IOEM - ontology engineering methodology for large systems*

Joanna Sliwa, Kamil Gleba[1],
Wojciech Chmiel, Piotr Szwed, and Andrzej Glowacz[2]

Military Communication Institute, Zegrze, Poland
{j.sliwa,k.gleba}@wil.waw.pl
and AGH University of Science and Technology, Krakow, Poland
{wch,pszwed,aglowacz}@agh.edu.pl

Abstract. The paper presents IOEM, a methodology for ontology development elaborated for the INSIGMA project. Although prepared for a particular use, the methodology is quite general and can be used in a large variety of IT projects requiring ontology components. It is particularly suitable for large and geographically distributed software projects. The methodology is oriented towards applications of ontologies in various phases of a software lifecycle: development and run-time.

Keywords: methodologies for building ontology, ontology engineering methodology, approaches for building ontologies for complex systems, ontology requirements, ontology life cycle.

1 Introduction

IOEM is an ontology engineering methodology designed for the Intelligent System for Global Monitoring Detection and Identification of Threats. The system is being developed within the INSIGMA project carried out by four Polish academic, research and commercial bodies (University of Science and Technology (AGH) - the consortium leader, Military Communication Institute (MCI) , Military University of Technology (WAT) and University of Computer Engineering and Telecommunication (WSTKT)). The effect of the project will be a complex monitoring system used to identify objects in the monitored environment and, based on the stored information and advanced algorithms, identify threats related to both - the traffic and suspicious behaviour of people. The system can be used for traffic management and route planning for individual users and for the public safety services as well. The route planning will also take into account complex parameters that provide the possibility to select the route in special circumstances, e.g. after a road accident or a natural disaster, in difficult weather conditions, etc.

* Work has been co-financed by the European Regional Developement Fund under the Innovative Economy Operational Programme, INSIGMA project no.POIG.01.01.02-00-062/09.

INSIGMA system will store and process large amounts of various types of data. Some of them will be raw data flowing from the sensors, some will have to be fused and processed in order to provide additional information. Moreover, the identification of threats requires methods for automatic recognition and classification of events in the system. One of the main tasks within the INSIGMA project is thus to define ontology that would help to manage the information and, at the same time, would help in automatic identification and classification tasks.

The objective of the article is to present INSIGMA Ontology Engineering Methodology (IOEM) designed for the large system with geographically dispersed groups of developers.

2 Application of ontologies in INSIGMA system

One of the key assumptions about the INSIGMA system is that it will be developed as an ontology driven information system. The role of ontologies for such systems can be classified in two dimensions: temporal and structural [1]. The temporal dimension is related to the stage in the software lifecycle: development time or run time. The structural dimension concerns the usage of ontologies in particular components: databases, user interface and business logic layer.

Ontologies developed within INSIGMA project fall into two categories: domain ontologies, specifying concepts of a particular domain (e.g. routes, vehicles, elements of a monitoring infrastructure, threats, weather) and task ontologies related to tasks, activities, processes (e.g. threats detection, services configuration, route planning). A particular software component can use both types of closely related domain and task ontologies.

Ontologies at the run time are used in situations where the domain model cannot be fully elaborated during the system development (some domain aspects are unknown or uncertain), or a kind of reasoning is required. In case of INSIGMA system, main uses are: classification of threats, their properties, reasoning about incidents, dynamic configuration of the system architecture to fit particular needs of an end-user, provision of semantic interoperability between components, that cannot be defined during the design stage and integration with external systems.

3 Ontology engineering process and related work

Methodologies for ontology engineering have been subject of research for a number of years. In general, ontology development depends on the context and the purpose of particular project. Therefore, each project team is very often trying to build its own methodology. The basis for the methodology development for INSIGMA system was the analysis of the most known existing approaches in that field.

There are many different ontology engineering methodology proposals. We analyzed typical methodologies used to build ontologies from scratch or by reusing other ontologies. In particular, the approaches dealt with were: On-ToKnowledge Methodology [2], Ontology development proposed by Noy and McGuinness [3], Enterprise Ontology [4], TOVE (TOronto Virtual Enterprise) [5], HCONE (Human Centered ONtology) [6], UPON (Unified Process for ON-tology building) [7], Holsapple [8], METHONTOLOGY [9].

After analyzing the above methodologies it was possible to distinguish some common activities (processes) for all methodologies which form an ontology life cycle:

– management process: consists of scheduling, control and quality assurance,
– ontology development process: consists of environment study, feasibility study specification, conceptualization, formalization, implementation, maintenance and use,
– support process: consists of activities run in parallel to the ontology development process. It includes knowledge acquisition, evaluation, integration, documentation, merging, configuration and alignment.

Each methodology defines its individual approach to carrying out the complete ontology life cycle. The above-mentioned groups of processes were incorporated in the proposed IOEM.

4 Requirements for INSIGMA ontology

Generally, it was assumed that the ontology for INSIGMA system should be explicit, coherent and extensible.

Being explicit means that an ontology should efficiently communicate intended meaning of defined terms. Definitions of terms should be objective and insusceptible to unintended interpretations.

Being coherent means that an ontology should allow for drawing meaningful inferences that are consistent with definitions and axioms.

Being extensible means that an ontology developer should take into account future extensions of created ontology without the need for revising definitions. New terms in ontology should be able to be defined based on the existing vocabulary. It should also be open to other existing ontologies. In this case it is important to specify the upper ontology that would enable definition of additional domain ontologies not colliding with the existing ones.

5 INSIGMA ontology engineering methodology

5.1 The approach

The IOEM is focused on the ontology application process. It is not intended to develop upper ontologies or specific ontologies used to reflect knowledge in particular domain, but ontologies used to realize the goal of the project.

The ontology engineering process presented in this chapter is in many cases similar to the object-oriented analysis and requirements management. It is a natural consequence of the fact, that for a given set of applications UML (Unified Modeling Language) [10] specifications and semantic models have similar role on software engineering process. Following steps of IOEM have been presented in the subsequent subchapters: Specification, Initial Conceptualization, Modularization, Module Conceptualization, Formalization, Deployment, Evaluation and Maintenance. These basic steps have been presented in Fig. 1.

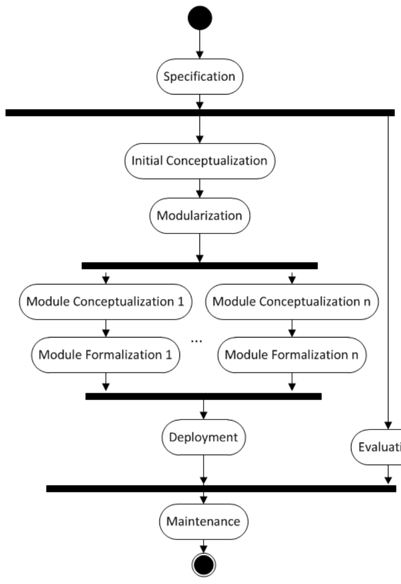

Fig. 1. IOEM stages.

5.2 Specification phase

The objective of the Specification phase is to determine the domain, scope of the ontology and aim of its application. The basis for this phase is the identification of the problem that the ontology application is to solve. This is very important in large scale systems since, at the beginning of the project, some of the developers do not know what is the final result they want to achieve and tend to support the idea of developing very broad and extensive ontology, which in turn may be inconvenient and rarely used. Therefore it is a good practice therefore to define a questionnaire for the developer teams consisting of a set of questions and exercises. This is to help to define a strong foundation for the ontology. An example questionnaire defined for INSIGMA consists of the following elements:

– *Ontology foundation*

    - What is the purpose of creating an ontology ?
    - What is the domain of the developed ontology?
    - For what types of questions should the ontology provide answers?
    - Who will be end-users of the ontology?
    - Who will be responsible for the ontology maintenance?
    - What are ontology use cases in the INSIGMA system?
- *Defining competency questions and motivating scenarios with use cases*
  Questions that an ontology should be able to answer are called competency questions. After the ontology is built, competency questions enable it to be verified in terms of its completion.
  Motivating scenario describes requirements that the created ontology should satisfy expressed as use cases. They provide examples for ontology application in the system.
- *Examples of existing ontologies application*
  Large number of ontologies have been developed and published. Using them (importing) into the INSIGMA ontology may accelerate the work, reduce costs and support interoperability. The potential candidates should be evaluated to check their validity and possibility for refinement and extension for particular domains and tasks.

The elements of the questionnaire are very important in the process of ontology development. For instance, competency questions determine the ontology application, which is crucial, e.g. for rule-based systems and expert systems focused on solving particular problems. Two ontologies describing the same domain of knowledge may give answers to completely different questions. This task should be therefore considered very important since it strongly influences the resulting ontology and can considerably scope it further usage. Reusing existing ontologies is also beneficial and should be taken into account. This supports the interoperability, enables reusing of the results of other projects and can decrease the amount of work for ontology development.

5.3 Initial Conceptualization phase

The goal of the Initial Conceptualization phase is to determine the scope of the developed ontology by establishing a glossary of key concepts, their hierarchy and relations. The result of this phase is usually an informal ontology description that may take: a narrative form, partial UML class diagrams or mind map diagrams. At this stage it is also possible to create a small monolithic ontology coded in a formal language, e.g OWL (Web Ontology Language) [11], covering only main concepts and relations from the modeled domain or providing a template that can be applied while building the whole model.

In particular, the elements of the domain model do not have to be anchored in an upper ontology, many concepts can be omitted or not linked by relations. An important activity related to the Initial Conceptualization is the assessment whether the scope of the model is sufficient to answer the competency questions and support the use cases realization. This activity is considered to be a part of the Evaluation process running in parallel with other processes.

At the end of this phase, the team responsible for the ontology development should attain a consensus concerning the ontology scope (detailing what should be included in the ontology) and guidelines how it should be expressed. The first issue is subject of interest of domain experts, while the second one of knowledge engineers who are responsible for ontology formalization carried out in further stages.

5.4 Modularization phase

Large ontologies are rarely constructed in a monolithic form, they are rather distributed into several modules (compound ontologies). Modularization is a natural method for management of complex models [12, 13]. Depending on a starting point, it is achieved by decomposition (top-down approach) or composition (bottom-up approach). Decomposition is more often used while describing a behavior, e.g. functional decomposition depicted by DFD (Data Flow Diagrams), whereas composition is usually used to describe structural relations, e.g. building a class from compound classes (attributes), organizing classes into packages, modules or libraries and using them from outside the organization unit.

In case of ontologies expressed in OWL language, a composition of ontologies is supported by the import mechanism. It allows to include external ontologies published in the web or stored in a local library into a master ontology and gain access to their content. Imported ontologies may contain a taxonomy of classes, a taxonomy of relations, axioms, individuals, rules or any coherent combinations of these elements.

The result of import is an import closure. The import closure for an ontology O is a set containing all entities of O and entities of all ontologies that are directly or indirectly imported by O. It should be noted however, that if two ontologies O1 and O2 are imported into a master ontology, they should not contain different versions of the same concepts or relations (it is forbidden to import different versions of the same ontology) or explicit indication, that they are incompatible (with use of the annotation owl:incompatibleWith referring to the URI of another ontology).

The goal of Modularization is:

- reuse of existing components,
- logical distribution of the ontology building process into a set of partially independent activities,
- giving an opportunity to parallel development of compound modules; this issue is particularly important for a team being distributed among several geographical locations, which is the case of the INSIGMA project,
- providing a possibility of scaling (several useful closures can be created by importing selected components, they may have different applications),
- testing of developed ontologies (e.g. by creating an ontology for testing purposes that imports entities from a developed ontology and adds several individuals to validate rules or to reason about relations).

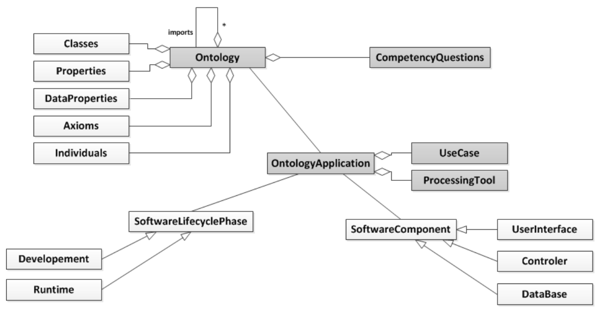

Fig. 2. INSIGMA Ontology and its applications.

Fig. 2 shows a metamodel specifying entities relevant to the Modularization phase. An ontology may contain classes (concepts), properties (relations), etc. An ontology may import other ontologies and may have some CompetencyQuestions specified. The class OntologyApplication describes a particular application of ontology important from the perspective of the system development or deployment. The application is related to a selected ontology (more precisely an import closure). For a set of ontologies, there may exist several applications.

An ontology application:

- is related to a software lifecycle phase (Development or Runtime),
- concerns a particular software component (a user interface, a data base, a controller module),
- may require an additional tool for ontology processing or deployment.

Formally, in IOEM methodology the elements of the metamodel are represented by an artifact: the Ontology Application Chart. The document taking a form of questionnaire is filled out by two independent teams: the one developing an ontology and the second deploying it. Ontology Application Chart contains the following elements:

- architecture specification: indication of the master and compound ontologies,
- specification of the ontology role in reference to the software lifecycle and the software architecture,
- refinement of use cases (it should be mentioned that for ontologies used during the run-time, their use cases correspond to the system use cases, whereas for ontologies used at the development phase, the use cases are related to supporting tools for building and processing ontologies),
- the status of the supporting tools (if applicable),
- methods of ontology storing,
- project roles responsible for ontology development, deployment and maintenance.

5.5 Module Conceptualization phase

Conceptualization is an activity that should be carried out before the Formalization process. It provides a domain ontology model in an informal language with terms and their relationships presented in ordered taxonomy. This process consists of the following actions: building a glossary of terms, building taxonomies of concepts, defining relations (between concepts and data types), building concept dictionary, defining axioms, defining rules, creating instances of particular classes.

To support the Conceptualization phase the authors proposed the second questionnaire, which helps to build conceptual dictionary and to define rules that describe the given domain.

5.6 Formalization phase

This phase transforms the conceptual model into the formal model, which enables to automatically process the knowledge and verify its consistency. The phase includes the building of a sematic model in an ontology language. It is important to choose an appropriate formal model for the purpose of the project. It needs to have a proper expressiveness for the purpose of inference and extensive software support at the stage of ontology edition, visualization, verification and further implementation in the system software components (in the Deployment phase).

In case of IOEM Formalization phase XML-based semantic languages, i.e.: RDF (Resource Description Framework) [14] and OWL will be used. They have appropriate expressiveness, are easy to use and are supported by available software development tools (e.g. Protégé, Jena library).

5.7 Deployment phase

According to the metamodel presented in Figure 2, an ontology application can occur in the phase of system development or during the run -time. An ontology used in the development phase can be treated as the system development artifact. In case of an ontology application during the run-time, it can be considered as a software component that should be subject of verification and testing.

Decision on using OWL for formal ontology specification to some extend limits platforms that can be used during deployment. Currently, the most advanced tools and libraries have been developed for the Java platform, e.g. Jena, OWL API, Jess (as a rule engine). Thus, components using ontologies in the INSIGMA system are usually coded in Java; if an integration with another platforms is needed, a wrapper in form of web services is used.

5.8 Evaluation phase

Evaluation is a process aimed at validating and verifying the ontology in terms of its scope, consistency and expressiveness. It relates the created ontology to the requirements defined in the Specification phase (possibility to answer competency questions, use cases coverage) but it lasts as long as the whole process of IOEM (see Fig.1). Evaluation relates therefore to all the products of subsequent phases and is carried out periodically even in parallel to the ontology lifecycle.

Even though the methodology consists of steps, it is always possible to take a step back and make corrections that will make the ontology tailored to the needs of the project. Moreover, it is assumed that creating the complete set of classes, their relations and axioms within one course of the Conceptualization and Formalization phases is impossible. It is necessary to run at least 2 iterations after which the resulting model is evaluated against semantics, syntactic, coherence, coverage and, what is crucial, adherence to the competency questions and use cases.

In terms of the Specification phase, evaluation covers the aim and scope of the ontology and use cases. Initial conceptualization is verified against specification, mostly on the basis of review of the informal descriptions. The architecture of the ontology set in the Modularization phase is evaluated against the possibility of multiple usage of the components, possibility to work in parallel on their development and identifying possible limitations on implementation.

Evaluation process related to the Deployment phase employs techniques not strictly related to ontologies but regular mechanisms of software validation and verification.

5.9 Maintenance phase

Maintenance is a process that corrects and updates the ontology. This phase is usually performed at the stage of ontology utilization in particular system components. Due to the fact, that the duration of the system development phase was established to 5 years, several software components will be successively created within this period. Particular domain ontologies will be though maintained by different task groups and in different timeframe. It is necessary for the ontology creator to actively supervise the ontology utilization and support the ontology maintenance. Although the Specification phase bases on the questionnaire that helps to develop the most appropriate assumptions, they may be not valid after the final system components are developed. The aim of the Maintenance phase is though to react on changes in system functionality, operation and implementation in order to provide the biggest benefit for the project.

In such a broad project as INSIGMA, particular domain ontologies are strongly related to their applications and deployment in particular components of the system. The changes included by a group of developers responsible by particular component must not negatively influence other modules and domain ontologies. It is very important though to assess the scope of ontology that supports only particular objective and parts of the ontology that have impact on many processes.

6 Summary

The paper presents IOEM, a methodology for ontology development elaborated for the INSIGMA project. Although prepared for a particular use, the methodology is quite general and can be used in a large variety of IT projects requiring ontology components. It is particularly suitable for large and geographically distributed software projects.

The methodology was proposed as a result of analyses of various approaches and known methodologies for building ontologies. The goal of IOEM methodology is to provide a defined process comprising phases, supporting tools and well determined results. The methodology implementation is a complex process including several activities that involve different experts and work groups. At present, a number of ontologies within the INSIGMA project are concurrently built. In general, their progress ranges from phase 1 to 4. So far, no significant corrections in the methodology was required. At the end of the project, the authors plan to assess the overall methodology and formulate lessons learned.

References

1. Guarino, N.: Formal Ontology and Information Systems. Proceedings of FOIS'98, Trento, Italy (1998)
2. Sure, Y., Studer, R.: On-To-Knowledge Methodology - Final Version. On-To-Knowledge EU IST-1999-10132 (2002)
3. Noy, N.F., McGuinness, D.L.: Ontology Development 101: A Guide to Creating Your First Ontology. Standford University (2001), `http://www.ksl.stanford.edu/people/dlm/papers/ontology101/ontology101-noy-mcguinness.html`
4. Uschold, M., King, M.: Towards a Methodology for Building Ontologies. Proceedings of IJCAI95s Workshop on Basic Ontological Issues in Knowledge Sharing (1995)
5. Gruninger, M., Fox, M.S.: Methodology for the Design and Evaluation of Ontologies. IJCAI'95, Workshop on Basic Ontological Issues in Knowledge Sharing (1995)
6. Kotis, K., Vouros, G.A.: Human Centered Ontology Management with HCONE. Proceedings of the IJCAI'03, Ontologies and Distributed Systems Workshop, CEUR-WS.org, vol. 71, (2003)
7. De Nicola, A., Missikoff, M., Navagli, R.: A proposal for Unified Process for Ontology building: UPON. Proceedings of Database and Expert Systems Applications, 16th International Conference, DEXA 2005, Copenhagen, Denmark (2005)
8. Holsapple, C.W., Joshi, K.: Handbook of Knowledge Management, A knowledge management Ontology, Chap. 6, ISBN 3-540-20005-3, Springer Verlag (2003)
9. Gómez-Pérez, A., Fernández-López, M., Juristo, N.: METHONTOLOGY: From Ontological Art Towards Ontological Engineering. Proceedings of Symposium on Ontological Engineering of AAAI, Spring Symposium Series, pp. 33-40, Stanford (1997)
10. Booch, G., Rumbaugh, J., Jacobson, I.: Unified Modeling Language User Guide, 2nd Edition. Addison-Wesley Professional (2005)
11. OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax, W3C Recommendation (2009), `http://www.w3.org/TR/owl2-syntax/#Imports`
12. IBM Certified Solution Designer - IBM Rational Unified Process V7.0, http://www-03.ibm.com/certify/certs/38008003.shtml
13. Leffingwell, D., Widrig, D.: Zarzadzanie wymaganiami. WNT Warszawa (2003)
14. Klyne, G., Carroll, J. J.: Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation (2004), `http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/`