This article presents a software development project carried out by a two-person startup enterprise. The objective of the project was to develop a social networking website for travelers. The new ISO/IEC 29110 standard developed specifically for very small entities and startups was used to develop the software for a Web application to allow users to collaborate with all members of a network of friends, planning their trips and sharing the details in a simple and easily accessible way. Managing this project and developing the software for it took about 1,000 hours. Using proven software engineering practices set out in the ISO/IEC 29110 standard enabled the two-person team to plan and execute the project—expending only 13 percent of the total project effort on rework (i.e., wasted effort). About 9 percent of the total project effort was expended on prevention tasks and 6 percent on evaluation tasks, such as desk check peer reviewing and testing. The last section of this article presents lessons learned and a list of recommendations for future users of the ISO/IEC 29110 standard and guides.

Key words

cost of quality, deployment package, desk check, ISO/IEC 29110, peer review, process, rework, standard, startup, very small entity (VSE)

Development of a Social Network Website Using the New ISO/IEC 29110 Standard Developed Specifically for Very Small Entities

CLAUDE Y. LAPORTE, CHARLES HÉBERT, AND CHRISTIAN MINEAU
Ecole de technologie supérieure

INTRODUCTION

Worldwide, startups and very small entities (VSEs), that is, enterprises, organizations, departments, or projects having up to 25 people, are major contributors of valuable products and services. In Europe, for instance, as illustrated in Table 1 on the next page, more than 92 percent of enterprises have fewer than 10 employees. Another 6.5 percent have between 10 and 49 employees.

In Canada, close to 98 percent of businesses are small businesses with fewer than 50 employees. About 32 percent

of these have between one and 19 employees (Statistics Canada 2008). These small businesses spend about 5.8 percent of their revenues on research and development (R&D). The standard and guides presented in this article have been specifically developed with the needs of startups and VSEs in mind.

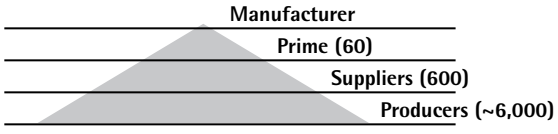
Now, more than ever, system integrators depend on their many suppliers to deliver subsystems that meet evolving requirements correctly, predictably, rapidly, and cost effectively. The supply chain of a large system often has a pyramidal structure. If an undetected defect is left in a low-level component, it may remain undetected once it has been integrated into a higher-level component. For example, as illustrated in Figure 1, a large manufacturer integrated into one of its products a component produced by one of its lowest-level suppliers that contained an undetected software error. This defective component cost the manufacturer millions of dollars. A vast majority of these low-level suppliers are VSEs.

TABLE 1 Size of enterprises in Europe (Moll 2013)

Type of enterprise	Number of employees	Annual turnover (EURO)	Number of enterprises (% of overall)	Number of enterprises
Micro-enterprises	1-9	≤ 2 million	92.2%	19,968,000
Small enterprises	10-49	≤ 10 million	6.5%	1,358,000
Medium enterprises	50-249	≤ 50 million	1.1%	228,000
SMEs, total	87,100,000		99.8%	21,544,000*
Large enterprises	> 250	> 50 million		
Large enterprises, total	42,900,000		0.2%	43,000

* Independent companies only, excluding legally independent companies that are part of large enterprises.

FIGURE 1 Example of the supply chain of a large manufacturer (adapted from Shintani 2006)



Essentially, startup software companies are organizations without an established product, customer base, or revenue stream. Some of the attributes of these enterprises are the following (adapted from Ruokolainen 2007; Nambisan 2002; Sutton 2000):

- Small
- Relatively young and inexperienced personnel
- Focus on outward-looking activities: getting the product into the marketplace
- Entrepreneur driven
- Technology oriented
- Few tangible assets
- Limited access to investments and loans
- Limited cash flow
- Focus on managing human resources judiciously and gaining valuable experience in product coding and testing
- Lacking discipline in product development tasks
- Projects typically of low complexity and limited scope
- Lacking process rigor, but without serious consequences, owing to the low complexity and highly resource-intensive nature of most of the projects
- Successfully evolve to star status only if other development tasks (e.g., version control, domain knowledge reuse) are performed, possibly along with the adoption of additional process control measures, which calls for investment in the process discipline by both utility developers and expert coders

A typical software startup VSE would most likely be at the bottom of the pyramid illustrated in Figure 1. Then, as the startup grows, it might climb up the pyramid and eventually become a large manufacturer. A VSE might also decide not to be part of the pyramid of a supplier and remain a standalone organization. This is the type of startup described in this article.

Since most standalone VSEs developing software are, by definition, small and aiming to grow, putting in place a minimal project management process and a software development process can provide a solid basis on which to establish the enterprise and enable it to accommodate a growing number of customers and hire new employees.

ISO/IEC 29110 is a new standard aimed at helping VSEs develop their products better, faster, and cheaper (Laporte, O'Connor, and Fanmuy 2013; O'Connor and Laporte 2014). The purpose of this article is to present

the development of social networking-oriented trip-planning software using the ISO/IEC 29110 standard as a reference. The project, which was carried out by a team of two individuals (co-authors of this article), resulted in an initial release of the software that included 25 functional requirements. Since the founders of the startup were already employed on a full-time basis, the software development was carried out on a part-time basis over a period of one year and required nearly 1,000 hours of effort. This was the first time the new ISO/IEC 29110 standard had been used by a startup information technology (IT) business involving two individuals.

The objective of the website, called SwiceTrip, is to help a traveler throughout the life cycle of a trip, from the initial planning stage to the ongoing monitoring and sharing of the travel experience. The site offers an attractive and easy-to-use interface to allow users to create travel itineraries and share useful information with fellow travelers, as well as keep track of the details of the trip, such as dates, names, hotels, and so on. The description of a trip can include cities, activities, accommodations, and photos, all of which can be accessed by the traveler's friends, who can also view its history. This small startup organization offers their travel services free of charge. The revenues will come from the advertising displayed on the site.

In the next section, the authors present an overview of the ISO/IEC 29110 standard and the guides used by this startup, followed by a description of the approach adopted to perform the management and engineering activities required by ISO/IEC 29110. They then present the results obtained and discuss the positive outcomes of the method used, as well as areas for improvement. Finally, the authors present the lessons learned and a list of recommendations for the use of ISO/IEC 29110 and the deployment packages developed to help implement the standard.

OVERVIEW OF THE ISO/IEC 29110 STANDARD

Many international software engineering standards, such as ISO/IEC/IEEE 12207 (ISO 2008), have been developed to capture proven software engineering practices. Unfortunately, these standards were not written for very small development organizations and are consequently difficult to apply in such settings (Laporte, Alexandre, and Renault 2008a).

TABLE 2 Road map of the generic profile group (adapted from ISO 2011d)

Generic profile group			
Entry	**Basic**	**Intermediate**	**Advanced**

An ISO working group, ISO/IEC JTC1 SC7 Working Group 24, has been mandated to develop a set of international standards and technical reports to provide software development VSEs with a four-stage road map, as illustrated in Table 2. These stages are also called "profiles." The four profiles are: Entry, Basic, Intermediate, and Advanced. The VSEs targeted by the Entry profile are startups and VSEs working on small projects (at most six person-months of effort). The Basic profile describes the software development practices of a single application by a single project team. The Intermediate profile targets VSEs developing multiple projects within the context of the organization taking advantage of them. The Advanced profile targets VSEs wanting to sustain their growth as independent competitive software development businesses.

Since a vast number of VSEs are not involved in the development of safety-critical software and because this type of software is often developed using domain-specific standards (e.g., DO-178 for aerospace, ISO 26262 for automobile), WG24 decided to focus initially on VSEs not involved in the development of safety-critical software. A VSE developing safety-critical software could use ISO/IEC 29110 and add the requirements imposed by a domain-specific standard.

There is a wide range of development approaches designed for organizations developing software, as illustrated in Figure 2 on the next page on two axes. On the horizontal axis, from left to right, is the level of ceremony, from a low ceremony approach with little documentation (e.g., the agile approach) to a high ceremony approach with comprehensive documentation (e.g., the plan-driven CMMI® approach). On the vertical axes are the approaches based on the level of risk. At the top of this axis is a low-risk linear project using a waterfall approach, while at the bottom of the axis is a risk-driven project using an iterative approach. As the authors explain next, ISO/IEC 29110 is positioned at about the intersection of the two axes.

ISO/IEC 29110 is not intended to dictate the use of a particular life cycle, such as waterfall, iterative, incremental, evolutionary, or agile. The target audiences of this standard and the associated technical reports are described in Table 3. Briefly, the content of the five ISO/IEC 29110 documents described in Table 3 (adapted from ISO 2011b) is as follows:

- ISO/IEC TR 29110-1 defines the terms common to the set of ISO/IEC 29110 documents. It introduces processes, life-cycle and standardization concepts, the taxonomy (catalog) of ISO/IEC 29110 profiles, and the ISO/IEC 29110 series.
- ISO/IEC 29110-2 introduces the concepts for systems and the standardized software engineering profiles for VSEs. It establishes the logic behind the definition and application of profiles, and it specifies the elements common to all the profiles (structure, conformance, assessment).
- ISO/IEC TR 29110-3 defines the process certification scheme, the assessment guidelines, and the compliance requirements needed to achieve the purposes of the defined profiles. ISO/IEC 29110-3 also contains information that can be useful to developers of certification and assessment methods, and also developers of certification and assessment tools.
- ISO/IEC 29110-4 provides the specifications for all the profiles in one profile group that are based on subsets of appropriate standards elements.
- ISO/IEC TR 29110-5 provides a management and engineering guide for each profile in one profile group.
- All ISO/IEC 29110 technical reports (TR) are available at no cost from ISO. A few countries have already translated the English set of ISO/IEC 29110 documents into Spanish, Portuguese, French, and Japanese. Brazil, Japan, Peru, and Uruguay have adopted ISO/IEC 29110 as a national standard.

FIGURE 2 Positioning of ISO/IEC 29110 (adapted from Kroll and Kruchten 2003)

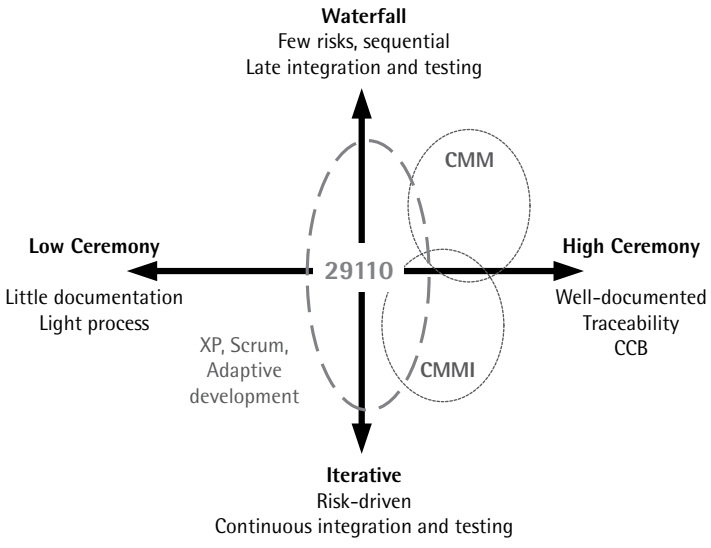


TABLE 3 ISO/IEC 29110 target audience (ISO 2011c)

ISO/IEC 29110	Title	Target audience
Part 1	Overview	VSEs and their customers, assessors, standards producers, tool vendors, and methodology vendors
Part 2	Framework	Standards producers, tool vendors, and methodology vendors Not intended for VSEs
Part 3	Certification and assessment guide	VSEs and their customers, assessors, accreditation bodies
Part 4	Profile specifications	VSEs, customers, standards producers, tool vendors, and methodology vendors
Part 5	Management and engineering guide	VSEs and their customers

Overview of the ISO/IEC 29110 Basic Profile

Since the Basic profile was used by the startup in this study, the authors present an overview of its structure in this section. Then, the ISO/IEC 29110 project management process is presented, followed by the project management tasks conducted during the development of the software. Finally, the authors present an overview of the engineering process of ISO/IEC 29110 used to develop the website.

The Basic profile of ISO/IEC 29110 is divided into two processes, as illustrated in Figure 3 on the next page: a project management (PM) process, and a software

implementation (SI) process. Each process is composed of a few activities and tasks, and the documents to be produced.

As illustrated in Figure 3, the customer's statement of work (SoW) is used to initiate the PM process. The project plan is used to guide the execution of the software requirements analysis, software architectural and detailed design, software construction, software integration and test, and product delivery activities. Verification, validation, and test tasks are included in the SI process. The PM process closure activity delivers the software configuration (i.e., a set of software products) and then obtains the customer's acceptance to formalize the end of the project.

For illustration purposes, one task of the project planning activity is expanded in Table 4. The roles involved in the task are listed on the left-hand side of the table. The PM and the customer (CUS) are involved in these two tasks.

FIGURE 3 Processes and activities of the ISO/IEC 29110 Basic profile (adapted from Laporte and O'Connor 2014b)

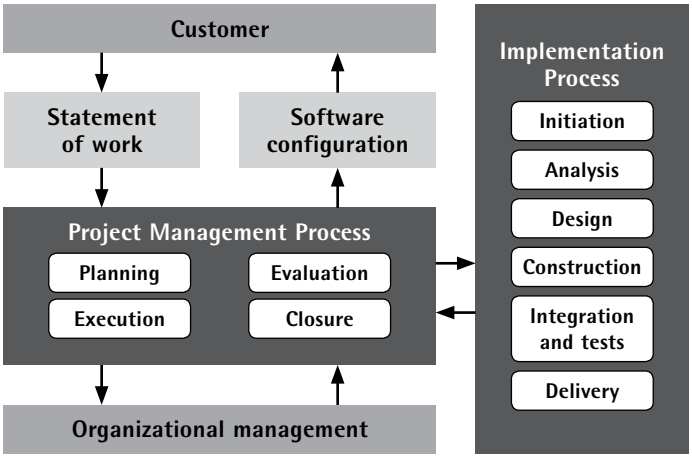


TABLE 4 Example of two tasks of the project planning activity (ISO 2011c)

Role	Task	Input products	Output products
PM CUS	PM.1.2 Define with the customer the delivery instructions of each of the deliverables specified in the statement of work	Statement of work (reviewed)	Project plan – Delivery instructions

ISO/IEC 29110 Project Management Process

Figure 4 on the next page shows the flow of information among the four activities in the PM process of the ISO/IEC 29110 standard, including the most relevant work products and the relationship between them. Each activity includes a description of tasks, roles, inputs, and outputs. The notation used to document the processes presents the activities sequentially, although it is possible to use other development approaches, such as iterative, incremental, evolutionary, or agile.

WEBSITE PROJECT PLANNING

Since this project was aimed at developing a website for travelers, it did not initially involve "real" customers. Friends were consulted to collect and validate the functionalities that such a site could offer. Then, the two-person development team played the role of a client and prepared an SoW with the following high-level functionalities:

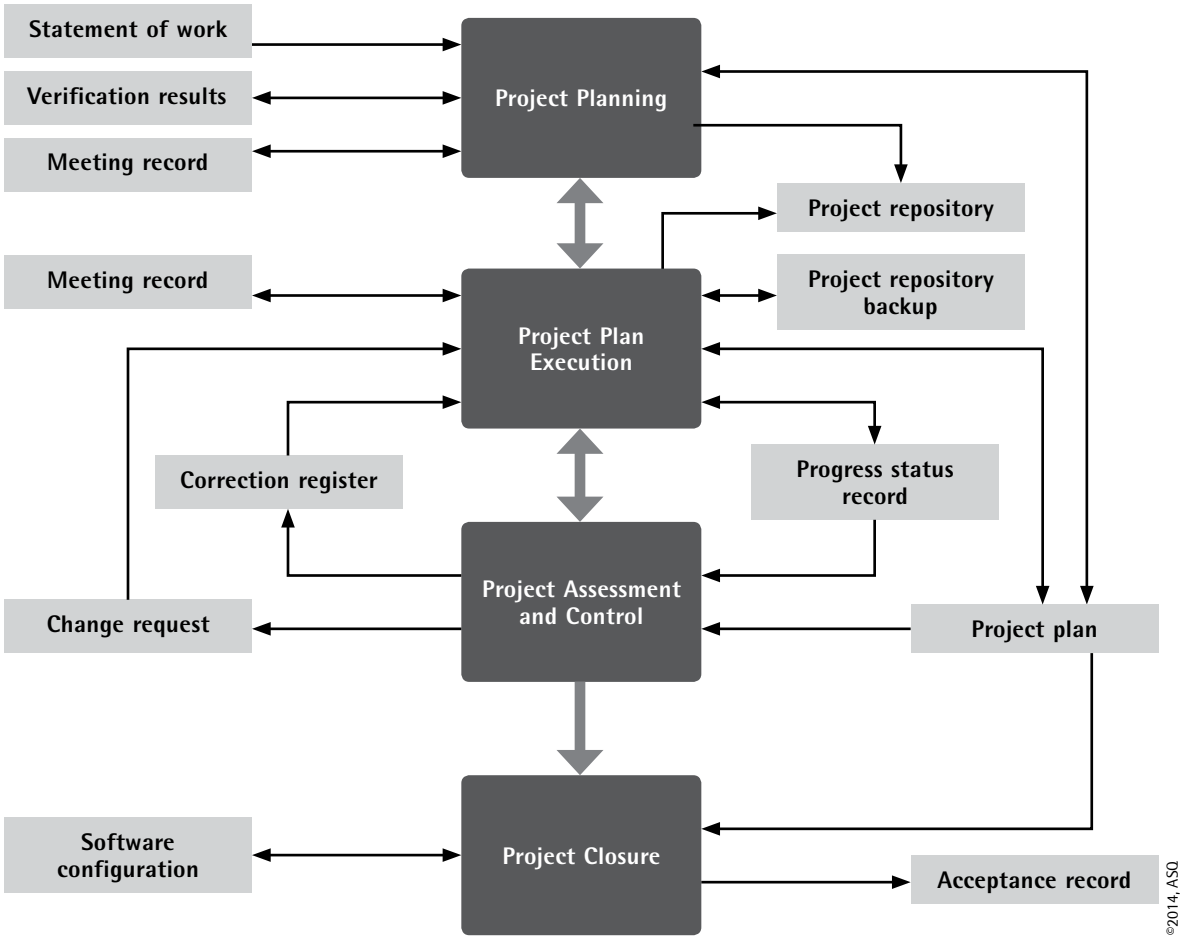
- User management
- Travel management
- Trip viewing
- Travel research
- Collaboration on a trip
- Website administration

Table 5 on the next page lists the user management features required for high-level functionality, with their estimated level of effort and their priority. A high priority is associated with the functionalities considered essential for the first version of the product.

The main nonfunctional requirements, also called performance requirements, of this project are:

- Security: The new Web application will meet the standards of the industry with respect to Web security.
- Performance: The website will provide a response time of less than two seconds.
- Reliability: The proposed infrastructure will ensure at least 99 percent availability.
- Audit: The website will keep track of user activities (who did what and when).

FIGURE 4 Project management process of the ISO/IEC 29110 Basic profile (ISO 2011c)



List of Deliverables

Figure 5 on the next page depicts the delivery instructions form, which lists the artifacts that will be delivered at the end of the project. The delivery instructions form is one element of the project plan; it is approved by the customer when the customer and the project manager approve the project plan.

Project Budget

Table 6 on the next page lists the top-level cost elements and an estimate of each for the development of the first version of the SwiceTrip website. To illustrate the size of the project, effort estimates were expressed in dollars. The only cost elements that were not performed by the two-member development team were the graphic design task and the communication task. The costs listed in Table 6 were calculated in order

TABLE 5 User management functionalities, with their estimated level of effort and priority

User management features required for high-level functionality	Estimated level of effort	Priority
Add a user	High	High
Modify a user	Medium	High
Authenticate a user	Low	High
Disconnect a user	Low	High
Consult a user	High	High
Delete a user	Low	Low
Consult the portfolio of a user's trip	Medium	High
Consult the recent changes made by a user	High	Medium
Send a private message to a user	Medium	Low

FIGURE 5 Delivery instructions form

Project name or name of customer: SwiceTrip

Prepared by: Christian Mineau

Date (yyyy-mm-dd): 2014-03-19

Identification of deliverables (e.g., hardware, software, and documentation):

Software components	Version information	Date accepted by customer	Signature of customer
Project plan	1.2	2013-03-25	
Specifications document	1.1	2013-05-08	
Architecture and design document	1.1	2013-06-12	
Traceability matrix	1.0	2013-12-12	
Change requests	1.0	2013-12-20	
Correction register	1.0	2013-12-20	
Test plan	1.0	2013-11-01	
Test results	1.0	2013-12-20	
Product operation guide and maintenance document	1.0	2013-12-20	
Website deployed and in operation Domain name: "swicetrip.com"	1.0	2013-12-15	

Delivery requirements:

- The website must be hosted by a commercial organization and accessible via the Internet.
- All documents must be in Microsoft Word format.

Signatures:

Project Manager **Customer or Customer Representative**

Date (yyyy-mm-dd): 2014-03-19

to illustrate the amount of effort required to develop and host the website.

The SoW document was used as an input to the drafting of the project plan proposed by the ISO/IEC 29110 standard. The main requirements of the project were extracted from the SoW, in order to define the scope of the project and describe the overall product. In the initial step, the roles of the two stakeholders in the project were identified and the responsibilities of each of them were documented. The assignment of the roles, as described in ISO/IEC 29110, for each of the team members was documented in a table on the project plan (see Table 7 on the next page).

The ISO/IEC 29110 Basic profile management and engineering guide lists the documents that have to be developed during a project and their typical content. Table 8 on the next page reveals which team member was either an author or a reviewer of each document produced during the project.

To initiate the development of the project plan, the authors began by prioritizing the features listed in the SoW, and then converted them into a set of requirements and assigned a unique identification number to each requirement. They subsequently estimated the effort required to perform the tasks listed in ISO/IEC

TABLE 6 SwiceTrip project budget (in Canadian dollars)

Cost element description	Cost
Software development (500 hours x 2 people x $90/hour)	$90,000
Graphic design (50 hours x $75/hour)	3,750
Communication (50 hours x $75/hour)	3,750
Hosting of the website	600
Legal fees associated with the creation of the enterprise	900
Miscellaneous	500
Total	**$99,250**

TABLE 7 Allocation of ISO 29110 roles to the two-member team

Role	Identification of team member
Analyst	A
Designer	B
Programmer	A/B
Project manager	B
Technical leader	A
Work team	A/B

29110. To prepare the schedule, the two-person team estimated that they could invest about 10 hours per week on the development project. The effort required to perform each activity was estimated based on their experience. They realized that it would not be possible to expend more effort per week, since they were both already employed full time.

To help with project planning, the team used an open-source GanttProject software tool (http://www.ganttproject.biz). With this tool, they developed a work breakdown, estimated tasks, identified dependencies between tasks, and assigned a resource to each task.

The next part of the project planning focused on the budget (i.e., human resources, office equipment, and tools), and also included risk management. The risks were documented, and a qualitative assessment of their probability of occurrence and potential impacts was performed. A brief risk mitigation plan was developed as well. The five main risks associated with the project are presented in Table 9 on the next page.

Management of the versions of the artifacts is documented in the project plan, and includes the tree, the location, the access mechanism, the backup/restore process, and artifact identification. The open source software Apache Subversion SVN was used to manage the artifact versions. Figure 6 on the next page shows two pages of the website developed.

EXECUTION OF THE PROJECT PLAN

Once the project plan was finalized, contrary to what happens in many VSEs and even in larger organizations, the plan was used to guide the execution of the project. Time sheets were used to collect the effort, in person-hours, associated with each task listed in the project plan.

TABLE 8 Distribution of responsibilities in the two-member team

Title of document	Main author	Reviewer (if applicable)
Change request	A	B
Correction register	B	A
Maintenance documentation	B	A
Meeting record	A	
Product operation guide	B	B
Progress status record	B	
Project plan	B	A
Project repository	B	
Project repository backup	B	
Requirements specification	A	B
Software	A/B	
Software components	A/B	
Software configuration	A/B	
Software design	B	A
Software user documentation	A	B
Statement of work	A	B
Test cases and test procedures	A	B
Test report	A	
Traceability record	B	A
Verification results	A/B	
Validation results	A/B	

TABLE 9 Five main project risks

Risk description	Probability	Impact
Underestimation of the complexity of the software components to be developed	Medium	High
Underestimation of the effort required to develop the components	Medium	Medium
Loss of a team member (e.g., unavailable owing to a high workload at work)	Low	High
Arrival of a competing product on the market	Low	Medium
Complexity of integrating external components (e.g., Google Maps)	Low	High

FIGURE 6 Example of pages of the website (in French) (www.swicetrip.com)

EVALUATION AND CONTROL OF THE PROJECT

ISO/IEC 29110 includes an evaluation and monitoring activity to assess the performance of the plan against the commitments documented.

Progress meetings were held during the first week of each month. As described in ISO/IEC 29110, during these meetings the progress on each of the tasks defined in the project plan was reviewed, and a completion percentage was determined for each task. All the risks identified in the project planning phase were reviewed to assess them, mitigate their impact, or both. A brief project progress report was documented by the project manager. This report included a list of tasks, their progress expressed as a percentage, their status (e.g., OK, risky, in danger), and a comment explaining the reason for the status when a task was assigned either "risky" or "in danger" status.

If a task deviated significantly from the project plan, corrections were proposed to return as much as possible to the initial project plan. Modifications were documented using the GanttProject software tool. The corrections were registered in the project progress report and monitored at monthly review meetings.

SOFTWARE QUALITY ASSURANCE

The two members of the team agreed that all the tasks described in ISO/IEC 29110 would be carried out in the project. For example, verification tasks, such as peer reviews, were performed on the requirement specification and architecture documents. The team used the desk check to review their documents.

The desk check (Wallace 1996), sometimes referred to as the "pass around," is not described in published standards. This type of peer review is inexpensive and easy to implement in any organization. It can be used to detect anomalies and omissions, improve a document, or present and discuss alternative solutions. This type of review is used to review low-risk documents or when the schedule does not include a more comprehensive review, such as a walk-through or inspection, as set out in the IEEE-1028 standard (IEEE 2008). Figure 7 on the next page shows a schematic view of the desk check review.

The following checklist, called a "generic checklist," was used to review most of the documents (adapted from Gilb and Graham 1993):

- GD1 (COMPLETE). All information relevant to the purpose of the document must be included or referenced.
- GD2 (RELEVANT). All information must be relevant to the purpose of the document and to the section in which it resides.
- GD3 (BRIEF). Information must be stated succinctly.
- GD4 (CLEAR). Information must be clear to all checkers.
- GD5 (CORRECT). Information must be free of technical errors.
- GD6 (CONSISTENT). Information must be consistent with all other information in the document and its source documents.
- GD7 (UNIQUE). Ideas should be stated once only and thereafter referenced.

A template to record anomalies detected during reviews, which would apply to documents required by ISO/IEC 29110, was developed early in the project. Figure 8 on the next page shows an example of this template.

Since the team consisted of only two members, the author of a deliverable had to make the corrections, while the other member was responsible for reviewing the document and making a list of the anomalies detected. The reviewer had the responsibility of identifying defects and indicating the outcome of the review as either "accept the document as is" or "make corrections." Some documents, such as the requirement specification document, required several rounds of reviews before the reviewer accepted the document. The various versions of the documents were managed using a software versioning tool.

ISO/IEC 29110 SOFTWARE IMPLEMENTATION PROCESS

As discussed previously, the second process in ISO/IEC 29110 is entitled, "Software implementation process." Figure 9 on the next page shows the flow of information between the six activities associated with this process, including the most relevant work products and their relationships.

Next, the authors present the activities of the software implementation process of ISO/IEC 29110, except for the first one, entitled, "Software process initiation," the main objective of which is to present the software developers with the project plan, along with their roles, their tasks, and the tools they will use.

Software Requirements Analysis

As defined in ISO/IEC 29110, the software requirements analysis activity analyzes the requirements agreed to by the customer, and establishes the validated project requirements. This activity provides (ISO 2011e):

- Review of the project plan by the work team to determine task assignment
- Commitment to the project plan by the work team and project manager
- Establishment of an implementation environment

The SoW is an important input to the development of the specifications document, and includes a general description of the product and the high-level

FIGURE 7
Overview of the desk check review

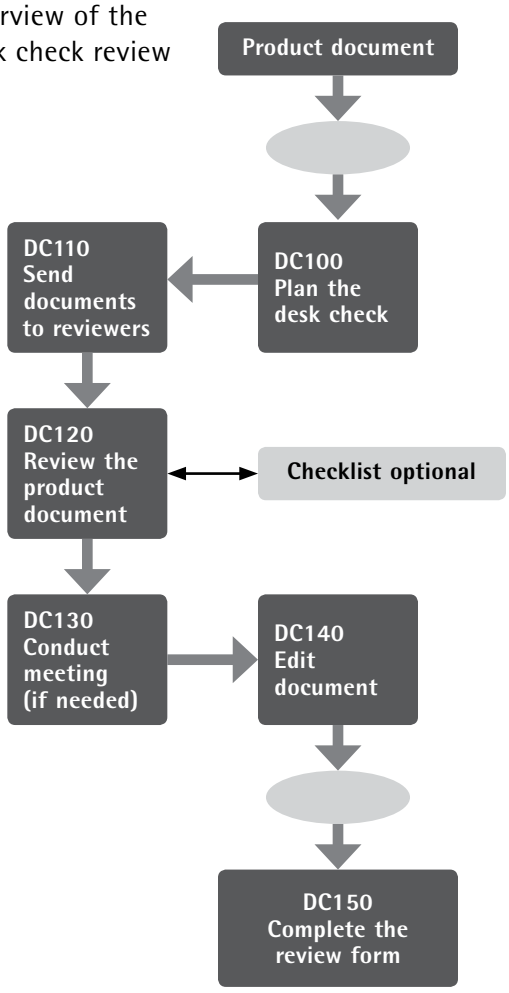
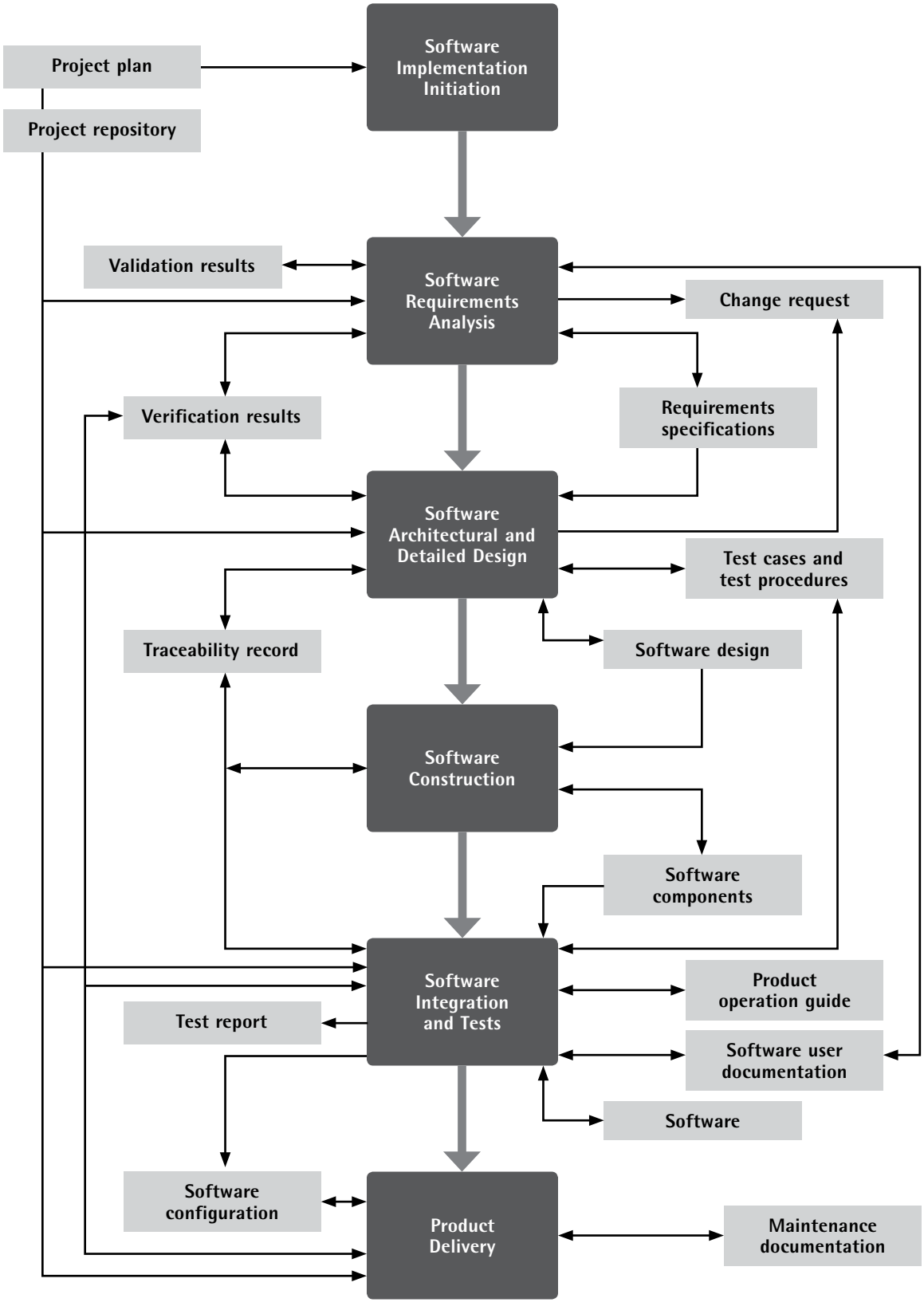


FIGURE 8 Anomaly registration form

Document title: Requirements Specifications					**Date of review:** 2014-05-02	
Document version: 0.5					**Name of author of document:** Charles Hébert	
					Name of reviewer: Christian Mineau	
Item ID	**Page number**	**Section number**	**Checklist ID**	**Reviewer initials**	**Description (key words)**	**Author's remark**
1	8	2.1	GD1	CM	The actor "Administrator" is missing from the list of actors. Add this actor.	Done
...	...	...	...	...	...	...
					Total number of anomalies	11
					Effort to review the document (hours)	2.5
					Effort to correct the document (hours)	4
					Decision: [ ] Accept as is [X] Make corrections [ ]	

FIGURE 9 Software implementation process of ISO/IEC 29110 (ISO 2011c)



characteristics of the software to be developed. The final version of the SoW had 28 pages, and the features described in it were described in more detail in the specification document. A template to capture the requirement specifications as use cases was developed by the team.

When the requirement specifications were documented, many questions were raised about how to meet the initial requirements, which sparked a great deal of discussion and allowed the team to identify the features that were the most complex to produce. Also, they realized that there were elements that had not been specified in the SoW: e.g., the development of a management console for the website. The analysis activity was very useful during the monthly project reviews. Some features were removed from the project, owing to their complexity, and others were added. The final requirement specification document contained 48 pages.

The requirement specifications were verified and validated several times to ensure their accuracy and testability, as well as their consistency with the product description. The results of the verification and validation tasks revealed major anomalies and reduced the amount of rework effort that would have been required downstream. Data on the verification and validation tasks and on the rework effort are presented next.

Software Architecture and Detailed Design

As defined in ISO/IEC 29110, the software architectural and detailed design activity transforms the software requirements into the software system architecture and detailed software design. This activity provides (ISO 2011c):

- Work team review of the project plan to determine task assignment
- Design of the software architecture, software components, and associated interfaces
- Detailed design of the software components and interfaces
- Work team review of the requirement specifications
- Verification of the software design and correction of defects
- Verification of the test cases and test procedures for integration testing
- Traceability of the software requirements to the software design, test cases, and test procedures
- Design of the products and documents under version control

This activity began following correction of the anomalies detected during the verification and validation of the requirement specification document. The architecture and the design document provide an overview of the software architecture and the details of each of its components, and of the development standards that were used during the construction activity. The database was also modeled during the architecture development phase.

A template, available in the architecture deployment package (Guillemot and Champagne 2009), was used to develop the software architecture and detailed design document. The resulting document consisted of three sections: the high-level software architecture, the detailed software design, and the external interfaces. The final version of the architecture and detailed design document contained 33 pages.

During this activity, a traceability matrix was developed, as described in the ISO/IEC 29110, to connect the software requirements, defined in the requirement specification document, to the software components. Since, in most projects, requirements, which are defined in the requirement specification activity, are never finalized at the end of this activity, a traceability matrix is very useful—one advantage being the possibility of rapidly identifying the software components impacted when software requirements are modified, added, or deleted during a project. A fragment of the project traceability matrix is presented in Figure 10 on the next page.

Software Construction

As defined in ISO/IEC 29110, the software construction activity develops the software code and data from the software design. This activity provides (ISO 2011c):

- Work team review of the project plan to determine task assignment
- Work team review of the software design to determine the software construction sequence
- Coded software components and unit testing applied
- Traceability between the software components and the software design

FIGURE 10 Fragment of the project traceability matrix

Traceability Matrix

Date (yyyy-mm-dd): 2012-11-29 Version Number: 1.3

Project Title: SwiceTrip.com

Name (Print) Signature Date (yyyy-mm-dd)

Verified by: _____

Approuved by: _____

Serial Number	Requirement Identififfication	Requirement Description	Identification of Architecture Component	Component Description	Test Case Numbers
1	E1	Add a User	C6	Profile Component	1,1; 1,2; 1,3; 1,4; 1,5; 1,6; 1,7; 1,7,1; 1,7,2; 1,7,3
2	E2	Modify a User	C6	Profile Component	2,1; 2,2; 2,3; 2,4; 2,5; 2,6; 2,7; 2,8; 2,9; 2,10; 2,11; 2,12; 2,13; 2,14; 2,15
3	E3	Authenticate a User	C6	Profile Component	3,1; 3,2; 3,2,1; 3,2,2; 3,2,3; 3,2,3; 3,2,4; 3,2,5

This activity began following verification and validation of the architecture and design documents. Components to develop (i.e., code), were assigned to the two-person team in accordance with the project plan. Since both the requirements and the architecture had been submitted for review, the construction phase went very smoothly.

Software Integration and Testing

As defined in ISO/IEC 29110, the software integration and testing activity ensures that the integrated software components satisfy the software requirements. This activity provides (ISO 2011c):

- Work team review of the project plan to determine task assignment
- Understanding of test cases and procedures, as well as the integration environment
- Integration of software components, correction of defects, and documentation of results
- Traceability of requirements and design to the integrated software product
- Documentation and verification of the operational and software user documentation
- Verification of the software baseline

A test plan was developed to verify the site's features, as documented in the requirement specification document, prior to the publication of the website. A fragment of the test plan is presented in Table 10. For each test case, the authors defined its relationship with the use case, a test description, the steps involved, and the expected results.

To manage the defects detected, the team used a tracking tool. This software allowed them to take an inventory of the problems found during the integration and testing activity, to track problems and classify them, and to determine a priority for each defect found. In this project, the open source Bugzilla software tool (http://www.bugzilla.org) was used to manage the defects.

The test report presents the results of the tests carried out using the test plan. These results are used to illustrate the number of problems found and the progress of the resolution of anomalies. The test plan includes 112 cases that have been successfully completed, with the exception of test cases connected to one type of defect: the wrong format for a date manually entered by a user. Since this defect was classified as minor, the decision was made not to correct it during the first development cycle. Table 11 on the next page summarizes the defects classified by their seriousness using the following defect classification:

- Blocker: prevents function from being used, no work-around, blocking progress on multiple fronts
- Critical: prevents function from being used, no work-around
- Major: prevents function from being used, but a work-around is possible
- Normal: makes a function difficult to use, but no special work-around is required

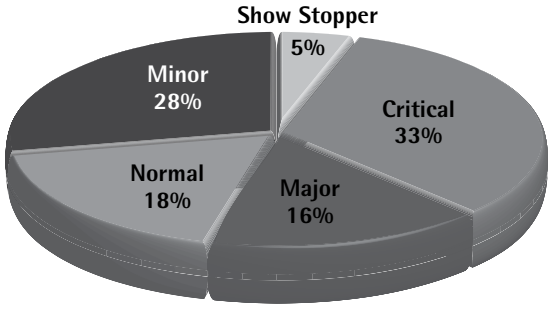
TABLE 10 Fragment of the test plan

Requirement	Test case ID	Test description	Test procedure	Success criteria	Results	Comment	Date of test
R1 – Add a user	1,1	Validation of obligatory fields	1. Click on "Register a user" 2. Click on "I want to subscribe"	A message is displayed beside each field	Successful		2013-11-20
	1,2	Validation of the length of the text in each field	1. Click on "Register a user" 2. Enter the maximum length of the text in each field	No. of characters: • First name: 20 • Name: 20 • Email: 50 • Password: 15 • Confirmation of password: 15	Successful		2013-11-20

TABLE 11 Number and types of defects detected through testing and corrected

Seriousness	No. of defects detected	No. of defects corrected	Percentage of defects corrected
Blocker	3	3	100%
Critical	22	22	100%
Major	11	11	100%
Normal	12	12	100%
Minor	19	6	32%
Total	67	54	81%

FIGURE 11 Percentage of defects detected for each category of defects



- Minor: does not affect function, but the behavior is not normal

Figure 11 illustrates the percentage of defects detected during the execution of the tests for each category of defects.

All defects classified as show stoppers, critical, major, and normal were corrected. The only defects that were not all corrected were the minor ones. These defects do not prevent the successful execution of the website, and will be addressed in a subsequent development phase.

PRESENTATION AND ANALYSIS OF EFFORT DATA

Table 12 on the next page presents the efforts of the team recorded on the timesheets as the number of hours expended for each task. The total effort devoted to this project was 990.5 hours. The effort devoted to prevention, such as the installation of the environment (e.g., server, tools), was 89 hours, and the execution of the tasks took 716 hours. These figures do not include the effort required to review artifacts (60.5 hours) and to correct defects (i.e., rework) (125 hours).

The desk check peer reviews, an evaluation task, detected a total of 39 defects in the documents produced. The defect breakdown is as follows:

- Project plan: 5
- Requirement specifications: 15
- Architecture and detailed design: 14
- Maintenance and product operation guides: 5

There was no effort required for review or correction for initiation activity environments, project management, or deployment (installation of the website with the external host). One can see that the remediation

TABLE 12 Effort to prevent, execute, detect, and correct errors by the two-member team

Task	Prevention (hours)	Execution (hours)	Review (hours)	Correction of defects (hours)
Environment installation (server, work stations, tools)	**89**			
Project plan development		**35**	**3**	**4**
Project plan execution and project assessment and control		**47**		
• Project plan execution		21		
• Project assessment and control		26		
Specification development and prototype development		**199.5**	**7**	**18**
• Statement of work		34	2	13
• Requirements specification		54	2	6.5
• Prototype development		93	3	17
Architecture development		**42.5**	**1.5**	**3.5**
Test plan development		**12.5**	**1**	**2**
Code development and code testing		**361**	**47**	**96.5**
• Home page		94		
• Research		27.5		
• Portfolio		28		
• Trip		78.5		
• City		41		
• Activity		56.5		
• Profile		29.5		
• Administration tools		6		
User guide and maintenance document development		**8**	**1**	**1**
Website deployment		**8.5**		
Project closure		**2**		
Total (hours)	**89**	**716**	**60.5**	**125**

effort (rework), which was greater, was mainly related to the definition of customer requirements (SoW) and the detailed specifications. In fact, a great deal of time was spent on reviewing and correcting specifications before embarking on the other phases, knowing that when more defects are found early in the development cycle, the effort to correct them will be less than if they are detected during testing.

The percentage of prevention task effort was 8.9 percent (i.e., 89 hours/990.5 hours) and the percentage of rework effort was 12.6 percent (i.e., 125 hours/990.5 hours) for this project, which is close to the performance of a Capability Maturity Model (CMM)® maturity level of 3 (Paulk et al. 1993), in a comparison of the authors' results with those attained in a study on the impact of CMM on quality, as illustrated in Table 13 on the next page.

Krasner has published a table, illustrated in Table 14 on the next page, presenting similar numbers (Krasner 1998). Again, one can see that a startup VSE with a rework percentage of 12.6 percent is performing slightly better than a CMM level 3 organization.

The percentage of rework associated with website development is lower than the percentage of rework required in a level 3 organization. The percentage of defects is low because the development team knew that, in most software development projects, a large number of the defects are introduced during the development of the requirements, and that it is much more efficient to detect and correct them in the requirement specification phase than later in the process. Figure 12 on the next page illustrates that, for a U.S. company, nearly 50 percent of software defects are found during the requirement

specification phase. Consequently, the development team put more effort into this activity to minimize the rework effort that would be required during a subsequent phase.

It is not unusual for software projects developed in immature organizations to require rework in the 40 to 50 percent range. For example, the authors of this article have collected data on the cost of software quality (CoSQ) in their environment from professional engineers, managers, and graduate students in the master's degree program in software engineering at a Montréal 7,500-student engineering school, the ETS. As illustrated in Table 15 on the next page, the estimated cost of this rework is about 30 percent of the total development cost. Most of the industrial data were collected at two large multinational enterprises: one involved in the transportation sector and the other in the aerospace sector. The numbers in parentheses indicate how many people responded to the CoSQ survey.

In most startups, the rework or wasted effort for a project similar to this one would have added at least another 400 hours. If one subtracts the numerous interruptions (phone calls, answering personal emails, interruptions from a colleague, discussions in corridors, and so on)

TABLE 13 CMM maturity level and the percentage of rework (Diaz and King 2002)

CMM maturity level	Percentage of rework
2	23.2%
3	14.3%
4	9.5%
5	6.8%

TABLE 14 Relationship between process maturity and the percentage of rework (adapted from Krasner 1998)

Process maturity (characteristic)	CMM maturity level	Percentage of rework
Immature	1	≥ 50%
Project controlled	2	25% - 50%
Defined organizational process	3	15% - 25%
Management by fact	4	5% -15%
Continuous learning and improvement	5	≤ 5%

FIGURE 12 Origin of software defects (Selby and Selby 2007)

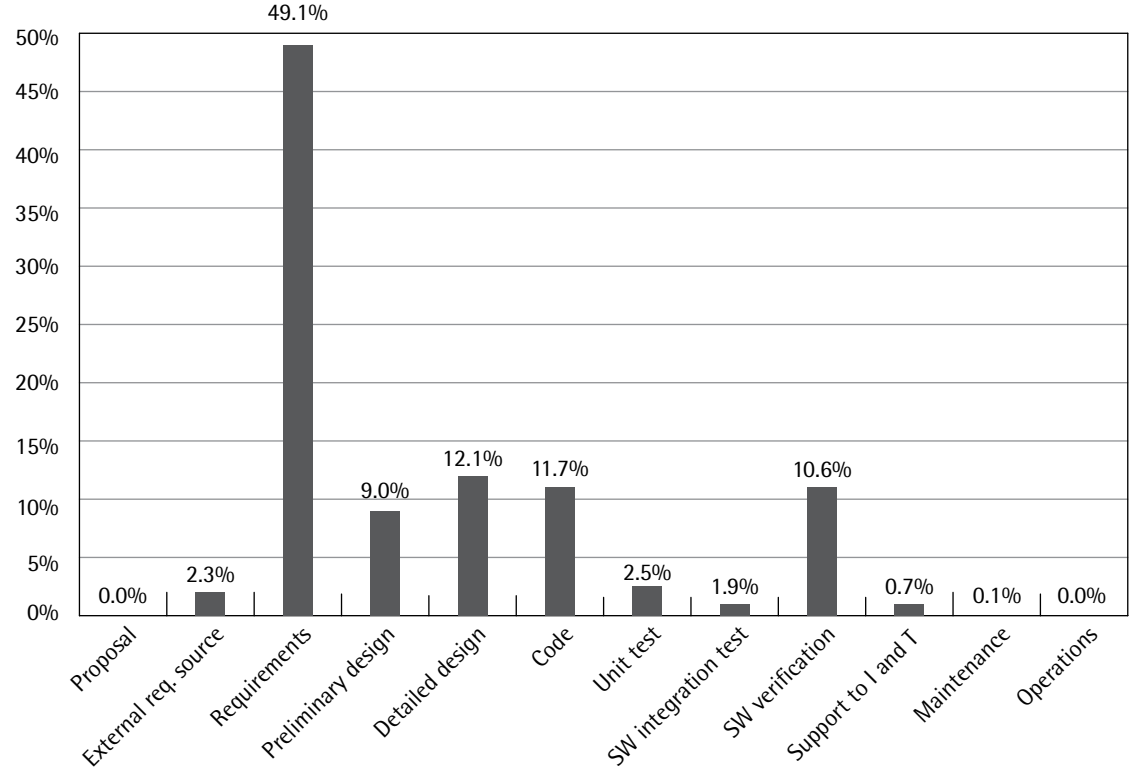


TABLE 15 Cost of software quality data from software professionals and managers (Laporte 2014b)

	Cost of performance	Cost of rework	Cost of appraisal	Cost of prevention	Quality (defects/KLOC)
Site A American Engineers (19)	41%	30%	18%	11%	71
Site A American Managers (5)	44%	26%	14%	16%	8
Site B European Engineers (13)	34%	23%	32%	11%	23
Site C European Engineers (14)	31%	41%	21%	8%	35
Site D European Engineers (9)	34%	34%	26%	7%	17
Course A 2008 (8)	29%	28%	24%	14%	403
Course B 2008 (14)	43%	29%	18%	10%	19
Course C 2009 (11)	45%	30%	14%	11%	48
Course D 2010 (8)	45%	25%	20%	10%	35
Course E 2011 (15)	34%	32%	27%	8%	60
Course F 2012 (10)	40%	31%	20%	9%	55
Course G 2013 (14)	44%	25%	19%	12%	72

FIGURE 13 Improvement data (Dion 1992; Haley 1996)

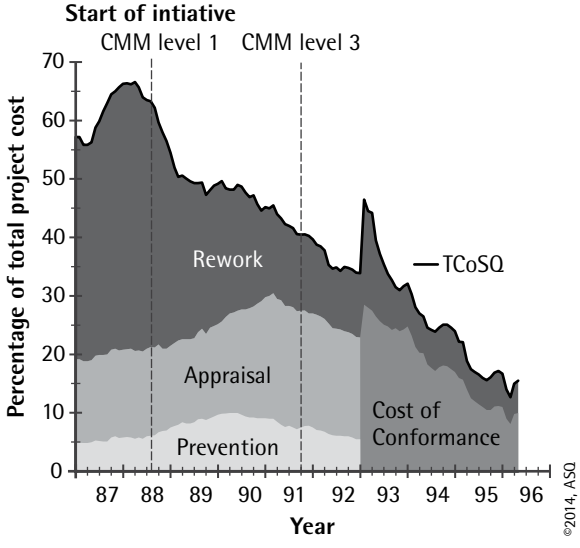
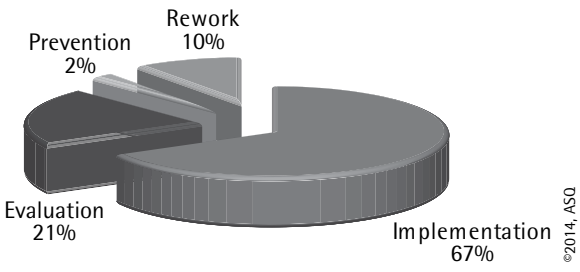


FIGURE 14 Distribution of effort in the 88,000-hour project (Laporte et al. 2012)



from an eight-hour day, this means that for an effort of about six hours per team member per day, the website would have been ready for operation about 30 days later than with a project with only 12.6 percent waste.

From Table 12, one can also calculate the percentage of evaluation effort. Since the two-member team spent 60.5 hours on evaluation tasks, such as the desk check, the percentage of effort expended on evaluation tasks was 6.1 percent.

One can compare the values for this project to the data published a few years ago by Raytheon (Haley 1996). Haley illustrated the relationships between the investments/benefits and the maturity levels of the CMM® for Software (Paulk et al. 1993). The study at Raytheon showed (see Figure 13) that, at level 3, the percentage of rework was about 11 percent and the percentage of effort invested in evaluation tasks was about 7 percent.

In a study performed in a large organization, 1,100 software tasks in a software development project, representing 88,000 hours of effort, were analyzed to measure the cost of quality (Laporte et al. 2012). The distribution of the development costs in the various categories of software quality and implementation is illustrated in Figure 14. The figure also reveals that the cost of rework is 10 percent, the cost of prevention is 2 percent, and the cost of evaluation is 21 percent of the total cost of development. At the time the cost of quality study was performed, this organization was at level 3 of the CMM maturity model.

When compared to the data presented in Tables 13 and 14, the cost-of-quality data (i.e., prevention, evaluation, rework) for the two-member startup enterprise presented in Table 10 positions the startup VSE at a maturity level close to 3. A VSE, using ISO/IEC 29110, can rapidly perform at a high level of maturity and then it can quickly and easily improve its processes.

Maintenance Document

The purpose of this document is to provide information on the operation and maintenance of the SwiceTrip software. This document describes the configuration of the software environments used (that is, the development and production environments), the maintenance procedures (start, stop, and roll), and the monitoring procedures of the system. The maintenance document allows maintainers to support this application. The final version of the maintenance document consisted of 18 pages.

Development Environment

The development environment can be configured on a laptop or a desktop computer. Table 16 lists the software technologies that were used, as well as the version identification of each tool used during the development of the website.

Identification of the version of the technologies used will be helpful whenever the startup wants to modify the software developed. It is very probable that some open source software tools will also have been modified. Knowing the version, the startup will be in a better position to analyze the impacts of the "improved" open source tools when launching improvements to the website software. For example, a new version of an open source tool might not support the features of a previous version, or the code produced with a new version of an open source tool is not compatible with other components of the website software.

LESSONS LEARNED

Even though the project closure activity of ISO/IEC 29110 does not require a lessons learned session to be conducted, the development team decided to hold one, so as to be more effective in the future development of the site's features. The two questions that were used to develop the lessons learned were: 1) What has the team done that should be repeated in a future project? 2) What has the team done that should either not be done or improved in a future project?

Development of a Prototype

The design of a prototype early in the development of the website was a factor in the success of the project, as the prototype enabled the team to identify areas for modification and to obtain comments and recommendations early in the development cycle. In fact, they developed and evaluated a small prototype for each group of website specifications. This approach reduced the overall development costs, as user feedback was considered prior to the coding phase.

Selecting an ISO/IEC 29110 Profile

The authors compared the two profiles of ISO/IEC 29110 available at the start of the project: the Entry profile (ISO 2012) and the Basic profile (ISO 2011c). The Entry profile targets startup VSEs (this VSE was founded less than three years ago), and VSEs with projects involving an effort of six person-months or less. Table 17 shows the differences between the two profiles. At the beginning

TABLE 16 Software tools used

Technology	Version ID
Apache ANT	V1.8.3
Eclipse	Indigo
Glassfish	V3.1
GoogleAdsense	N/A
GoogleMap	v3
Java Persistence API (JPA)	V2.0
JEE	V1.6.0.35
JSF	V2.0
Log4J	v1.2.16
PostgreSQL	V9.1
Putty	0.62
Turtoise SVN	1.7.6
WinSCP	4.3.6

TABLE 17 Comparison between the Entry and Basic profiles of ISO/IEC 29110

Profile	Entry	Basic
No. of project management process tasks	18	26
No. of software implementation process tasks	22	41
No. of roles	3	7
No. of documents to produce	14	22

of the website development project, the two-person team found that the differences between these two profiles were not too great, and they believed the additional workload for the basic profile would be a good fit for this project.

In the middle of the project, the team consulted the public website of ISO/IEC JTC1/SC7 Working Group 24 and discovered a video in French describing ISO/IEC 29110. According to the video, the Entry profile targets startup VSEs, among others. The team wondered whether they should, in fact, have selected the Entry profile for this project, since they were starting a new enterprise. The additional practices and documentation required by the Basic profile could have been set aside for the first iteration of the project. The team decided to complete the development using the Basic profile.

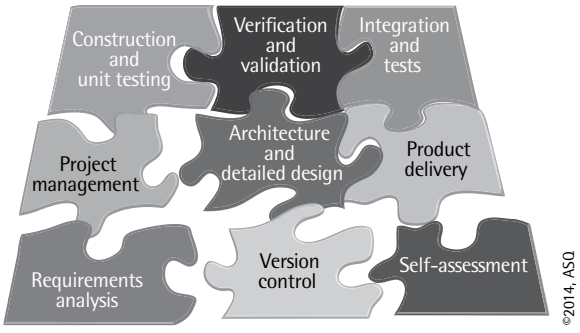
Using Open Source Software

The use of open source software facilitated the execution of many tasks of the ISO/IEC 29110 standard. For example, for project management, a free GanttProject software tool helped define the tasks, the effort, the delivery dates, and the resources to track ongoing development. For the software integration and testing activity, the team chose the Bugzilla software tool, which allowed them to record the defects identified and their descriptions, to define the components affected, to determine a priority for each defect, and to track the corrections for each defect. The version management software tool Apache Subversion SVN (http://subversion.apache.org) was used throughout the project. Finally, a repository was created, using the SVN tool, to manage the documents produced and their versions right from the start of the project.

Using the Tools Developed to Support the ISO/IEC 29110 Standard

The two-person team benefited from the tools developed to support the new standard. As described in Table 3 and the text below it, ISO/IEC 29110 Part 2 (Framework) and Part 4 (Profile specifications) are international standards. Part 4 defines what needs to be done, that is, the specifications of each profile, in this case the Basic profile specifications. This startup VSE could have used this document to define the software processes they needed to develop their website. Instead, the team used Part 5 (Management and engineering guide), which defines how to do what is described in Part 4. In addition to the description of the management and software implementation processes, this document describes the roles and the typical content of each document used as inputs or outputs to the two processes.

FIGURE 15 DPs developed to support the Basic profile



To provide additional guidance to busy VSEs on the implementation of the management and engineering guidelines, a series of deployment packages (DPs) was developed to explain the processes and activities defined in ISO/IEC 29110 in more detail. The typical content of a DP includes a description of processes, activities, tasks, steps, roles, and products, and provides templates, checklists, examples, references, mappings to standards and models, and a list of tools. The mappings show that a DP has explicit links to standards, such as ISO/IEC/IEEE 12207 (ISO 2008), or to models, such as CMMI® for Development (SEI 2010).

As illustrated in Figure 15, this set of DPs was designed to enable a VSE to implement its content, such as version control, without having to implement the complete set of activities of ISO/IEC 29110 all at once. A set of nine DPs was developed and is freely available on the Internet. Finally, all these tools are freely available on the Web in English, French, and Spanish.

RECOMMENDATIONS FOR FUTURE USERS OF ISO/IEC 29110

Select the Appropriate Profile

Future users should take the time to select the profile that is best suited to the characteristics of their project.

At this time, the first two profiles have been published by ISO: the Entry profile (ISO 2012) and the Basic profile (ISO 2011c). Working Group 24 is busy developing the next two profiles: Intermediate and Advanced. Once all four profiles are available, it will become important to choose the appropriate profile for a particular project. The authors recommend using the self-assessment tool available on the website (http://profs.etsmtl.ca/claporte/VSE/Groupe24-menu.html) to help make this choice. For anyone establishing a new company and developing a new product, the authors suggest considering the Entry profile to start with, and advancing to the Basic profile after completion of one or two development iterations. Since the Basic profile is based on the Entry profile, the jump from the Entry profile to the Basic profile should be an easy one.

Select the Right ISO/IEC 29110 Standard Language

The authors recommend using the version of ISO/IEC 29110 that is the most familiar to the VSE principals. The barrier posed by the use of ISO/IEC 29110 in a language that is difficult to understand is considerable. The management and engineering guides are available in English, French, Portuguese, and Spanish.

Take Advantage of Open Source Software Tools

There are software tools available free of charge that can readily implement some of the standard activities and tasks outlined in this ISO/IEC standard. The authors believe the use of these tools, such as GanttProject, Bugzilla, and Subversion, would be of significant value to VSEs in developing their products.

Use the Deployment Packages

When time is an issue, DPs, available on a public website of Working Group 24, can help with the implementation of the most important tasks and activities of the management and implementation processes of ISO/IEC 29110, while the remaining tasks and activities can be implemented gradually using other DPs.

Other documents available on this website are implementation guides, checklists, assessment tools, descriptions of software reviews, videos, and short case studies.

Adapt the ISO/IEC 29110 Standard to the Organization

ISO/IEC 29110 part 5 is a management and engineering guide developed to facilitate the implementation of ISO/IEC 29110 formally defined in Part 4 (that is, the profile specifications). It is possible, for example, to adapt the terminology of the management and engineering guide to the terminology used in a particular organization. Some of the documents described in the guide can also be combined or split.

Although the tasks and activities described in the guides are listed sequentially, a VSE can, and should, adapt them to other approaches as needed, such as iterative, incremental, evolutionary, or agile. DPs to illustrate the use of the guide using an agile approach are currently under development. DPs are also under development for the application of ISO/IEC 29110 in specific domains, such as gaming, medical, and automotive.

An organization seeking ISO/IEC 29110 certification will not have much latitude for eliminating tasks or documents of the standard. Part 4 describes the requirements for the audit process. An ISO/IEC 29110 document describing the certification is under development (ISO 2014c).

CONCLUSION

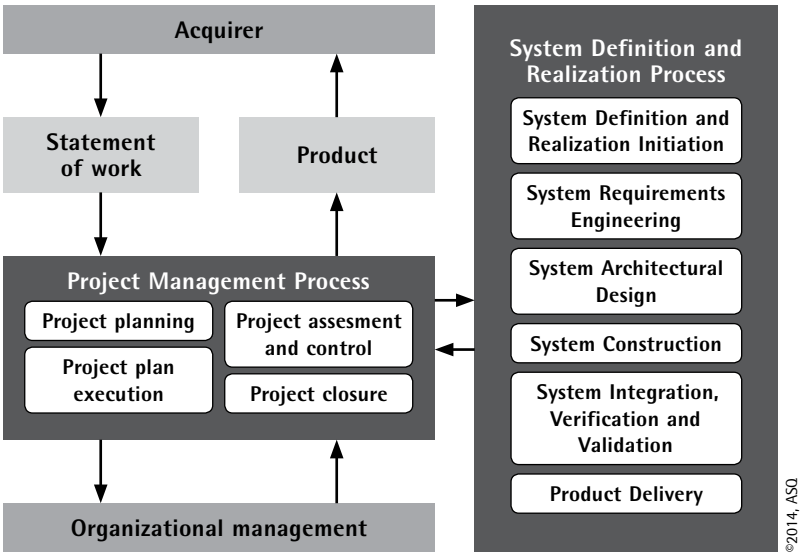
This software was developed in accordance with the effort planned for the project and delivered on time, according to the schedule set while developing the baseline software requirements. The authors have shown with this project that it is possible to effectively plan and execute a software development project using the management and engineering guides of the ISO/IEC 29110 standard.

The authors recommend the use of ISO/IEC 29110 for all VSEs wishing to improve their management and software engineering practices. It provides small organizations, and larger organizations with small projects, the ability to use a framework adapted to their needs. A VSE can gradually increase its performance and enhance its image by developing quality software while meeting the needs, timetable, and budget of the client. The application of ISO/IEC 29110 has resulted in a very low percentage, that is, 12.6 percent, of wasted effort, which is similar to the figures encountered in high-maturity organizations. Finally, VSEs seeking investors, partners, or customers could apply for ISO/IEC 29110 certification.

This project has demonstrated that it is possible to properly plan and execute a project, and develop a software product, using the proven software practices documented in ISO/IEC 29110, without interfering with the creative process during the development of the website. Those who think of standards as a burden, unnecessary overhead, or a threat to creativity should look at this startup project and revisit their assumptions.

There is a systems engineering version of ISO/IEC 29110 (Laporte and O'Connor 2014a; Laporte, Houde, and Marvin 2014; ISO 2014a; ISO 2014b) for those interested in the development of systems with hardware components. The two processes involved in developing these systems, along with the activities of the Basic systems engineering profile, are illustrated in Figure 16.

FIGURE 16 Processes of the ISO/IEC 29110 systems engineering Basic profile (Laporte and O'Connor 2014a)



There are similarities between the software and the system Basic profiles. In fact, the project management process of the two profiles is almost identical. The engineering processes of both profiles, at a high level (i.e., at the activity level), are quite similar. They differ mostly at the task level. In the systems engineering profile, if the system requires the development of software, then the systems engineering profile suggests using the Basic software profile. An Entry profile, for systems engineering, is currently being finalized and should be published by ISO in 2015.

REFERENCES

Diaz, M., and J. King. 2002. How CMM impacts quality, productivity, rework, and the bottom line. *Cross Talk: Journal of Defense Software Engineering* (March):9-14.

Dion R. 1992. Elements of a process improvement program. *IEEE Software* 9, no. 4 (July):83-85.

Gilb T., and D. Graham. 1993. *Software inspection.* Boston: Addison-Wesley.

Guillemot, F., and R. Champagne. 2009. Software architectural and detailed design deployment package. ISO/IEC 29110 Basic Profile. Canada: École de technologie supérieure.

Haley, T. J. 1996. Software process improvement at Raytheon. *IEEE Software* 13, no. 6:33-41.

IEEE. 2008. IEEE Standard 1028-2008: IEEE standard for software reviews and audits. New York: Institute of Electrical and Electronics Engineers (IEEE).

ISO. 2008. ISO/IEC/IEEE 12207:2008: Information technology – Software life cycle processes. Geneva, Switzerland: International Organization for Standardization/International Electrotechnical Commission.

ISO. 2011b. ISO/IEC TR 29110-1:2011, Software engineering - Lifecycle profiles for very small entities (VSEs) - Part 1: Overview. Geneva, Switzerland: International Organization for Standardization (ISO). Available at: http://standards.iso.org/ittf/PubliclyAvailableStandards/c051150_ISO_IEC_TR_29110-1_2011.zip.

ISO. 2011c. ISO/IEC TR 29110-5-1-2:2011- Software engineering - Lifecycle profiles for very small entities (VSEs) - Part 5-1-2: Management and engineering guide - Generic profile group: Basic profile. Geneva, Switzerland: International Organization for Standardization/International Electrotechnical Commission. Available at: http://standards.iso.org/ittf/PubliclyAvailableStandards/c051153_ISO_IEC_TR_29110-5-1_2011.zip.

ISO. 2011d. ISO/IEC 29110-2:2011 Software engineering - Lifecycle profiles for very small entities (VSEs) – Part 2: Framework and taxonomy. Geneva, Switzerland: International Organization for Standardization (ISO). Available at: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=51151.

ISO. 2012. ISO/IEC TR 29110-5-1-1:2012- Software engineering - Lifecycle profiles for very small entities (VSEs) - Part 5-1-1: Management and engineering guide - Generic profile group: Entry profile. Geneva, Switzerland: International Organization for Standardization/International Electrotechnical Commission. Available at: http://standards.iso.org/ittf/PubliclyAvailableStandards/c060389_ISO_IEC_TR_29110-5-1-1_2012(E).zip.

ISO. 2014a. ISO/IEC TR 29110-6-5-2:2014 - Systems and software engineering – Systems engineering lifecycle profiles for very small entities (VSEs) - Management and engineering guide: Generic profile group: Basic profile. Geneva, Switzerland: International Organization for Standardization/International Electrotechnical Commission. Available at: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=63371.

ISO. 2014b. ISO/IEC PDTR 29110-6-5-1:2014 - Systems and software engineering – Systems engineering lifecycle profiles for very small entities (VSEs) - Management and engineering guide: Generic profile group: Entry profile. Geneva, Switzerland: International Organization for Standardization/International Electrotechnical Commission.

ISO. 2014c. ISO/IEC CD 29110-3-2:2014 - Systems engineering lifecycle profiles for very small entities (VSEs) – Part 3-2: Conformity certification scheme. Geneva, Switzerland: International Organization for Standardization/International Electrotechnical Commission.

Krasner, H. 1998. Using the cost of quality approach for software. *Crosstalk - The Journal of Defense Software Engineering* 11 (November):6-11.

Kroll, P., and P. Kruchten. 2003. *The Rational unified process made easy: A practitioner's guide to the RUP*. Boston: Addison-Wesley Longman Publishing Co.

Laporte, C. Y., S. Alexandre, and R. O'Connor. 2008b. A software engineering lifecycle standard for very small enterprises. In *Proceedings of EuroSPI Springer-Verlag*, CCIS 16:129-141.

Laporte, C. Y., N. Berrhouma, M. Doucet, and E. Palza-Vargas. 2012. Measuring the cost of software quality of a large software project at Bombardier Transportation. *Software Quality Professional Journal* 14, no. 3 (June):14-31.

Laporte, C. Y., R. O'Connor, and G. Fanmuy. 2013. International systems and software engineering standards for very small entities. *CrossTalk - The Journal of Defense Software Engineering* 26, no. 3 (May/June):28-33.

Laporte, C. Y., and R. O'Connor. 2014a. A systems process lifecycle standard for very small entities: Development and pilot trials. In *Proceedings of the 21st European Software Process Improvement Conference* (Euro SPI 2014), Luxembourg, 25-27 June.

Laporte, C. Y., and R. O'Connor. 2014b. Systems and software engineering standards for very small entities implementation and initial results. In *Proceedings of QUATIC'2014, 9th International Conference on the Quality of Information and Communications Technology*, Guimarães, Portugal, 23-26 September.

Laporte, C. Y., R. Houde, and J. Marvin. 2014. Systems engineering international standards and support tools for very small enterprises. Paper presented at the 24th Annual International Symposium of INCOSE (International Council on Systems Engineering), Las Vegas, NV, June 30 – July 3.

Moll, R. 2013. Being prepared – A bird's eye view of SMEs and risk management. ISO Focus+. Geneva, Switzerland: International Organization for Standardization.

Nambisan, S. 2002. Software firm evolution and innovation–orientation. *Journal Engineering Technology Management* 19:141–165.

O'Connor, R., and C. L. Laporte. 2014. An innovative approach to the development of an international software process lifecycle standard for very small entities. *International Journal of Information Technology and the Systems Approach* 7, no. 1:1-22 (January-June).

Palza, E. 2009. Verification and validation deployment package, École de technologie supérieure, July 31. Available at: http://profs.etsmtl.ca/claporte/VSE/Trousses/DP_V&V_rev1.doc.

Paulk M., B. Curtis, M. B. Chrissis, and C. Weber. 1993. Capability Maturity Model for Software, version 1.1 (CMU/SEI-93-TR-24). Pittsburgh: Software Engineering Institute, Carnegie Mellon University.

Ruokolainen. 2007. Constructing a market domain model for startup software technology companies: A case study. *Journal of Engineering Technology Management* 24:186–202.

Selby, P., and R. W. Selby. 2007. Measurement-driven systems engineering using Six Sigma techniques to improve software defect detection. In *Proceedings of 17th International Symposium, INCOSE*, June, San Diego, CA.

Shintani, K. 2006. Empowered engineers are key players in process improvement. Presentation at the First International Research Workshop for Process Improvement in Small Settings, Software Engineering Institute (CMU/SEI-2006-SR-01), Pittsburgh, PA.

SEI. (2010). CMMI for Development, version 1.3 (CMU/SEI-2010-TR-033). Pittsburgh: Software Engineering Institute, Carnegie Mellon University.

Statistics Canada. 2008. Available at: http://www.ic.gc.ca/sbstatistics.

Sutton, S. M. 2000. The role of process in a software startup. *IEEE Software* (July/August):33-39.

Wallace, D. 1996. Reference information for the software verification and validation process, National Institute of Standards and Technology (NIST), U.S. Department of Commerce, Special Publication 500-234.

BIOGRAPHIES

Claude Y. Laporte has been a professor since 2000 at the École de technologie supérieure (ÉTS), a 7,500-student engineering school, where he teaches software engineering. His research interests include software process improvement in small and very small enterprises, as well as software quality assurance. He has worked in defense and transportation enterprises for more than 20 years. He received a master's degree in physics from the Université de Montréal, a master's degree in applied sciences from the École Polytechnique de Montréal, and a doctorate from the Université de Bretagne Occidentale (France). In addition, he was awarded an honorary doctorate by the Universidad de San Martin de Porres (Peru) in 2013. He is the editor of ISO/IEC JTC1 SC7 Working Group 24, tasked to develop ISO/IEC 29110 lifecycle standards and guides for very small entities. He is the co-chair of the INCOSE Systems Engineering for Very Small Entities Working Group. He is a member of INCOSE, IEEE, PMI, and the professional association of engineers of the Province of Québec (Ordre des ingénieurs du Québec). Laporte is the coauthor of two French books on software quality assurance and one English textbook, on the same topic, to be published in 2014.

Charles Hébert is co-founder of the website SwiceTrip.com. Hébert has 10 years of experience in software development and design. He has a bachelor's degree in computer science and a master's degree in software engineering, both completed in Canada. Hébert also holds the credential of Microsoft Certified Applications Developer (MCAD). He has worked as an SQL server database administrator in an engineering firm for three years. For the last seven years, his work has focused mainly on the design of Web applications using Microsoft.NET technologies. He currently holds the position of computer analyst in a regulatory agency in the Canadian province of Quebec. He is working on the development of an administration system for pension plans reserved for workers in the construction industry. Hébert has a particular interest in the improvement of business processes based on software engineering standards. He can be reached by email at charles.hebert@sohosme.com.

Christian Mineau is co-founder of the website SwiceTrip.com. Mineau is a senior analyst programmer with more than 10 years of experience in industries. He focuses on good development practices in software engineering and IT solution architecture. He worked on multiple Java EE software development projects in the banking, telecommunications, and consulting industries. Mineau completed his master of software engineering at the École de technologie supérieure of Montreal in 2013. He can be reached by email at Christian.mineau@gmail.com.