

We thus decided to use a description logic variant as the representation language of IMAMO, given its properties of decidability, subsumption, and its inference possibilities.

For the encoding of our class diagram in *ALCQHI*, we used the same approach as in (Berardi, Calvanese, & De Giacomo, 2005). While all associations are binary in our ontology's UML model, they can be translated in the same way as an aggregation, with the extra assertions for an association ASSOC between the classes C1 and C2: $\exists\, \mathrm{ASSOC} \sqsubseteq \mathrm{C1}$ and $\exists\, \mathrm{ASSOC} \sqsubseteq \mathrm{C2}$.

For example, the association between the relation *has-top-model* and between the *Physical-equipment* and *Equipment-model* concepts will be translated as follows:

$$\exists \mathrm{PhysicalEquipment\_Has\_EquipmentModel} \sqsubseteq \mathrm{EquipmentModel}$$

$$and$$

$$\exists\, \mathrm{PhysicalEquipment\_Has\_EquipmentModel} \sqsubseteq \mathrm{PhysicalEquipment}$$

$$\mathrm{PhysicalEquipment\_Has\_EquipmentModel} \sqsubseteq \mathrm{EquipmentModel} \sqcap \mathrm{PhysicalEquipment}$$

We notice that in most cases when defining the relation's name we compose this one of a combination of a verb with the two related concepts as in the example presented above.

3.4 Implementation

After formalizing, we subsequently translated the formalized model of IMAMO into PowerLoom. Despite the current availability of version 4.0, we chose to work with version 3.2.0 since it is stable whereas version 4.0 is as yet a beta version.

In addition, it should be noted that a PowerLoom exporter for the Protégé editor has been implemented. It can write ontologies using the Protégé frame language in PowerLoom, either fully native or with support for the system concepts from Protégé. Moreover, the PowerLoom GUI (or knowledge editor), a Java-based graphical client for PowerLoom, is now a standard feature and available with PowerLoom starting with version 4.0.

In this section we present a part of the structural model of the equipment implemented by PowerLoom. Each UML class is translated into a PowerLoom concept using the ″DEFCONCEPT″ command. Associations and attributes of classes are translated into a PowerLoom relation or function using the ″DEFFUNCTION″ and ″DEFRELATION″ commands.

```
(DEFMODULE "/PL-KERNEL-KB/PL-USER/ONTOLOGIE-MAINTENANCE"
:DOCUMENTATION "Module for Maintenance"
:INCLUDES ("PL-USER"))
(IN-MODULE "/PL-KERNEL-KB/PL-USER/ONTOLOGIE-MAINTENANCE")
(IN-DIALECT: KIF)
(DEFCONCEPT PHYSICAL-EQUIPMENT)
(DEFRELATION PHYSICAL-EQUIPMENT-ID ((?C PHYSICAL-EQUIPMENT) (?EQUIPMENET-ID
STRING)))
(DEFRELATION PHYSICAL-EQUIPMENT-COAST ((?C PHYSICAL-EQUIPMENT) (?COAST DOUBLE)))
(DEFRELATION PHYSICAL-EQUIPMENT-CONSTRUCTOR ((?C PHYSICAL-EQUIPMENT) (?CONSTRUCTOR
STRING)))
(DEFRELATION PHYSICAL-EQUIPMENT-CONSTRUCTION-DATE ((?C PHYSICAL-EQUIPMENT)
(?CONSTRUCTION-DATE DATE)))
(DEFRELATION PHYSICAL-EQUIPMENT-PURSHASE-DATE ((?C PHYSICAL-EQUIPMENT) (?PURSHASE-
DATE DATE)))
(DEFRELATION EQUIPMENT-HAS-TOP-MODEL ((?E PHYSICAL-EQUIPMENT) (?MG EQUIPMENT-
MODEL)))
(DEFRELATION EQUIPMENT-COMPOSED ((?E PHYSICAL-EQUIPMENT) (?COM PHYSICAL-
EQUIPMENT)))
(DEFFUNCTION FUNCTIONNAL-DEGREE ((?C PHYSICAL-EQUIPMENT)) :->(?N INTEGER))
(DEFCONCEPT EQUIPMENT-MODEL)
(DEFRELATION EQUIPMENT-MODEL-INHERITS ((?MG1 EQUIPMENT-MODEL) (?MG2 EQUIPMENT-
MODEL)))
    (assert (forall (?x ?y)
       (=> ( and (EQUIPMENT-COMPOSED ?x ?y) (not equivalent (?y ?x))))))
(DEFCONCEPT EXPLOITATION-MODE)
(DEFCONCEPT COMPONENT (? EM EXPLOITATION-MODE))
(DEFRELATION  HAS-EXPLOITATION-MODE  ((?PE  PHYSICAL-EQUIPMENT)  (?EM  EXPLOITATION-
MODE)))
(DEFCONCEPT SENSOR (?C PHYSICAL-EQUIPMENT))
```

3.5 Evaluation

Let us remember that the evaluation activity in METHONTOLOGY is a support activity that must be applied within the main development activities. Its role is to evaluate this investigation and thus we present it in a separate section including different evaluations undertaken throughout the development process of IMAMO.

In fact, evaluation activity can lead us to propose an ontology of good quality according to conceptual, operational and functional levels.

However, these different levels of evaluation do not guarantee that user expectations will be met when the ontology is in use since it is precisely at that moment that its performance or shortcomings will appear. Ontology evaluation is not entirely satisfactory at present. There are metrics and approaches that qualify this, but cannot attest to the real quality of the ontology.

Sensitive to the evaluation problem, Brank et al. (Brank, Grobelnik, & Mladenić, 2005) conducted a study of evaluation approaches and have identified four types:

- the first is based on comparing the ontology to a "gold standard", hence the necessity of its existence;
- the second is based on using the ontology in an application and evaluating the results;
- the third involves comparisons with a source of data in the domain to be covered by the ontology;
- finally, the fourth approach, in which evaluation is carried out by humans who try to assess how well the ontology meets a set of predefined criteria, standards, requirements, etc.

Brank et al. stated that the selection of a suitable evaluation approach depends on the purpose of the evaluation, on the application for which the ontology is to be used, and on which aspect of the ontology is being evaluated (Brank, Grobelnik, & Mladenić, 2005). In contrast, another survey, presented by Obrst et al. (Obrst, Werner, Inderjeet, Steve, & Smith, 2007), describes evolution strategies and highlights current ontology evaluation techniques such as criteria, questions and aspects.

However, while several techniques and methods for evaluating ontologies have been developed aiming at estimating and evaluating well-domain ontologies, there is no standardized, objective and widely-accepted evaluation method.

The aim of our evaluation is, firstly, to verify[28] and validate[29] IMAMO. Secondly, our aim is to assess the quality of this ontology and to highlight its additional value for maintenance systems and actors, as well as to provide new users with sufficient information to promote the use of this ontology (the extent of the maintenance domain coverage). Another aim of this evaluation is to focus on its weak points in order to facilitate maintenance and evolution tasks.

We evaluate IMAMO using the four types of approach identified by Brank et al., and endeavor to give the most comprehensive answers possible.

Moreover, given the absence of a gold standard in the maintenance domain (refers to the third approach identified by Brank et al.) and with IMAMO already reusing various concepts from standards and

[28]According to NeOn glossary, ontology verification is the ontology evaluation which compares the ontology against the ontology specification document (ontology requirements and competency questions), thus ensuring that the ontology is built correctly (in compliance with the ontology specification) (Suárez-Figueroa & Gómez-Pérez, 2008).

[29]According to NeOn glossary, ontology validation is the ontology evaluation that compares the meaning of the ontology definitions against the intended model of the world aiming to conceptualize (Suárez-Figueroa & Gómez-Pérez, 2008).

models of the domain such as MIMOSA-CRIS, SMAC model and SMO, our evaluation of IMAMO includes two main steps related to three other approaches:

- Evaluation of the quality of the conceptual model according to certain metrics, and

- Business-oriented evaluation based on the added value of the ontology by respectively using evaluation approaches by humans and implementation. This evaluation falls within the aim of respectively validating and verifying the ontology by:

    o Checking the functionalities of IMAMO via a question/answer method;

    o Evaluating applicability and knowledge exploitation by querying the reasoning engine associated to the ontology and evaluating the results.

3.5.1 Quality of the conceptual model

3.5.1.1 Background

In accordance with Tartir et al. (Tartir, Arpinar, Moore, Sheth, & Aleman-Meza, 2005), assessing the quality of an ontology is important for several reasons, including allowing the developer to automatically recognize areas that might need more work, and to know what parts of the ontology might cause problems. Different dimensions are available for assessing the quality of an ontology. We are interested in the quality metrics presented by (Tatir & Budak Arpinar, 2007). We use the metrics of schema evaluation to evaluate the success of the ontology's UML model of the real-world domain of maintenance: how classes are organized, how the depth, richness, breadth, and height balance of the ontology schema inheritance tree can play a role in quality assessment.

To understand the metrics that have been used and the discussion below, it is important to know the following as referred to (Tatir & Budak Arpinar, 2007):

Ontology structure (schema): An ontology schema is a sextuple $O := \{C, P, A, H^C, prop, att\}$, consisting of two disjoint sets C and P whose elements are called concepts and relationships, respectively, that is to say a concept hierarchy H^C. H^C is a directed, transitive relation $H^C \subseteq C \times C$ which is also called concept taxonomy. $H^C(C_1, C_2)$ means that C_1 is a sub-concept of C_2, a function $prop$: $P \rightarrow C \times C$, that relates concepts non-taxonomically. The function att: $A \rightarrow C$ relates concepts with literal values.

Relationship Richness: This metric reflects the diversity and the placement of relations in the ontology. An ontology that contains many relations other than class-subclass ones is richer than a taxonomy that has only class-subclass relationships. Formally, the relationship richness (*RR*) of a schema is defined as the ratio of the number of relationships (*P*) defined in the schema, divided by the sum of the number of subclasses (*SC*) (which is the same as the number of inheritance relationships) plus the number of relationships.

Attribute Richness: The number of attributes that are defined for each class can indicate both the quality of ontology design and the amount of information pertaining to instance data. In general, we assume that the more slots there are defined, the more knowledge the ontology conveys. Formally, the attribute richness (AR) is defined as the average number of attributes (slots) per class. It is computed as the number of attributes for all classes (*att*) divided by the number of classes (C).

Inheritance Richness: This measure describes the distribution of information across different levels of the ontology's inheritance tree, or the fan-out of the parent classes. This is a good indication of how well knowledge is grouped into different categories and subcategories in the ontology. This measure can distinguish a horizontal ontology from a vertical one, or an ontology with different levels of specialization. A horizontal (or flat) ontology is one that has a small number of inheritance levels, each class having a relatively large number of subclasses. In contrast, a vertical ontology contains a large number of inheritance levels where classes have a small number of subclasses. This metric can be measured for the entire schema or for a sub-tree of the schema. Formally, the inheritance richness of the schema (*IRs*) is defined as the average number of subclasses per class.

3.5.1.2 Interpretations of IMAMO

Before applying these metrics to IMAMO, we note that its UML class diagram contains:

- 200 Relations (P),
- 110 Concepts (classes) (C),
- 61 Subclasses (SC) and
- 91 Attributes (att).

In Table 11, we include a summary of the calculation of IMAMO's metrics and their associated interpretation.

3.5.2 Business evaluation of the ontology

The business evaluation of the ontology deals with the application of various classes of reasoning for checking the consistency of the ontology, checking whether the concepts and their subsumption are satisfactory, and checking the classification of instance. Thus, these classes of reasoning are used to evaluate the answers of conjunctive queries over a knowledge base. The first part of this evaluation involves checking business and technical functionalities, and the second deals with querying the knowledge base according to different use cases.

3.5.2.1 Ontology validation: checking functionalities

This part is based on the "question asking" approach including in the human verification approach, to check whether certain of the ontology's functionalities are enabled on IMAMO. These questions are summarized in Table 12 where we try to respond and argue in favor of the "Does IMAMO allow…?" type of question.

Questions are divided into two categories. Firstly, business questions related to the four fields of the maintenance process identified by Rasovska et al. (Rasovska I. , 2006) concerning the equipment analysis, diagnosis and expertise, resource management and maintenance strategy. The second category deals with the technical functionalities that can be operated in the ontology.

3.5.2.2 Ontology verification: use cases and querying

IMAMO has already been already tested in a real environment. It is integrated in a software platform that manages the maintenance of SISTRE a "System of Industrial Supervison of pallet TRansfEr" located in the AS2M laboratory of the Femto-ST Institute (see Fig.11)

In this section, the application of and tests for IMAMO were performed on SISTRE. The latter represents a flexible production system and is composed of five robotized work stations which are served by a transfer system of pallets organized into double rings (internal and external). Each station is equipped with pneumatic actuators (pushers, pullers and indexers) and electric actuators (stoppers) as well as a certain number of inductive sensors (proximity sensors). An inductive read/write module enables identification and location of each pallet and provides information relative to the required operation at a specific station. The pallets are moved by friction on belts run by electric motors. Each pallet has a magnetic label that is used as an onboard memory that can be read at each work station by means of magnetic read/write modules (Balogh) and that allows the product assembly sequence to be memorized. These labels thus enable determination of the pallet's path through the system. The pallets are conveyed on the interior ring which ensures transit between the various stations. When the pallet must be handled by a robot at the concrete work station (information read on the pallet's label), the latter is shifted onto the external ring where the appropriate work station is located. The work station is situated on the external ring and contains pneumatic and electric actuators (puller, pusher, indexer, and stopper) as well as inductive sensors (Rasovska I. , 2006).

Since equipment is central in this field, we will focus on three important cases in maintenance:

- Equipment expertise,

- Maintenance intervention on the equipment,

- Exploitation of equipment's history of failure in the reuse of knowledge.

A- The equipment expertise

The following PowerLoom code specifies how we can assert the `PHYSICAL-EQUIPMENT` SISTRE, and the `EQUIPMENT-MODEL` PLATEFORME, as well as the associations `PHYSICAL-EQUIPMENT-CONSTRUCTOR` "Bosch" of SISTRE, the top-model `EQUIPMENT-HAS-TOP-MODEL` of SISTRE which is PLATEFORME and the association `EQUIPMENT-COMPONENT-COMPOSED` in order to describe the composition of the physical equipment SISTRE:

```
(ASSERT (PHYSICAL-EQUIPMENT SISTRE))
(ASSERT (PHYSICAL-EQUIPMENT-CONSTRUCTOR SISTRE "Bosch"))
(ASSERT (EQUIPMENT-MODEL PLATEFORME))
(ASSERT (EQUIPMENT-HAS-TOP-MODEL SISTRE PLATEFORME))
(ASSERT (EQUIPMENT-COMPOSED SISTRE ROBOT))
…
(ASSERT (EQUIPMENT-COMPOSED SISTRE CONVOYEUR))
(ASSERT (EQUIPMENT-COMPONENT-COMPOSED SISTRE CAMERA-DE-SERVEILLANCE))
(ASSERT (COMPONENT ENTRETOISE))
(ASSERT (EQUIPMENT-COMPONENT-COMPOSED CONVOYEUR ENTRETOISE))
(ASSERT (EQUIPMENT-COMPOSED CONVOYEUR COURROIE))
```

The set of defined concepts and assertions are included in the knowledge base of the maintenance platform. Some constraint rules can be added to enrich this knowledge base and to give greater precision concerning concepts such as, for example, the definition of constraint for `CRITICAL-COMPONENT` which is `PHYSICAL-EQUIPMENT` having more than 6 relations of `HAS-INTERVENTION`:

```
(DEFCONCEPT CRITICAL-COMPONENT ((?P PHYSICAL-EQUIPMENT))
 :<<=>> (AND (PHYSICAL-EQUIPMENT ?P) (> (HAS-INTERVENTION ?P ?i) 6)))
```

Also, there is some flexibility for the manipulation of PowerLoom. For example, this expression is equivalent to the following two expressions:

```
(DEFCONCEPT CRITICAL-COMPONENT ((?P PHYSICAL-EQUIPMENT))
(DEFRULE CRITICAL-COMPONENT (> (Has-Intervention ?P ?i) 6))
```

The java API of PowerLoom makes it possible to query the knowledge base (e.g. assertion of the ontology). This allows us to check the consistency of the ontological model and to verify the correctness of query answers.

In this example we query the list of all physical equipment. The given answer is not just the physical equipment SISTRE but its composition. In our ontology we specified that physical-equipment can be composed of items of physical-equipment having components as exploitation mode (see rules defined in section 3.2.4 and in Fig. 3).

```
PL-USER |= (load "ontologie-maintenance.plm")
PL-USER |= (in-module "ONTOLOGIE-MAINTENANCE")
ONTOLOGIE-MAINTENANCE |= (RETRIEVE ALL (PHYSICAL-EQUIPMENT ?PE))
There are 28 solutions:
#1: ?PE=COURROIE
  #2: ?PE=CONVOYEUR
  #3: ?PE=DETECTEUR
  #4: ?PE=ACTIONNEUR
  #5: ?PE=SISTRE
  …
#26: ?PE=BAL0
#27: ?PE=TAP-EXT
  #28: ?PE=TAP-IN
```

B- The maintenance intervention on the equipment

In addition, the implicit relations between concepts can be retrieved via PowerLoom's Java API. The first and second queries concerning interventions can be as expressive examples.

First Query:

```
ONTOLOGIE-MAINTENANCE |= (RETRIEVE ALL (Has-Intervention TEST02 ?I))

There are 2 solutions:

#1: ?I=INTERVENTION-5

#2: ?I=INTERVENTION-9
```

Second Query:

```
ONTOLOGIE-MAINTENANCE |= (RETRIEVE ALL (EXISTS (?D)

(AND   (EQUIPMENT-SCADA ?D TEST02)

(EXISTS (?M) (AND  (DATA-ACQUISITION-SYSTEM-CATCH-MEASURE ?D ?M)  (EXISTS (?T)
(AND (MEASURE-TRIGGER-EVENT ?M ?T)
(EXISTS (?WR) (AND     (WORK-REQUEST-ORIGIN ?WR ?T)
(HAS-WORK-REQUEST ?I ?WR)))
))  )) )) )

There are 2 solutions:

#1: ?I=INTERVENTION-5

#2: ?I=INTERVENTION-9
```

PowerLoom's reasoning engine can infer from IMAMO ontology the various maintenance interventions carried out on equipment through explicit and implicit relations.

C- Third use case: knowledge exploitation and reuse

In this section we present a simple example checking the possibility of exploiting existing knowledge to extract new knowledge. Our case study deals with knowledge about reusable components after disassembly of SISTRE. Knowledge necessary to the finding of reusable components exists in the knowledge base, but it is not exploited. We thus simply need to edit a new "abstract relation" defining reusable components. Then via the PowerLoom reasoning engine we can retrieve the list of reusable components in SISTRE. The following commands define a reusable component as one that has a functional period of less than 60000 hours, or fewer than three failures in operating mode. We consider that the functional period of a component is the same as that for physical equipment. Fig.12 shows the part of the ontology used to define this rule.

```
DEFRELATION  REUSABLE-COMPONENT-OF  ((?PE  PHYSICAL-EQUIPMENT)  (?COM  PHYSICAL-
EQUIPMENT))  :=>
      (OR
            AND(
                  (HAS-EXPLOITATION-MODE ?COM ?EM)
                  AND ((HAS-FUNCTIONAL-PERIOD ?EM ?FP)
                        (> (SUM (NUMBER-HOURS ?FP ?NH))60000)
                  )
            )
            (AND (HAS-PERIOD ?EM ?P)
                  AND ((DURING ?P ?OM)
```

```
                    AND((TRUE(FAILURE-STATE ?OM))
                        (>(COUNT(STATE_ID ?OM ?X))3)
                    )
                )
            )
        )
```

The following query gives the list of all reusable components of SISTRE:

```
ONTOLOGIE-MAINTENANCE |= (RETRIEVE ALL (REUSABLE-COMPONENT-OF SISTRE ?RC))
There is 1 solution:
  #1: ?RC= ACTIONNEUR
```

Thus, we note that "ACTIONNEUR" is the only reusable component of SISTRE. This information was unknown, but, through the defined relation that can be used on any physical equipment, new knowledge concerning the reusability of components can be extracted from the knowledge base. This type of extracted knowledge gives an added value to the maintenance system in support of decisions for cost decreases.

3.6 Maintenance guidelines

According to Yildiz (Yildiz, 2006) ontologies change and maintenance can take several forms: modification, versioning and evolution.

It is to be noted that defining maintenance operations for ontologies is not easy (Yildiz, 2006) since all possible resulting effects for the components of an ontology must be taken into account when a change is made. Klein (Klein, 2004) identifies three kinds of changes: conceptual (e.g. changing concept relations), specification (e.g. adding new properties [attributes] to a concept) and representation (i.e. formalization and or implementation phases through the use of another language for ontology representation).

Consequently, IMAMO can evolve according to two kinds of change: conceptual and specification. As mentioned in the evaluation section, one of the main evolutions needed for IMAMO is the addition of concept attributes. Also, as mentioned in the integration section, other ontologies can be integrated and reused to enrich IMAMO.

In addition to these aims, an initiative has been launched to create a website for IMAMO after the end of the SMAC project. This will render the ontology more widely accessible to the academic and industrial communities, ensure greater sharing and thus initiate collaborative evolution and versioning.

Concerning the versioning that Klein (Klein, 2004) defines as "the ability to manage ontologies changes their effects by creating and maintaining different variants of the ontology", this form of change is very important in the life cycle of IMAMO, considering that the major benefits of this ontology are its reuse and interoperability. This functionality is essential while the ontology is generic and will be used in different applications in the maintenance field. Two cases are to be noted: in the first, new versions of IMAMO can be shared amongst all these applications; in the second, with each new application a specific local version of IMAMO will be developed. Despite the differences between local versions, interoperability among them must also be ensured (Karray, Chebel-Morello, & Zerhouni, 2010).

4 Discussion and future study

As mentioned above, the creation of IMAMO favors two points, both the exploitation and the sharing of knowledge, as well as semantic interoperability on the maintenance platform.

Concerning the first point, we can exploit the capabilities of the PowerLoom classifier so as to reason in terms of the relations between IMAMO's concepts for the generation of new knowledge models for the support maintenance actors, for example the generation of the behavior model of equipment using relations between concepts presenting the structure of the equipment, and those presenting functions and requirements shown in the functional and dysfunctional view of the UML model. We are also currently working to integrate a self-learning module exploiting this ontology so that the platform's behaviors can evolve. In addition, and in perspective, we plan to investigate the development of an intelligent module of diagnosis based on equipment behavior models generated from IMAMO by means of the concepts and relation presented in the functional and dysfunctional views.

In order to resolve interoperability problems, our aim is also to share IMAMO for exploitation by the maintenance support applications that will be integrated within the maintenance platform. This exploitation can be simple, which means that the ontology is to be exploited as it is. For other cases, where IMAMO evolves locally in these integrated applications, we have developed an initial version of a semantic mediator system ensuring interoperability between the different local versions (Karray, Chebel-Morello, & Zerhouni, 2010) and we are working to enhance this system and to integrate it within the maintenance platform.

In contrast, within the field of semantic interoperability, in a large-scale maintenance information system, the mapping between domain ontologies would be easier if the ontologies to be mapped were derived from a standard upper ontology (Obrst, Semy, & Pulvermacher, 2004). It would be useful if we could link our ontology to an upper and already existing one. Two approaches exist for the use of upper ontologies: top-down and bottom-up (Uschold & Gruninger, 1996). Since IMAMO's concepts were not derived from an upper ontology, we must adopt the bottom-up approach and proceed to map our ontology with an upper one. This approach also capitalizes on the knowledge built into the upper ontology, but one would expect the mapping to be more challenging, as inconsistencies may exist between the maintenance domain and upper ontology to be adopted. In fact, a variety of upper ontologies are available such as Suggested Upper Merged Ontology (SUMO)[30], Upper Cyc Ontology[31], Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE)[32] and WordNet[33].Indeed, after a manual research through the SUMO and WordNet glossaries, we observed that several IMAMO concepts have already been defined as concepts in WordNet: equipment, actor, period, activity, process, resource and lubricant are a few examples. Consequently, we manually mapped all IMAMO concepts with WordNet and found that forty one (41) of the 110 concepts defined in IMAMO are already defined as concepts in this upper ontology, thirty three concepts can be classed among the sub-concepts of concepts defined

[30] http://www.ontologyportal.org/

[31] http://www.cyc.com/

[32] http://wonderweb.semanticweb.org/deliverables/documents/D18.pdf

[33] http://wordnetweb.princeton.edu

(e.g. maintenance strategy is considered as a sub-class of strategy, equipment group is considered as a sub-class of group, equipment mode is considered as a sub-class of mode, etc.).

Finally, the creation of IMAMO allowed us to take a close look at the delicacy and the difficulty of the tasks of ontology development: meeting the expectations of users, covering all the domain's aspects and respecting its requirements and standards.

The METHONTOLOGY methodology, by way of the very good structure of its activities, has guided us well. Nevertheless, we must insist on the importance of adding some kind of formal generic competency questions that will help ontology developers to better specify their needs and requirements before undertaking development, questions to guide them during the specification activity and also to help them produce better specification documents. Such a test might enable them to better identify the sources of knowledge. Given the importance and the impact of the specification activity throughout the development process, we recommend that it be enhanced and formalized through the addition of a guide for good practices.

5 Conclusion

In the landscape of industrial maintenance today, systems integration and collaboration are believed to be the key enabling technologies that drive industrial maintenance management systems towards improved productivity and efficiency. From the pursuit of these aims emerges the prospect of integrated maintenance platforms aiming to provide maintenance actors with the right information exploitable at the right time. To reach these aims, the integrated maintenance platform features of semantic interoperability and knowledge sharing must be reinforced.

Indeed, ontologies provide a basic conceptual structure that is semantically unambiguous, requiring a formal and explicit representation of the domain that can be shared and reused. We thus adopted an ontology engineering approach that meets both of these requirements.

Due to the lack of a domain ontology covering all aspects of industrial maintenance, despite the presence of some standards in the field and different specific ontologies, in this study we have built a domain ontology called IMAMO (Industrial Maintenance Management Ontology) that we have presented in its development life cycle.

After an overview of different state-of-the-art methodologies of ontology creation, we adopted METHONTOLOGY due to its stability and maturity in comparison to other methods, as well as for its structured activities covering the entire life cycle of ontology development. In addition, a look at some state-of-the-art languages and ontology creation tools led us to choose the description language PowerLoom and its Java API for the development of IMAMO.

Furthermore, in the IMAMO life cycle, support activities such as "knowledge acquisition" and "integration" enabled us to better perform development activities such as "specification" and "conceptualization". In these support activities we took into account previous works on standards and projects produced in the field of maintenance, such as the MIMOSA-CRIS conceptual model of the maintenance process (Rasovska, Chebel–Morello, & Zerhouni, A mix method of knowledge capitalization in maintenance, 2008) and the ontology of

diagnosis and help system repair (Rasovska, Chebel -Morello, & Zerhouni, 2005), or those related to maintenance such as the SOM model from the PROMISE project, as well as software maintenance ontologies.

Regarding the conceptualization activity, we chose the UML class diagram for a semi formal conceptualization, motivated by the expressiveness of that language.

Thus, to obtain a readable and understandable model, particularly concerning the relations between concepts, we broke the class diagram down into eight views according to their points of focus.

To transform the conceptual model into a formal model, we used a presentation system allowing the encoding of class diagrams in ALCQHI (a variant of description logic), after which we implemented this formal model using PowerLoom 3.2.0.

While the evaluation activity supports the creation of high quality ontologies at the conceptual, operational and functional levels, these different levels of evaluation do not guarantee a perfect match with the expectations of the user operating the ontology. Indeed, it is when the ontology is actually used that any performance gaps are detected.

Thus, knowing that there is no standardized, objective nor any widely accepted methods for ontology evaluation, to evaluate IMAMO we based our work on and sought answers within the four types of approach identified by Brank et al.

An initial evaluation of the quality of the conceptual model according to the metrics describing ontology conceptualization allowed to us to come to some conclusions. Hence, our ontology is not a hierarchy; it is rich in its inheritance and other relationships. Also, it is a hybrid, striking a balance between generality and explicitness, but it is poor in terms of attributes. In contrast, a second, business-oriented, evaluation focused on the added value of the ontology by respectively using the human evaluation and application approaches. Thus, this business evaluation allowed us to observe the wealth of manipulations via the PowerLoom classifier as well as the possibilities of ensuring semantic interoperability and knowledge generation.

In the final development activity in METHONTOLOGY which is the maintenance activity, we discussed the possible evolution that IMAMO may undergo as to specification and versioning.

The bottom-up approach to mapping was manually applied to IMAMO, mapping it to the upper ontology WordNet. We observed that 30 concepts (equipment, actor, period, activity, process, resource, lubricant, etc.) have already been defined in that ontology and, consequently, IMAMO can be mapped to other ontologies derived from it.

ACKNOWLEDGMENT

This work was carried out and funded in the framework of SMAC project (Semantic-maintenance and life cycle), supported by Interreg IV programme between France and Switzerland.

REFERENCES

Bangemann, T., Rebeuf, X., Reboul, D., Schulze, A., Szymanski, J., Thomesse, J., et al. (2006). PROTEUS - Creating Distributed Maintenance Systems through an Integration Platform. *Computers in Industry* , 539-551.

Berardi, D., Calvanese, D., & De Giacomo, G. (2005). Reasoning on UML class diagrams. *Artificial Intelligence, 168* , 70–118.

Bézivin, J. (2000). De la programmation par objets à la modélisation par ontologie. . *Journal of Ingénierie de connaissances* .

Brank, J., Grobelnik, M., & Mladenić, D. (2005). A survey of ontology evaluation techniques. *Proceedings of Conference on Data Mining and Data Warehouses* .

Campos, J. (2009). Development in the application of ICT in condition monitoring and maintenance. *Computers in Industry 60 (1)* , 1-22.

Chalupsky, H., MacGregor, R., & Russ, T. (2010). *PowerLOOM manual Powerful knowledge representation and reasoning with delivery in Common-Lisp, Java, and C++ Version: 1.48 16.* University of Southern California. .

Charlet, J., Bachimont, B., & Troncy, R. (2004). Ontologies pour le Web sémantique. *Revue I3, numéro Hors Série «Web sémantique»* .

Corcho, O., Fernández, M., Gómez-Pérez, A., & López-Cima, A. (2005). Building legal ontologies with methontology and webode. Dans L. a. Web, *R. Benjamins, P. Casanovas, J. Breuker, and A. Gangemi* (pp. 142–157). Berlin: Springer-Verlag.

Corcho, O., Fernandez-Lopez, M., & Gomez-Perez, A. (2003). Methodologies, tools and languages for building ontologies: Where is their meeting point? *Data & Knowledge Engineering, Vol. 46* , 41-64.

Cranefield, S. (2001). Networked Knowledge Representation and Exchange using UML and RDF. *Journal of Digital Information, Vol 1, No 8* .

Cranefield, S., & Purvis, M. (2000). Extending agent messaging to enable OO information exchange. *Proceedings of the 5th European Meeting on Cybernetics and Systems Research.*

Fernandez-Lopez, M., & Corcho, O. (2004). *Ontological Engineering.* Berlin: Springer-Verlag.

Fernández-López, M., Gómez-Pérez, A., & Juristo, N. (1997). Methontology: from ontological art towards ontological engineering. *Proceedings of Symposium on Ontological Engineering of AAAI* .

Fox, M. (1992). The TOVE Project, Towards a Common Sense Model of the Enterprise. *Enterprise Integration* .

Frankovic, B., & Budinska, I. (2006). The role of ontology in building of knowledge systems for industrial applications. *Proceedings of the 4th Slovakian - Hungarian Joint Symposium on Applied Machine Intelligence*, (pp. 15-25).

Gómez-Pérez, A. F. (1996). Towards a Method to Conceptualize Domain Ontologies . *Proceedings of the Workshop on Ontological Engineering. ECAI'96. (pp. 41-52).* Budapest. Hungary.

Grüninger, M., & Fox, M. S. (1994). The Role of Competency Questions in Enterprise Engineering. Dans A. Rolstadas, *Benchmarking—Theory and Practice* (pp. 83–95). London: Chapman and Hall.

Guarino, N. (1998). Formal Ontology and Information Systems. *Proceedings of FOIS'98: Formal Ontology and Information Systems* (pp. 3-15). Trento: Italy: IOS Press.

Guarino, N., & Welty, C. (2002). Evaluating Ontological Decisions with OntoClean. *Communications of the ACM. 45(2)* , 61-65.

Hung, M., Chen, K., Ho, R., & Cheng, F. (2003). Development of an e-diagnostics / maintenance framework for semiconductor factories with security considerations. *Advanced Engineering Informatics* , 165-178.

Jardine, A. K., Daming, L., & Banjevic, D. (2006). review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing* , 1483-1510.

Kaffel, H. (2001). *La maintenance distribuée: concept, évaluation et mise en oeuvre.* Quebec: Thèse de doctorat, Université Laval,.

Kahn, J. (2003). Overview of MIMOSA and the Open System Architecture for Enterprise Application Integration. *Proceedings of the 8th Condition Monitoring and Diagnostic Engineering Management (COMADEM)*, (pp. 661-670). Växjö University, Sweden.

Karray, M. H., Chebel-Morello, B., & Zerhouni, N. (2010). A contextual semantic mediator for a distributed cooperative maintenance platform. *Proceedings of 8th IEEE International Conference on Industrial Informatics (INDIN'10).* Osaka: Japan.

Karray, M. H., Chebel-Morello, B., & Zerhouni, N. (2009). Toward a maintenance semantic architecture. *Proceedings of the Fourth World Congress on Engineering Asset Management (WCEAM)* (pp. 98-111). Athens: Springer-Verlag London .

Kitamura, Y., & Mizoguchi, R. (1999). An Ontological Analysis of Fault Process and Category of Faults. *Proceedings of Tenth International Workshop on Principles of Diagnosis, (pp. 118-128).*

Kitamura, Y., & Mizoguchi, R. (1998). Functional Ontology for Functional Understanding. . *International Workshop on Qualitative Reasoning (QR-98)* (pp. 77-87). Cape Cod, USA: AAAI Press.

Kitchenham, B., on Mayrhauser, A., Niessink, F., Schneidewind, N., Singer, J., Takada, S., et al. (1999). Towards an Ontology of Software Maintenance. *Journal of Software Maintenance: Research and Practice 11* .

Klein, M. (2004). *Change Management for Distributed Ontologies.* PhD thesis, Department of Computer Science, Vrije Universiteit Amsterdam.

Lee, J., Liao, L., Lapira, E., Ni, J., & Li, L. (2009). Informatics Platform for Designing and Deploying e-Manufacturing Systems. *Collaborative Design and Planning for Digital Manufacturing* , 1-35.

Levrat, E., Iung, B., & Crespo-Marquez, A. (2008). e-Maintenance: review and conceptual framework. *Production Planning & Control 19 (4)* , 1-22.

Liyanage, J., & Kumar, U. (2003). Towards a value-based view on operations and maintenance performance management. *Journal of Quality in Maintenance Engineering,Vol. 9* , 333–350 .

March, S., Hevner, A., & Ram, S. (2000). Research commentary: An agenda for information technology research in heterogeneous and distributed environment. *Information System Research 11, 4* , 327–341.

Matsokis, A., & Kiritsis, D. (2009). An Ontology-based Approach for Product Lifecycle Management. *Computers in Industry. Special Issue: Semantic Web Computing in Industry* .

Matsokis, A., Karray, M., Morello-Chebel, B., & Kiritsis, D. (2010). An Ontology-based Model for providing Semantic Maintenance. *Proceedings of the 1st IFAC workshop on Advanced Maintenance Engineering, Services and Technology (A-MEST'10).*

Mizogouchi, R., & Bourdeau, J. (2004). Le rôle de l'ingénierie ontologique dans le domaine des EIAH. *Revue STICEF, Volume 11* .

Mizoguchi, R. (2004). Tutorial on ontological engineering. . *New Generation Computing 22(2)* , 198-220.

Muller, A., Marquez, A. C., & Iung, B. (2008). On the concept of e-maintenance: Review and current research. *Journal of Reliability Engineering and System Safety* , 1165–1187.

Obrst, L., Semy, S., & Pulvermacher, M. (2004). Upper Ontology Distinctions for Use in U.S. Government and Military Domains. *Third International Conference on Formal Ontology in Information Systems, FOIS-04.* Torino, Italy.

Obrst, L., Werner, C., Inderjeet, M., Steve, R., & Smith, B. (2007). The Evaluation of Ontologies: Toward Improved Semantic Interoperability. Dans J. O. Christopher, Baker, & K.-H. Cheung, *Semantic Web: Revolutionizing Knowledge Discovery in the Life Sciences.* Springer.

OntoWeb Consortium. (2002). *Deliverable 1.3: A survey on ontology tools.*

Pinto, H. S., Tempich, C., & Staab, S. (2004). Diligent: Towards a ¯ne-grained methodology for distributed, loosely-controlled and evolving engingeering of ontologies. *In Proceedings of the 16th European Conference on Arti¯cial Intelligence (ECAI).*

PROMISE. (2008). *SOM FP6 project. www.promise.no.*

Rasovska, I. (2006). *Contribution à une méthodologie de capitalisation des connaissances basée sur le raisonnement à partir de cas : Application au diagnostic dans une plateforme d'e-maintenance.* Thèse de doctorat, Université de Franche-Comté.

Rasovska, I., Chebel -Morello, B., & Zerhouni, N. (2005). Process of s-maintenance: decision support system for maintenance intervention. *Proceedings of the 10th IEEE International Conference on Emerging Technologies and Factory Automation.* Catania, Italy.

Rasovska, I., Chebel–Morello, B., & Zerhouni, N. (2008). A mix method of knowledge capitalization in maintenance. *Journal of Intelligent Manufacturing Volume 19, Number 3* , 347-359 .

Rasovska, I., Morello-Chebel, B., & Zerhouni, N. (2007). Classification des différentes architectures en maintenance . *Proceedings du 7ème Congrès international de génie industriel.* Trois-Rivières, Canada.

Retour, D., Bouche, M., & Plauchu, V. (1990). Où va la maintenance industrielle. *Problèmes Économiques, No. 2.159* , 7-13.

Ruiz, F., Vizcaino, A., Piattini, M., & García, F. (2004). An Ontology for the management of software maintenance projects. *International Journal of Software Engineering and Knowledge* .

Suárez-Figueroa, M. (2010). *NeOn Methodology for Building Ontology Networks: Specification, Scheduling and Reuse.* PhD Thesis. Universidad Politécnica de Madrid. Available at "http://oa.upm.es/3879/".

Suárez-Figueroa, M., & Gómez-Pérez, A. (2008). First Attempt towards a Standard Glossary of Ontology Engineering Terminology. *The 8th International Conference on Terminology and Knowledge Engineering.* Copenhagen, DENMARK.

Tartir, S., Arpinar, I. B., Moore, M., Sheth, A. P., & Aleman-Meza, B. (2005). OntoQA: Metric-based Ontology quality analysis . *Proceedings of IEEE Workshop on Knowledge Acquisition from Distributed, Autonomous, Semantically Heterogeneous Data and Knowledge Sources.*

Tatir, S., & Budak Arpinar, I. (2007). Ontology Evaluation and Ranking using OntoQA. *Proceedings Int. Conf. on Semantic Computing (ICSC)* .

Uschold, M., & Gruninger, M. (1996). Ontologies: Principles, Methods and Applications. *Engineering Review 11 No. 2* , 93-113.

Uschold, M., & King, M. (1995). Towards a Methodology for Building Ontologies. *Workshop on Basic Ontological Issues in Knowledge Sharing* .

Van Der Straeten, R., Simmonds, J., Mens, T., & Jonckers, V. (2003). Using description logic to maintain consistency between UML models. *Lecture Notes in Computer Science vol. 2863* , 326–340.

Welty, C., & Andersen, W. (2005). Towards OntoClean 2.0: a framework for rigidity. *Journal of Applied Ontology 1(1)* , 107-116.

Yildiz, B. (2006). *Ontology Evolution and Versioning.* Technical Report, TU Vienna, .

Table1-ONTOLOGY REQUIREMENT SPECIFICATION DOCUMENT

Domain	Industrial maintenance
Name	IMAMO: Industrial MAintenance Management Ontology
Date	2010
Conceptualized by	Mohamed-HediKarray, Brigitte Morello, Thibault Bobyck
Implemented by	Mohamed-HediKarray, Thibault Bobyck
Purpose	Ontology concerns most concepts of industrial maintenance when information about all technical, administrative and managerial activities and actions is required in maintenance information systems. This ontology can be used to ascertain decision making throughout the life cycle of maintenance activities from failure detection to intervention and repair.
Level of Formality	Formal
Scope	Structure of equipment to be maintained, spare parts, monitoring activity, failure detection, events, material resources, maintenance actors, technical documents, administrative documents, intervention, maintenance reports, equipment states, equipment life cycle.
Sources of Knowledge	Standards (AFNOR, MIMOSA..), projects, experts

Table 2–EXAMPLE OF A REUSE TABLE

Ontologies and models sources	Mapped Concepts	Concepts Reused in IMAMO
PROMISE // MIMOSA-CRIS	*Product* -- *Asset*	Physical equipment
MIMOSA-CRIS // PROMISE	*Model*-- *As-designed-product*	Equipment Model
MIMOSA-CRIS // PROMISE	*Asset type*-- *product group*	Equipment group
MIMOSA-CRIS // SMAC-Model	*site* -- *Location site*	Site
MIMOSA-CRIS // PROMISE	*Asset Event* + *Event Type* +*Measurement Event* – *Event*	Triggering event
MIMOSA-CRIS // PROMISE	*Measurement Event*-- *Field Data*	Measure
MIMOSA-CRIS	*Geoposition*	Equipment location + Geo-location system
MIMOSA-CRIS // SMAC-Model	*Alarm type* -- *Alarm*	Alarm
MIMOSA-CRIS // SMAC-Model	*Work Management Type*-- *Process*	Process pattern
MIMOSA-CRIS // SMAC-Model	*Work Task Type*-- *Process*	Intervention type
MIMOSA-CRIS // PROMISE	*Work Step* -- *Activity*	Step
MIMOSA-CRIS // PROMISE	*Work Order* -- *Document resource*	Work Order
MIMOSA-CRIS // SMAC-Model	*Work Request*-- *Process*	Work request process
MIMOSA-CRIS // SMAC-Model // SMO	*Agent* -- *personal resource* -- *Human resource* + *software resource*+ *hardware resource*	Actor
MIMOSA-CRIS // PROMISE // SMO	work step -- Activity -- Maintenance activity	Maintenance task
MIMOSA-CRIS // PROMISE // SMO	*Logistic Resource* -- *resource* -- *resource*	Resource

MIMOSA-CRIS // SMAC-Model	Asset Function + Model Function -- *function + function group*	Function + sub-function

Table 3 – DATA DICTIONARY OF THE STRUCTURAL VIEW

Concept Name	Synonyms	Description
domain		A specific field of knowledge or expertise (e.g. hydraulics).
physical equipment	Asset Physical- product Machine Device Item	A tangible, instantiated, serialized object, component, device, subsystem, functional unit, equipment or system which can be individually considered to be in maintenance. Physical equipment may be an entire facility, an entire functioning platform (such as a CH-47 Tail Number XYZ helicopter), or a component piece of equipment, such as a specific instance of a bearing.
transportation equipment		Specific physical equipment conveyance. A conveyance which may contain one or more area (s) of production, a set of maintenance teams, and a set of stores. For example: A fishing vessel off the coast has its own decomposition (a motor allowing it to move, etc., as well as a production department that cleans and freezes the fish).
maintenance tool		Specific physical equipment used as a tool to perform maintenance activities. This type of physical equipment must also undergo maintenance.
equipment model	As-designed-product Model	Conceptual view of physical composition of the equipment. It is comprised of the various component models of the components of the physical equipment.
component model		Conceptual view of a component (e.g. model of an electrical motor).
component		Component is an exploitation mode that can be played by physical equipment. It has the particularity of being found within superior physical equipment (e.g. motor3X57H).
exploitation mode		Abstraction of a role played by equipment. It presents the state of exploitation that can take physical equipment. It can be exploited as a component, production equipment, a spare part or be under repair.
equipment under repair		Specific exploitation mode affected to physical equipment while it is being repaired or is in a maintenance center awaiting repair.
production equipment		Specific exploitation mode affected to physical equipment while it is exploited in production tasks and/or located in a production area.
spare part		Specific exploitation mode affected to physical equipment intended to replace corresponding physical equipment in order to restore the original required function of the physical equipment. Generally, it is located in a store.
equipment location		Position of physical equipment in a production area (to locate and track the equipment's position).
area		Particular geographical region (for multisite management).
sub area		Region that makes up part of an area.
site		Place or setting of something. An area or plot of ground with defined limits on which a building, project, park, etc., is located or proposed to be located.
maintenance center	Maintenance workshop	Specific area for maintenance tasks.
Store		Stock or supply reserved for physical equipment for future use.
production area		Specific area for production tasks.
period		Time interval.
functional period		Typical period during which the equipment must perform certain functions.

Table 4 – DATA DICTIONARY OF THE EVENT VIEW

Concept Name	Synonyms	Description
Measure	Measurement	Number or measure or quantity captured by a sensor.
Magnitude		Greatness of size or amount. It presents the property of relative measure.
data acquisition system		Software system (abbreviated with the acronym DAS or DAQ) that typically converts analog waveforms generally retrieved from sensors into digital values for processing.
Condition		Environmental or functional requirement defined to supervise (monitoring task) specific physical equipment or a place (e.g. site) by the use of sensors and data acquisition systems.
triggering event		Something that happens to physical equipment at a given time that triggers a specific maintenance process which is a work request process.
Alarm		Type of triggering event launched from a data acquisition system indicating that there is a measure from a sensor violating some conditions concerning a specific equipment or environment.
improvement request		Triggering event concerning a specific or general request for the improvement of physical equipment. An improvement is defined asthe combination of all technical, administrative and managerial actions, intended to improve the dependability of physical equipment, without changing its required function.
event observed by user		Type of triggering event concerning a dysfunction of physical equipment observed by the user who is a human resource.
Notification		Type of triggering event giving notice of future events such as planned maintenance or the prognostic RUL.
Prognostic		Type of notification consisting of the health status at a future time and the remaining useful life (RUL) of physical equipment. It is the output of the prognostic tool.
prognostic tool		See Table 6.
maintenance scheduler		See Table 6.

Table 5 – DATA DICTIONARY OF THE FUNCTIONAL AND DYSFUNTIONAL VIEW

Concept Name	Synonyms	Description
function		Function or a combination of functions of physical equipment which are considered necessary to provide a given service.
sub function		Elementary function in a combination of functions.
functional requirement		Need required by physical equipment to perform a particular function. It references the input or the output of physical equipment (e.g. the printer needs paper, electrical energy and ink in order to print).
functional environment		Type of functional requirement presenting the environmental needs (e.g. temperature, humidity, etc) of the physical equipment for it to perform its functions.
matter		Type of functional requirement presenting the raw material (e.g. paper, oil, etc.) needed as input by physical equipment to perform a function, or the material product provided as output of the physical equipment during the performance of a function.
information		Type of functional requirement presenting the information (e.g. a value, PLC commands, etc.) needed as input by physical equipment to perform a function, or the information provided as the output of the physical equipment during the performance of a function.
energy		Type of functional requirement presenting a thermodynamic quantity equivalent to the capacity of a physical system (e.g. electric, hydraulic, etc) needed as input by physical equipment to perform a function or the energy provided as output of physical equipment while performing a function.
functional flow		Relational flow between two components defined by input/output flow.
equipment input		Input needed by physical equipment to perform a function. It enables management of the functional flow.
equipment output		Output provided by physical equipment while performing a function. It enables management of the functional flow.
operating mode	Availability performance	Ability of physical equipment to be in a state to perform a required function under given conditions at a given time or within a given time interval, assuming that the required external resources are provided.
normal state	Operating state	State when an item is performing a required function.
degraded state		State of physical equipment in which it continues to perform a function within acceptable limits but which are lower than the specified values, or continues to perform only some of its required functions.
programmed stop	Stand by state	Non-operating up state during the required time.
failure state	Fault	State of physical equipment characterized by inability to perform a required function, excluding inability during preventive maintenance or other planned actions, or due to lack of external resources.
trouble		Source of difficulty, a problem.
dysfunction		Particular trouble presenting an anatomy of function; it means that the physical equipment is not able to perform a required function.
causes		Relation between troubles. It is the reason which leads to trouble. The reasons may be the result of one or more of the following: design failure, manufacturing failure, installation failure, misuse failure, mishandling failure, and maintenance- related trouble.
effects		Relation between troubles. Consequence that follows and is caused by

		previous effects.
degradation		An irreversible process in one or more characteristics of an item with either time, use or an external cause.
failure		Termination of the ability of an item to perform a required function.
action		Physical action taken to recover and eliminate trouble.
skill	Competence	Ability acquired through deliberate, systematic, and sustained effort to smoothly and adaptively carryout complex activities or tasks involving ideas, things (technical skills) and/or people (interpersonal skills).
requirement not complied		Particular trouble caused by a non-respect of functional requirement needed to ensure the function of the physical equipment.

Table 6 – DATA DICTIONARY OF THE INFORMATIONAL VIEW

Concept Name	Synonyms	Description
resource	Maintenance support	Resources, services and management necessary to carry out maintenance.
actor		Person or a computer system that interacts in the maintenance process.
role		Prescribed or expected behavior associated with a particular position or status in the maintenance process.
manager		Role played by an actor in the maintenance process. It is responsible for the managerial tasks in the maintenance process.
operator		Role played by an actor in the maintenance process. It is responsible for the maintenance tasks and especially the intervention process and repair actions.
expert		Role played by an actor in the maintenance process. It is responsible for the maintenance tasks of monitoring, diagnosis, prognostics analyses and especially identification of causes of dysfunction and the corresponding repair actions necessary.
human resource		Human actors contributing to the all technical, administrative and managerial actions in the maintenance process.
document		Item containing some information concerning the maintenance domain, controlled and identified with a number in a document management System.
contract		Specific type of document defining a service agreement between a service provider and a physical equipment owner or exploiter for the maintenance of a set of physical equipment. The contract may be applicable to an entire model of equipment (e.g. for all engines) in one or more areas (area) (e.g. Peugeot site in Belfort), or to a set of production equipment (e.g. robot station 1).
financial document		Specific type of document containing financial information concerning all technical, administrative and managerial actions.
technical documentation		Specific type of document containing technical information about physical equipment or software resources (such as the user manual, SADT, etc).
equipment draw		Type of technical documentation containing the design information of the physical equipment.
work order	Job order Job ticket Work ticket	An order edited by an actor to plan an intervention concerning a work request. It is considered as a specific type of document containing information concerning the resources allocated to the intervention. It is received by actors designated to ensure activities comprising the intervention.
work request		A specific type of document edited by an actor when a triggering event is detected. Each work request is covered by a maintenance contract. The edition of a work request document launches the work request process.
software resource		Software actors contributing to all technical, administrative and managerial actions in the maintenance process.
maintenance scheduler		Type of software resource that allows planning, allocation of a significant amount of time and a high degree of coordination between different departments, and is typically initiated through a work order. It is considered a software resource.
resource scheduler		Type of software resource that allows maintenance actors to reserve any type of resource such as spare parts, a human resource, maintenance tools, company cars, and more! It allows the checking of resource availability and the reservation of online resources. It is considered a software resource.
geo-location		Type of software resource (GPS type, tag ...) that identifies the exact or

system		approximate location (e.g. zone) of production equipment (even mobile equipment).
e-doc system		Type of software resource called document management system (DMS) which is a computer system (or set of computer programs) used to track and store electronic documents and/or images of paper documents.
CMMS		Type of software resource. CMMS stands for Computerized Maintenance Management System, also known as Enterprise Asset Management and Computerized Maintenance Management Information System (CMMIS). The purpose of CMMS is to simplify the planning and administrative functions of maintenance, purchasing, and inventory management.
ERP		Type of software resource. Enterprise resource planning (ERP) is an integrated computer-based system used to manage internal and external resources including tangible assets, financial resources, materials, and human resources.
prognostic tool		Software tool or system allowing prediction and estimation of time remaining before failure and the risk of subsequent existence of one or more failure modes, having a confidence level which is a value indicating the degree of certitude that the prognosis is correct.
diagnostic tool		Type of software resource used to identify trouble recognition and localization, characteristic of a particular dysfunction and its causes. In certain cases it may provide or recommend actions for repair of the trouble.
diagnostic		Result provided by a diagnostic tool. It is mainly the tuple composed of the trouble, localization, cause, and actions.
external service		Considered a type of resource or service provided to the company by an external organization.
lubricant		Consumable substance such as grease or oil used to reduce friction between components in the maintenance of physical equipment. It is a type of resource.

Table 7 - DATA DICTIONARY OF THE INTERVENTION VIEW

Concept Name	Synonyms	Description
intervention		Specific type of process concerning the main technical part of maintenance which is the core of maintenance.
activity		Organizational unit for performing a specific action ensured by an actor.
intervention report		Specific type of document edited, containing information about the intervention such as observations and remarks on the part of actors. This type of document is exploited in the experience feedback process.

Table 8 - DATA DICTIONARY OF THE STRATEGY VIEW

Concept Name	Synonyms	Description
maintenance strategy		A long-term plan, covering all aspects of maintenance management, setting the direction for maintenance management and containing firm action plans for achieving a desired future state for the maintenance function with respect to a maintenance type.
technical indicator		Value calculated from different technical factors of maintenance concerning physical equipment maintained under a contract. It presents a specific technical evaluation of the contract.
financial indicator		Value calculated from different financial and managerial factors of maintenance concerning physical equipment maintained under a contract. It presents a specific financial evaluation of the contract.

Table 9 – DATA DICTIONARY OF THE PROCESSES VIEW

Concept Name	Synonyms	Description
goal		Objective to be reached. It is followed by maintenance strategy and fulfilled by a set of tasks (e.g. 80% availability).
goal requirement		Needs required to reach a goal (e.g. duplicate spare parts of critical physical equipment).
process		Sequence of interdependent and linked activities which, at every step, consume one or more resources (employee time, energy, machines, money) to convert inputs (data, material, parts, etc.) into outputs while respecting a process pattern (i.e. the process pattern presents the general model of the process). These transition outputs then serve as transition inputs for the next step until a known goal is reached.
process pattern		Pattern which describes a proven, successful approach and/or series of actions. A pattern is a description of a general solution to a common problem or issue from which a detailed solution to a specific problem may be determined.
maintenance type		Specific type of process pattern concerning the method for carrying out maintenance and the orchestration of processes used in order to achieve the maintenance strategy goals. There are 12 possible types of maintenance which are: Preventive maintenance, Scheduled maintenance, Predetermined maintenance, Condition-based maintenance, Predictive maintenance, Corrective maintenance, Remote maintenance, Deferred maintenance, Immediate maintenance, On-line maintenance, On-site maintenance and Operator maintenance.
maintenance task		Specific type of task concerning one part of maintenance work (e.g. repair, replace, inspect, lubricate, etc).
intervention type		Specific type of process pattern. It is the principle method of conveying the appropriate activities to all parties involved in an intervention on physical equipment. It presents the generic model of an intervention.
task		Work assigned or performed as part of one's duties. The task is evaluated by looking at its outcome in terms of completeness, accuracy, tolerance, clarity, error, or quantity.
repair action		Specific type of maintenance task defined as a physical action taken to restore the required function of faulty physical equipment.
production task		Specific type of task that terminates in a discrete product or outcome that is observable and measurable.
constraint		Restrictive condition for the control of transitions between steps.
activity		Organizational unit for performance of a specific action. An activity is the execution of a task, whether a physical activity or the execution of code. It presents the activity performed by an actor in the real world.
step		Maneuver under taken as a part of the progress made towards the progress of a process. It is referenced by an activity.
work request process		Specific type of process launched automatically or by an actor when receiving a work request. It allows management of the work request until resolution of the original problem triggering the event and the end of the intervention process, including edition of the intervention report.
transition		Passage from one step to another in the course of a process. It is the connection between two steps in a process. A transition ensures the move from one step to another according to a particular event.

Activity Input Output		Parameters, values and / or input events triggering the launching of an activity. These parameters can be the outputs from other activities. The execution result of an activity (output) is the input from another activity. Activity Input Output is the passing link from one activity to another. Therefore, each Activity Input Output can refer to a Transition.

Table 10–DATA DICTIONARY OF THE MIDDLE-OF-LIFE VIEW

Concept Name	Synonyms	Description
life record	life-Cycle-Phase	From a PLM approach, life record is added as a concept containing the pieces of information concerning the middle of life phase of physical equipment (e.g. all during its exploitation, from purchase to disassembly).

Table 11– METRICS OF IMAMO EVALUATION

Metric Name	**Metric Formula**	**IMAMO's Metric Result**	**Interpretation**
Relationship Richness	RR= $\frac{\|P\|}{\|SC\|+\|P\|}$	IMAMO_RR= 200/(200+61) = 0.76 = 76%	The RR of our ontological model largely exceeds the average. This means that our ontology is not a hierarchical one. It is not just a hierarchy of subclasses but it is rich with role associations. This is due to the inclusion of domain concepts and proves that the ontological model is business-oriented and responds to the maintenance needs of businesses.
Inheritance Richness	IRs= $\frac{\sum\|Hc(C1, Ci)\|}{\|C\|}$	IMAMO_$\Sigma H^C (C^1, C^i)$ = 61; IMAMO_IRs = 61/110 = 0.55.	This result is near the average of 0.5. This shows that in the context of knowledge details, our ontological model is hybrid; it is neither vertical (which might reflect a very detailed type of knowledge that the ontology represents) nor horizontal (which means that it represents a wide range of general knowledge). We consider this result as a target reached, because our first goal was to build a generic ontological model for the maintenance domain, but one that was simultaneously strong enough to cover as many maintenance aspects (concepts) as possible.
Attribute Richness	AR= $\frac{\|att\|}{\|C\|}$	IMAMO_AR =91 /110 = 0.82	The result obtained shows the poverty of the ontological model in terms of attributes. The result gives as an average of 0.81 attributes per concept which is very low. We have known this from the beginning because when constructing the model we chose to define concepts and not attributes so as to be more general. This choice was made in order to favor the reuse and the exploitation of these concepts by the different applications integrated into the platform, especially since we are concerned with all aspects of maintenance. However, for our future work, and in order to enrich these concepts, we envisage collaboration with business experts so as to include certain attributes (e.g. details of classes of the IMAMO model).

Table 12- FUNCCTIONALITIES ENSURED BY IMAMO

Category	Functionality	Does IMAMO allow…?
Questions about business aspects	Description of equipment as it is	Yes. By means of its structural model.
	Tracking the history of equipment	Yes, it is possible to retrieve information about current and past uses of the physical equipment through concepts "exploitation mode" and "functional period". Moreover, it retrieves the different modes of operation during the life cycle of equipment through the concepts of "operating mode", "triggering event", “intervention” and “life record."
	Management of maintenance data	Yes, it is possible to link information with the maintenance process through the concepts of process view, the concept "trouble" and the concepts of resources view, namely "intervention report", "work request", "diagnosis" and others.
	Extension of Coverage on PLM	Yes. It is able to connect this information with the different life-cycle phases via the ‘life record’, ‘equipment model’ and ‘period’ concepts.
	Automatic Data Processing for Events	Yes, via the event view via PowerLoom’s API and by means of the description classifier.
	Resources management	Yes, via the resource view including all types of resources.
	Monitoring and prognostic management	Yes, via the manipulation of event view and resource view, as well as the functional and dysfunctional view.
	Diagnosis management	Yes, via the manipulation of event view and resource view, as well as the functional and dysfunctional view.
	Strategy management	Yes, via the manipulation of strategy view and resource view, as well as the intervention view.
Questions about technical aspects	Execution	Yes, via PowerLoom API.
	Loading of Data	Yes. It can be instantiated via PowerLoom API.
	Consistency	Yes, via PowerLoom’s description classifier.
	Equivalencies	Yes, via PowerLoom’s description classifier.
	Reclassification	Yes, via PowerLoom’s description classifier.
	Inference	Yes, via PowerLoom’s description classifier.
	Importation/exportation of Data	Yes, via PowerLoom’s API and GUI.
	Importation of multiple Models under one source	Yes, via PowerLoom’s API and GUI.
	Merger of Models	Yes/No, depending on which tools are used.
	Simple Calculations	Yes, via PowerLoom’s API and thanks to description classifier.

	Restriction ofAll the classes	Yes, via PowerLoom's API and by means of the description classifier.

Figures captions

Figures

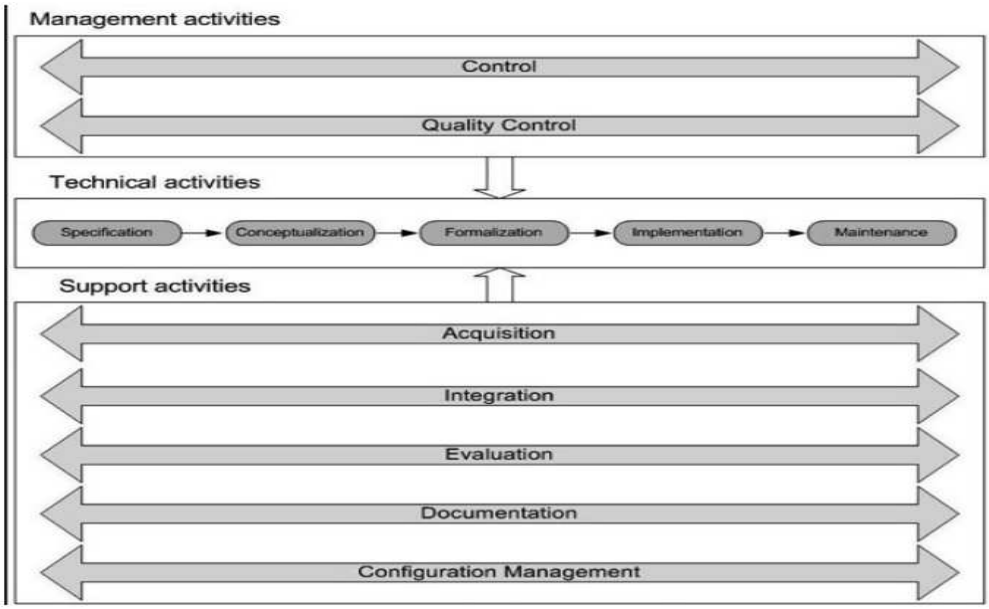

Fig. 1 -Components of METHONTOLOGY (Corcho, Fernández, Gómez-Pérez, & López-Cima, 2005)

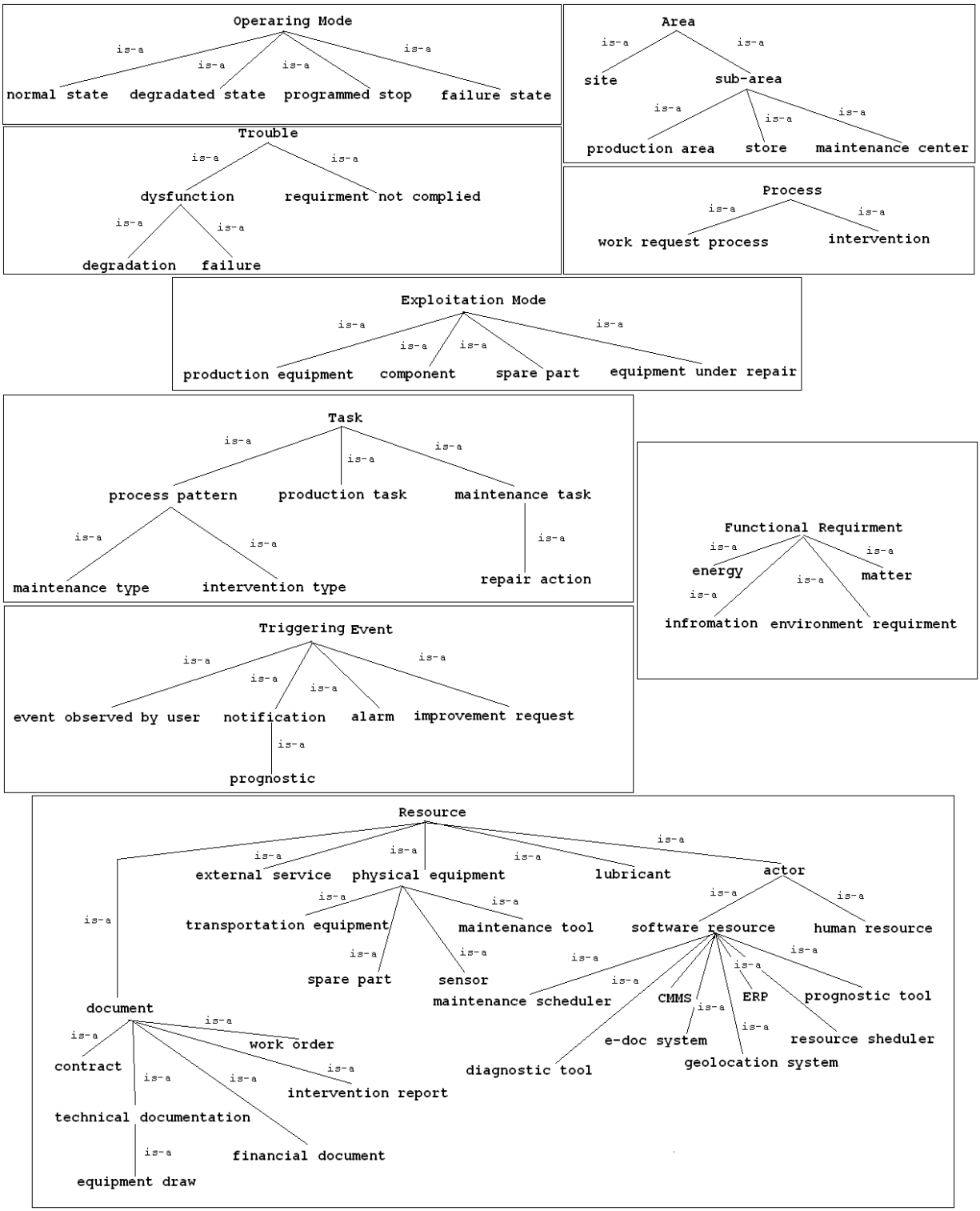

Fig.2 some examples of classification tress

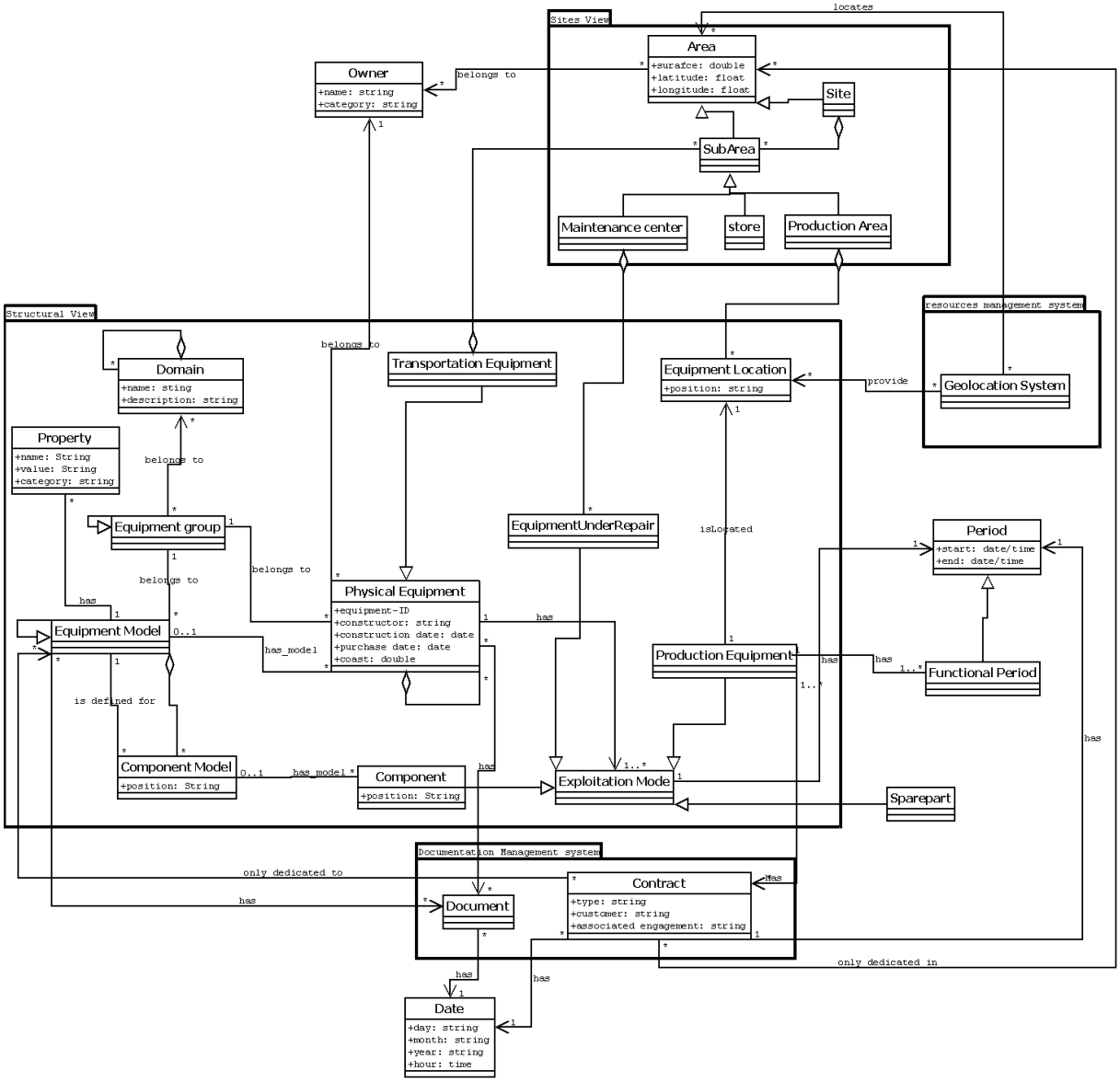

Fig 3. Structural view

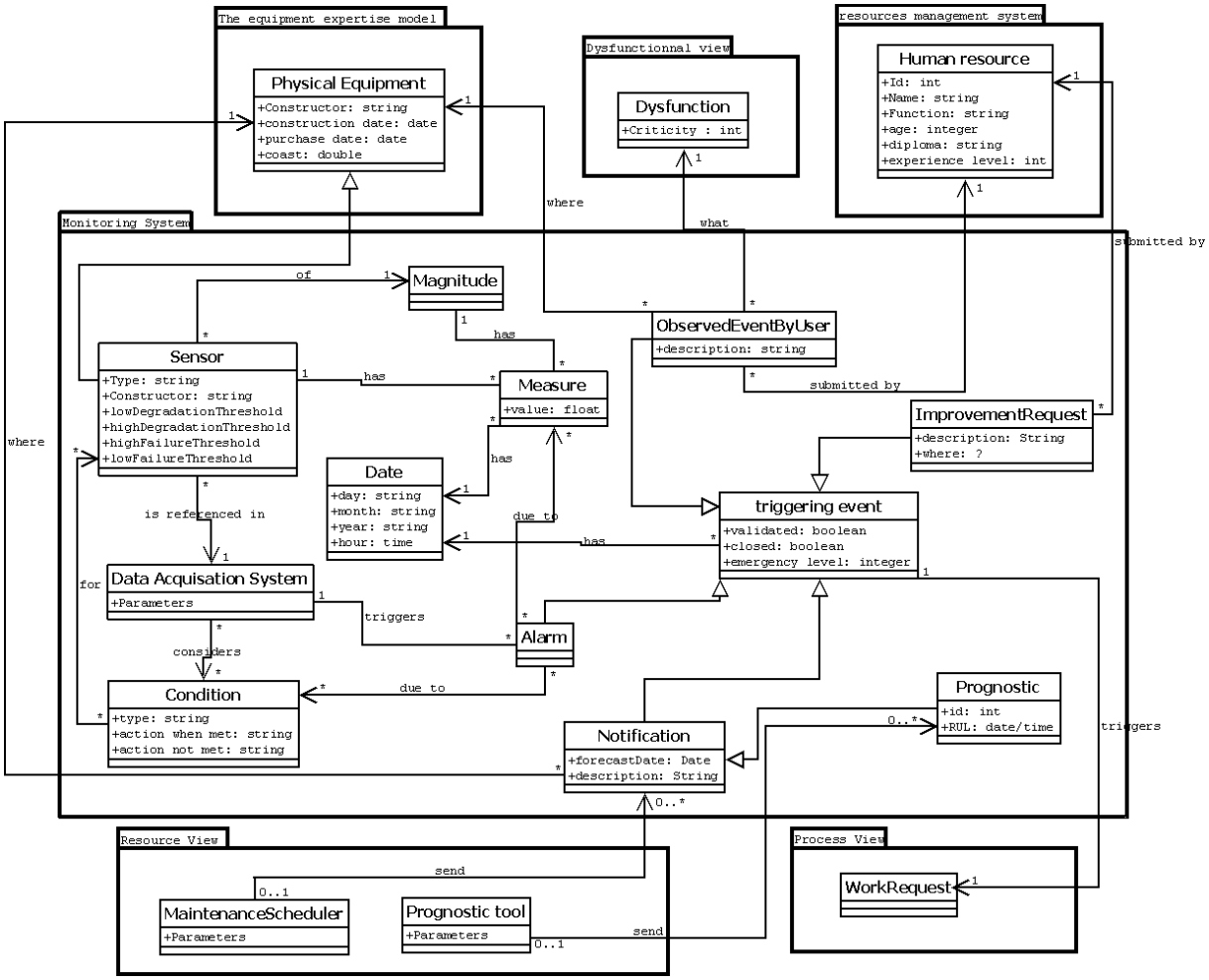

Fig 4. Event view

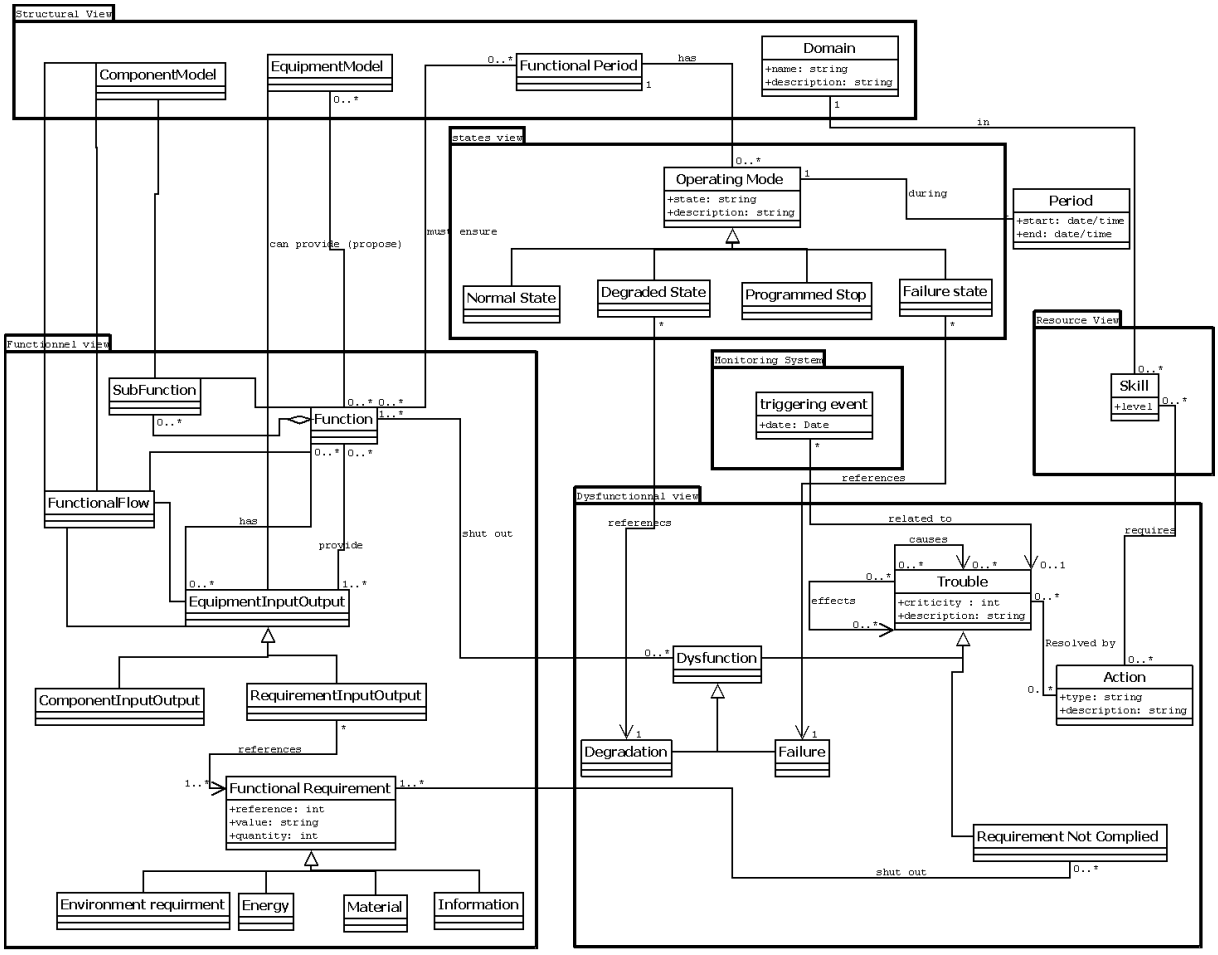

Fig 5. Functional and dysfunctional view

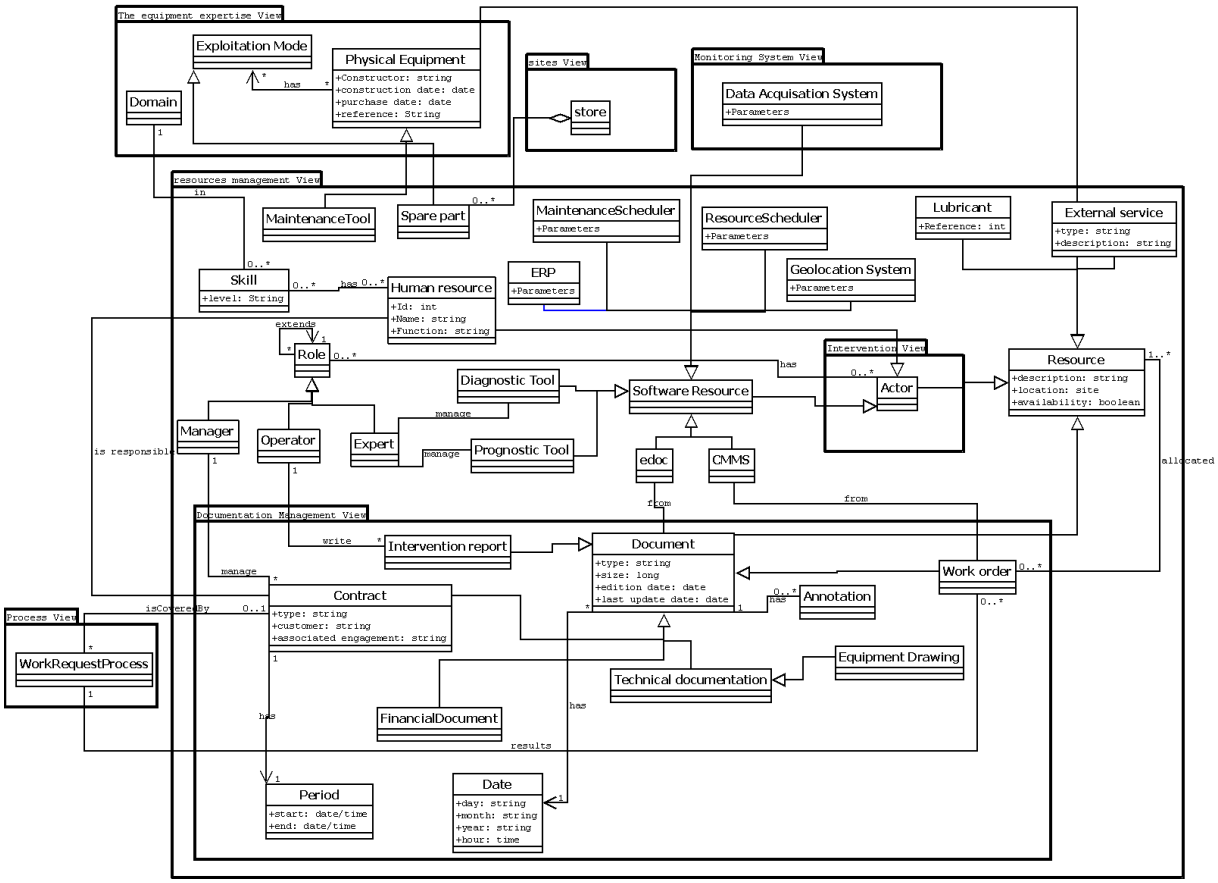

Fig 6. Informational view

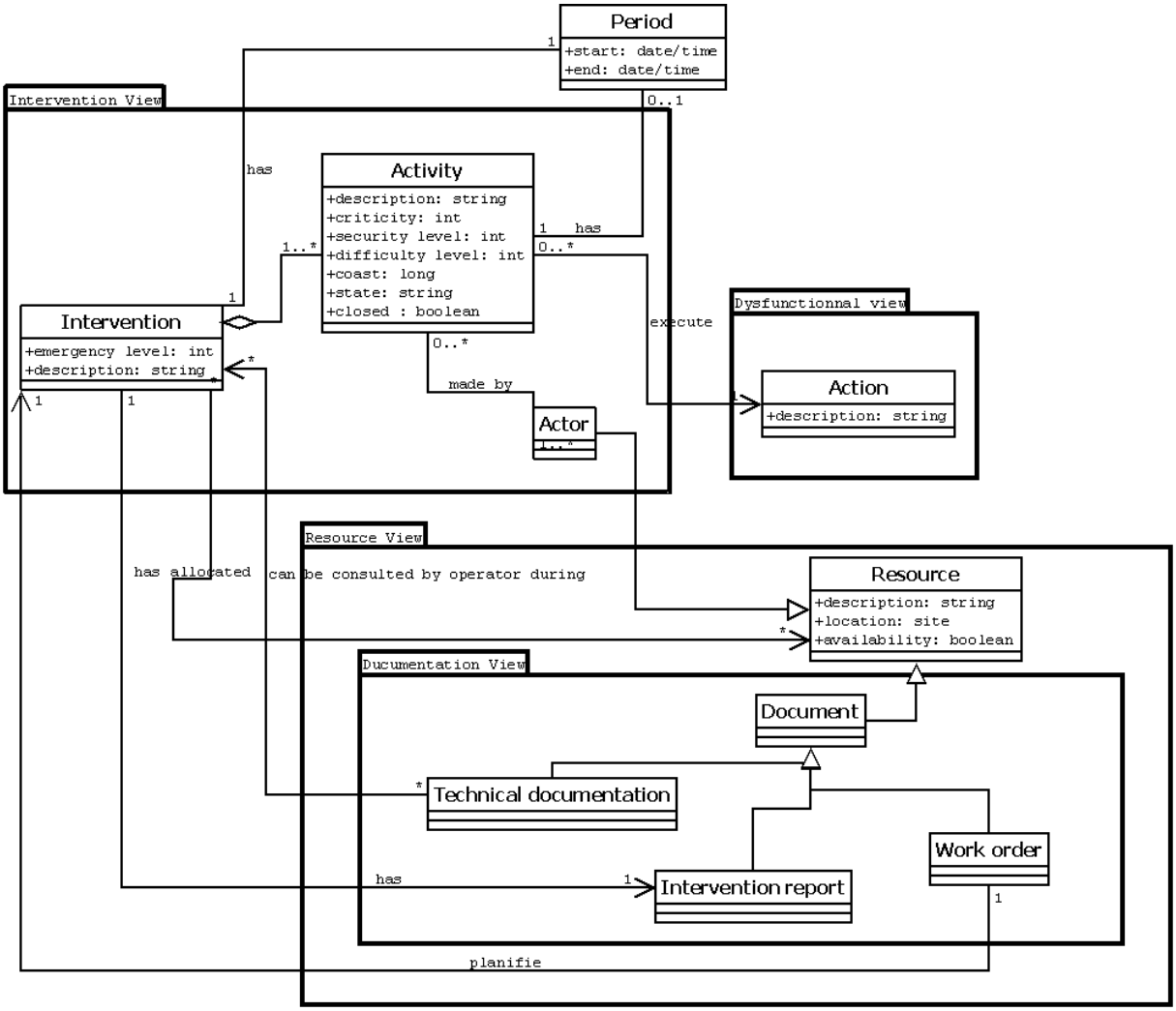

Fig 7. Intervention view

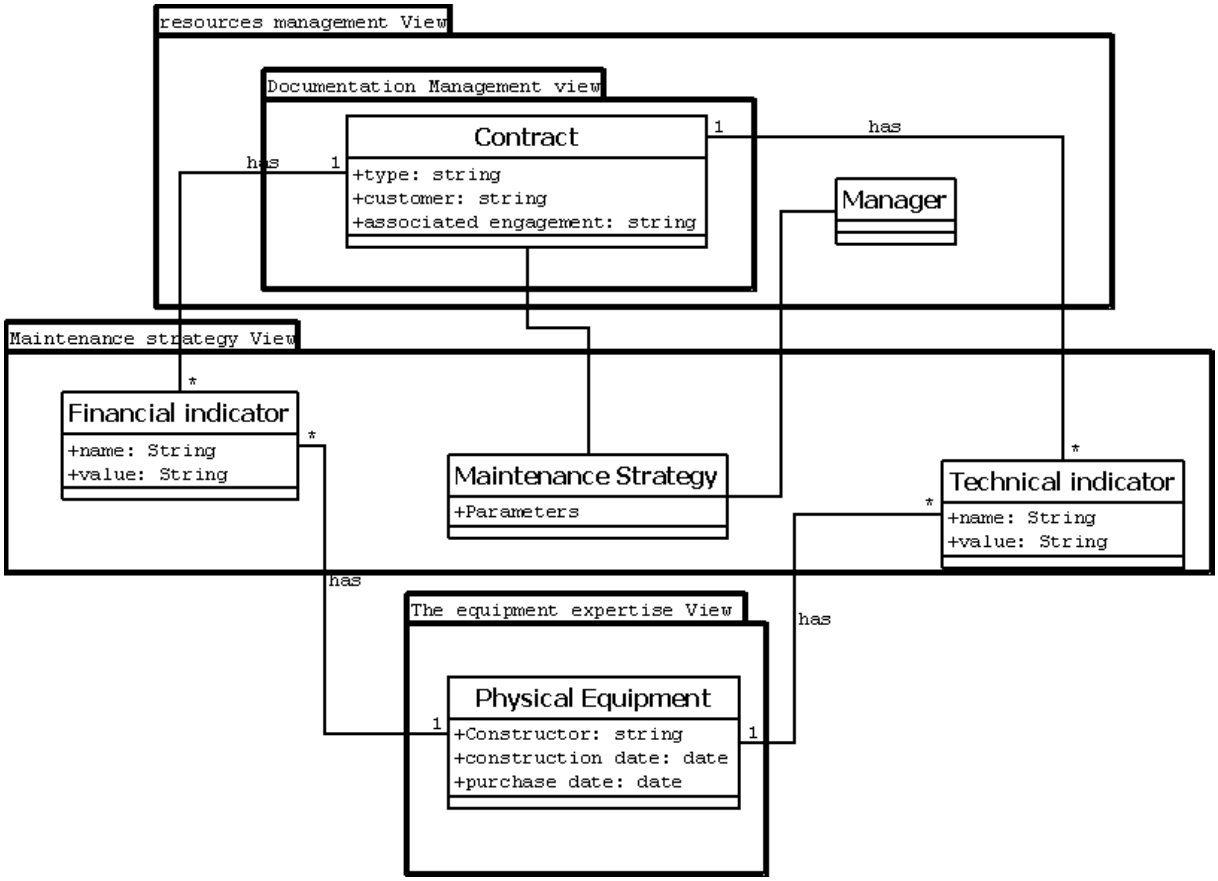

Fig 8. Strategy view

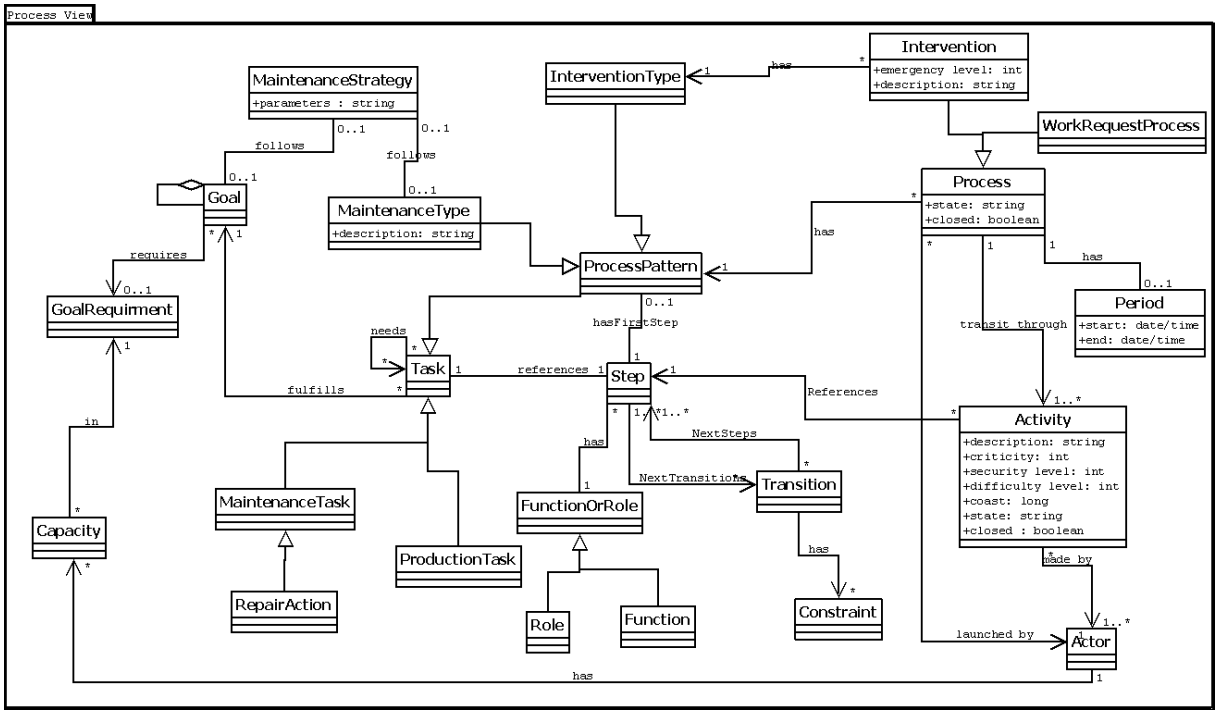

Fig 9. Processes view

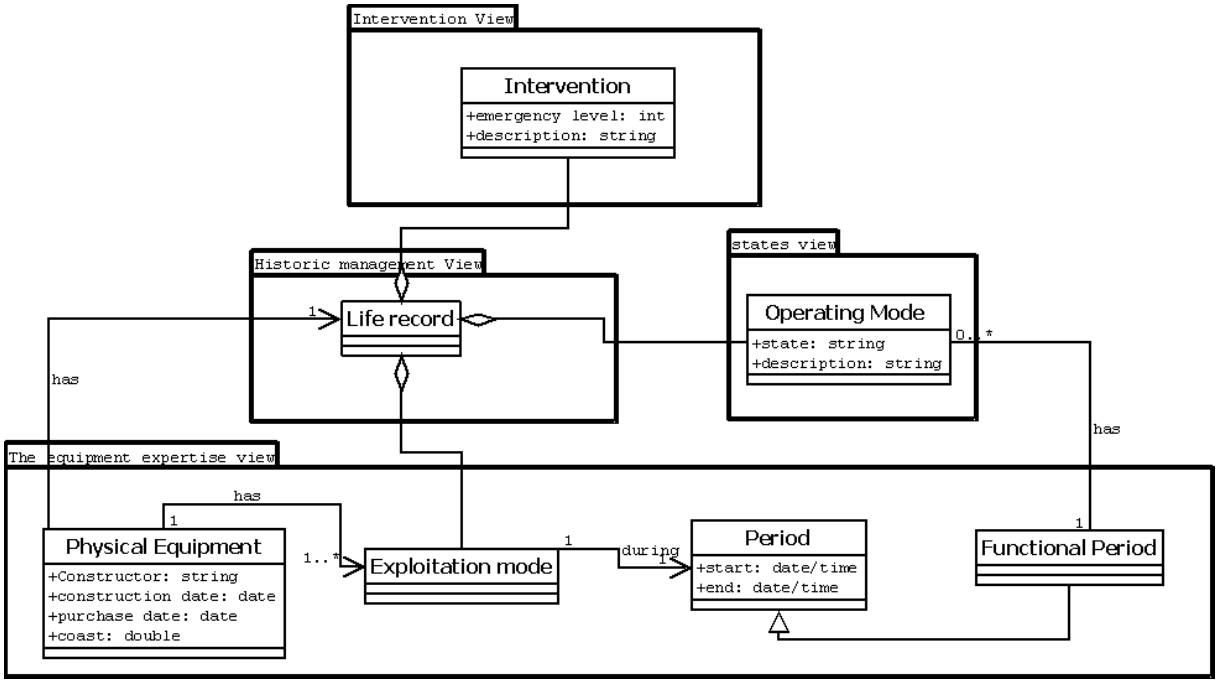

Fig 10. Middle of life view

Fig 11. The pallet transfer system “SISTRE”

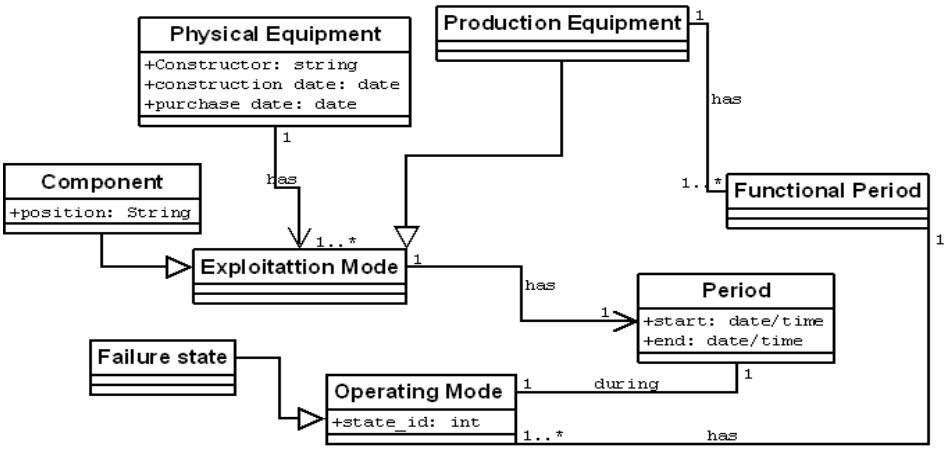

Fig 12. A part of the ontology used to define reusable component