ISO-Standardized Requirements Activities for Very Small Entities

Philippe Saliou, Vincent Ribaud

Post-print de Requirements Engineering in Small Companies 2010,
Institut für Informatik und Wirtschaftsinformatik (ICB),
Universität Duisburg-Essen

ISO-Standardized Requirements Activities for Very Small Entities

Philippe Saliou and Vincent Ribaud

Université de Brest, LISyC, CS 93837, 29238 Brest Cedex, France
Université européenne de Bretagne, France
{ Philippe.Saliou@univ-brest.fr, Vincent.Ribaud@univ-brest.fr }

Abstract. The use of Software Engineering standards may promote recognized and valuable engineering practices for Very Small Entities (VSEs) but these standards do not fit the needs of VSEs. The ISO/IEC Working Group 24 (WG24) is developing the ISO/IEC 29110 standard "Lifecycle profiles for Very Small Entities"; this standard is due for approval in June 2010.
A pilot project about ISO 29110 use has been established between our Software Engineering group and a 14-person company building and selling counting systems about the frequentation levels of public and private sites. The pilot project aims to help VSEs deliver the Software Requirements Specification, Test Cases and Test Procedures for a new web-based system intended to manage fleets of counting systems. As the project goes along, it appears that the 29110 set of documents was not up to the task of sustaining this VSE in its engineering activities. We supported the VSE in two ways: (i) a Training Session based on the 29110 Requirements Analysis activity, and (ii) Self-Training Packages - a set of resources intended to develop experience and skills in Requirements Identification and SW Requirement Specification (SRS). Our inspiration stems from the 15504-5 standard with a desire to provide software engineers with an exemplar set of base practices providing a definition of the tasks and activities needed to fulfil the process (e.g. requirements) outcomes. Task definition is collected on a task card. The results of this pilot study provide the VSE with a roadmap through the Requirements activity, which is compatible with the ISO/IEC 29110 standard.

Keywords: Very Small Entities, Requirements Specification, ISO/IEC 29110.

1 Introduction

Very Small Entities (VSEs) are recognized as being very important to the software economy, and produce stand-alone or integrated software components in large software systems. The use of Software Engineering standards may promote recognized and valuable engineering practices - but these standards do not fit the characteristics of VSEs. The term 'Very Small Entity' (VSE) was defined by the ISO/IEC JTC1/SC7 Working Group 24 (WG24) as being "an entity (enterprise, organization, department or project) having up to 25 people". This definition has

subsequently been adopted for use in the ISO response to VSEs' specific needs: the emerging ISO/IEC 29110 standard "Lifecycle profiles for Very Small Entities" [1]. The 29110 standard defines a group of Standardized Profiles, including the ISO/IEC IS 29110-4-1 Basic profile [2] which applies more specifically to a VSE that is involved in software development of a single application by a single project team with no special risk or situational factors.

A VSE claiming compliance with ISO/IEC IS 29110-4-1 will implement and use all the profile elements, as identified in Clause 7 of the profile specification [2]. The profile elements concerning requirements are: Project Plan Execution (PM.2) and Project Assessment and Control (PM.3) - producing the *Change Request* work product, and Software Requirements Analysis (SI.2) - producing work products *Change Request* and *Requirement Specification*.

This paper reports some of the conclusions reached by a pilot project the authors conducted with a 14-person VSE that builds and sells counting systems about the frequentation of private and public sites. Only 3 of the employees are software developers, and the VSE asked for assistance with software project management – mainly managing requirements and establishing a disciplined test process. Deployment Packages (DP) are expected to be particularly helpful. A DP is "a set of artefacts developed to facilitate the implementation of a set of practices, for the selected framework, within a VSE [3]". As the project goes along, it appears that the 29110 set of documents (including DPs) was not up to the task of sustaining this VSE in its engineering activities. One idea defended here is that implementing standardized software engineering activities in a VSE requires specific and operational materials and mechanisms. We are proposing to provide VSE employees with Self-Training Packages intended to help the engineer carry out [and learn] the task.

Section 2 presents related work, and offers an overview of a SE standard for VSEs. Section 3 introduces the pilot project, presents Self-Training Packages, and evaluates the system's efficiency. We conclude with brief perspectives.

2 Related work

2.1 Requirements engineering for small software companies

In 2007, IEEE Software published a special issue on the theme "SE Challenges in Small Software Companies". The guest editors' introduction presents common challenges faced by large and small software development companies: "They need to manage and improve their software processes, deal with rapid technology advances, maintain their products, operate in a global software environment, and sustain their organizations through growth [4]". Yet VSEs also have specific characteristics and needs.

J. A. Calvo-Manzano et al. [5] presented an SPI solution called MESOPYME for small and medium-size enterprises (SME). MESOPYME is based on the Action Package concept - a mechanism which assists faster and affordable SPI program implementation for SMEs. Experimentation with this package has been carried out in

the Requirements Engineering domain. The structure of an Action Package (such as the Requirements Engineering Action Package) presents similarities to our own structure of Self-Training Packages. Training is provided using the Action Package Training component. This component basically comprises four courses: software process model (CMM), the improvement method (MESOPYME), team building, and training in the process selected for improvement (e.g. Requirements Engineering). Our approach is different in that MESOPYME is a Software Process Improvement method for SMEs, whereas we aim to implement a Lifecycle Standardized Profile in VSEs.

The REDEST project [6] aimed to develop a selection of innovative Requirements Engineering methodologies to act as Best Practice Cases for 14 independent software development companies. REDEST disseminated results via a Best Practice Case Booklet [7]. Case Study 8, carried out by a VSE named SignalKomplex, aimed to experiment with the following features: introduction of a systematic RE process; a more thorough understanding of customer requirements; basic tracking of changes in requirements. The size (24 employees) and the products and services (vehicle traffic control equipment) provided are very similar to the VSE case study reported in this paper. SignalKomplex baseline project (development of a vehicle sensor card) presents similarities with the VSE project (a web-based system intended to manage fleets of counting systems). The RE approach selected by SignalKomplex was a method called PAISLEY, which is an approach whose focus couples Requirements Elicitation with the processes of the object being developed. SignalKomplex selected this approach because it was equally operable for hardware and software requirements, a key issue from the SignalKomplex point of view. As SignalKomplex reported in the REDEST Best Practice Case Book [7, p. 114], the RE solution also required input from other areas of the company, such as the sales and business departments. Combining pure, technical specifics with other inputs was mostly achieved by exploiting spreadsheet features. The ISO/IEC 29110 Basic Profile is applicable to VSEs which do not develop critical software products, and the traceability tool provided with the Deployment Package associated with requirements is a spreadsheet-based tool. Our proposal is to perform a preliminary Requirements Elicitation through the building of a Services Identification List (see Figure 1) which is also supported by spreadsheets. Keeping a powerful requirements management tool as simple as possible is a key issue for a VSE.

2.2 SE Standards for Very Small Entities

ISO initiative. Software engineering standards and methods often neglect the needs and problems of the small and medium-sized organizations which constitute a major part of the software industry. The ISO/IEC Working Group 24 (WG24) is developing the emerging ISO 29110 standard, which is a set of technical specifications and guides for use by very small software enterprises. This set is based on the concept of VSE profile [1]. The purpose of a VSE profile is to define a subset of ISO/IEC standards relevant to the VSE context - for example, selected processes and outcomes of ISO/IEC 12207 [8] and selected products of ISO/IEC 15289 [9].

ISO/IEC 29110 Set of Documents. The ISO/IEC 29110 Set of Documents comprised multiple documents (overview, profiles and guides) with different purposes and audiences. The overview document (Part 1) [1] introduces processes, lifecycle and standardization concepts. Part 2 [10] introduces the framework and the

taxonomy. Part 3 [11] defines the process assessment guidelines and compliance requirements needed to meet the purpose of the defined VSE profiles.

The document ISO/IEC 29110-4-1 [2] provides the specification for all the Generic Profile Group profiles. The Generic Profile Group is applicable to VSEs which do not develop critical software products [1]. The Basic Profile describes the software development of a single application by a single project team with no special risk or situational factors [2]. The ISO/IEC 29110-5-1-2 document [12] provides an implementation management and engineering guide for the Basic Profile.

The ISO 29110 Set of Documents is due for approval in June 2010. It is possible that VSEs may be intimidated by this set. Moreover, this set includes ISO standards, submitted to copyright fees. However, guides are targeted at VSEs, and should be VSE-accessible, in terms of both style and cost [1].

2.3 Basic Profile

Basic Profile Processes: Objectives and Tasks Decomposition. The Basic Profile establishes VSE characteristics, needs and suggested competencies, and uses it to define process objectives. For instance, objectives related to requirements are: the SI.O2 objective "Software requirements are defined, analyzed for correctness and testability, approved by the Customer, baselined and communicated [2, p. 7]", the SI.O3 "[…] Consistency and traceability [of the design] to software requirements are established [2, p. 8]", and the SI O.4 "[…] Traceability [of the software components] to the requirements and design are established [2, p. 8]".

The Basic Profile consists of 2 processes: Project Management (PM) and Software Implementation (SI). A process is defined as "a set of interrelated or interacting activities which transforms inputs into outputs [8]". An activity is "a set of cohesive tasks of a process [8]". For each activity of the PM and SI processes, the Basic Profile details the tasks to be performed: role, description of the task, input and output products. For instance, the starting point of the 29110 use for requirement is the SI.2 "Software Requirements Analysis" activity, its list of tasks: SI.2.1 to SI.2.7 and the associated roles. Roles are: TL Technical Leader, WT Work Team, AN Analyst, and CUS Customer. Table I provides a tasks breakdown for the activity SI.2 [2, pp. 15].

Table 1. SI.2 Software requirements analysis - tasks and roles. * means (if appropriate).

Task List	Role
SI.2.1 Assign tasks to the Work Team members in accordance with their role, based on the current *Project Plan.*	TL, WT
SI.2.2 Document or update the *Requirements Specification.*	AN, CUS
SI.2.3 Verify the *Requirements Specification.*	AN
SI.2.4 Validate the *Requirements Specification*	CUS, AN
SI.2.5 Document the preliminary version of the *Software User Documentation* or update the present manual. *	AN
SI.2.6 Verify the *Software User Documentation*	AN
SI.2.7 Incorporate the *Requirements Specification*, and **Software User Documentation* to the *Software Configuration* in the baseline.	TL

Basic Profile Products. Part 29110-4-1 provides Work product specifications, and Activity input & output specification. For instance, SI.2.1 to SI.2.7 tasks have associated output products: *Requirements Specification*, *Verification Results*, *Change Request*, *Validation Results*, and [preliminary] *Software User Documentation*.

2.4 Deployment Package

Significant help is expected from Deployment Packages (DP). C. Laporte, the editor of the ISO/IEC 29110 defines a DP as "a set of artefacts developed to facilitate the implementation of a set of practices, of the selected framework, in a VSE [3]". The elements of a typical deployment package are: process description (activities, inputs, outputs, and roles), guide, template, checklist, example, presentation material, reference and mapping to standards and models, and list of tools [13]. Packages are designed in such a way that a VSE is able to implement its content without having to implement the entire framework at the same time.

Regarding requirements, the Deployment Package - Software Requirement Analysis [14] adds depth to the standard, providing guidance through a simplified breakdown of the SI.2 SW requirements analysis activity. The DP sums up the SI.2 activity in 4 tasks: requirement identification, requirements refinement and analysis, requirements verification and validation, requirements change management. For each of theses 4 tasks, the DP describes a step-by-step method.

This DP follows the SPEM approach promoted by OMG in [15]. In this DP, the tasks required for performing SW requirements analysis are defined through textual step-by-step explanations, describing how specific fine-granular development goals are achieved, through which roles, and with which resources and results. The DP also provides several templates (including a simplification of IEEE 830 [16]) of a Software Requirement Specification Document.

Training materials and an Excel-based Traceability tool can be downloaded from the public WG24 web site http://profs.logti.etsmtl.ca/claporte/English/VSE/index.html.

3 A Pilot Project on Requirements

3.1 Overview

Context of the VSE. A VSE of 14 people (with 3 software engineers) requested our help in Spring 2009. This VSE designs, builds, develops and sells a counter system intended to collect and analyze frequentation of public or private sites. Counting systems are based on stand-alone counter boxes (including sensors, power supply, data storage, and data exchange) and a software chain able to collect, analyze, present, and report counting data. The data set was downloaded from counters via infrared link or GSM, stored on PC and exchanged via a file transfer utility.

The new software project. The VSE started a complete reconstruction of its software chain in order to transform it into a web-based system called Eco-Visio, intended to host data from fleets of counting systems for each client, and able to process statistics and generate analysis reports on counting. At the end of June 2009,

the VSE hired an Information Technology graduate from our university. At the same moment in time, we initiated a pilot project intended to help the VSE implement just one part of the 29110.

The pilot project. The absence of requirement traceability and systematic testing was rapidly recognized by all stakeholders. Both authors also agreed that project management was in need of improvement, but we deliberately omitted this point. We proposed a 2-stage plan of action: - 1- implementation of the “Software Requirements Analysis” Deployment Package and - 2 - implementation of the “Software Testing” Deployment Package. The first stage is complete, and reported on in this paper.

Deployment Package. The starting point of the 29110 use for requirement is the SI.2 “Software Requirements Analysis” activity, its list of tasks - SI.2.1 to SI.2.7 - and the associated roles. A step-by-step approach to perform the required SI.2 tasks is given in the Deployment Package - Software Requirement Analysis [14]. One VSE employee received a short training course, using the training material associated with this DP, and downloaded the Traceability Tool provided with the DP. Despite all this assitance, the VSE engineer was unable to proceed with 29110 Requirements Engineering. He therefore attended a Training Session on requirements, based on the 29110 materials. A description of this session is presented in section 3.2.

Self-training packages. During the training session, the VSE engineer – like his co-trainees – attained an initial level of proficiency in using the 29110 for Requirements Specification – yet trainees asked for further assistance and guidance. We therefore constructed a dedicated assistance approach, which is presented in section 3.3. This approach relies on Self-Training Packages - a set of resources intended to develop experience skills in SE activities, e.g. Requirements Identification and SW requirement specification.

Assessment. We built 2 groups: a control group of 9 people and a study group of 10 people performing the 29110 training. We intended to measure the efficiency of the training system by comparing requirements competencies between both groups.

3.2 Training session

Training session context. We scheduled a training week on 29110 Software Requirements Analysis in December 2009. 10 young engineers (including our VSE engineer) attended the session. The 29110 Training Session comprises a course on requirements and a case study using the DP - Software Requirement Analysis [14].

Content of the training session. The session begins with an introductory lecture on requirements, but trainees are plunged into 'doing' with the preparation of a peer-review on a requirements analysis guide. This guide is issued by an ISO-9001 major software company (at which both authors had been employed for about ten years). The SW Requirements Specification (SRS) Document is issued by the DOD-STD-2167A software development standards [17]. This guide is intended to facilitate the writing of the SRS. Peer-reviewing this guide provided trainees with initial exposure to standardized requirements management.

During the second phase of the session, trainees have to contribute to the writing of a similar guide, based only on the 29110 standard. Authors provide trainees with a preliminary version of the guide, written in a top-down manner, starting from the 12207 standard processes devoted to requirements (6.4.1 Stakeholder Requirements Definition, 7.1.2 SW Requirements Analysis) to the 29110 Basic Profile SI.2 “Software Requirements Analysis” activity. Trainees have to incorporate both the DP - Software Requirement Analysis and its step-by-step approach into the guide.
Finally, trainees have to apply the enhanced guide to a 'real' SRS and update this SRS to satisfy compliance with the guide. The 'real' SRS is for eCompas - an existing system developed by the first author and former graduate students.

3.3 Towards requirements management capability

Objectives. Despite the path traced in the standard (including the guidance provided by the DP), some young engineers (and this is true of the VSE engineer in particular) may be unable to find their way through the managing requirements. Below, we present the step-by-step path proposed by the DP Requirement Analysis.
Task 1. Requirements identification. The objective is to clearly define the scope of the project and identify key requirements of the system. Steps are: (i) Collect information about the application domain; (ii) Identify project scope; (iii) Identify and capture requirements; (iv) Structure and prioritize requirements.
Task 2. Requirements refinement and analysis. The objective is to detail and analyze all the requirements identified. Steps are: (i) Detail requirements; (ii) Produce a prototype.
Task 3. Requirements verification & validation. The objective is to verify requirements and obtain validation from the customer or his representative. Steps are: (i) Clarify fuzzy requirements (verification); (ii) Review SRS (Software Requirements Specification); (iii) Validate requirements.
Task 4. Requirements change management. The objective is to manage requirements change in line with a process agreed upon with the customer. Steps are: (i) Track changes to requirements; (ii) Analyze impact of changes; (iii) Identify changes that are beyond the project scope; (iv) Prioritize changes.

The core of requirements gathering and specification must be performed in tasks 1 and 2. We decided to build two Self-Training Packages aimed at helping young engineers with: A - Requirements Identification and B - SW Requirements Specification. A discussion of Self-Training Packages is beyond the scope of this paper, but we will say that one objective of our research group is to provide VSEs with a training complement to the 29110 set of documents called the 'Self-Training Package'. Self-training packages are intended to be performed autonomously by VSE employees, requiring (almost) no interaction with a coach - except at the time of package delivery to the VSE.

The inspiration stems from the 15504-5 standard [19, Part 5] with a desire to provide software engineers with an exemplar model of software engineering activities together with complementary self-training material. While we are designing self-training for an SE activity (such as Requirements Analysis) and its required tasks

(such as Requirements identification or Requirements refinement and analysis), we aim to prescribe the engineer's tasks broken down into small units. Task definition is collected on a task card.

Fig. 1. Example of a task card.

N° 24	**Date**:	**Origin**:	**Roles assignment**	
Project :	**TASK CARD**		ANalyst	**Employee X** **Employee Y**
Process: Software Implementation (SI) **Activity**: Software Requirements Analysis (SI.2)				**Task Title**: Requirements bootstrap

WORK DESCRIPTION

Objectives

The goal of this task is to collect and identify requirements using a structured and prioritized list of requirements, and to establish a synthesis of users' needs.

Objectives are strongly related to SI.2.1 task objectives: "*The objective of this activity is to clearly define the scope of the project and identify the key requirements of the system.*"

Step-by-step

1. Identify functional and technical needs
   Extract users' needs from the eCompas Statement of Work (call for tender) and the preliminary response to tender.
   Write a unique document "Needs Synthesis Document", gathering together any elements related to a functional or technical need.
2. Summarize required services (Services Identification List)
   Identify, classify and sum up users' needs through a list of high-level services required by the eCompas software.
   Each identified service (or sub-service) shall be documented with:
   - Identification number (could be temporarily left blank)
   - Type (Functional or Technical) and Domain (one of the five eCompas domain areas)
   - Service number (hierarchical numbering inside domains)
   - Actors (main users of the service)
   - Summary (a very short description of the service)
   - Origin (traceability to Statement of Work or Tender response)
   - Link to "Need Synthesis Document" (references to corresponding paragraphs)
3. Establish a glossary of the eCompas domain
4. Structure and prioritize the "Needs Synthesis Document"
   With the help of the "Services Identification List", rewrite a new version of the "Needs Synthesis Document" complying with the proposed hierarchy.
   Establish traceability.
   Number services with a hierarchical identification number.
5. Perform a peer-review of an existing SW Requirements Specification
   Prepare the review of the eCompas SRS following the instructions of the Reviewer Guide

Resources

- eCompas Statement of Work and Tendering answer
- SRS Writing Guide and Peer- Reviewer Guide

…

Output products

The main output product of this task is the "Needs Synthesis Document", which will be used in the next task - "SRS writing" as a preliminary version of the Software Requirements Specification.

Products	**V.**	**Milestone**
Needs Synthesis Document	A, B	
Services Identification List	A	

Task cards. The description of the task is designed as a theatre scene: the scene being the reference context in which the action takes place. The scene aims for unity

of place, time and action; it is a situation in which people do [and learn], a scenario of actions, a role distribution, an area mobilizing resources and means. The different components of a scene, along with their articulation, are depicted on a task card (see an example of the Requirements bootstrap card in Figure 1).
Its main elements are:

- Related 29110 Process / Activity

This reference (SI / SI.2 SW Requirements Analysis in this instance) provides a smooth link to the 29110 and through the ISP to the 12207 and 15504 standards.

- Role

Role (here ANalyst) is a quick reference to the 29110 Role

- Task Title and Objectives

Similar to Process Title, Process Purpose, and Process Outcomes as defined in ISO/IEC 12207

- Step-by-step

A comprehensive description of the work to be done - intended to be useful as a practical guide to completion of the task.

- Resources

The set of resources required. This may set up the context and/or be required to perform the task. It may include online courses that are affordable to a technology transfer centre, where the cost is beyond the reach of a VSE.

- Output products

This is generally a 29110 Work Product, or an intermediary product required to build this Work Product. A hidden goal is to initiate and develop a strategy of capitalizing on the activity, and transferring knowledge to VSE employees.

Self-training. For the self-training reported in this section, we built two task cards: Requirements bootstrap and SRS writing. Self-training is then performed as a case study: a set of resources is used to set up the context, and engineers have to perform tasks as they should do in a 'real' situation.
Our study group of 10 engineers performed both Self-Training Packages in January and February 2010. The first Training Package was intended to offer an initial level of maturity in ISO/IEC 29110 Requirements Management (through the study of SI.2 activity and a review of a ‘real’ WP11 Requirements Specification) and the second Training Package aims to perform a Requirements Analysis on a ‘real’ case. Very little interaction with the coach (the first author) occurred. Each engineer completed each package in roughly a week.

3.4 Process assessment

The Part ISO 29110-3 [11] is an Assessment Guide applicable to all VSE profiles. It is compatible with ISO/IEC 15504-2 and ISO/IEC 15504-3 [18]. As specified in [11], “a VSE-specific Process Assessment Model (PAM) can be derived by selecting only the assessment indicators in the 15504-5 Exemplar PAM, relevant to the corresponding process outcomes defined in ISO/IEC 29110-4.”

For instance, in the Basic Profile, the SI Process defines 7 objectives and SI.02 is the only one relevant to requirements: “Software requirements are defined, analyzed

for correctness and testability, approved by the Customer, baselined and communicated." [2] Then, reducing the 7.1.2 Software Requirements Analysis Process outcomes (15504 ENG.4) corresponding to the SI.02 objective will give:

1) requirements allocated to the software elements of the system and their interfaces are defined
2) software requirements are analyzed for correctness and testability
6) software requirements are approved and updated as needed
8) software requirements are baselined and communicated to all affected parties

If we apply the profile to the Base Practices of ENG.4, we can remove Base Practices that do not contribute to the selected outcomes (1, 2, 6, and 8). Hence, the list of profiled Base Practices of the ENG.4 Process is reduced to ENG.4.BP1 Specification of software requirements; ENG.4.BP3: Development of criteria for software testing; ENG.4.BP5: Evaluation and updating of software requirements; ENG.4.BP6: Communication of software requirements.

Clause 5 of ISO/IEC 15504-2 [19, Part 2] defines a measurement framework for the assessment of process capability, defined on a six point ordinal scale. Within this measurement framework, the measure of capability is based upon a set of process attributes (PA). Each attribute defines a particular aspect of process capability. The extent of process attribute achievement is characterized on a defined rating scale. Clause 6 of the 15504-5 [19, Part 5] presents the process capability indicators related to the process attributes associated with capability levels 1 to 5. Process capability indicators are the means of achieving the capabilities addressed by the considered process attributes.

ISO/IEC 15504 separates processes and capability levels in two dimensions whilst CMMI handles them in a single dimension. However, it should be pointed out that separate process and capability dimensions may discourage a VSE regarding process assessment. For instance, capability level 2 indicators applied to requirements relate to defining, planning, monitoring and adjusting the performance of the SI.2 Requirements Analysis activity and to identifying, defining, documenting, reviewing and adjusting each work product related to this activity. In our opinion, this kind of assessment will neither determine whether a VSE achieves the Basic Profile, nor help the VSE to improve its Requirements Engineering implementation. We would like VSE employees to understand the importance of the assessment principle, whilst performing regular self-assessment on a reduced set of major objectives. Such an objective should be formulated with a sentence in the "To be able to …" format. This proposal, applied to Requirements Engineering, is detailed in the following section.

3.5 Evaluation of the system efficiency

Since 2008, local employers in Brest have significantly increased take-up of a work placement system called "Contrat de professionnalisation" (professionalization contract) over a period of 12 months. During these 12 months, fully-paid employees attend university for approximately 250 hours of technical training (about 40 days over the whole year). This academic year, 19 young software engineers who graduated from our university in June 2009 after a 4-year programme in Computer

Science or Information Technology, are benefiting from this system. As mentioned above, 10 people chose the 29110 training; the other 9 chose to attend a UML-based analysis course. Thus, we have a population divided into 2 groups: a control group of 9 people and a study group of 10 people performing the 29110 training reported in previous sections. The UML-based analysis course and 29110 training were performed within a period of about 3 weeks between September 2009 and February 2010. Hence, we sought to measure the efficiency of the training system by comparing requirements competencies between both groups.
We defined three major objectives in requirements:

- To mobilize specification methods and tools in a real project
- To work under the control of a standardized baseline
- To produce a Software Requirement Specification (including traceability)

We decided to assess each objective on a self-assessment scale ranging from 0 to 5: - 0 - ? : Do not know anything about the topic; - 1 - Fog: has only a vague idea; - 2 - Notion: has a general idea but is unable to achieve the objective; - 3 - User: is able to achieve the objective with the help of an experienced colleague and has an initial experience of its achievement; - 4 - Autonomous: is able to work autonomously; - 5 - Expert: is able to act as an expert to modify, enrich or develop the knowledge area on which the objective focuses. We asked each of the 19 engineers to self-assess themselves three times: 1 – At job start: at the beginning of their (first) job, young engineers complete the first self-assessment; all participants did this in September 2009; 2 – At 6 months: after 6 months of employment, young engineers complete the second self-assessment; this was done in March 2010 for the whole group; 2 – At 9 months: in order to assess how software engineering practices are maturing, young engineers complete a third self-assessment in June 2010.
Table 2 presents average self-assessment scores for both groups.

Table 2. Base Practices average self-assessment scores.

	Control Group			**Study Group**		
Objectives	***Sep. 09***	***Mar. 10***	***Jun. 10***	***Sep. 09***	***Mar. 10***	***Jun. 10***
SI.2.1 To mobilize specification methods and tools in a real project	1.56	2.11	2.11	1.50	2.70	2.80
SI.2.2 To work under the control of a standardized baseline	0.78	1.44	1.44	0.70	2.60	2.60
SI.2.3 To produce a Software Requirement Specification (including traceability).	2.33	2.67	2.89	1.40	2.80	3.20

No statistical comparison was performed. Requirements training took place for both groups. However, there is evidence that self-assessment scores are increasing more significantly for the study group than for the control group.

Table 3 presents score frequency distribution for both sets.

Table 3. Base Practices self-assessment distribution.

September 2009	Control Group							Study Group						
Objectives	?	F	N	U	A	E	*Avg.*	?	F	N	U	A	E	*Avg.*
SI.2.1	2	3	1	3	0	0	*1.56*	1	4	4	1	0	0	*1.5*
SI.2.2	3	5	1	0	0	0	*0.78*	5	3	2	0	0	0	*0.7*
SI.2.3	0	1	4	4	0	0	*2.33*	2	4	2	2	0	0	*1.4*
March 2010	Control Group							Study Group						
Base Practice	?	F	N	U	A	E	*Avg.*	?	F	N	U	A	E	*Avg.*
Objectives	1	1	4	2	1	0	*2.11*	0	0	4	5	1	0	*2.7*
SI.2.2	2	2	4	2	1	0	*1.44*	0	0	5	4	1	0	*2.6*
SI.2.3	0	1	2	5	1	0	*2.67*	0	0	3	6	1	0	*2.8*
June 2010	Control Group							Study Group						
Objectives	?	F	N	U	A	E	*Avg.*	?	F	N	U	A	E	*Avg.*
SI.2.1	1	1	4	2	1	0	*2.11*	0	0	3	6	1	0	*2.8*
SI.2.2	2	2	4	1	0	0	*1.44*	0	0	5	4	1	0	*2.6*
SI.2.3	0	0	3	4	2	0	*2.89*	0	0	3	5	2	0	*3.2*

Empirical evaluation. The VSE engineer reported that he was now ready to apply the SI.2 SW Requirements Analysis on the Eco-Visio project. As the specifications were soon established by another VSE colleague, he only reviewed and rewrote some sections of the existing Requirements Specification, in order to establish compliance with the template provided in the DP - Software Requirement Analysis [14]. Once updated, the WP11 Requirement Specification [Validated] served as an input to the SI.5 SW Integration and Tests. The system has been deployed since April 2010 and load testing and application optimization should be soon complete. Defects have to be corrected through a short cycle of SI activities.

As an empirical measure of its satisfaction, the VSE asked for a similar approach for the SI.5 SW Integration and Tests. In particular, the VSE wants guidance and support in establishing a disciplined Change Request Process. A Self-Training Package is under construction, and we should start with the “Software Testing” DP [19] as a basis for the whole Training Package. Probably because Tests occur in many SE activities, this DP is organized so that it spans PM and SI tasks, raising a wealth of new questions.

5 Conclusion and future work

We reported on a system that was intended to help a VSE with requirements management. Two points are discussed (1) a Training Session based on 29110 materials; (2) Self-Training Packages intended to perform requirements definition and analysis through a step-by-step approach. We used self-assessment to establish a comparison between a control group of 9 people attending a UML-based analysis course and our 10-person study group performing our proposition. Self-assessment scores are increasing more significantly for the study group than for the reference set. The concept of the Self-Training Package seems to extend to other processes such as design or testing. Further work is required to determine how far the scope of this concept and its main tool - task cards - can be extended.

References

1. International Organization for Standardization (ISO): ISO/IEC DTR 29110-1 Software Engineering - Lifecycle Profiles for Very Small Entities (VSEs) -- Part 1: Overview. ISO, Geneva (2010)
2. ISO: ISO/IEC FDIS 29110-4-1 Software Engineering - Lifecycle Profiles for Very Small Entities (VSEs) -- Part 4: Specification-VSE Generic Profile Group. ISO, Geneva (2010)
3. Laporte, C. Y.: Contributions to Software Engineering and the Development and Deployment of International Software Engineering Standards for Very Small Entities. PhD thesis of the Université de Bretagne Occidentale, Brest (2009), http://tel.archives-ouvertes.fr/tel-00483255/fr/
4. Richardson, I. , Wangenheim, C. G.: Why are Small Software Organizations Different?. IEEE Software 24 (1), pp.18-22 (2007)
5. Calvo-Manzano J. A. et al.: Experiences in the Application of Software Process Improvement in SMES. Software Quality Journal 10 (3), pp. 261-273 (2002)
6. Hardiman, S.: REDEST - 14 Best Practice SME Experiments with Innovative Requirements Gathering Techniques. In: Proceedings of the IEEE Joint International Conference on Requirements Engineering (RE'02), pp. 191 (2002)
7. CARSA: Redest Requirements Engineering Best Practice Case Book - Experiment-based cross-fertilisation and dissemination of software requirements gathering techniques. http://www.redest.net/documentos/LibretoRedest.zip (2003)
8. ISO: ISO/IEC 12207:2008 Information technology -- Software life cycle processes. ISO, Geneva (2008)
9. ISO: ISO/IEC 15289:2006, "Systems and software engineering -- Content of systems and software life cycle process information products (Documentation)". ISO, Geneva (2008)
10. ISO: ISO/IEC FDIS 29110-2 Software Engineering - Lifecycle Profiles for Very Small Entities (VSEs) -- Part 2: Framework and Taxonomy. ISO, Geneva (2010)
11. ISO: ISO/IEC DTR 29110-3 Software Engineering - Lifecycle Profiles for Very Small Entities (VSEs) -- Part 3: Assessment Guide. ISO, Geneva (2010)
12. ISO: ISO/IEC DTR 29110-5-1-2 Software Engineering - Lifecycle Profiles for Very Small Entities (VSEs) -- Part 5: Management and Engineering Guide-Basic VSE Profile. ISO, Geneva (2010)
13. Laporte, C.Y., Alexandre, S., O'Connor, R.: A Software Engineering Lifecycle Standard for Very Small Enterprises. In: R.V. O'Connor et al (eds) EuroSPI 2008. CCIS, vol. 16, pp. 129-141. Springer-Verlag, Heidelberg (2008)
14. Alexandre, S., Laporte, C. Y.: Deployment Package - Software Requirement Analysis. Available from http://profs.logti.etsmtl.ca/claporte/VSE/Publications/DP-Software Requirements Analysis-V1_2.doc (2010)
15. OMG, "Software & Systems Process Engineering Meta-Model Specification Version 2.0", http://www.omg.org/cgi-bin/doc?formal/08-04-01.pdf (2004)
16. IEEE: IEEE Std 830:1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE (1998)
17. Department of Defence: Defense System Software Development, DOD-STD-2167A. 29 February 1998 (1998)
18. ISO: ISO/IEC 15504 Information technology -- Process assessment. ISO, Geneva (2004)
19. Gómez Arenas, L.: Deployment Package - Software Testing. Available from http://profs.logti.etsmtl.ca/claporte/VSE/Publications/DP-Software_Basic_Profile_Testing-CL00.doc (2010)