Real Reuse for Requirements

Executive Summary

A mobile device manufacturer in a highly competitive market needs to deliver the next generation of mobile device to its customers quickly, and at the lowest possible cost. The company wants to adopt a baseline set of requirements for the next-generation project, but must make necessary modifications to leap ahead of the competition.

An automotive supplier must produce embedded software components consistently and reliably for its OEM clients. To do so, the supplier's development process must account for the variations required by each manufacturer.

Requirements reuse provides organizations with the unique ability to share a requirement across projects without absorbing unnecessary duplication of artifacts within a repository. This is a critical capability that accelerates time-to-market and cuts development costs. Shared requirements may be read-only in cases where the requirement may not be changed such as regulatory requirements, or varied by each product version as necessary. Changes to shared requirements can then be propagated to all applicable product versions in an automated fashion.

The concept of reuse is a familiar notion within the software development realm, but less common when considered in the field of requirements management. There are various definitions and use cases that must be taken into consideration when implementing a solution to address requirements reuse.

This white paper discusses the elements that make up a requirement and establishes common understanding of how requirements evolve, and how organizations can reuse requirements to accelerate product innovation, reduce complexity and control costs.

Dissecting a Requirement

To understand the concept of requirements reuse, we must first look at the various parts of a requirement: requirement description, attributes and relationships.

Requirement Description

Describes and specifies the requirement via text, graphics, tables, pictures, etc.

Attributes

This is descriptive data about the requirement, which aids in organizing or using the requirement object within a process. Attributes typically describe the current state of the object and have the same scope as the requirement itself. For instance, attributes may describe the State/Stage within a requirement workflow (i.e., Approved, Rejected, Satisfied and Tested).

Relationships

This characteristic of a requirement allows you to model:

- structure (i.e., Consists Of, Includes)
- history (i.e., Revision Of, Derived From)
- conceptual links or traces (i.e., Satisfies)
- references (i.e., Defined By, Decomposes To)
- security (i.e., Authorized By, Enables)

Any given requirement can have information in each of the description, attributes and relationships categories. When requirements are reused, any or all of the information can also be reused. An organization's chosen requirements management tool needs to have an underlying architecture and user capabilities that support the strategic level of reuse dictated by the demands of the organization. Since reuse can occur at a number of different levels by leveraging the elements of a requirement, flexibility is also critical to solving the reuse challenge.

History, Versions and Baselines

When implementing a complex reuse scenario, or even a system where requirements persist release after release, one must be able to identify significant points in that requirement's evolution. In the development world, these significant points are called "versions." This term may mean different things to different people, so we will begin with a definition of the term "version" as it applies to requirements reuse and show how it relates to similar terms such as history, baselines and milestones.

Consider a system where requirements are captured within requirements documents but are stored as individual items within the repository.

History is the term used to describe the audit trail for an individual item or requirement. All changes made to the item, whether they pertain to the description, attributes or its relationships, are captured in its history. History answers the who, when and what questions with respect to changes to that item.

Version represents a meaningful point in an individual item's history. Not all changes to an item are significant and warrant a new version of the item. For example, the reassignment of a requirement from Nigel to Julia would not require a specific version identifier. The change is recorded to the item's history, but a new version is not created.

Baseline is a very similar concept to version, but has a much different scope. Individual items are often organized into groups or sets. In the requirements management domain, these sets are called "documents" and a baseline is a meaningful point in a document's history. Some organizations use a slightly different definition for baseline. Rather than being a snapshot -in-time for a given document, a baseline, as defined here in the context of requirements reuse, is a specified goal. For purposes of this discussion, we will call the goal-oriented baseline a "milestone" in order to distinguish between the two definitions.

Requirements management claims to allow for the versioning of individual requirements. Many tools support versioning by cloning or copying the entire requirement. Even fewer solutions relate the copy to the original requirement.

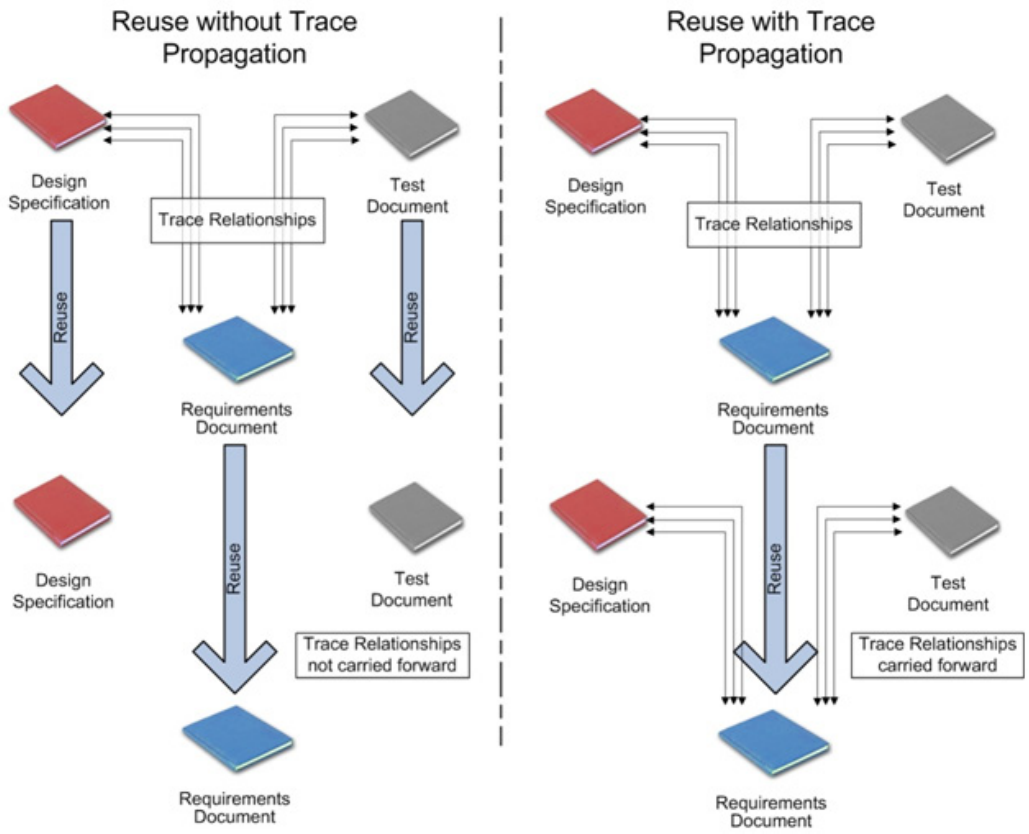
Although related, versioning and reuse are not the same. The concepts of versioning are often confused with those of reuse. In the next section, we will explore various reuse scenarios to illustrate the differences – and the benefits – of versioning and reuse.

Reuse or Not Reuse?
The Many Flavors of Requirements Reuse

Requirements Reuse without Reuse – Share

The ability to share an item between projects, documents or other work efforts could be considered a form of reuse. Under this definition, all of the projects that are sharing the item see – and can possibly even contribute to – the evolution of the item. The attributes on the item are shared, as are all the relationships and the description.

This is not actual reuse. It can be debated that this is not reuse at all, but it is included here for completeness.



Requirements Reuse without Heritage – Copy

As mentioned previously, copying an object from one place to another can also be considered a form of reuse. In fact, this is the form of reuse that Microsoft Word (or any other non-Requirements Management tool) supports. When analysts open a document, select some content and perform a copy/paste operation into another document, they are reusing that content for a new purpose. This form of reuse has no knowledge of heritage or "Where did I come from?" and, of course, changes in one document have no impact on changes in the other. In fact, changes are completely independent and one document has no knowledge that change occurred in the other, let alone what the change might have been.

This is not actual reuse. Any "flavor" of reuse must minimally include a pointer indicating the source of the original content.

Requirements Reuse with Heritage

Given the above scenarios, let us assume you can answer the "Where did I come from?" question. Augmenting the copy with the pointer back to its origin provides several options for reuse. It is the manner in which this link is leveraged that will differentiate each of the following reuse models. Most requirement management tools available today have some notion of links or relationships – if not at the individual requirement level, then at the document level. Unfortunately, document level links are not very powerful. In the long run, they don't answer the traceability question in sufficient detail to be meaningful. Having a link to an item's origin is the start of actual reuse, though it is certainly not the end.

Requirements Reuse with Change Notification

In this situation, a requirement and all related information, is reused in its entirety. Project state determines the state of the requirements at the time of reuse, and any change to requirements in a reuse scenario causes a ripple effect, flagging all artifacts related to those requirements as suspect.

Requirements Reuse with Change Control

Reuse with Change Control is similar to Reuse with Change Notification in that requirements are reused in their entirety. This seems quite similar to the Share topic discussed above. However, there is one significant difference: the two projects sharing the same requirement only share it until one project needs to change it. When the information changes, a new version/branch is created, and only items referencing that new version are declared suspect. All other projects or documents are unaffected.

Requirements Reuse with Annotations

In the two reuse paradigms immediately above, the requirements and related information are reused in their entirety. In Reuse with Annotations, only some of the information belonging to a requirement is identified as a candidate for sharing and reuse. The rest of the information is specific to the project or document. The shared information is held in the repository while the other information belongs to the project or document reference. Each instance of the requirement being reused has its own attributes and relationships. The project or document state is, or can be, independent of the state of the requirements that are contained within it. New versions of the requirement are automatically created when the shared information in the repository is changed. These changes that trigger new revisions can make other references suspect, as well as other items in the system, by the ripple effect of that change. For example, changes to requirements may affect test cases or functional specifications downstream.

Once you have project or document independence in terms of the metadata/attributes, you have the ability to model both a dynamic (share) and static (reuse) form of reuse at the same time. The project manager or analyst decides if they want to remain consistent with the evolving requirement in a dynamic way or if they want to lock the requirement down, such that the impact of change does not affect their project.

Requirements Reuse with Annotations and Change Management

Applying change and configuration management paradigms to the requirements management discipline in a single integrated, traceable solution can bring the power of reuse to a new level. Incorporating a process on top of reuse and controlling how and when requirements can be modified and reused enables you to experience these benefits without unnecessarily branching and versioning objects, unless it is authorized and appropriate to do so. Requests for Change (RFCs) come in, get filtered and are directed by various review boards. Some of these RFCs get approved and assigned to users to affect the requested changes. Ideally, this change management process can define what type of changes can be made – whether it is modification, branching, applying a baseline or other gestures. Only then can an engineer modify the requirement, causing the system to version and branch accordingly, and notify the related constituents appropriately.

Clearly, there are additional reuse models that are not described here. This paper provides only a sampling. The business needs will help determine which model is most effective for an organization.

Is Requirements Reuse Right for Your Organization?

Requirements reuse is not for everyone. There is a broad spectrum of need in terms of requirements management tooling in the market today, and organizations first have to know where they lie on the requirements maturity curve.

Many companies are still in the infancy of requirements management. They have not yet adopted a requirements management tool, and are currently using business productivity applications such as Word or Excel to capture and track requirements. These organizations are not yet at a point of requirements sophistication where reuse support is necessary. They may look for capabilities such as ease of document import, rich text support, and downstream traceability to ease business adoption.

However, if an organization is progressing on the maturity curve with respect to requirements management, managing multiple projects and thousands of requirements in parallel, and seeking to reduce complexity, lower cost of development, and shorten innovation cycles, then requirements reuse is a concept that should be investigated.

Integrity Business Unit Locations

North America
1 800 613 7535

United Kingdom
+44 (0) 1252 453 400

Germany
+49 (0) 711 3517 75 0

Asia Pacific
+65 6830 8338

Japan
+81 3 5422 9503

integrityinfo@ptc.com

For more information visit: PTC.com/products/integrity

© 2012, Parametric Technology Corporation (PTC). All rights reserved. Information described herein is furnished for informational use only and is subject to change without notice. The only warranties for PTC products and services are set forth in the express warranty statements accompanying such products and services and nothing herein should be construed as constituting an additional warranty. References to customer successes are based upon a single user experience and such customer's testimonial. Analyst or other forward-looking statements about PTC products and services or the markets in which PTC participates are those of the analysts themselves and PTC makes no representations as to the basis or accuracy thereof. PTC, the PTC logo, and all PTC product names and logos are trademarks or registered trademarks of PTC and/or its subsidiaries in the United States and in other countries. All other product or company names are property of their respective owners. The timing of any product release, including any features or functionality, is subject to change at PTC's discretion.

7512-Integrity: Real Reuse for Requirements-WP-EN-0312