gOntt, a Tool for Scheduling and Executing Ontology Development Projects

Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez, Oscar Muñoz, Martín Vigo
Ontology Engineering Group. Departamento de Inteligencia Artificial. Facultad de Informática. Universidad Politécnica de Madrid
{mcsuarez, asun, omunoz}@fi.upm.es, mvigo@delicias.dia.fi.upm.es

Abstract

Nowadays the ontology engineering field does not have any method that guides ontology practitioners when planning and scheduling their ontology development projects. The field also lacks the tools that help ontology practitioners to plan, schedule, and execute such projects. This paper tries to contribute to the solution of these problems by proposing the identification of two ontology life cycle models, the definition of the methodological basis for scheduling ontology projects, and a tool called gOntt that (1) supports the scheduling of ontology developments and (2) helps to execute such development projects.

1. Introduction

Planning and scheduling are related activities that are applied in different contexts, such as civil engineering, software engineering, etc. While planning[1] is the act of drawing up plans, a series of steps to be carried out to achieve an objective, scheduling[1] is defined as the activity to set order and time to planned events. Scheduling should be performed after planning; and both are crucial in any development project.

In Software Engineering, every development project has a life cycle [1], which is produced by instantiating a particular life cycle model. Life cycle models can be seen as abstractions of the phases through which a product passes along its life.

To properly manage software development projects, it is crucial to have knowledge of the entire software development life cycle [2]. Software engineers always plan and schedule every development project before starting it. The project plan devises the tasks to be done and the actors to perform them. To estimate the effort required to perform each task, techniques such as [2] PROBE and COCOMO II can be used.

The project schedule links the tasks to be done with the resources in order to support their performance. The most common form of representing schedules is to use a Gantt chart [2]; and the most popular tool for creating a project schedule is Microsoft Project [2]. To create a schedule, Microsoft Project provides three different ways: (a) from scratch; (b) from a project template (commercial template, report template, etc.) selected by the user using a template library; and (c) from an existing project. However, according to our knowledge, any tool for managing project schedules provides guidelines on how to execute the project.

Ontologies are used for making knowledge explicit and allowing it to be shared. One of the keys when building ontologies is to plan and schedule the ontology development. However, in Ontology Engineering, planning and scheduling are still in their early stages. Only METHONTOLOGY [3] defines the scheduling activity, but it does not provide guidelines for helping ontology developers to plan and schedule their projects. Other methodologies, such as On-To-Knowledge [4] and DILIGENT [5], do not include these activities in their developments. Regarding the calculation of cost estimation of projects, the only technique available is ONTOCOM [6], a model that predicts the costs of ontology development projects.

The ontology engineering field lacks methods for guiding ontology developers when planning and scheduling their ontology development projects. Furthermore, the template library of Microsoft Project does not have project templates oriented to ontology development projects; on the other hand, at present, no tool can provide ontology developers with ontology project schedules in the form of Gantt charts, nor can recommend developers which methodological guidelines and tools should be used for executing a process or an activity in the ontology development.

Thus, the innovation of this paper is that (1) it identifies two ontology life cycle models, (2) defines

[1] wordnet.princeton.edu/perl/webwn

the groundings for scheduling ontology projects, and (3) presents gOntt, a tool that supports the scheduling of ontology developments and their guided execution.

The paper is organized as follows: Section 2 presents the scheduling activity. Section 3 deals with ontology life cycle models. Section 4 summarizes the methodological groundings for the scheduling activity. Section 5 shows the main gOntt functionalities. Section 6 presents an evaluation of gOntt. Finally, Section 7 provides the conclusions.

2. Scheduling ontology developments

Scheduling [7] refers to the activity of identifying the different processes and activities to be performed during the ontology development as well as their arrangement, and the time and resources needed for their completion. An important task within this activity is the establishment of the *ontology network life cycle*, that is, the specific ordered sequence of processes and activities that ontology practitioners have to carry out during the life of the ontology network.

To establish the particular schedule for the ontology development, four questions have to be answered: (1) Which life cycle model is the most appropriate for the development?; (2) Which processes and activities should be carried out in the ontology development?; (3) Which order and dependencies exist among processes and activities?; and (4) How many resources are needed for the development of the ontology?

The first three questions are related to the establishment of the ontology life cycle, and their responses would provide a plan for the ontology development. The fourth question (out of the scope of this paper) is related to the inclusion of time and human resources restrictions, and its response would provide the concrete schedule for the ontology development. The information about how many people should be involved in the ontology development can be obtained using the ONTOCOM model [6].

3. Ontology network life cycle models

The *ontology network life cycle model* establishes in an abstract way how to develop an ontology network development project. The ontology network life cycle is created by mapping the processes and activities identified in the ontology development onto a selected ontology life cycle model. While the model refers to a general framework, the life cycle refers to the concrete sequence of processes and activities. In this regard, while METHONTOLOGY and DILIGENT propose a model based on evolutionary prototypes, On-To-Knowledge proposes an incremental and cyclic model.

Along this section we will try to show that there is not a unique life cycle model valid for all ontology development projects and that each life cycle model is appropriate for projects with different features.

For this reason, we propose two ontology network life cycle models; the *waterfall ontology network life cycle model* and the *iterative-incremental ontology network life cycle model*. These models were created by (1) adapting the life cycle models described in Software Engineering to the characteristics of the ontology development (that is, the acquisition of knowledge, the evaluation and assessment of the different phase outputs, project and configuration management and documentation should be performed in all of the phases); (2) reusing the ideas presented by Larman [8]; and (3) analyzing our experiences in different ontology development projects.

3.1. The waterfall life cycle model

The waterfall model represents the stages of the ontology development as a waterfall, where a concrete stage must be completed before the following stage begins and where backtracking is permitted from the maintenance phase to the phase that follows that of the requirements. Taking into account the importance of reusing and reengineering knowledge resources as well as ontology merging in the ontology network development, we define five different versions of the waterfall model (as presented in Figure 1).

a) 4-Phase Waterfall Model. It represents the stages of an ontology network, starting with the initiation phase and going through the design, the implementation and the maintenance.

b) 5-Phase Waterfall Model. It extends the 4-phase model with the reuse of existing ontological resources as they are.

c) 5-Phase + Merging Phase Waterfall Model. It is a special case of the 5-phase model. It includes the *Merging Phase* to obtain a new ontological resource from two or more ontological resources previously selected in the reuse phase.

d) 6-Phase Waterfall Model. It extends the 5-phase model with the *Reengineering Phase*. It allows the reengineering of knowledge resources (ontological and non-ontological).

e) 6-Phase + Merging Phase Waterfall Model. It extends the 6-phase model by including the *Merging Phase* after the reengineering of knowledge resources.

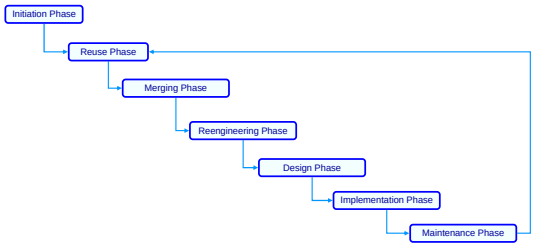

Figure 1. Waterfall model family

This model is recommended in projects that

- Have closed, non-ambiguous, unchangeable and completely known requirements at the beginning of the ontology development.
- Do not last long (e.g., 2 months).
- Re-implement an existing ontology or part of it in a different formalism or language.
- Transform a particular knowledge resource (e.g., ISO standards or thesauri) into an ontology.
- Cover a small and well-understood domain.

3.2. Iterative-incremental life cycle model

This model organizes the ontology development in a set of iterations (or short mini-projects with a fixed duration). Any iteration is scheduled as an ontology development project that uses one of the waterfall model versions shown in Section 3.1.

The model proposes the successive improvement and extension of the ontology by means of performing multiple iterations with feedback and adaptation. Thus, the ontology grows incrementally along the development. In each iteration, new or modified requirements are allowed for. The number of iterations will depend on the knowledge we may have of the requirements at the beginning of the project. The result of any iteration in this model is an ontology that meets the requirements identified in the iteration.

It should be noted that when using this model, no backtracking is allowed between the phases of a particular iteration because the refinement should be performed in the next iteration, and that in the initiation phase of each iteration revisions of ontology requirements and schedule should be carried out.

This model is recommended in ontology projects

- With large groups of developers in which complex scenarios are considered, such as reengineering non-ontological resources or aligning ontological resources.
- In which requirements are not completely known at the beginning or can change during the ontology development.
- In which requirements have different priorities.

4. Methodological groundings for scheduling ontology development projects

We propose to carry out the scheduling of ontology development projects based mainly on the scenarios identified in the NeOn Methodology [9]. For this reason and to be able to answer the first three questions presented in Section 2, we did some research that yielded the following results: (1) a set of questions that help to select the most appropriate version of the waterfall life cycle model; (2) the correspondences between the phases of the life cycle model and the processes and activities; and (3) the order and dependencies between processes and activities.

4.1. The most appropriate model version

To select a particular waterfall model version from those presented in Section 3.1, we propose the set of natural language questions displayed in Table 1. These questions are related to the different scenarios identified in the NeOn Methodology [9]. If one or more questions of those proposed are answered affirmatively, then several candidate models could be used. In that case, the model version selected should be the most specific one (e.g., 5-phase is more specific that 4-phase, 6-phase + merging phase is more specific than 6-phase). Otherwise, if all answers are negative, then the 4-phase waterfall model is selected by default.

Table 1. Questions and model versions

Will you use any non-ontological resource (NOR) in your ontology development?	6-Phase
Will you use any ontological resource in your ontology development?	5-Phase
Will you use and modify any ontological resource in your ontology development?	6-Phase
Will you use and merge a set of ontological resources in your ontology development?	5-Phase + Merging Phase
Will you use, merge, and modify a set of ontological resources in your ontology development?	6-Phase + Merging Phase
Will you use ontology design patterns in your ontology development?	5-Phase
Will you restructure your ontology?	4-Phase
Will you develop your ontology in different natural languages?	4-Phase

4.2. Model phases and processes and activities

Processes and activities should be carried out in a particular phase of the selected ontology network life cycle model to fulfill the purpose and outcome of that phase. Table 2 presents an excerpt of the matching between model phases and process and activities. The processes and activities are defined in the NeOn Glossary [7] whereas the phases are defined in our repository of models and these are the initiation phase, the reuse phase, the merging phase, the reengineering phase, the design phase, the implementation phase, and the maintenance phase.

Table 2. Model phases, processes and activities

Initiation Phase	O. Requirements Specification Scheduling O. Evaluation	
Reuse Phase	NOR Reuse O. Search O. Reuse	O. Statements Reuse O. Evaluation
Merging Phase	O. Aligning O. Evaluation	
Reengineering Phase	NOR Reengineering O. Modularization	O. Evaluation
Design Phase	O. Conceptualization O. Evolution	O. Localization O. Evaluation
Implementation Phase	O. Evaluation	
Maintenance Phase	O. Evaluation	

4.3. Dependencies of processes and activities

The preliminary order in which processes and activities should be performed is determined by the three following major factors:

1) The selected ontology network life cycle model dictates an initial ordering of processes and activities, based on the order in which model phases should be performed.
2) The availability of output information from one process or activity could affect the start of another process or activity. For example, the ontology requirements specification activity should be performed before the scheduling one.
3) Processes and activities might be executed in a parallel way. For example, the ontological resource reuse could be performed in parallel with the non-ontological resource reuse rather than in serial execution if there are enough developers to carry out the reuse in a parallel fashion.

These dependencies have been represented in scheduling templates in the form of Gantt charts. The scheduling templates show ontology project default plans based on the different and possible combinations among life cycle models, scenarios [9], and processes and activities. We have identified and represented 112 templates that can be used as preliminary schedules. Figure 2 shows one of the scheduling templates. This template is used when the model is the 6-phase waterfall and the reuse of non-ontological resources, ontological resources, and ontology design patterns, as well as the localization of the ontology are considered.

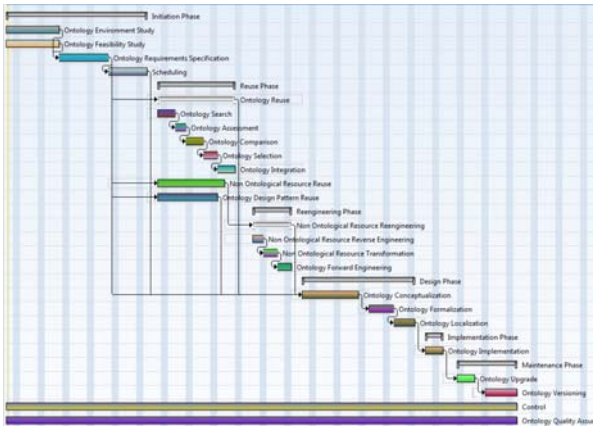

Figure 2. Example of a scheduling template

5. gOntt description

gOntt is a tool for scheduling and executing development projects. It is implemented as a NeOn Toolkit[2] plug-in with the following main features:

(a) It uses templates oriented to schedule ontology developments.

(b) It generates the scheduling of ontology developments in the form of Gantt charts following the basis presented in Section 4.

(c) It informs ontology developers about how to carry out a process or an activity using prescriptive methodological guidelines. It also informs about the specific NeOn Toolkit plug-ins to be used.

We describe gOntt functionalities for scheduling ontology projects and for helping in their executions.

5.1. Scheduling an ontology development

gOntt provides support to ontology developers so that they can (a) decide which ontology life cycle model is the most appropriate for building their ontologies (waterfall or iterative-incremental) and which processes and activities should be carried out and in which order (e.g., specifying ontology requirements before reengineering a knowledge resource into an ontology), and (b) create a graphical representation in the form of a Gantt chart with the processes and activities needed, including time restrictions between them. Schedules for ontology development projects can be created either *from scratch* or *in a guided way*.

From scratch: gOntt allows the developer to include processes, activities, phases, as well as restrictions among them, according to his needs.

[2] http://www.neon-toolkit.org/

In the guided way: gOntt creates a preliminary plan for the ontology development by means of (1) templates that schedule ontology projects and (2) a simple wizard that contains intuitive questions implicitly allowing the ontology developer to select the ontology life cycle model and the processes and activities needed in his development. To answer such questions the ontology developer should take into account the ontology requirements and the type of candidate knowledge resources to be reused.

gOntt uses internally the methodological foundations explained in Section 4.

gOntt main output is the initial plan for building the ontology in the form of a Gantt chart, which the developer can modify later on (a) by including, modifying, or deleting processes, activities and phases; (b) by changing order and dependencies among processes and activities; and (c) by including resource assignments and restrictions to the plan. This functionality to generate preliminary plans provides a great advantage over the tools that schedule projects.

5.2. Helping to execute an ontology project

With the aim of helping ontology developers to carry out a particular process or activity, gOntt provides ***prescriptive methodological guidelines*** by means of (1) a **filling card** that includes the process or activity definition, its goal, the inputs and outputs, the performer of the action, and the time required, and (2) a **workflow** that describes how the process or the activity should be performed with its inputs, outputs, tasks and actors involved. Figure 3 presents the methodological guidelines for the ontology requirements specification activity.

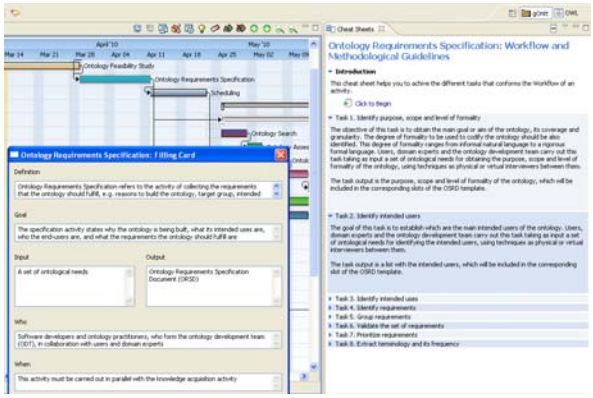

Figure 3. Example of methodological guidelines

In addition, gOntt provides a ***direct and automatic access to NeOn Toolkit plug-ins***[3] associated to each process and activity. Besides, gOntt displays a quick-start guide for using the plug-in launched.

6. Evaluation of gOntt

We conducted an experiment using gOntt in two settings that involved Master and PhD students. Both groups of students had chosen the 'Ontology Engineering and Semantic Web' subject. The students executed the experiment during the period of October-November 2009. The students (working in pairs) had to develop an ontology network in a free domain. They were provided with (1) a set of mandatory processes and activities (e.g., to specify requirements and to reuse different types of knowledge resources) and (2) a set of optional processes and activities (e.g., localization, evaluation, and modularization) to be carried out in the ontology development project.

The main goal of the experiment was to test that the students (1) scheduled their ontology development project with gOntt, (2) performed the ontology development project following the schedule created with gOntt and using the methodological guidelines provided by gOntt, and (3) used the necessary NeOn Toolkit plug-ins for every process and activity scheduled. The plug-ins could be triggered by gOntt in the right moment of the schedule.

At the end of the experiment, the students had to fill a questionnaire. Such a questionnaire was divided in two main parts: (1) one related to background knowledge, and (2) another related to the experiences gained by using gOntt. The second part included 10 questions with fixed answers and an open question in which they were asked to express other comments.

In order to analyze the students' perception of gOntt, we selected three questions from the questionnaire. The answers are displayed in Table 3.

Table 3. Answers to the questions selected

Question	Distribution of answers		
	Yes	No	Other
Is gOntt useful for scheduling ontology development projects?	14	1	1

Question	Distribution of answers			
	A useful idea?	A nice utility?	A useless idea?	I don't know
Is the idea of having methodological guidelines in gOntt:	10	6	0	0
Is the idea of triggering NeOn Toolkit plug-ins from gOntt:	6	4	0	6

[3] http://neon-toolkit.org/wiki/Neon_Plugins

Analyzing the answers in Table 3, we can say that

- The students perceived gOntt as a useful scheduling tool.
- The students in general perceived as a positive and useful issue the fact that gOntt provided them with methodological guidelines for each process and activity involved in the ontology development.
- Regarding the functionality of triggering NeOn Toolkit plug-ins, their opinion was divided, but in general, they tended to consider it as a good idea.

Additionally, the students provided general comments that can be summarized as follows:

- gOntt is considered a good scheduling tool that is based on activities and processes of the ontology engineering field.
- gOntt is considered a good help in providing a preliminary schedule for a development project.
- gOntt is considered one of the easiest tool for creating Gantt charts, thanks to its simple wizard.
- gOntt is seen as a centralized environment that allows the user to obtain methodological and technological guides without searching in external locations.

The students also commented that the tool established at random the length of the processes and activities planned and not according to previous experiences. They also missed the functionality of exporting the schedule to Microsoft Project.

7. Conclusions and future work

This paper outlines the methodological basis for scheduling ontology development projects and explains the technological infrastructure, gOntt, which supports the automatic generation of the initial ontology development plan using a Gantt chart.

gOntt, which is integrated within the NeOn Toolkit, is the first tool that has been created for systematizing the planning and scheduling of ontology projects. It uses templates and decision trees to create initial ontology development project schedules in the form of Gantt charts. In addition, gOntt is the first meta-tool that helps ontology developers to execute ontology development projects. It includes prescriptive methodological guidelines and informs about the recommended tools to use when performing a process or activity in an ontology development.

In short, the most important difference between gOntt and Microsoft Project is that while in gOntt the system generates automatically the initial ontology development plan, in Microsoft Project the user selects a particular project template from the library. Furthermore, gOntt is the only planning tool for creating and managing project schedules that provides developers with (a) the guidelines to how to execute the project and (b) the recommended tools to be used during the project execution.

The work described in this paper has important implications for our further research for two mains reasons. The first one, as soon as the tool is used by the community, we will conduct additional experiments. The second, we plan (a) to integrate gOntt with the works carried out by the ONTOCOM team for predicting the total costs of the ontology development project, and (b) to propose how long processes and activities should take by considering past and present experiences within ontology development.

Acknowledgments. This work has been partially supported by the projects NeOn (FP6-027595) and *BuscaMedia* (CENIT 2009-1026).

8. References

[1] Taylor, J.C.: Project Scheduling and Cost Control: Planning, Monitoring and Controlling the Baseline. J. Ross Publishing, 2008, ISBN: 9781932159110

[2] Stellman, A., Greene, J.: Applied Software Project Management. 2005. ISBN: 0-596-00948-8

[3] Gómez-Pérez, A., Fernández-López, M., Corcho, O.: Ontological Engineering. Springer Verlag. Advanced Information and Knowledge Processing series. ISBN 1-85233-551-3. November 2003

[4] Staab, S., Hans, P., Studer, R., Sure, Y.: Knowledge Processes and Ontologies. IEEE Intelligent Systems 16(1): 26–34. (2001)

[5] Pinto, H.S., Tempich, C., Staab, S.: DILIGENT: Towards a fine-grained methodology for DIstributed, Loosely-controlled and evolvInG Engineering of oNTologies. 16th European Conference on Artificial Intelligence (ECAI 2004), pp. 393-397

[6] Simperl, E., Popov, I., Bürger, T.: ONTOCOM Revisited: Towards Accurate Cost Predictions for Ontology Development Projects. In: Proceedings of the European Semantic Web Conference 2009 (ESWC '09)

[7] Suárez-Figueroa, M.C., Gómez-Pérez, A.: First Attempt towards a Standard Glossary of Ontology Engineering Terminology. 8th International Conference on Terminology and KE (TKE 2008). ISBN: 87-91242-50-9. Pp 1-15. Copenhagen, Denmark. August 2008

[8] Larman, C.: Applying UML and patterns: an introduction to object-oriented analysis and design and the unified process. Second edition. Pearson Education, Inc. 2002. ISBN: 0-13-092569-1

[9] Suárez-Figueroa, M.C., Gómez-Pérez, A.: NeOn Methodology for Building Ontology Networks: a Scenario-based Methodology. In Proceedings of the International Conference on Software, Services and Semantic Technologies (S3T 2009). ISBN: 978-954-9526-62-2. Pp: 160-167. Sofia, Bulgary. October 2009