



eXtreme  
Programming  
Centre

## Optional *scope* project

Marina Morgagni – Manager, eXtreme Programming Centre

Piergiuliano Bossi – Coach, eXtreme Programming Centre

Quinary SpA

Optional *scope* project

**Copyright© 2001 Quinary SpA**

All rights reserved.

The copyright to these materials and any accompanying software is owned, without reservation, by Quinary SpA. These materials and any accompanying software may not be copied in whole or part, without the express written permission of Quinary SpA..

## Optional *scope* project

### 1. Introduction

Quinary offers optional scope projects in order to satisfy the needs of customers who wish to develop systems whose technical specifications continuously evolve, where the priorities with which functions are to be implemented may change frequently, or where the specifications are to be defined *en route*.

An optional *scope* project contract entails purchasing the skills of a development team for one or more time intervals (iterations). The length of time is measured in weeks (from a minimum of one up to a maximum of four). Ideally the customer should give an indication of the overall duration at the beginning of the project, so that the team's operations may be planned and there are no overlaps with requests from other customers.

The customer decides the length of the iteration and, once it is over, plans how to proceed according to their own needs. Initially a customer may decide to purchase a contract in which the development team is available to them for two weeks. Later they may purchase a further three weeks, then another week, and so on. The customer may choose the length of the development period on the basis of two alternative considerations:

- they have a deadline which they cannot miss. The customer wishes to implement the largest number of functions possible in the time available from now right up to the deadline;
- they want the system to include a set of functions: the development period is estimated on the basis of the commitment necessary to implement them.

It should be made clear right from the start that this type of contract only applies to projects which may be implemented using object-oriented languages, such as Java or SmallTalk.

This type of project is managed in practice using a method known in computing literature as **eXtreme Programming** (see the White Paper “eXtreme Programming in a nutshell”).

### 2. Project structure

Unlike what happens in traditional projects where each activity is time-limited and follows a time sequence, in the case of an optional scope project all the activities last the entire length of the project and only end when the project ends.

#### Macro activities

<b>A. Exploration</b>
Definition and analysis of the functionality (user stories) and estimate of the commitment involved in implementing it.
<b>B. Planning</b>
Planning of the activities to be undertaken according to customer-defined priorities.
<b>C. Development</b>
Development of the functions chosen by the customers within the agreed times.
<b>D. Tests and Acceptance</b>

## Optional *scope* project

### Macro activities

---

Functional tests (following the checklist defined by the users themselves) together with the customer.

---

## 2.1 Exploration

This activity either involves choosing and estimating how many and which functions may be implemented by the date fixed by the customer, or fixing a possible date by which a set of functions chosen by the customer may be implemented.

### 2.1.1 Definition of the functionality

A Quinary engagement manager<sup>1</sup> assists the customer<sup>2</sup> in compiling and putting all the user stories (i.e. the technical/functional specifications) together on story cards, and assigning each one a title, a short descriptive analysis and functional tests to check their correctness. To do this, the customer should indicate data values for the tests which they have come across in their own daily experience. The functional tests are the way in which the customer exemplifies the specifications. A story without tests cannot be estimated by the team. The tests allow both the efforts needed to implement the user stories to be estimated, and the mechanisms and interrelationships underlying the stories to be determined; it is therefore very important for the tests to be detailed.

Functionality definition and analysis lasts for the whole duration of the project. The customer may alter and/or eliminate and/or define new specifications at any time.

---

<sup>1</sup> for details regarding roles, see chapter “3 Resources involved, roles and support tools”.

<sup>2</sup> the customer who is asked to describe the stories and tests may be either a single person or a team; in any case, the customer must be skilled in the real use of the system and, if represented by a team, they must act in a coordinated and consistent way.

## Optional *scope* project


<b>user story</b> title: chosen	date:  estimate	
story description: _____ _____ _____		
functional 1. _____ 2. _____ 3. _____ 4. _____		
notes: _____ _____		
engineering 1. _____ 2. _____ 3. _____		

Figura 1 - Story Card

### 2.1.2 Story Analysis and Estimation

The engagement manager passes the development team and its coach all the *user stories* together with any additional information (restrictions regarding the delivery date or the user story set which the customer deems necessary); the team breaks each story down into development activities (so-called **engineering tasks**), highlights any constraints (a card which the customer may consider irrelevant may be a prerequisite for an important part of the system) and then estimates the story cards.

During this analysis, the development team works together with the customer in order to investigate and discover all the fine points regarding any unclear aspects, until a sufficient degree of detail is reached to produce an estimate for each *story card*.

## 2.2 Planning

The aim of this activity is to choose the functions to include in an iteration.

With the help of the engagement manager, the customer decides which user stories to implement during the iteration under way on the basis of the user story estimates and restrictions (delivery date or functionality).

The customer decides on the length of the iteration (e.g. 2 weeks) and the engagement manager lets the customer know the corresponding total number of story points (the unit of measurement used to measure the development workload which can be carried out during an iteration; the efforts needed to implement a story are also expressed in story points) available (i.e. 200). Together with the customer, the engagement manager then chooses stories until the points are used up. The estimate field of each story shows how many points its implementation will cost.

## Optional *scope* project

The customer may also request a draft release plan, which the team keeps constantly up to date as regards its forecasts.

A release is a set of functions which is valuable to the customer's business (in their unquestionable opinion), or in other words, a release is a complete set of functions which may be used by the system user. Releases may be planned in advance or decided at the end of an iteration. Of course it is impossible to include unestimated stories in the plan. During the planning activity, the most important user stories are assessed with the customer; the functionality each release has is therefore defined on the basis of business priorities or any unpostponable deadlines.

If any stories are not completely developed or not accepted by the customer during an iteration, the team estimates them again (together with all those which have not yet been tackled). In this way the customer has increasingly precise forecasts. The customer assesses all the stories, together with their respective estimates, and gives priority to those which interest them most, so that new plans may be drafted for the next iteration.

## 2.3 Development

At the beginning of each development period, the customer is given a document which includes all the stories described and indicates the ones to be developed during the iteration itself.

Once it has been decided which user stories belong to the iteration (according to their size and the length of the iteration itself), the team proceeds iteratively and incrementally by first of all developing the unit tests (which check the correctness of the tasks), followed by the user stories, and finally the functional tests (which check the conditions described by the customers on the story cards). The whole activity takes place in the development environment (a set of PCs and servers assigned exclusively to development) and each development activity is alternated with continuous testing.

During this activity the software engineers may involve the customer by asking them to clarify the descriptions and/or tests, specify the meanings of the stories, and decide upon micro-priorities. For their part, the customer may add new stories at any time by passing them directly to the engagement manager; these stories will only be taken into consideration during a new iteration.

When a task has been completely developed, the new code is integrated in another environment, separate from the development environment, which is shared by all the software engineers. Its compatibility with the other developed components is checked and correct system operation is verified using the unit and functional tests respectively. Obviously, if errors occur, the development activity regarding the task continues until the malfunction is corrected.

Since testing is automatic, it is carried out upon each integration activity; in this way, it is simpler to identify the causes of any failure.

Every time a story is completed (when the respective functional tests are carried out), the development team may check its correctness with the customer, who decides whether to accept it formally or not.

In the event of estimation errors or problems regarding the specifications of a particular user story, the team may find it impossible to complete all the stories in a single iteration. In this case the customer is alerted in good time so that they may split one or more of the user stories and postpone the split parts to later iterations, making sure that the present iteration has a sufficient degree of cohesiveness.

## Optional *scope* project

The customer may check the progress of the project at any time through extremely precise tracking mechanisms. Reports are available relating to the development of single stories, iterations and releases.

In order to respect project times and safeguard design quality, it is essential for the customer to make themselves available to the team during the iteration to dispel all doubts and uncertainties which may arise as the system evolves.

### 2.4 Tests and acceptance

The system configuration and implementation acceptance procedure takes place in the test environment, where all the data and systems which are found in the production environment are reproduced as faithfully as possible. This procedure entails performing the functional tests defined on all the story cards in the iteration under way together with the customer.

The outcome of the tests is formalized by drafting a **Release and Acceptance Statement**, which is signed by the person with whom the procedure is carried out.

The statement describes the stories and lists the functional tests for each one; a positive outcome (i.e. the absence of blocking faults attributable to Quinary's work and the conformity of the stories with the customer's requests) is simply shown in the document by the word 'accepted'. If, on the other hand, the way in which one or more stories are implemented is deemed inadequate, the customer expresses their non-acceptance by defining the story/stories as rejected and describes the way its/their functional tests should have been interpreted.

This document marks the end of the iteration.

## 3. Resources involved, roles and support tools

Optional scope projects are carried out by a mixed group of people consisting both of members of Quinary's staff and the customer's.

The staff to which each person belongs and their responsibilities are specified below:

- **Engagement manager (Quinary)** - he/she helps the customer define the contents of each iteration and release, and therefore makes alignment between the customer's needs and the design team's estimates easier. He/she assists the customer in choosing and writing the functional tests. At the end of each iteration, he/she carries out the functional tests together with the customer and formalizes the outcome.
- **Software Engineer (Quinary)** - he/she estimates the time commitment involved in each individual story; he/she writes the code for automatic unit and functional tests and for implementing the stories.
- **Tester (Quinary)** - this is the same person as the software engineer. He/she performs the tests at each integration.
- **Coach (Quinary)** - he/she is the technical design manager. He/she supervises the step-by-step development of each component, takes part in defining the architecture, intervenes in critical cases and makes sure the design maintains its "extreme" characteristics.
- **Tracker (Quinary)** - this is the same person as the engagement manager. He/she analyses the project progress reports. Quinary has developed a tool which automatically compiles reports which outline any faults and the development of

## Optional *scope* project

each story, iteration and release on basis of data which the software engineers input at the beginning and end of each day. The tool also makes a few measurements in order to assess the quality of the project and test code.

- **User (Customer)** - he/she has real system use expertise. He/she is directly involved in all the project activities and may be a single person or a team. With the aid of the engagement manager, he/she describes the stories and functional tests and is available to cooperate actively with the design team.

## 4. Prerequisites

The ideal situation, for the overall economy of the project, is definitely the on-site presence of the customer, which, if not continuous, must at least be recurrent. If this is not possible, Quinary offers alternatives in order to ensure a high level of communication between the customer and the team in any case: videoconferences, logins on extranet, etc. With the aid of these technologies, efficiency both in terms of time and communicative content is ensured so that possible misinterpretations and misunderstandings are avoided.

Any special activities, integration and installation of the software developed in the design environment which are to take place at the customer's offices are estimated and planned by the development team as special deployment user stories (the story point costs of these activities depend on the complexity and peculiarities of the production environment). Their estimate presupposes that the customer provides all the necessary information and the reference staff needed to carry them out correctly.

In the same way, document drafting (e.g. user manuals), training and all other documentation activities are considered normal user stories, which are only chosen by the customer when they are considered priority, and scheduled for an iteration.

If these activities cannot be clearly assimilated to development team activities, they will be estimated and offered in the best possible way.

The customer may recede from the contract at any time: the only obligation is payment of the iteration in progress.

## 5. Software, documents and other supplied materials

Quinary releases all the documents produced during the project along with the application components as they are found on the date of final acceptance.

These documents consist of:

- user stories;
- reports;
- release and acceptance statements;
- source code.

Optional *scope* project



**Quinary spa**

via Pietrasanta 14  
20141 Milano - Italia  
t +39 02 3090 1500  
f +39 02 3090 1501

via Mar della Cina 193  
00144 Roma - Italia  
t +39 06 5228 21  
f +39 06 5228 2533

[www.quinary.com](http://www.quinary.com)