TESMA: Towards the Development of a Tool for Specification, Management and Assessment of Teaching Programs

Nicolas Guelfi
University of Luxembourg
6 rue Coudenhove Kalergi
Luxembourg
+352.46 66 44 5251
nicolas.guelfi@uni.lu

Benjamin Jahić
University of Luxembourg
6 rue Coudenhove Kalergi
Luxembourg
+352. 691 30 74 38
benjamin.jahic.001@student.uni.lu

Benoît Ries
University of Luxembourg
6 rue Coudenhove Kalergi
Luxembourg
+352.46 66 44 5267
benoit.ries@uni.lu

ABSTRACT
Defining and managing teaching programs at university or other institutions is a complex task for which there is not much support in terms of methods and tools. This task becomes even more critical when the time comes to obtain certifications w.r.t. official standards. In this paper, we present an on-going project called TESMA whose objective is to provide an open-source tool dedicated to the specification and management (including certification) of teaching programs. This tool has been engineered using a development method called Messir for its requirements elicitations and introduces a domain-specific language dedicated to the teaching domain. This paper presents the current status of this project and the future activities planned.

Categories and Subject Descriptors
K.3.2 [**Computers and Education**]: Computer and Information Science Education – *accreditation, curriculum, self-assessment*.

General Terms
Documentation, Design, Languages.

Keywords
Teaching Programs Development, Tool-support, Software Engineering, Domain-Specific Languages, Automatic Generation.

INTRODUCTION
The University of Luxembourg is a young university (created in 2003). In this "start-up" context, we have been setting up new programs at bachelor, master and doctorate levels providing different education certificates. All those programs are offered to our students by three faculties. The need for a tool to support the task to define and manage (including certification) the education program came rapidly. The market analysis for this category of tools showed that no tool was available. A project has been started to engineer a method and a tool to support those needs. This project has been conducted following a software engineering process that had the following main steps:

- Requirements analysis: to provide the initial requirements for the TESMA tool, a requirement specification document has been produced using the Messir method [1].
- Design: to state the main choices concerning the TESMA architecture and interfaces.
- Implementation: to reach an operational system usable for validation w.r.t. the requirements.
- Those steps have been performed iteratively to produce the TESMA tool in an incremental way.

The content of this paper provides details on the requirements analysis, design and implementation of the TESMA tools. It also ends with a short related work section which summarizes what we have found when we did our market analysis.

1. TESMA REQUIREMENTS
Following part of the Messir method [1], we have elicited the actors that are concerned by teaching programs. They are:

- *The institution director* who represents the institution and validates new programs, courses and course modifications (e.g. the dean of a faculty, the head of a teaching unit, ...).
- *The program director* who specifies his programs and validates course modifications made by instructors.
- *The instructor,* who specifies, manages and maintains the courses he gives.
- *The student,* who tunes his curriculum (elective courses, ...) and receives information about his curriculum.
- *The secretary*, who is a delegate of any of the institutional actors (*institution director, program director or instructor)* and also ensures interoperability with other institution information systems.
- *The quality officer* who evaluates and validates the programs with respect to the universities internal laws. He's also responsible of program certification processes.

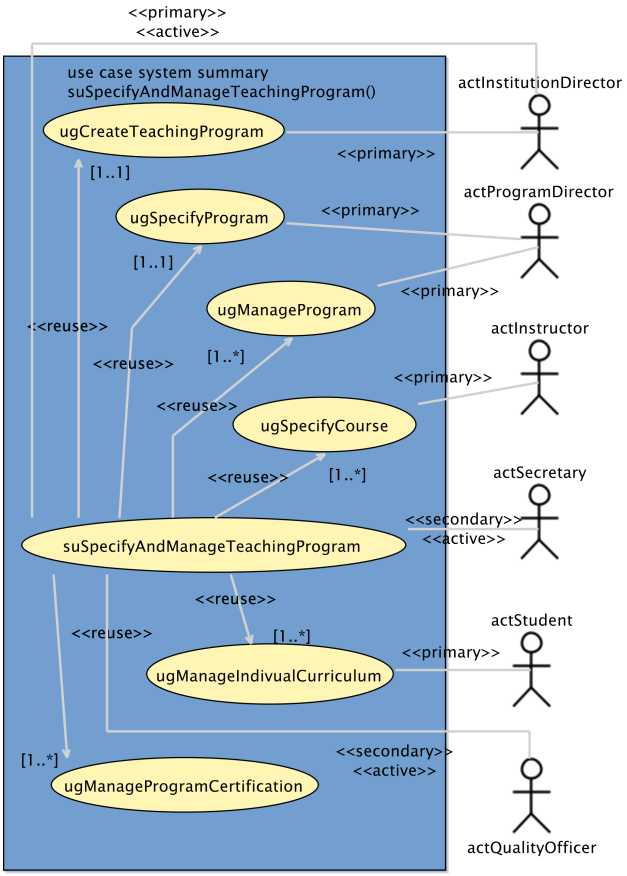

Figure 1. TESMA summary use-case.

You can find above a use-case model made in the context of the Messir method that displays the actors contributing to the high-level summary use-case dedicated to managing a teaching program.

The concepts managed by the TESMA actors are analysed and specified in the Messir concept model which is a UML class diagram. Among all the concepts that are necessary to specify the operations executed by the actors we have:

- concepts related to the actors and for which TESMA has to handle an internal representation: students, instructors, ...
- concepts related to the programs: course details, teaching periods, course evaluation, ...
- concepts related to program certification: standard description, standard coverage by an existing program, ...

In Messir those concepts are specified using an UML class diagram. Figure 2 provides a partial diagram view that models the concepts related to a program (institution, program, courses).

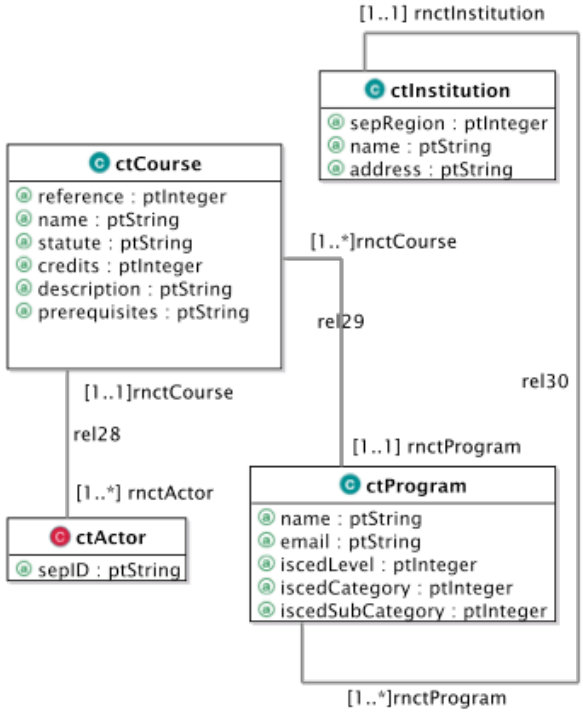

Figure 2. TESMA concepts.

The requirements analysis phase has allowed to determine a first version of the functionalities and data that should be handled in the first increment. The next section presents the design and implementation of this first version.

2. TESMA DESIGN AND IMPLEMENTATION

After having analysed the TESMA actors, concepts and functionalities. We have started to design a first version of the tool. A major design choice made is to allow the specification of programs using a domain-specific language (DSL) defined using the Xtext [2] framework. Thus, we designed TESMA as a plugin to the Eclipse workbench [3]. It is composed of three main architectural components illustrated in Figure 3 at the top of the diagram. Our components are based on stable Eclipse plugins themselves based on the Eclipse Modelling Framework (EMF) [4].

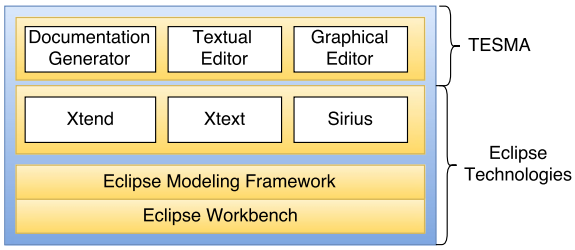

Figure 3. Architectural components overview.

2.1 Textual Editor

The main feature of the *Textual Editor* is to allow the specification of the teaching programs (this specification is called TESMA model in the remaining part of this paper) with the TESMA DSL. It also offers other supporting features, as for instance: syntactical validation rules, syntax highlighting, templates proposal, etc.

Xtext is an open-source framework that eases the development of domain-specific languages and offers features to provide a textual editor to the TESMA DSL. Xtext is based on EMF, which is the underlying-core library handling the TESMA model.

The TESMA DSL is designed to be intuitive, customizable and loosely coupled. In order to have an intuitive DSL, we have chosen to design its grammar using mainly keywords in natural language. Institutions may use different terminologies for the concepts used in our approach, this is why we designed the grammar of our DSL to be customizable. The institutions have the possibility to choose their own naming conventions. Lastly, the rules of the grammar are loosely coupled, i.e. optional cross-references are mostly used instead of containment relations.

2.2 Graphical Editor

The Graphical Editor provides a representation of the TESMA Model in a tabular view and offers the possibility to modify the TESMA model. The graphical editor provides typical table handling features like data sort, import/export from/to Excel sheets, hide/show columns, multiple rows selections.

The technology used to develop our graphical editor is Sirius [5], an open-source software Eclipse project that eases the creation of custom graphical modelling workbenches. Both Xtext and Sirius are based on EMF, which allows the TESMA tool-support to interact between Xtext and Sirius using EMF as underlying-core library for the TESMA model as represented in Figure 4.

Thanks to our tabular format, the graphical editor is intuitive and usable by non-computer experts. All the program's attributes are easy to access and modify. The modifications can be performed directly inside the graphical editor view.

2.3 Documentation Generator

The main feature of the *Documentation Generator* is to generate documents of different types, like Excel sheets, CSV files and PDF files. The *Documentation Generator* may be configured to produce a customized PDF file, e.g. by not generating some of the sections inside the pdf files.

The technologies used to develop the documentation generator are Xtend [2], Latex and the apache.poi library, for handling Excel sheets. Xtend is a programming language based on Java. It provides a compact syntax and eases the generation of natural language text. Latex is a document preparation system, which uses libraries, keywords and plaintext for writing scientific documents in pdf format. Finally, the apache.poi library provides the necessary tools for generating Excel sheets, which are used as teaching material.

The Documentation Generator has been designed to ease information retrieval in the generated Latex files. Additionally, it is designed to automatically update the final report, when the user manually adds data into the reserved appropriate folders. Finally, the different Latex files are imported inside one Latex file, which is compiled into a pdf file containing the program description.

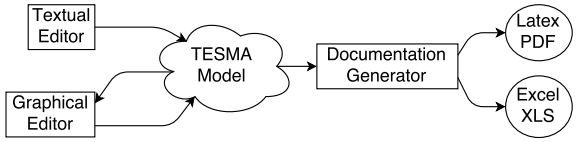

Figure 4. TESMA process overview.

3. ILLUSTRATION

We illustrate the TESMA approach with a course of a Master program named "Software Engineering Environment" (SEE) at the University of Luxembourg. Figure 5 is a screenshot of the TESMA tool-support in the Eclipse environment.

The TESMA model describing the SEE course have been specified using the approach described in this paper. The teaching program description of this example includes a number of textual files using the TESMA DSL syntax. The course run information is illustrated in Figure 5 by specifying the course teaching team organisation and dividing the teaching term into small periods and defining the tasks, tests for each period. The tasks and tests are defined in separate folders and referenced to the teaching period. At that point, TESMA is able to generate *a part of the Teaching Material*, like evaluation criteria, task lists and course information.

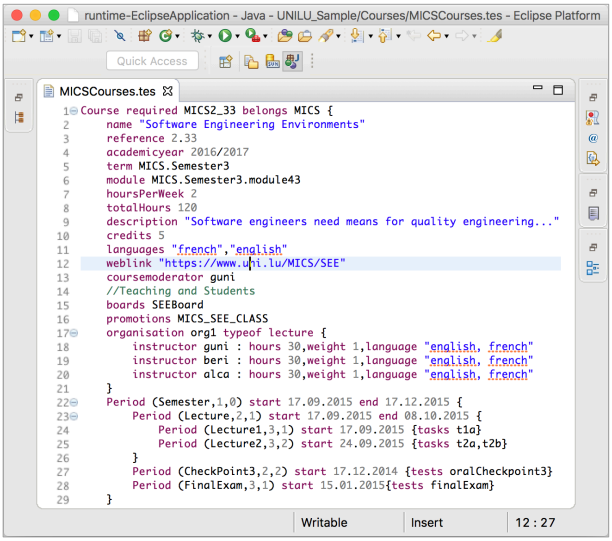

Figure 5. SEE specification in TESMA tool.

All other concepts can be specified using the TESMA DSL including the coverage of an education standard by an education program.

In this case study for the SEE course, we created for each category of TESMA model element a file containing all information related. In this case, one institution, one program and one course have specified, which represent about 10 textual files. We defined 10 instructors and 7 students for this example case, which are grouped in a single file. The total description in our case needs about 500 lines of specification text (>1000 in

case of certification). The specification text size mostly depends on the preciseness of the specifier. If the specification is done in details, the number of lines increases quickly. In general, it could vary from 100 to 1500 lines.

4. RELATED WORK

A number of related works have been performed in the past in the field of education programs and course specifications, especially for K-12 classes in high schools. However, we could not find methods, which are supported by tools, which help to design detailed teaching programs. On the one hand, some university guidelines are available online, e.g. [6,7], without offering supporting software applications. On the other hand, a few software applications are available, that do not offer comprehensive and customizable guidelines. We present in the following, three of these tools.

PDF Syllabus builder [8] is an open-source tool, which only offers a template PDF form for writing course syllabi. TESMA covers many additional features, for instance it generates automatically reports in PDF format. The design of the reports is standardized and generated by the TESMA tool.

Jump-Rope [9] is a proprietary tool, based on a web application. It supports a curriculum design tool, standards based gradebook, accurate attendance, administrator tools. TESMA has similar features and is more flexible in terms of customization of its textual language. Moreover, the use of MDE techniques allows the automatic reconfiguration of the user interface and the generated documents.

Build-Your-Own-Curriculum [10] is a proprietary tool, based on web-application developed for K-12 classes in the United States. It supports the feature of defining standardized courses, classroom managements, evaluations and assignments. TESMA is similar to this kind of tools, but has two major advantages: firstly, it is based on MDE technique, which allows customizing the documentation generation process in a flexible way. Secondly TESMA is an open-source project, which allows adapting freely its code to the institution's taste.

5. CONCLUSION AND FUTURE WORK

In this paper, we have shown how our project of engineering a tool for educational program and course specification based on a textual domain-specific language with a graphical editor has been developed. It generates documents out of the specification, which can be used by the institution's staff. Our tool and method has been successfully used on a small, yet real, example. As a future work we plan to iterate the process to stabilize the requirements and the tool design and implementation. We also plan to study the automated generation of a web application from the language grammar in order to provide a user-friendly front-end that is mapped to our textual language grammar.

6. REFERENCES

[1] Guelfi, N. 2016. *The Messir Scientific Approach to Requirements Engineering*. Laboratory for Advanced Software Systems Technical Report, TR-LASSY-16-01. University of Luxembourg.

[2] Bettini, L. 2013. *Implementing Domain-Specific Languages with Xtext and Xtend.* Packt Publishing Ltd.,Birmingham, UK.

[3] Beck, K., Gamma, E. 2003. *Contributing to eclipse – Principles, Patterns, and Plugins*. Addison-Wesley Professional.

[4] Steinberg, D., Budinsky, F., Paternostro, M., Merks, E. 2008. *EMF: Eclipse Modeling Framework, 2nd Edition*. Addison-Wesley Professional.

[5] Viyović, V., Maksimović M. and Perisić B. 2014. Sirius: A rapid development of DSM graphical editor. In *Proceedings of the IEEE 18th International Conference on Intelligent Engineering Systems* (Tihany, Hungary, July 3-5, 2014). INES 2014. IEEE, 233-238. DOI= http://dx.doi.org/10.1109/INES.2014.6909375.

[6] University of Washington. Course and Syllabus Design. Online, retrieved on July 2016. http://www.washington.edu/teaching/teaching-resources/preparing-to-teach/designing-your-course-and-syllabus/.

[7] Cornell University Center for Teaching Excellence. *Writing a Syllabus*. Online, retrieved on July 2016. http://www.cte.cornell.edu/teaching-ideas/designing-your-course/writing-a-syllabus.html.

[8] Joeckel, G., Longhurst, M. *PDF syllabus builder: open source tool for online instructors, course developers and instructional designers*. Online, retrieved on July 2016. http://olc.onlinelearningconsortium.org/effective_practices/pdf-syllabus-builder-open-source-tool-online-instructors-course-developers-and-i.

[9] Olsen, J., Meyer, J. *Curriculum Design Tool. Jump Rope*. Online, retrieved on July 2016. https://www.jumpro.pe/features/curriculum-design-tool/.

[10] School Software Group. *Build Your Own Curriculum (BYOC): Evaluating a K–12 online curriculum management system*. ProQuest. Online, retrieved July 2016. http://www.schoolsoftwaregroup.com/.