Processing Requirements by Software Configuration Management

Ivica Crnkovic[1], Peter Funk[1], Magnus Larsson[2]
[1]Mälardalen University, Department of Computer Engineering, S-721 23 Västerås, Sweden
{ivica.crnkovic, peter.funk}@mdh.se
[2]ABB Automation Products AB, 721 67 Västerås magnus.ph.larsson@se.abb.com

Abstract

Short development life cycles, the importance of time-to-market and fast changes in technology influence the requirements engineering process. Requirements are exposed to changes during the entire development life cycle, and decisions related to requirements and system design are moving toward developers. For this reason it is important to keep requirement changes under control during the entire development process. This control can be achieved by utilizing Configuration Management (CM) functions and in particular Change Management.

This paper describes a model for managing requirements using CM functions. A requirements specification is defined as an hierarchic structure, in which elements of the structure are isolated requirements designated Requirements Specification Items. Having items under version control it is possible to get a better overview of the requirements change process. In the implementation phase, requirement items are associated with Change Requests which define implementations to be introduced in the system. When using Change Requests as links between requirements and the implemented functions we achieve a greater awareness of requirements and a better overview over the requirement process. Furthermore it provides a foundation for reuse of requirements when new systems are built.

1. Introduction

Requirements are exposed to changes during entire development lifecycle. They are often incomplete and inconsistent when first defined.

The process of the requirement clarification does not finish with a well-defined requirement specification. The process continues also during the development process, especially when using software development models such as incremental development, prototyping, etc. In managing requirements we are now facing new problems, such as:

- Faster changing technologies, methods and tools used in the design, implementation and change processes as well as changing technologies for implementing the system (hardware and software), frequently even during he development of the system.
- The time-to-market becoming a particularly important factor. The use of external components, such as COTS (commercial-off-the-shelf), influences system design and requirements.
- Those working with requirements not always completely understanding the new technologies, and developers, mastering development technologies, not being familiar with the background to requirements.

This means that the requirements are far from being correctly defined at the beginning of the implementation process.

All these factors influence the relation between the system design and requirements. The final decisions about requirements tend to become the responsibility of the developers.

Requirements are increasingly exposed to changes, and the following problems become more apparent:

- Inadequate use of requirements and inadequate guidance by requirements during software design and development.
- Inadequate awareness of changes in requirements during system development.
- Inadequate control over which requirements are under implementation, which are fully implemented.
- Inadequate validation and verification of implementation against the requirements.

The basic idea in this paper is to keep requirements under Configuration Management (CM). This paper presents a model, in which requirements are under strict CM control. Using features from CM, especially from Change Management, it is possible to achieve better control and

awareness of the requirements. It simplifies tracing of requirements by means of reuse or by identification of problems.

For efficient CM control, every requirement is saved as a separate configuration item. By means of this separation it is easier to control particular requirements.

The structure of a requirements specification and the interaction of requirements with CM features are described in chapter 2. Chapter 3 describes the behavior of the requirements during the implementation phase. Their relation to Change Requests, documents which describe a change to be implemented in the system, are discussed. Using Change Requests as links between the requirements and implementations promotes awareness of the requirements and the entire development process. Chapter 4 discuses certain limitations of the model. Finally, chapter 5 provides a conclusion of the paper.

2. Requirements Specification Structure and Configuration Management

The idea is to keep the requirements under CM control. The same CM process and tools which are used in the implementation phase can also be used for versioning requirements.

2.1 Requirements Specification Structure

If a requirements specification is prepared as one document, we can manage different versions of it, but we cannot control the requirement items separately.

For this reason we define a requirements specification as a “virtual” document which consists of Requirements Specification Items (RS Items), organized in a structure as shown in Figure 1.

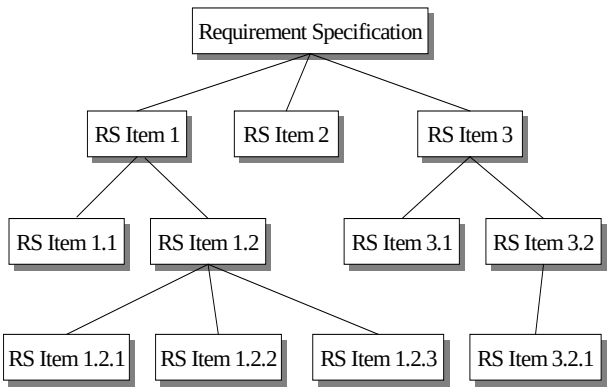

FIGURE 1. Requirements Specification

An RS Item is the description of one particular requirement. The top level of the structure contains basic requirements and these requirements are refined on the lower levels.

RS Items can be described in a natural language, but there are other alternatives, such as: forms, design description languages, graphs, or formal specifications [5].

A predefined format of an RS Item makes it possible to extract particular information from the entire requirements specification or from a part of it (i.e. from the entire tree or from a sub-tree).

To process a requirements specification efficiently, we need an application, a so-called “RS browser”, which can manage individual RS Items, a group of items, or the entire requirements specification.

2.2 Versioning Requirements Specification

As RS items are separate entities we can place them under version control. Every RS Item is treated as an CM item. We can create new items, or modify or remove old items. We get new versions on those items which we change.

The identification of each item version is a part of the CM tool: Each Item has an identity, title and other features related to each version, such as version identity, author, creation date, and a set of other attributes, depending on which CM tool we use.

The hierarchic structure is also managed by the CM tool. A logical relation between different RS items, such as requirements on inputs or outputs, can be a part of the RS item data. The requirements specification structure is however defined within the CM environment, and not within the RS items themselves. It is thereby easier to move RS Items within the structure.

A particular version of the requirements specification is generated from specified versions of RS items, i.e. from a specific configuration of RS items, designated a baseline [4].

2.3 Processing Requirements

We also use other CM features to manage the requirements change process. Using CM report facilities we can easily trace:

- Which requirements, or parts of them, have been changed since the latest baseline.
- Who has made the change.
- When the change was made.

Version attributes can be used to describe possible states of RS Items, as shown in Figure 2.

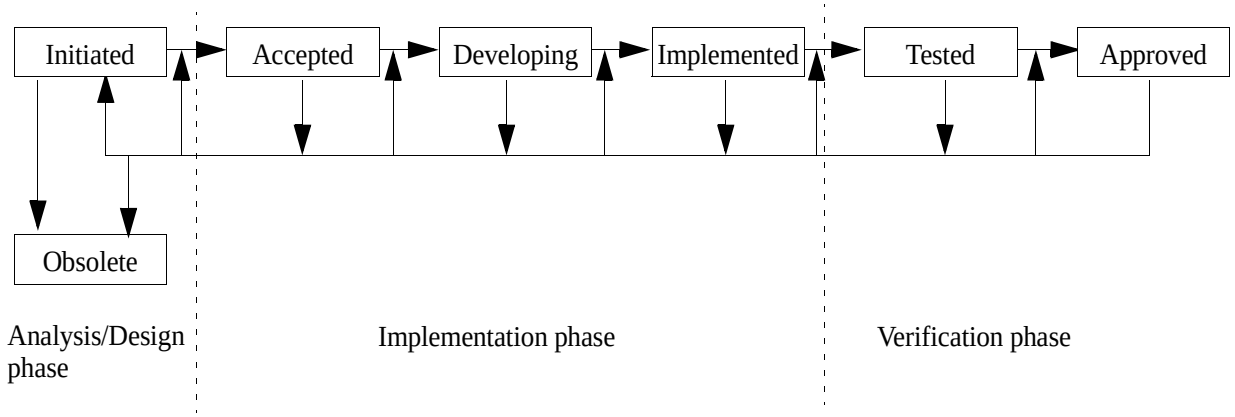

FIGURE 2. RS Item States

As RS Items are organized in a hierarchic structure, the functions we can apply to them can affect individual RS Items or a sub-tree of Items. For example, we may want to designated all RS Items lying under a main RS Item as Tested. Such functions are provided by certain CM tools.

3. Processing Requirements during the Implementation Phase

When we place Requirements Specification under version control, we obtain control over the requirements changes. This control is however not sufficient, we also need a mechanism to relate requirements to the implementation parts. This relation is often not clearly defined, especially in the beginning of the development process when requirements are not clear and not completely understood.

In general, we have many-to many-relations between RS Items and system modules[1]: A module can be affected by several requirements, and, on the contrary, a requirement can be implemented in different modules.

These relations are even more complicated if we take into considerations that some requirements are related to other requirements, and the same applies to modules. System modules can consist of a number of items, such as documents and source files, and in such a complex relationship keeping track of requirements is a challenge.

We wish to make requirements more visible during the implementation. Developers should be aware of the requirements they are expected to implement, and they should be aware of possible changes in the requirements, of which requirements are being implemented and of which new requirements have appeared.

One possibility of relating requirements with system modules is to use Change Management support from CM. CM provides a basic support for change process operations and the CM functions can be utilized for providing links between requirements and the final result of the development process.

The basic purpose of the change-oriented CM tools is to control changes instead of files. Change management controls the connections between logical changes and physical changes implemented in the files within a system. The term Change Request is used to refer to a logical change [1], [2], [6], [7], [9], [11]. Any kind of change can be addressed by Change Request, for example a request to solve a problem or improve a function or implement a new function. To allow changes to be managed at the logical level rather than the physical level, it is essential to associate physical changes, i.e. changes performed on files, with Change Requests.

SDE, a CM tool developed and used at ABB [3], uses Change Requests and, differing from other CM tools, keeps Change Requests under version control [4].

As requirements are starting points for the design and implementation of a system, it is natural to relate them to Change Requests which initiate activities in the implementation phase.

Change Requests can be also used for implementation of completely new functions, assuming that they change an empty set to something new. A more appropriate name in this case would be Implementation Requests.

1. By a system module we mean a part of the system which can be designed, developed and maintainend relatively independent of other modules. For example, a component library is a system module.

3.1 Associating Requirements Specification Items with Change Requests

We treat requirements specifications in the same way as any other item under version control. We define a tree structure under CM control which we use for RS items. For example, if we use a check in/check out CM model, we check out and check in items from the tree structure or those RS items which we wish to modify. The check in/out procedure is performed by an RS Item Browser, or simply by CM functions.

In addition to the standard version management of the requirements, we also define a bidirectional relation between RS Items and Change Requests.

To avoid a risk that the relation to Change Requests remains unclear, it is important to define:

- Which types of relations exist between Change Requests and system modules on the one side, and on the other side between Change Requests and RS items.
- How the process of information exchange between Change Requests and RS Items works, and how to ensure a consistent description of the entire process.

We can group Change Requests in the same way as RS Items. If a CM tool does not support this grouping, it is possible to develop additional functions managing this grouping. Change Requests grouped together are related to the same logical change. This means that they are related to the same RS Item. Each Change Request can refine the requirement, or can refer to another part of the system. For this reason, we can relate Change Requests from one group to an RS item. When it is decided to begin with the implementation of the requirement, one or several Change Requests are created.

Change Requests uniquely address system modules where the requests are to be implemented (Figure 3).

The relation between RS Items and Change Requests is bidirectional. An RS Item refers to the associated Change Request, and, if a Change Request is generated from an RS Item, it contains a reference to the corresponding RS Item.

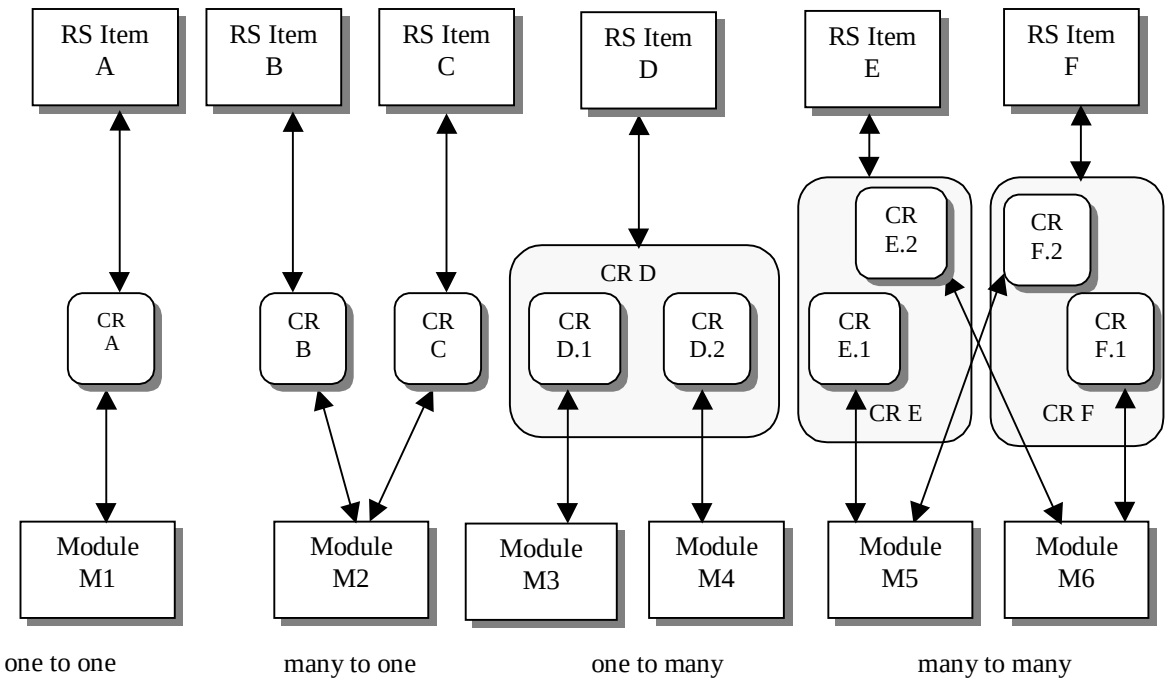

FIGURE 3. Relation between RS Items, Change Requests and system modules

One of the intentions of this model is to increase the awareness of the developers of requirements during the implementation process. This can be achieved by making requirements more visible to them. Since developers deal with Change Requests - they refer to Change Requests when performing check out/in - they can access RS Items via Change Requests. The CM tool should provide this possibility. RS Items specified in Change Requests can in fact be treated in a way similar to any other item (source code, document). All versions of items related to a Change Request should be easily accessible and from it.

3.2 Processing RS Items and Change Requests

During the requirements engineering process and during the implementation process both RS Items and Change Requests are being changed, which means that new versions of them will be created. This implies that only particular versions of RS Items and Change Requests make a consistent combination. The consistency between different versions can be treated in the same way as the consistency between source code files. Taking care of the consistency is a typical CM function - one possibility is to relate the latest versions of all items and generate a baseline at the synchronization points. A baseline in this case will comprise source code, RS Items and Change Requests into a manageable entity.

Figure 4 shows a comprehensive process of managing RS Items, Change Requests and system modules.

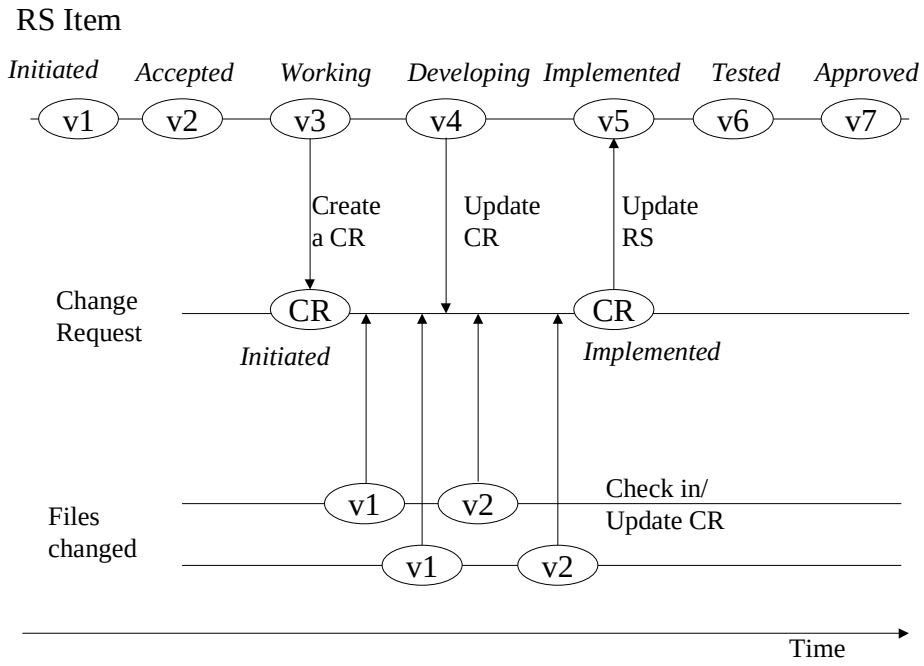

FIGURE 4. Processing RS Items, Change Requests and modules

In the first phase of the requirements engineering process the RS items will be changed (or removed and new added) by means of to analysis and validation, problem identifications and requirements negotiations [8], until they reach the Accepted state.

The second phase, the implementation, begins with the initiation of Change Requests. The RCS Item reaches the Developing state. In this state RS Items are being implemented. A new Change Request is generated from an RS Item. The new Change Request inherits some parts of the RS Item such as title, identity, etc. It also contains a reference to the RS Item and its version. The implementation phase includes implementation of the requirement, i.e. creation and modification of items of modules. Information about which files and versions have been modified is saved in the Change Request. A developer is responsible for correct updating of Change Requests with files being modified and placed under CM control. When performing these actions the developer can easily refer to the related RS Items and in that sense become aware of it.

It can happen that RS items are modified even during the implementation phase. In such a case, the engineers responsible for the requirements can easily determine the current state of the RS Items and how of the implementation has been performed. This data can be suitable input for an analysis of the implications the change can have. If an RS Item has been changed, the associated Change Request will be updated - a new version of the RS Item and a short change description will be recorded in the Change Request. The Change Request owner will be informed simultaneously (via e-mail) of the change in the RS Item and Change Request.

When the required changes are implemented the developer responsible for the Change Request closes it. This action triggers an updating procedure which sets the corresponding RS Item in the Implemented state. The tests and verification activities now performed conclude with approval and closing both of Change Request and RS Item.

4. Limitations of the Model

The presented model contains certain limitations which originate in the nature of requirements.

We recognize two types of requirements [10]:

- *Functional requirements* are statements of services which the system should provide, and how the system will respond to particular inputs.
- *Non-functional requirements* are constraints on the system in general, for example standards, timing constraints, quality constraints, etc.

While functional requirements can be more or less directly mapped to the functions which should be implemented, the non-functional requirements usually concern the entire system. For this reason it is difficult to relate Change Requests to the later.

The non-functional requirements are also specified as RS items, but their internal format will not be the same as those of functional requirements. It is difficult to relate them to particular Change Requests. This implies that we can not use the same mechanism for referring to RS Items, and we do not have the same support in achieving a higher degree of awareness of requirements. It must be obtained in another way.

Another problem originates in ambiguities and lack of clarity regarding requirements. If a requirement is not precisely defined it is difficult to relate it to a specific Change Request, which describes a requirement for implementation in a particular system module. Such requirements are more often changed, even during the implementation phase and this causes difficulties in maintaining consistency in relation to Change Requests. The old Change Requests, perhaps partly implemented, become irrelevant, and new Change Requests must be created. Of course such a RS Item indicates the possible problems with the requirement's design. These indications can be detected by measurement - if there are many versions of an RS item and changes in references to Change Requests, a more serious analysis of the RS Item is called for.

5. Conclusion

The goal of the model presented is to manage requirements in a controlled way. This is performed by placing requirements under Configuration Management and by utilizing Change Management functions.

The use of requirements management and its combination with CM has the following advantages:

- It provides support in controlling how and when requirements are changed, if they are under implementation or if they have already been implemented.
- It provides designers and programmers with direct and easy access to related requirements information.
- It simplifies keeping of requirements consistent with implementation and free of redundancy.
- It simplifies reuse of requirements previously used and implemented.

The model can be only partially implemented by using of standard tools. While most of the functions related to versioning and configuring are available in most of the CM tools, the functions related to requirements management must be implemented.

6. References

[1] Continuus Software Corporation, Task-Based Configuration Management, Version 2.0, http://www.continuus.com/developers/developersACEA.html

[2] Continuus Software Corporation, http://www.continuus.com/homepage.html, 1998

[3] Ivica Crnkovic, Experience with Change Oriented SCM Tools, Software Configuration Management SCM-7, Springer Verlag, ISBN 3-540-63014-7, 1997, pages 222-234

[4] Ivica Crnkovic, A Change Process Model in a SCM tool, Proceedings Euromicro 98, IEEE Computer Society, OSBN 08-8186-8646-4, pages 794-799

[5] P.J., Robertson D., Case-Based Support for the Design of Dynamic System Requirements, Advances in Case-Based Reasoning, Selected Papers, Keane M., Haton J.P., Manago M. (eds.), Springer-Verlag, 1995, pp 211-225.

[6] David B. Leblang, The CM Challenge: Configuration Management that Works, Configuration Management, edited by Walter F. Tichy, Jown Wiley & Sons, ISBN 0 471 94245-6

[7] David B. Leblang, Managing the Software Development Process with ClearGuide, Software Configuration Management SCM-7, Springer Verlag, ISBN 3-54063014-7, 1997, pages 66-80

[8] Pete Sawyer, Ian Sommerville and Stephen Viller, Improving the Requirements Process, Technical report: CSEG/30/1997, Lancaster University http://www.comp.lancs.ac.uk/computing/research/cseg/97_rep.html

[9] Rational http://www.rational.com/products/clearquest/index.jtmpl, 1998

[10] Ian Sommerville, Software Engineering, Addison Wesley, ISBN 0-201-42765-6

[11] Darcy Wiborg Weber, Change Sets versus Change Packages, Software Configuration Management SCM-7, Springer Verlag, ISBN 3-54063014-7, 1997, pages 25-35