Learning systems development using reusable standard-based requirements catalogs

Ambrosio Toval, Juan M. Carrillo-de-Gea, Joaquin Nicolás, Jose L. Fernández-Alemán
Dept. Informática y Sistemas
Universidad de Murcia
Murcia, Spain
atoval, jmedg1, jnr, aleman @um.es

Rosa Toval
Facultad de Educación, Primary Teaching Postgraduate
Universidad de Murcia
Murcia, Spain
rous_11 @hotmail.com

Abstract— Requirements engineering (RE) is a discipline of critical importance in software development. This paper provides a process and a set of software artifacts to help in the production of e-learning systems with emphasis on reuse, standards and globalization issues.

Keywords- Global Development; Internationalization and localization; Learning Systems Development; Requirements Engineering; Reusability; Standardization.

I. INTRODUCTION

System and software requirements documents play a crucial role in software engineering (SE) in that they must both communicate requirements to clients in an understandable manner and define requirements in precise detail for system developers. A learning environment can be seen as a complex socio-technical system with many different stakeholders involved [1] e.g. software developers, learners, content providers, subject tutors, authors and accreditors. Besides common requirements concerning any kind of software (basic functionality, quality or security) there exist specific Learning Technology (LT) domain requirements (constraints imposed by learning standards, expected or desired specific functionality for a particular product). Following the ITEA recommendations [2] a systematic and rigorous requirements engineering (RE) approach can contribute importantly to obtain interoperable, high-quality learning systems in a productive way.

Using standards encourages interoperability (common terminology, concepts and procedures) but when these standards are used in a software development context, it is very useful –often necessary- to adapt, refine and express their contents in the form of explicit software and system requirements [3] [4] [5].

In addition, globalization imposes new challenges to the development and use of LT that must be also taken into account. Thus, main contributions of our proposal are:

• Definition of an infrastructure for a reusable requirements repository in the LT realm, integrating common requirements imposed by software engineering standards (e.g. Quality Software Product

This research is part of the projects PEGASO-PANGEA (TIN2009-13718-C02-02), financed by the Spanish Ministry of Science and Innovation (Spain) and MELISA- GREIS (PAC08-0142-335), financed by the Regional Science and Technology Ministry of Castilla-La Mancha (Spain).

II. REQUIREMENTS REUSE AND E-LEARNING STANDARDS

A. SIREN, the current model for requirements reuse

In order to take advantage of the benefits of reuse at the reqs (requirements) level, our group proposed the SIREN method [5] [6]. This is a practical approach for creating, selecting and specifying the reqs of a software system based on reuse and software engineering standards. SIREN encompasses a spiral process model, reqs documents templates, a reusable reqs repository which is organized by catalogs and a

ISO/IEC 9126 or SQuaRe ISO/IEC 25000), Learning Technology standards (provided by international standardisation bodies and consortia active in e-learning) and new product requirements.

• Identification and management of globalization issues involving the process (risks and safeguards repository in Global Software Development -GSD) and the product (LT localization and internationalization reqs).

• Using the results above, definition of an RE process specific for LT development.

The repository of reusable requirements catalogs provides the formal starting point for the subsequent software development, and supports the definition of LT software product lines (SPL). This approach is based upon previous work of our research group regarding a broad approach to requirements reuse, named SIREN (Simple REuse of RequiremeNts) [5] [6]. The contributions mentioned in the bullets above are new results w.r.t. that previous work, provided as an outcome of the needs identified by our group in several experiences [7] [8], in particular the application of an ordinary RE process in the development of educational software and e-book editor (MTO®, Multimedia Time Organizer, a commercial e-learning platform), performed under contract with our research group, and the definition of SPLs in other scope [9].

After this introduction, section II briefly introduces SIREN and the use of formal sources of information; section III presents the infrastructure proposed (artifacts and process) and section IV discusses the globalization issues. Finally, section V summarizes conclusions and further work.

supporting CARE tool (SirenTool). These catalogs are organized in a hierarchy of reqs specification documents, which are structured according to IEEE standards [3] [4]. The textual information of SIREN reqs is complemented by a set of attributes. There is a set of attributes common to all reqs (including priority, rationale, source, state, etc.), and additional attributes can be defined. Besides attributes, different traceability relationships can be defined to relate reqs: to sum up, these are inclusive, exclusive and parent-child traceability relationships. SIREN also deals with variability, e.g. through parameterized reqs, which contain some parts that have to be customized and that have to be instantiated when reused, and generic reqs, used as reqs patterns.

Our previous papers on SIREN proposed this method as a broad, but co-localized, approach to reqs reuse. In this paper, critical reqs regarding globalization are considered both in the product and the process (see section IV).

B. E-learning and general purpose software standards

There are several international organizations working on the standardization of e-learning technologies: ADL, AICC, CEN, DCMI, IMS Global Learning Consortium, ISO/IEC JTC1 SC36 and LTSC from the IEEE, among others. Each one address different issues related with learning technologies standards and specifications [10] such as: architectures, interfaces, content aggregation, metadata, runtime, assessment, accessibility, competency definitions, learner information, localization and internationalization, etc. The main problem in e-learning is not so much the identification of suitable standards and specifications as their adoption and effective application in e-learning practice [11].

An application profile is a refinement or implementation reference of an existing specification to make it more suitable for its application by a particular community of practice. For instance Shareable Content Object Reference Model (SCORM) is a well-known application profile based on a combination of the most relevant e-learning specifications and standards (IEEE, IMS, AICC, and so on) for sharable learning object packaging, delivering and sequencing. SCORM is the result of the ADL Initiative, established in 1997 by the U.S. Department of Defense. SCORM provides a runtime environment, a content aggregation model, a metadata model and a sequencing and navigation model to facilitate the interoperability among different e-learning systems. Common Cartridge (CC) is proposed as an enhancement of SCORM offering more flexibility [12]. System developers should consider using [24]: (1) a comprehensive application profile such as SCORM or CC, (2) a standard specification such as IMS CP (see table I), or (3) simpler conventions that are more uniquely suited to the particular needs of an application.

There is also a plethora of general purpose software standards, proposed by international standardization bodies such as CEN, UNE, ISO/IEC, IEEE, etc. Regarding, software quality, we can quote e.g. ISO/IEC 25012, ISO/IEC 9126 or its evolution and integration to SQuaRE (see table I).

Other formal documents, for example methods, guidelines, regulations or legislation on a topic of interest for the LT systems developer, could be also used as a source of useful, reusable reqs. For example: federal laws (e.g. [14], requiring federal agencies to provide access to electronic information to disabled individuals); international, European or national directives (e.g. [15] [16], regarding personal data and privacy in the electronic communications industry). The discussion on the problems of identifying this kind of candidate sources and selecting the appropriate ones for the kind (family) of LT systems is out of the scope of this paper for the sake of space.

### Table I. Examples of Usable Standards

<table>
<thead>
<tr>
<th>E-learning</th>
<th>Acronym</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS Content Packaging (IMS CP)</td>
<td>ISO/IEC 12785:2009</td>
</tr>
<tr>
<td>IMS GLC Common Cartridge Version 1.0 - Final</td>
<td>IMS CC</td>
</tr>
<tr>
<td>Software quality</td>
<td></td>
</tr>
<tr>
<td>Software Engineering -- Software product Quality Req and Eval, 2005</td>
<td>SQuaRE 25000</td>
</tr>
</tbody>
</table>

Standards contain technical specifications which ensure that materials, products, processes, services, systems, or persons are fit for their intended purpose, consequence of the experience both in Industry and Academy. Standards also provide common concepts and practices that encourage interoperability and technology transfer. However, in many cases these sources express the information in a too general form, sometimes introducing problems associated with natural language usage (e.g. ambiguity, imprecision or inconsistencies), sometimes with very different abstraction level information (all in one place or just lack of detail), making the adoption of these standards and their application in e-learning practice difficult [11]. Thus, we think it is necessary to adapt, refine and express their contents in the form of explicit software and system reqs. SIREN catalogs and SIREN reqs structure provide the necessary means to achieve this goal, as explained below.

III. Basis for an E-Learning Systems SPL Infrastructure

The Software Factory (SF) concept [17] refers to those software development organizations which can group together an important number of their products in the so called Software Product Lines (SPL) [18]. Typically, SFs are able to systematize their production systems, mainly based upon reuse of a variety of artifacts (such as reqs, analysis or design models, code, and documentation). The current trend is towards SPLs in particular domains (such as banking or automotive fields). It is worth noting that domains are not necessarily disjoint and even inclusion relationships between related domains is usual. An SPL is defined by a set of systems sharing common features, which satisfy the specific needs of a
particular sector. These systems are developed within the SPL from a pre established set of assets or reusable artifacts. SPL development consists usually of two complementary processes (Fig. 1): domain and product engineering. Reqs in a SPL define the common and variable features (domain level) as well as concrete products in such line (product level).

![SPL Development](image)

Figure 1. SPL Development

### A. E-learning domains and products

In the educational context, several e-learning families can be found such as: learning management systems (LMSs), e-learning products, learning objects (LOs) and learning object repositories (LORs). SPLs can be found in any of these e-learning families. For example, automated assessment tools and simulators are e-learning products that can be considered SPLs. Several domain features and SPLs (i.e. EDA, FORM or FOSD). SIREN is compatible with such techniques [9], but using them is not essential. Variability can be specified and managed in SIREN directly by parameterization, generality and patterns. Our approach is compatible with any software development model process, provided that a sufficient attention to reqs is paid. Thus, although the adoption of an SPL strategy to apply our proposal is not strictly necessary, SPL and, in general, SF adoption provide LT systems development with a model process provided. Although we define these processes sequentially, these are performed in an iterative and incremental way.

#### 1) The Domain Engineering process

The activities in this group are carried out at the beginning of defining a new SPL and each time we wish to generate new catalogs within this SPL (see Fig. 2). For example, if we consider in the following an SPL for automated assessment of programming assignments, then a set of reqs catalogs of automated assessment tools [19] [20] [25] within the e-learning domain can be created.

a) **SPL Definition**: A list of textual descriptions regarding high level issues which should be solved by means of the products of the SPL. Business processes potentially involved are also identified and textually described in a few paragraphs, to better describe a context.

b) **Problem Domain scope**: In short, it consists of

\[
\text{REQLT} = \text{REULT} \cup \text{PRODLT} \tag{1}, \text{where}
\]

\[
\text{REULT} = \text{REU} (\text{Sources}) \cup \text{REU} (\text{myown}) \tag{2}, \text{where}
\]

#### REU(Sources) = REU(Std) \cup REU(Leg) \cup REU(Reg) \cup REU(Oth) \tag{3}, and

\[
\text{REU} (\text{Std}) = \text{REU} (\text{StdLT}) \cup \text{REU} (\text{StdSE}) \tag{4}
\]

In the expressions above: \(\text{REU} (\text{Std})\) denotes the set of catalogs created from international standards, distinguishing – just for legibility and organization issues- specific LT standards, \(\text{REU} (\text{StdLT})\), from general purpose software ones, \(\text{REU} (\text{StdSE})\); \(\text{REU} (\text{Leg})\) is the set of catalogs created from legislation and, similarly for \(\text{REU} (\text{Reg})\) (regulations) and \(\text{REU} (\text{Oth})\) (other existing sources). \(\text{REU} (\text{myown})\) denotes the set of possible catalogs built, not from existing sources but by our own, from our personal knowledge on the domain. These may include, for example, common, reusable functional reqs concerning a kind of e-learning system which configures part of a particular SPL (such as simulation objects).

Within this infrastructure \(\text{REULT}\), if we call \(\alpha\) to the instantiation function (see Fig.1) and \(\mathcal{P} (\text{A})\) denotes the set of parts of a set \(\text{A}\), then \(\alpha: \mathcal{P} (\text{REULT}) \to \text{PRODLT}\) and we can express the instantiation function as follows:

\[
\text{Let } \text{reu} \in \mathcal{P} (\text{REULT}), \text{ then } \alpha (\text{reu}) = \text{rs}, \text{ where } \text{rs} \in \text{PRODLT} \text{ is a concrete PRS, resulting from the instantiation process potentially including: (1) generic reqs in catalogs of reu reused ‘as such’ (copied); (2) generic reqs in catalogs of reu reused and modified; (3) instantiated reqs from parameterized reqs in catalogs of reu; and (4) their corresponding meta information and traces relationships. Usually, we will need to add new reqs for a given product (called deltas), not previously considered in the SPL (REULT).}
\]

#### B. A process for building and using requirements catalogs

Actually, two sub processes can be identified: one for building reusable catalogs (Domain Engineering) and another one for using them (Product Engineering). Although we define these processes sequentially, these are performed in an iterative and incremental way.

1) **The Domain Engineering process**

The activities in this group are carried out at the beginning of defining a new SPL and each time we wish to generate new catalogs within this SPL (see Fig. 2). For example, if we consider in the following an SPL for automated assessment of programming assignments, then a set of reqs catalogs of automated assessment tools [19] [20] [25] within the e-learning domain can be created.

a) **SPL Definition**: A list of textual descriptions regarding high level issues which should be solved by means of the products of the SPL. Business processes potentially involved are also identified and textually described in a few paragraphs, to better describe a context.

b) **Problem Domain scope**: In short, it consists of

1) Identifying main sources related to the domain to which the SPL belong: LT standards, SE standards, legislation, problem-specific documents, etc. For example, the Spanish PDF/RMS [16] and the ISO/IEC 12785:2009 (IMS CP) can be selected for the construction of an
automated assessment tool. Both regulations also establish suitable reqs for other SPLs and domains. Then, select, prioritize and schedule the generation of the detailed catalogs in REULT.

b.2) Generating first version catalogs, for each one of the sources selected in the previous step. Typically, these will consist of a mapping from relevant text in the sources to software reqs, where common and variable features in the domain are described.

These catalogs will contain and refine the part of the reqs at the REULTperv level, corresponding exactly to the kind of products provided by this SPL. Common and variable features, not just in the domain but in the particular SPL, are now described and detailed. Specific techniques, algorithms or procedures (even keeping variability) for implementing particular products in the SPL are now considered. Examples of refined reqs at this level are:

**C1.** The automated assessment tool shall allow the content author to compose a content package manifest that contains absolute references to the files in the repositories. (Refined from req S1, and included in the IMS CP catalog).

**C2.** The automated assessment tool shall accept programs written in [non empty set of progr. langs.] programming language. (Refined from req S2, with parameter [non empty set of progr. langs] and included in the Automated Assessment Tool (AAT) catalog).

**C3.** The person in charge of the file will use [hardware backup] for the backup. (Refined from the Spanish PDP domain req S3, with parameter [hardware backup], and included in the PDP catalog).

**C4.** The automated assessment tool used shall provide a [computer compiler] to compile and run computer programs. (Inclusive traceability relation with the req C2, that is, the reuse of C2 will imply the reuse of C4, with parameter [computer compiler]).

At this point, reqs attributes such as Project Unique Identification (e.g. C1), priority (e.g. high), source (e.g. Section 4.1 IMS CP) and person in charge (e.g. John) are incorporated. Both the reqs attributes and traceability relationships can also be reused.

2) The Product Engineering process

The activities in this group are carried out each time we wish to develop a new, specific e-learning product or evolve an existing one. For example, let us suppose that we intend to build an automated assessment tool, addressed to degree and/or vocational training programming students. The computer-based automated assessment systems allow students to solve programming exercises and to submit their solutions. These systems analyze the program across a number of program metrics (e.g. number of comments, percentage of methods declared abstract, typographic or lexical structure). They usually support online marking, provide an interface for delivering feedback, and incorporate plagiarism detection software. Three of the most important web-based tools that support automated assessment for several languages are CourseMarker, BOSS and Mooshak. These activities include (Fig.2):

a) **Product Definition:** Decide the main required features of the product (such as privacy, accessibility,etc), selecting them from a pre defined form and providing weights within an homogeneous scale. This form identifies high-level abstraction functional and non-functional reqs, both educational (e. g. Interoperability: “HLR1. The learning objects can be imported into and exported from a variety of repositories”), and general purpose (e. g. “HLR2. The product will manage sensitive

personal data”) ones. A first specification, the so called “Initial product specification”, including a prioritized list of main required features, is generated.

b) Select Sources and Catalogs: Select the available catalogs in REULT, related to features in the “Initial product specification”. For example, the available IMS CP catalog might be reused to describe how to package the learning objects mentioned in HLR1. Then, for those features not considered in our existing repository of catalogs, identify related and applicable sources of interest. For example, when migrating from a proprietary format LMS to an e-learning standard compliant LMS, the feature “The automatic assessment tool shall allow to import and export programming problems (learning objects) in the proprietary format” is not in the SPL catalogs. In this case, information sources related with the proprietary format should be selected. At this point, and according to the existing budget, we can decide either it is worthy to build new catalogs corresponding to these features / sources (e.g. if we think they can be reused in further projects) or just to use the sources directly to obtain new reqs for our PRS (part of the so called deltas, at the end of III.A).

c) Instantiate / Reuse requirements: By using the selected, available or newly built, catalogs in REULT, obtain a first version of the PRS populated with reused reqs. Examples of product reqs at this abstraction level are:

R1. The automated assessment tool shall allow the content author to compose a content package manifest that contains absolute references to the files in the repositories. (Used as it was in previous step).

R2. The automated assessment tool shall accept programs written in Java programming language. (from parameterized req C2, with alternatives Java, C/C++ and Ada compiler).

R3. The person in charge of the file will use a DVD for the backup. (from parameterized req C3, with alternatives: DVD, CD and USB memory).

R4. The automated assessment tool used shall provide a Java compiler to compile and run computer programs. (From req C4, alternatives Java, C/C++ and Ada compiler. The trace from R2 would also be instantiated).

d) Elicitate requirements: Add to the PRS new, specific reqs for this product (deltas).

e) Analyse and Negotiate: Check possible inconsistencies and problems coming from the integration of reused and newly elicitedated (deltas) reqs. Solve possible different views and interests of participating stakeholders.

f) Validate&Verify reqs : To ensure the quality of the PRS created, check it to guarantee both that the resulting product will perform as expected in the user’s environment, and whole compliance w.r.t. the RE processes and standards used for the PRS (e.g. [3]).

Activities d, e, and f are typical of any RE process, while the others help to configure a specific LT SPL. Note that these steps are applied iteratively and incrementally until an approved PRS is achieved (the so called baseline, see Fig.2).

C. Tool support

A requirements management (RM) IBM Rational RequisitePro® plug-in, named SIRENTool is already running [6]. It supports the particular reuse features to manage reqs catalogs, and complements the ordinary RE process support provided by RequisitePro.

IV. INTRODUCING THE GLOBAL PERSPECTIVE

Nowadays, globalization has reached a high level of development and penetration in many human activities. Regarding e-learning systems, two complementary views arise: the process (global software development-GSD) and the product (internationalization, global education support -GES).

A. E-learning global software development

This point applies when the e-learning software is developed in a distributed way, involving different countries. This new paradigm implies the dispersion of the stakeholders across countries placed in different continents, so that geographical, temporal and even cultural distances have to be taken into account. The current rise of this trend is due to the advantages that GSD brings in comparison to the traditional, non-distributed software development process: lower cost of qualified work forces in certain countries, up to a 24 hour-workday available, access to a globally accessible pool of highly-qualified resources and improved proximity to clients.

1) A GSD risk and safeguards repository

A Systematic Literature Review (SLR) [21], extending our work in [22], has led to the compilation of a more complete repository which gathers the risks, and their corresponding safeguards, that concern RE when developed in a GSD environment. Based on the SLR, a taxonomy composed by six different types of concerns or areas of interest was compiled: Communications and distance, Knowledge management and awareness, Cultural differences, Management and project coordination, Tools, Clients

B. Internationalization reqs and global education issues

This point applies to the final e-learning product, independently of the process used to develop it. We propose using the infrastructure and processes described above (sect. II) to identify, select and manage internationalization and localization (I/L) reqs concerning LT products. Special attention must be paid to the CEN Workshop on LT and the group JTC 1/SC 36/WG 7 working on “ITLET - Culture, language and individual needs”. This support, in addition to specific global education reqs (the so called deltas above), enables the definition of new platforms, conforming current trends to give systematic, reusable support to the new challenges imposed by globalization in education and the related LT systems. To adapt our proposal to these issues, it will be enough to include specifically the I/L sources, relevant to LT systems (considering similar clusters: standards, legislation, regulations, etc.), obtaining the corresponding

catalogs, adding them to REULT, and using these new I/L reqs, following the process explained in III.B.

C. Tool support for the globalization issues

Regarding the specific GSD issues, the three points below (or a combination of them) can be considered. In order to make easier the construction of LT products supporting a wide range of internationalization reqs, the last two ones apply:

- Extending SIRENTool with new catalogs and functionalities, accomplishing the specific GSD reqs, so extending the basic RE capabilities of RequisitePro\textcircled{R}. A similar schema can be followed using any RM tool, other than RequisitePro\textcircled{R}.
- Using wikis or semantic wikis as support technology, which are frequently proposed in the literature to enable distributed collaboration. For example, SoftWiki (http://softwiki.de/netzwerk/) has been used to define an ontology for RE, SWORE.
- Using Social Networks, which provides a global, popular infrastructure for both GSD and GES activities. However, their real strengths and weaknesses must be analyzed in depth. We have some preliminary study done, for supporting the design of teams for software development projects [23]. We are running an in-depth analysis to find out real chances of these tools for the aim set out.

V. CONCLUSIONS AND FUTURE WORK

This approach aims at helping in the production of LT systems according to SE and RE best practices. Independently of any SE or RE particular process model or method, the proposal can be integrated with the common practice in the organization, just adapting the structure of the reqs documents. Catalogs provide quality and well organized set of reqs, improving the related information from the original sources (as explained at the end of sec. II.B).

The identification and selection of suitable sources may be a difficult task, because of their number and variety. We recommend that this process, as well as the generation of new catalogs for the organization, be addressed gradually, starting from the top priority and most used or more important sources (e.g. those that are mandatory, by law). The more catalogs we have in our repository, the better rates we will obtain both in productivity (reusing and instantiating reqs is faster than defining them from the scratch), quality (reused reqs are improved, potentially in each iteration) and interoperability (catalogs based upon standards). Also regarding this limitation, we plan to define a classification (taxonomy) of LT systems, together with a mapping of the different groups found, with the corresponding standards and sources, and updating procedures, in order to help in those identification and selection tasks.

REFERENCES