Deployment Package
Security Requirements Considerations
Basic Profile + Security

Authors	Perla Maciel – CIMAT A.C. Jezreel Mejía – CIMAT A.C. (México)
Editors	Perla Maciel – CIMAT A.C. Jezreel Mejía – CIMAT A.C. (México))
Creation date	24/06/21
Last update	24/06/21
Version	0.1

Version History

Date	Version	Description	Author
24/06/2021	0.1	Document creation	Perla Maciel

Abbreviations/Acronyms

Abre./Acro.	Definition
DP	Deployment Package - a set of artefacts developed to facilitate the implementation of a set of practices, of the selected framework, in a Very Small Entity.
VSE	Very Small Entity – an enterprise, organization, department or project having up to 25 people.
VSEs	Very Small Entities.
SA	Security Advisor

Table of Contents

1. Technical Description

Purpose of this document

This Deployment Package (DP) supports the Basic Profile as defined in ISO/IEC TR 29110-5-1-2: 2011 Management and Engineering Guide[1]. The Basic Profile is one profile of the Generic profile group. The Generic Profile Group is a collection of four profiles (Entry, Basic Intermediate, Advanced), providing a progressive approach to satisfying a vast majority of VSEs. The Generic profile group is applicable to VSEs that do not develop critical software, commercial off the shelf software products. The Generic profile group does not imply any specific application domain.

A DP is a set of artefacts developed to facilitate the implementation of a set of practices in a Very Small Entity (VSE). A DP is not a process reference model (i.e. it is not prescriptive). The elements of a typical DP are: description of processes, activities, tasks, roles and products, template, checklist, example, reference to standards and models, and tools.

The content of this document is entirely *informative*.

This document has been produced by Perla Maciel and Jezreel Mejía of CIMAT A.C. (México).

Why Security Requirements Considerations are Important?

Several studies clearly underlined the importance of requirement management in software engineering. Figure 1 illustrates that close to 50% of the software defects are produced during the requirements phase (Selby 2007)[2]. That established, the requirement phase brings an opportunity area to prevent the security holes that can surge after the development of the software application, then defining security requirement may reduce the number of defect and vulnerabilities introduced to the project.

The Security Requirement help in the identification of the security functionalities of a Software Project. Standard security criteria allow developers to reuse the specification of security controls and best practices rather than building a custom approach to security for each application.

[1] Available at no cost at:
http://standards.iso.org/ittf/PubliclyAvailableStandards/c051153_ISO_IEC_TR_29110-5-1_2011.zip

[2] Selby, P., Selby, R.W., Measurement-Driven Systems Engineering Using Six Sigma Techniques to Improve Software Defect Detection, Proceedings of 17th International Symposium, INCOSE, June 2007, San Diego.

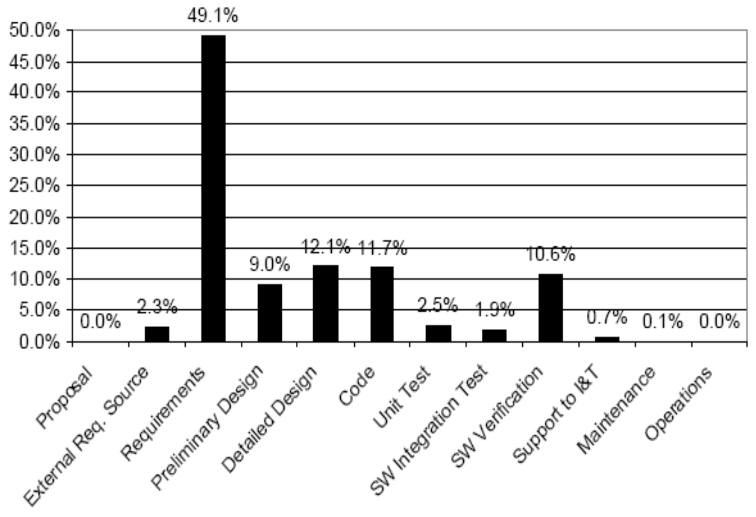


Figure 1 Origins of software defects (Selby 2007)

2. Definitions

In this section, the reader will find two sets of definitions. The first set defines the terms used in all Deployment Packages, i.e. generic terms. The second set of terms used in this Deployment package, i.e. specific terms.

Generic Terms

Process: set of interrelated or interacting activities which transform inputs into outputs [ISO/IEC 12207].

Activity: a set of cohesive tasks of a process [ISO/IEC 12207].

Task: required, recommended, or permissible action, intended to contribute to the achievement of one or more outcomes of a process [ISO/IEC 12207].

Sub-Task: When a task is complex, it is divided into sub-tasks.

Step: In a deployment package, a task is decomposed in a sequence of steps.

Role: a defined function to be performed by a project team member, such as testing, filing, inspecting, coding. [ISO/IEC 24765]

Product: piece of information or deliverable that can be produced (not mandatory) by one or several tasks. *(e. g. design document, source code)*.

Artefact: information, which is not listed in ISO/IEC 29110 Part 5, but can help a VSE during the execution of a project.

Specific Terms

Requirement: *1.* a statement that identifies what a product or process must accomplish to produce required behaviour and/or results. *IEEE 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process*. 3.1.16. *2.* a system or software requirement that specifies a function that a system/software system or system/software component must be capable of performing. *ISO/IEC 24765, Systems and Software Engineering Vocabulary.* *3.* a requirement that specifies a function that a system or system component must be able to perform. [ISO/IEC24765]

Requirements analysis: The process of studying user needs to arrive at a definition of system, hardware, or software requirements. [ISO/IEC 24765]

Requirements document: a document containing any combination of recommendations, requirements or regulations to be met by a software package. [ISO/IEC 24765]

Requirements phase: the period of time in the software life cycle during which the requirements for a software product are defined and documented. [ISO/IEC 24765]

Software Requirements Specifications: The SRS is a specification for a particular software product, program, or set of programs that performs certain functions in a specific environment. The SRS may be written by one or more representatives of the supplier, one or more representatives of the customer, or by both. [IEEE830-98]

The SRS document contains both functional and non functional requirements.

The SRS can be materialized in a word document but it can also be managed in a database or in an Excel file.

Non Functional Requirement: a software requirement that describes not what the software will do but how the software will do it. ISO/IEC 24765, Systems and Software Engineering Vocabulary. Syn. design constraints, non-functional requirement. See also: functional requirement. NOTE for example, software performance requirements, software external interface requirements, software design constraints, and software quality attributes. Non functional requirements are sometimes difficult to test, so they are usually evaluated subjectively. [ISO/IEC24765]

Security Requirement: a security requirement cover functional security and emergent characteristics that ensure that one of many different security properties of software is being satisfied. As OWASP Top Ten Proactive Controls 2018 states that for an application, security requirements offer a foundation of approved security functionality. Standard security criteria allow developers to reuse the specification of security controls and best practices rather than building a custom approach to security for each application. Security issues that have happened in the past can be solved using the same established security requirements. Requirements exist to prevent security failures in the future. [OWASP Top Ten Proactive Controls 2018]

Prototype: ***1.***an experimental model, either functional or non functional, of the system or part of the system. *IEEE 1233, 1998 Edition (R2002) IEEE Guide for Developing System Requirements Specifications.* 3.12. ***2.*** a preliminary type, form, or instance of a system that serves as a model for later stages or for the final, complete version of the system. *ISO/IEC 24765, Systems and Software Engineering Vocabulary.* ***3.*** model or preliminary implementation of a piece of software suitable for the evaluation of system design, performance or production potential, or for the better understanding of the software requirements. *ISO/IEC 15910:1999 Information technology -- Software user documentation process.* 4.41. [ISO/IEC24765]

Traceable*:* having components whose origin can be determined. [ISO/IEC24765]

Traceability matrix: a matrix that records the relationship between two or more products of the development process. [ISO/IEC24765]

3. Relationships with ISO/IEC 29110

This deployment package covers the activities related to requirements analysis of ISO/IEC TR 29110 Part 5-1-2:2011 for Very Small Entities (VSEs) – Generic Profile Group: Basic Profile [ISO/IEC 29110].

In this section, the reader will find a list of applicable Project Management (PM) and Software Implementation (SI) process, activities, tasks and roles from Part 5 are directly related to this topic. This topic is described in details in the next section.

- **Process:**
  - **Project Management**
- **Activities:**
  - **PM.1 Project Planning**
- **Tasks and Roles:**

Tasks	**Roles**
PM.1.3 Identify the specific Tasks to be performed in order to produce the Deliverables and their Software Components identified in the Statement of Work. Include Tasks in the SI process along with verification, validation and reviews with Customer and Work Team Tasks to assure the quality of work products. Identify the Tasks to perform the Delivery Instructions. Document the Tasks.	PM, TL

- **Process:**
  - **Software Implementation**
- **Activities:**
  - **SI.2 Software requirements analysis**
- **Tasks and Roles:**

Tasks	**Roles**
SI.2.2 Document or update the *Requirements Specification.* Identify and consult information sources (customer, users, previous systems, documents, etc.) in order to get new requirements. Analyse the identified requirements to determinate the scope and feasibility. Generate or update the *Requirements Specification.*	AN, CUS
SI.2.3 Verification and obtaining approval of the *Requirements Specification.* Verify the correctness and testability of the *Requirements Specification* and its consistency with the *Product Description*. Additionally, review that requirements are complete, unambiguous and not contradictory. The results found are documented in a *Verification Results* and	AN, TL

Tasks	Roles
corrections are made until the document is approved by AN. If significant changes were needed, initiate a *Change Request.*	

4. Description of Processes, Activities, Tasks, Steps, Roles and Products

Process: Project Management

The purpose of the Software Implementation process is the systematic performance of the analysis, design, construction, integration and tests activities for new or modified software products according to the specified requirements.

The Software Requirements Analysis activity analyses the agreed customer's requirements *and establishes the validated project requirements.*

- **Process:**
  - **Project Management**
- **Activities:**
  - **PM.1 Project Planning**
- **Tasks and Roles:**

Tasks	**Roles**
PM.1.3 Identify the specific Tasks to be performed in order to produce the Deliverables and their Software Components identified in the Statement of Work. Include Tasks in the SI process along with verification, validation and reviews with Customer and Work Team Tasks to assure the quality of work products. Identify the Tasks to perform the Delivery Instructions. Document the Tasks.	PM, TL

Create Quality Gates/Bug Bars

Objectives:	Quality gates and bug bars are used to establish minimum acceptable levels of security and privacy quality.
Rationale:	Defining these criteria at the start of a project improves the understanding of risks associated with security issues and enables teams to identify and fix security bugs during development.
Roles:	PM – Customer
	TL – Technical Leader
	SA – Security Advisor
Artefacts:	Quality Gates/Bug Bars
Steps:	1. Define quality gates for each development phase
	2. Negotiate which quality gates must be integrated and which not
	3. Approve the quality gates negotiated

Step Description:	***Step 1. Define quality gates for each development phase***: A Project team must define quality gates to each development phase of the project (for example, all compiler warnings must be triaged and fixed prior to code check-in). ***Step 2. Negotiate which quality gates must be integrated and which not*** A project team must negotiate quality gates (for example, all compiler warnings must be triaged and fixed prior to code check-in) for each development phase. ***Step 3. Approve the quality gates negotiated*** Have the quality gates approved by the security advisor, who may add project-specific clarifications and more stringent security requirements as appropriate.

Process: Software Implementation

The purpose of the Software Implementation process is the systematic performance of the analysis, design, construction, integration and tests activities for new or modified software products according to the specified requirements.

The Software Requirements Analysis activity analyses the agreed customer's requirements *and establishes the validated project requirements.*

- **Process:**
  - **Software Implementation**
- **Activities:**
  - **SI.2 Software requirements analysis**
- **Tasks and Roles:**

Tasks	Roles
SI.2.2 Document or update the *Requirements Specification.* Identify and consult information sources (customer, users, previous systems, documents, etc.) in order to get new requirements. Analyse the identified requirements to determinate the scope and feasibility. Generate or update the *Requirements Specification.*	AN, CUS
SI.2.3 Verification and obtaining approval of the *Requirements Specification.* Verify the correctness and testability of the *Requirements Specification* and its consistency with the *Product Description*. Additionally, review that requirements are complete, unambiguous and not contradictory. The results found are	AN, TL

Tasks	Roles
documented in a *Verification Results* and corrections are made until the document is approved by AN. If significant changes were needed, initiate a *Change Request.*	

- **Tasks and Roles:**
  - **Security Requirements identification**

Security Requirements identification

Objectives:	The objective of this activity is to clearly define the scope of the project and identify the key requirements of the system.
Rationale:	It is important to clearly define the project scope (boundaries) and to identify key functionalities of the future system with the customer to avoid problems like forgotten key functionalities or requirements creep.
Roles:	CUS – Customer
	AN – Analyst
	TL – Technical Leader
	SA – Security Advisor
Artefacts:	Use Cases – scenarios
	Abuse/Misuse Cases – scenarios
	Requirements Document
Steps:	1. Collect information about the application domain (e.g. finance, medical)
	2. Identify project's scope
	3. Identify and capture requirements
	4. Structure and prioritize requirements
Step Description:	***Step 1. Collect information about the domain***: During this Step, analyst captures the key concepts of the business domain of the customer. The customer assists the analyst by giving him all the information (existing documentation or explanation) that will facilitate this understanding. Key concepts are listed in a glossary section in the *Software Requirements Specification Document outline* document. ***Step 2. Identify project's scope*** Software analyst, helped by the person in charge of the contractual aspects of the project (sales manager) clearly identifies main functionalities that are included in the project scope.

	With the help of a Security Advisor, the Software analyst must identify the security requirements (classified as non-functional requirements in some cases) taking in consideration the assets list to protect. *Tips*: Identifying functionalities that are OUT of scope is also very valuable to clarify differences of understanding with your customers. ***Step 3. Identify and capture requirements*** Having in mind key concepts related to the customer business domain, analyst can start requirements identification. None of the situations in IT projects are identical. In some cases, most of the requirements are already identified in a document (call for tender in case of fixed priced projects). However, in most of the cases, key requirements are just (orally) mentioned by the customer. Analyst must identify and list the key requirements of the system to be built. During this Step, analyst should not start detailing identified requirements. The main goal is to gain a comprehensive view of the system requirements. ***Step 4. Structure and prioritize requirements***: Using requirements identified in the previous Step, the analyst has to organise and structure identified requirements accordingly (e.g. by business processes or by system functions). A priority must be identified by the customer for system key functionalities. Priorities can be stated like • '*High*' – a functionality that *shall be* implemented • 'Medium' - a functionality that *should be* implemented • 'Low' - a functionality that *could be* implemented The output of this Step is a list of requirements that are organized in the *Requirements Document.*

Requirements refinement and analysis

Objectives:	The objective of this Step is to detail and analyse all the requirements identified.
Rationale:	It is important to go through identified requirements in order to detect requirements that seem easy to implement but hiding a business complexity that will cause problems in the project.
Roles:	AN – Analyst
	CUS - Customer
	PR - Programmer
	SA – Security Advisor
Artefacts:	Use Cases – scenarios

	Abuse/Misuse Cases – scenarios
	Requirements Document
	Software prototype
Steps:	1. Detail requirements
	2. Produce a prototype
Step Description:	***Step 1. Detail requirements:*** The analyst goes through a set of identified requirements and performs a more detailed analysis. Business complexity can be implicit for some requirements. This must be clarified at this point before any implementation. The analyst will interact with customer representatives in order to clarify ergonomic question, if relevant (e.g. if a Graphical User Interface must be developed). This Steps results in a new version of the *Software Requirements Specification Document.* ***Step 2. Produce a prototype*** Producing a prototype can facilitate requirement understanding from all project participants (i.e. customer side and development team side). A prototype may only implement some of the functionalities.

Role Description

This is an alphabetical list of the roles, abbreviations and list of competencies as defined in Part 5.

	Role	**Abbreviation**	**Competency**
1.	Analyst	AN	Knowledge and experience eliciting, specifying and analyzing the requirements. Knowledge in designing user interfaces and ergonomic criteria. Knowledge of the revision techniques and experience on the software development and maintenance. Knowledge of the editing techniques and experience on the software development and maintenance.
2.	Customer	CUS	Knowledge of the Customer processes, ability to explain the Customer requirements and experience in the application domain. The Customer (representative) must have the authority to approve the requirements and their changes. The Customer includes user representatives in order to ensure that the operational environment is

			addressed.
3.	Programmer	PR	Knowledge and/or experience in programming, integration and unit tests. Knowledge of the revision techniques and experience on the software development and maintenance. Knowledge of the editing techniques and experience on the software development and maintenance
4.	Project Manager	PM	Leadership capability with experience making decisions, planning, personnel management, delegation and supervision, finances and software development.
5.	Security Advisor	SA	Knowledge to advise the businesses to identify potential security weaknesses, create security policies, and reduce risks to their IT systems.
6.	Technical Leader	TL	Knowledge and experience in the software process domain.
7.	Work Team	WT	Knowledge and experience according to their roles on the project: TL, AN, DES, and/or PR. Knowledge on the standards used by the Client and/or by the VSE.

Product Description

This is an alphabetical list of the input, output and internal process products, its descriptions, possible states and the source of the product.

	Name	Description	Source
1.	Change Request	It may has the following characteristics: Identifies purpose of change Identifies request status (new, accepted, rejected) Identifies requester contact information Impacted system(s) Impact to operations of existing system(s) defined Impact to associated documentation defined Criticality of the request, date needed by The applicable statuses are: initiated, evaluated and accepted.	Software Implementation Customer Project Management
2.	Project Plan	It Includes the following elements which may have the characteristics as follows: - Product Description o Purpose	Project Management

		o General Customer requirements - Scope description of what is included and what is not - Objectives of the project - Deliverables - list of products to be delivered to Customer - Tasks, including verification, validation and reviews with Customer and Work Team, to assure the quality of work products. Tasks may be represented as a Work Breakdown Structure (WBS). The task also includes the identification of security requirements, list of assets, threats, information security risk and incidents. - *Relationship and Dependence of the Tasks* - *Estimated Duration* of tasks - *Resources* o perform the project both in development and security (humans, materials, equipment and tools) including the required training, and the schedule when the resources are needed. - *Composition of Work Team* - *Competences pf personal Record* - *Role Matrix* - *Incidents Response of the project* - Schedule of the Project Tasks, the expected start and completion date, for each task. - Estimated Effort and Cost - Identification of Project Risks - Version Control Strategy - Product repository tools or mechanism identified - Location and access mechanisms for the repository specified - Version identification and control defined - Backup and recovery mechanisms defined - Storage, handling and delivery (including archival and retrieval) mechanisms specified - Delivery Instructions - Elements required for product release identified (i.e., hardware, software, documentation etc.) - Delivery requirements - Sequential ordering of tasks to be performed - Applicable releases identified - Identifies all delivered software components with version information - Identifies any necessary backup and recovery procedures	

		The applicable statuses are: verified, accepted, updated and reviewed.	
3.	Project Repository	It may have the following characteristics: - Stores project work products - Stores released deliverables products - Storage and retrieval capabilities - Ability to browse content - Listing of contents with description of attributes - Sharing and transfer of work products between affected groups - Effective controls over access - Maintain work products descriptions - Recovery of archive versions of work products - Ability to report work products status - Changes to work products are tracked to Change Requests The applicable statuses are: recovered and updated.	Project Management
4.	Requirements Specification	It may have the following characteristics: - Introduction –general description of software and its use within the scope of the customer business; - Requirements description: - Functionality – established needs to be satisfied by the software when it is used in specific conditions. Functionality must be adequate, accurate and safe. - User interface – definition of those user interface characteristics that allow to understand and learn the software easily so the user be able to perform his/her tasks efficiently including the interface exemplar description; - External interfaces – definition of interfaces with other software or hardware; - Security – specification of the software execution level concerning the level of security implemented - Reliability – specification of the software execution level concerning the maturity,	Software Implementation

		fault tolerance and recovery; - Efficiency – specification of the software execution level concerning the time and use of the resources; - Maintenance – description of the elements facilitating the understanding and execution of the future software modifications; - Portability – description of the software characteristics that allow its transfer from one place to other; - Design and construction limitations/constraints – needs imposed by the customer; - Interoperability – capability for two or more systems or software components be able to change information each other and use it. - Reusability – feature of any product/sub-product, or a part of it, so that it can be used by several users as an end product, in the own software development, or in the execution of other software products. - Legal and regulative – needs imposed by laws, regulations, etc. Each requirement is identified, unique and it is verifiable or can be assessed. The applicable statuses are: verified, validated and baselined.	
5.	Software User Documentation	It may have the following characteristics: - User procedures for performing specified tasks using the Software - Installation and de-installation procedures - Brief description of the intended use of the Software (the concept of operations) - The supplied and required resources - Needed operational environment - Availability of problem reporting and assistance - Procedures to access and exit the Software - Lists and explains software commands and system-provided messages to the user	Software Implementation

		- As appropriate for the identified risk, it includes warnings, cautions, and notes, with corrections - It includes troubleshooting and error correction procedures. It is written in terms understandable by users. The applicable statuses are: preliminary, verified and baselined.	
6.	Verification Results	It may include the record of: - Participants - Date - Place - Duration - Verification check-list - Passed items of verification - Failed items of verification - Pending items of verification - Defects identified during verification	Project Management Software Implementation
7.	Validation Results	Documents the validation execution, It may include the record of: - Participants - Date - Place - Duration - Validation check-list - Passed items of validation - Failed items of validation - Pending items of validation - Defects identified during validation	Software Implementation

Artefacts Description

This is an alphabetical list of the artifacts that could be produced to facilitate the documentation of a project. The artifacts are not required by Part 5, they are optional.

	Name	Description
1.	Abuse/Misuse Cases - scenarios	Describe the system's behaviour under attack; building abuse cases requires explicit coverage of what should be protected, from whom, and for how long. An Abuse case is generated as a flow diagram that contains the interactions and possible attack that can be identified form the use cases.
2.	Requirements Document	Document in which all identified requirements are centralized. See Software Requirements Specification definition in section 0.
3.	Software prototype	Working piece of software produced during the early phases in order to demonstrate/validate a functionality of a system.

	Name	Description
4.	Use Cases – scenarios	Description of a sequence of interactions between the user and the future systems. Uses cases can be written as prescribed by UML but can also be text scenarios.

5. Template

The templates provided with this deployment package should be customized for your project.

SRS Template Table of Content –Basic List of Requirements

To be used in an Excel sheet structured, for example, as:

ID	*Requirement*	*Description*	*Priority*

SRS Template Table of Content –Adapted from IEEE 830

1. Introduction

1.1 Purpose

1.2 Document conventions

1.3 Intended audience

1.4 Additional information

1.5 Contact information/SRS team members

1.6 References

2. Overall Description

2.1 Product perspective

2.2 Product functions

2.3 User classes and characteristics

2.4 Operating environment

2.5 User environment

2.6 Design/implementation constraints

2.7 Assumptions and dependencies

3. External Interface Requirements

3.1 User interfaces

3.2 Hardware interfaces

3.3 Software interfaces

3.4 Communication protocols and interfaces

4. System Features

4.1 System feature A

4.1.1 Description and priority

4.1.2 Action/result

4.1.3 Functional requirements

4.2 System feature B

5. Other Non functional Requirements

5.1 Performance requirements

5.2 Safety requirements

5.3 Security requirements

5.4 Software quality attributes

5.5 Project documentation

5.6 User documentation

6. Other Requirements

Appendix A: Terminology/Glossary/Definitions list

Appendix B: To be determined

SRS Template Table of Content -Construx[3]

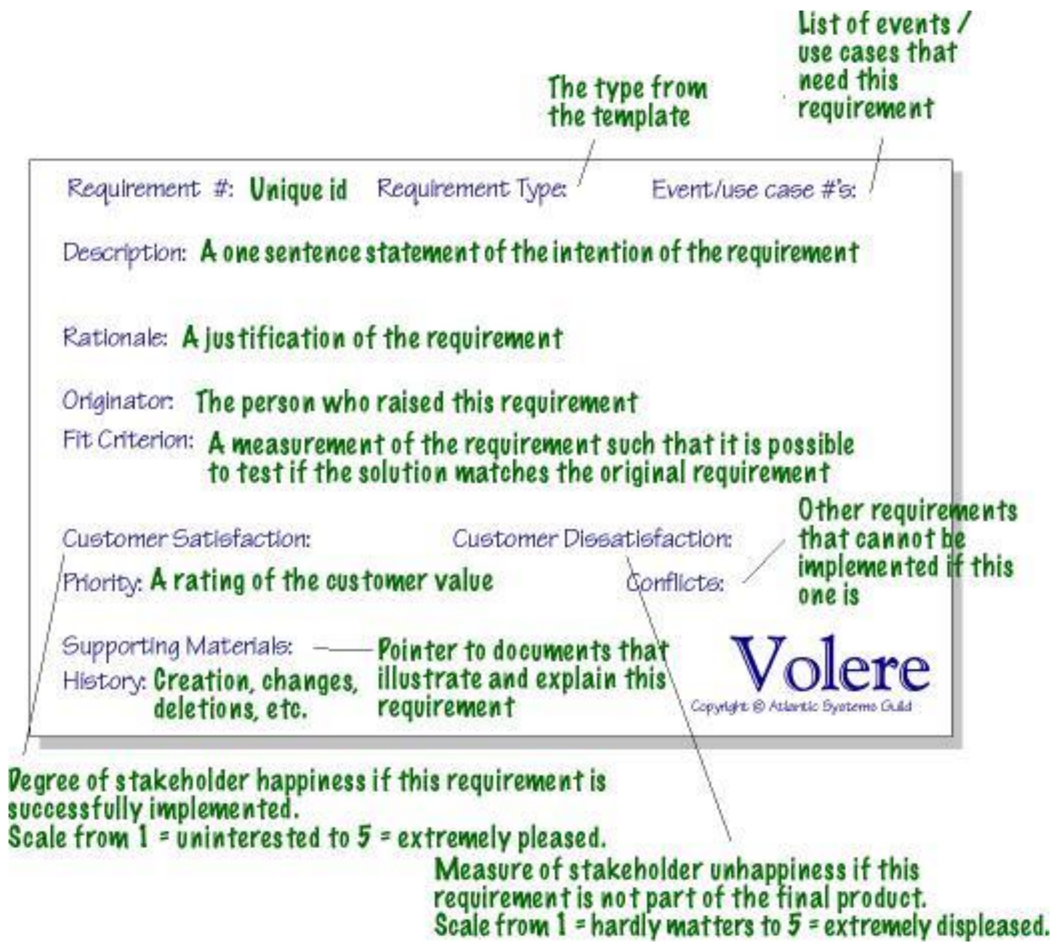
[3] www.construx.com

SRS Template Table of Content – Volere[4]

Contents

[4] http://atlsysguild.com/GuildSite/Robs/Template.html

SRS Template Table of Content – Volere Requirement Shell



Requirement #: Unique id Requirement Type: Event/use case #'s:

The type from the template

List of events / use cases that need this requirement

Description: A one sentence statement of the intention of the requirement

Rationale: A justification of the requirement

Originator: The person who raised this requirement

Fit Criterion: A measurement of the requirement such that it is possible to test if the solution matches the original requirement

Customer Satisfaction: Customer Dissatisfaction:

Priority: A rating of the customer value Conflicts:

Other requirements that cannot be implemented if this one is

Supporting Materials: Pointer to documents that illustrate and explain this requirement

History: Creation, changes, deletions, etc.

Volere

Copyright © Atlantic Systems Guild

Degree of stakeholder happiness if this requirement is successfully implemented.
Scale from 1 = uninterested to 5 = extremely pleased.

Measure of stakeholder unhappiness if this requirement is not part of the final product.
Scale from 1 = hardly matters to 5 = extremely displeased.

6. Example of Lifecycle

This section provides some graphical representation of example of requirement practices lifecycle. These examples are provided to help the reader to implement his own requirement lifecycle fitting his IT project's context and constraints.

Example 1 of Requirement Practices Lifecycle

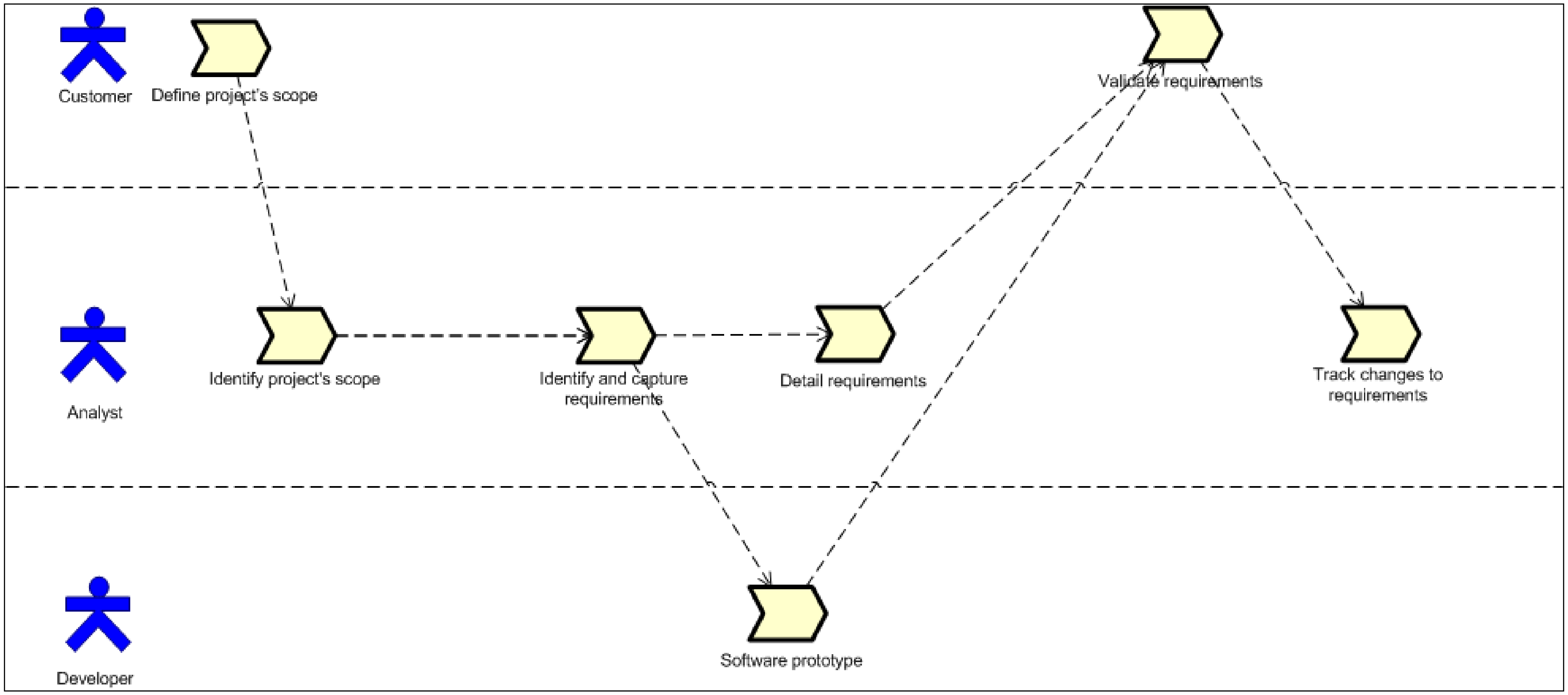


Figure 2 Example 1 of Requirement Practices Lifecycle

Example 2 of Requirement Practices Lifecycle

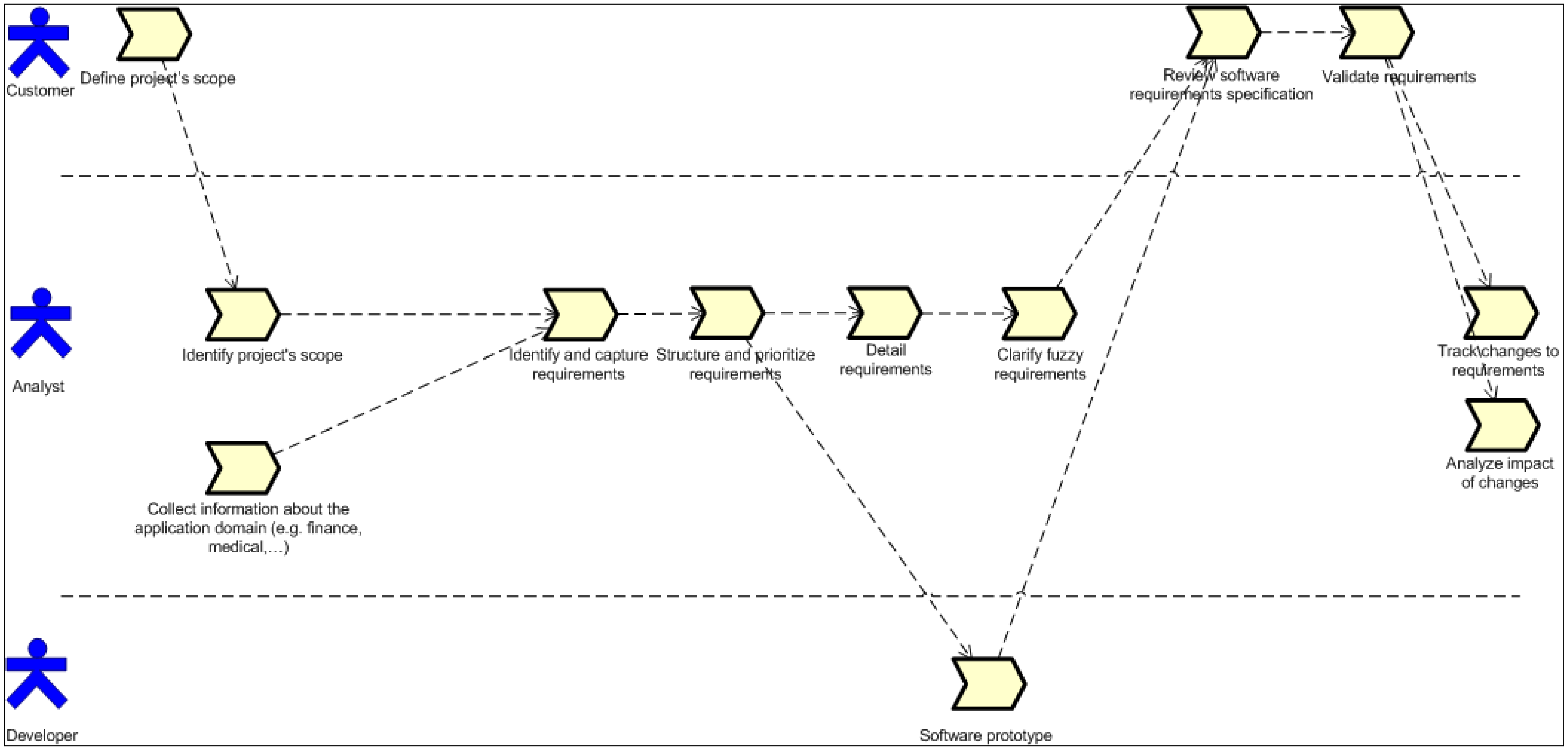

Customer
Define project's scope
Review software requirements specification
Validate requirements
Analyst
Identify project's scope
Identify and capture requirements
Structure and prioritize requirements
Detail requirements
Clarify fuzzy requirements
Track changes to requirements
Analyze impact of changes
Collect information about the application domain (e.g. finance, medical,...)
Developer
Software prototype

Figure 3 Example 2 of Requirement Practices Lifecycle

Example 1 of Abuse/Misuse Cases

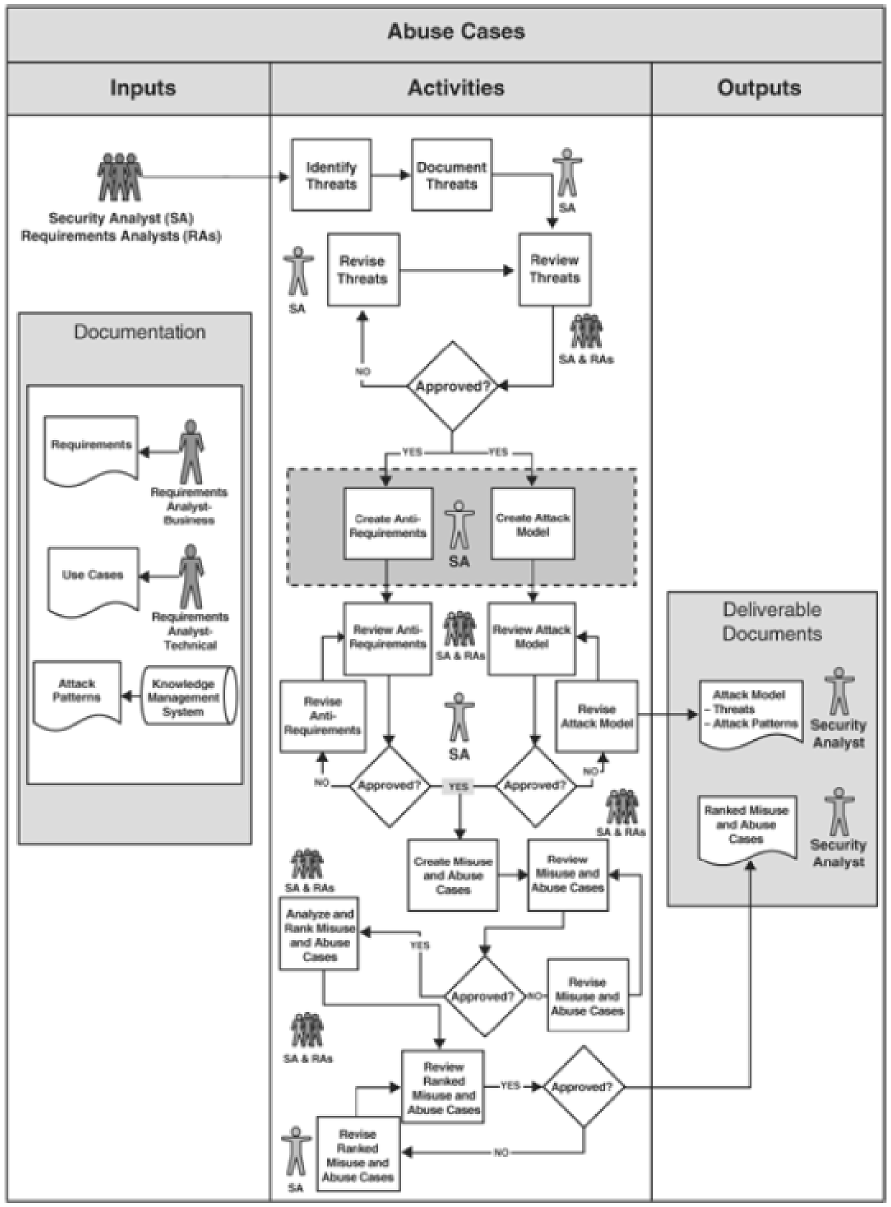

Abuse Cases
Inputs
Activities
Outputs
Security Analyst (SA)
Requirements Analysts (RAs)
Identify Threats
Document Threats
SA
Revise Threats
Review Threats
SA & RAs
NO
Approved?
YES
Documentation
Requirements
Requirements Analyst-Business
Use Cases
Requirements Analyst-Technical
Attack Patterns
Knowledge Management System
Create Anti-Requirements
Create Attack Model
Review Anti-Requirements
Review Attack Model
Revise Anti-Requirements
Revise Attack Model
Deliverable Documents
Attack Model – Threats – Attack Patterns
Security Analyst
Ranked Misuse and Abuse Cases
Create Misuse and Abuse Cases
Review Misuse and Abuse Cases
Analyze and Rank Misuse and Abuse Cases
Revise Misuse and Abuse Cases
Review Ranked Misuse and Abuse Cases
Revise Ranked Misuse and Abuse Cases

Figure 4 Example 1 of Misuse/Abuse Cases

7. Checklist

.

Requirement checklist

This Requirement checklist is based on [Constr07]

RS 1 Testable	All requirements are verifiable (objectively)
RS 2 Complete	Are the requirements complete?
RS 3 Traceable	All requirements must be traceable to a systems specification, contractual/proposal clause.
RS 4 Correct	Requirements must be correct (i.e. reflect exactly customer's requirements)
RS 5 Unique	Requirements must be stated only once
RS 6 Elementary	Requirements must be broken into their most elementary form
RS 8 High Level	Requirement must be stated in terms of final need, not perceived means (solutions)
RS 9 Quality	Quality attributes have been defined.
RS 10 Unambiguous	
RS 11 Hardware	Hardware environment is completely defined.
RS 12 Solid	Requirements are a solid base for design

8. References

Key	Reference
[ISO/IEC 29110]	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. International Organization for Standardization/International Electrotechnical Commission: Geneva, Switzerland. Available at no cost at: http://standards.iso.org/ittf/PubliclyAvailableStandards/c051153_ISO_IEC_TR_29110-5-1_2011.zip
[OWPL-EN]	Renault A., Habra N., Alexandre S., Deprez J.-C., OWPL. Software Process Improvement for VSE, SME and low maturity enterprises. Version 1.2.2, FUNDP-CETIC, 2000. (http://www.cetic.be/internal393.html)
[CMMI 2010]	CMMI® for Development, Version 1.3, CMU/SEI-2010-TR-033, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, 2010.
[IEEE830-98]	IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications, IEEE, 1998.
[ISO/IEC12119]	ISO/IEC 12119:1994 Information technology – Software packages -- Quality requirements and testing. International Organization for Standardization/International Electrotechnical Commission: Geneva, Switzerland.
[ISO/IEC12207]	ISO/IEC 12207:2008 Systems and software engineering - Software life cycle processes. International Organization for Standardization/International Electrotechnical Commission: Geneva, Switzerland.
[ISO/IEC24765]	ISO/IEC 24765:2010, Systems and Software Engineering Vocabulary. International Organization for Standardization/International Electrotechnical Commission: Geneva, Switzerland.
[ConstSoft02]	Construx Software – Checklist for Software Requirements Specifications, 2002.
[SELB07]	Selby, P., Selby, R.W., Measurement-Driven Systems Engineering Using Six Sigma Techniques to Improve Software Defect Detection, Proceedings of 17th International Symposium, INCOSE, June 2007, San Diego.
[STAN02]	Standish Group – Chaos report 2002.
[SPEM05]	Software Process Engineering Metamodel Specification, OMG, 2005.
[VOLE07]	Volere, Requirements Resources - http://www.volere.co.uk
[OWASP Top Ten Proactive Controls 2018]	OWASP Top Ten Proactive Controls 2018, https://owasp.org/www-project-proactive-controls/v3/en/c1-security-requirements