

# An Innovative Approach to Managing Requirements

## Executive Summary

Requirements may be the most critical aspect of the product development lifecycle. Effective requirements management practices ensure that accurate requirements are readily available to all project team members and are only changed under controlled circumstances. Requirements that are agreed upon and approved help ensure that business objectives are met. A common source of easily accessible, up-to-date requirements enables members of the project team to work more efficiently. Analyzing the impact of changes to requirements before they are made, and alerting project team members when changes are made, enables change to be managed more effectively. Leveraging the requirements assets and all related artifacts between projects over time minimizes rework and allows project teams and the organization as a whole to streamline processes, increase productivity and dramatically reduce time to market.

This paper provides a thorough and detailed examination of a comprehensive requirements management solution. The paper begins by summarizing the high level benefits of an effective requirements management approach, the role of requirements management within Application Lifecycle Management (ALM) and how Integrity, a PTC product, can seamlessly and effectively manage requirements through its single platform, single architecture approach. This paper examines how requirements are authored, captured and traced through the downstream lifecycle, how companies can utilize best practices such as parallel development and reuse in relation to requirements, and how configuration management concepts such as versioning and baselining can be leveraged to achieve advanced requirements management capabilities.

## First Generation Requirements Management Tools

First generation requirements management solutions are disconnected from the development process. When separate tools are used for requirements management and development, data does not flow smoothly between the analysts who generate the requirements and the developers who build the end product. Separate repositories for requirements and development data also limit the ability of analysts to get a view into the progress of their requirements or upper management to get a cohesive picture of activities across the organization.

Although many organizations are interested in requirements management and recognize it as a critical capability, the overall market share for requirements management tools is comparatively small. This discrepancy can be partly attributed to the fact that most of the requirements management tools available on the market are too restrictive, forcing organizations into development processes that don't fit their unique situation. Existing solutions prescribe traditional waterfall processes or Agile processes, while most development environments consist of modified or hybrid processes or multiple teams using mixed processes. Flexibility to accommodate this and evolve as processes improve and adapt is critical in a requirements management solution.

As the pace of software development accelerates, the need to support parallel development projects and leverage configuration management concepts and practices is moving outward from the development group to the other domains in the application lifecycle. Highly dynamic projects need to leverage project assets and stay connected to the activities of related projects as changes now ripple across project boundaries. Tools which support traceability, which traditionally is essential from requirement to code, need to extend this support to include project-to-project, version-to-version, and all the elements that make up each deliverable and asset--be it source code, test case or requirement--in the system.

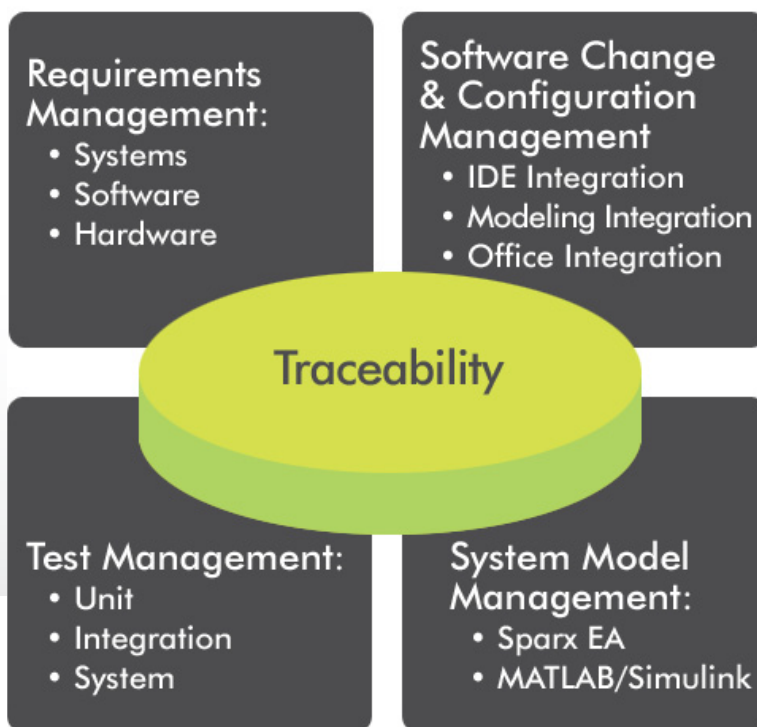
## Integrity for Requirements Management: A Unified Approach

Integrity for Requirements Management enables you to capture, store and manage requirements as a unified part of the development process. Integrity's requirements management ability is built as an extension of Integrity's powerful process management and workflow engine. Analyst, development, quality, and build teams collaborate using a single lifecycle management platform for requirements capture, traceability, change management, and management of development, testing, and deployment tasks.

This integrated approach is not only more cost effective, it also facilitates intra-team and intraproject communication. Analysts can easily determine the impact of a proposed change by reviewing the status of all work in progress to implement the requirement. The development team has easy access to up-to-date requirements and is automatically notified of any changes. Project managers can assess the impact of change, as it occurs, within reused assets. Senior management can get a connected view into all phases of their development projects using powerful querying, charting, reporting, and dashboarding features.

Integrity provides a clear and visible connection between development objects such as source code or documentation and their associated requirements. This traceability satisfies and compliance initiatives and better leverages the assets across the organization.

## Integrity SSLM



Integrity includes all of the features demanded of a powerful requirements management solution but takes each to the next level providing users with the control to manage change effectively through:

- **Document view and rich text support** – You can prescribe and capture requirements into the system in a familiar document-style interface in addition to the traditional flat list and hierarchical tree views of the data;
- **Named relationship fields** – You can define how your requirements, test cases and development activities are related;
- **Suspect link support and automatic notification** – You can set up rules that determine what changes to requirements result in related development activities being automatically flagged for further investigation;
- **Traceability and impact analysis through relationships** – You can navigate the hierarchy of requirements and their associated development activities for traceability and impact analysis of changes; and,
- **Historical navigation and reporting** – You can see the contents of a single requirement or an entire document at an arbitrary point in its history, you can see how the contents changed between any two points in time and you can start new work, branching the document and its contents, from any baseline or selected point in time.

Integrity's advanced single ALM architecture offers capabilities far beyond those found in first generation requirements management offerings that include:

- **Requirements change management** – Integrated change management capabilities allow you to control requirements churn, keep a handle on project scope and more effectively delegate, authorize and assign work across team members;
- **Requirements reuse and persistence** – You can logically associate groups of requirements and re-use them in a parallel development scenario while maintaining full traceability, history and genealogy;

- **Requirements versioning** – You can tap into a data model that enables configuration management of requirements, test cases and other artifacts within the system which not only mirrors how source code assets are managed but also enables full traceability to those assets; and,
- **Requirements baselines** – You can securely identify a document or a set of requirements at any point in history and not only navigate the system based on that identifier but begin new work from those existing baselines.

The remainder of this paper provides a more in depth review of core capabilities of Integrity's advanced requirements management offering.

## Authoring and Capturing Requirements

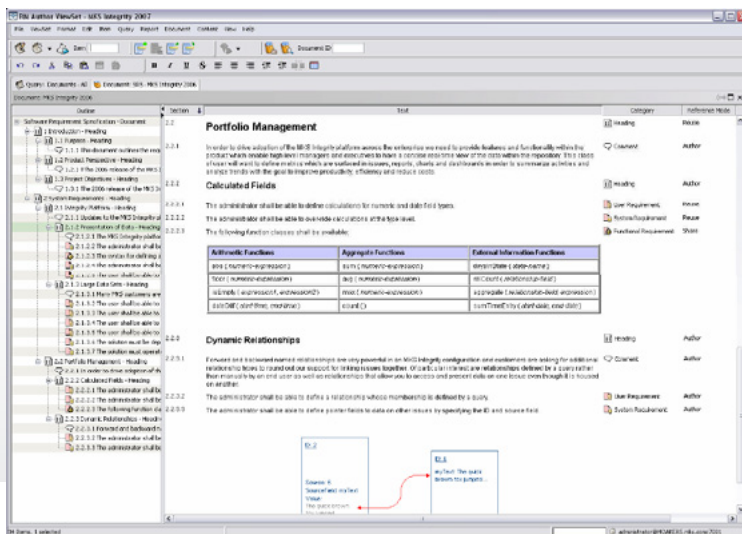
Authoring and Capturing Requirements Requirements are recorded in Integrity using special item types. There are a number of ways that requirements can be captured and viewed within the Integrity system thereby enabling users to work in an environment and style that is comfortable for them.

### Document View

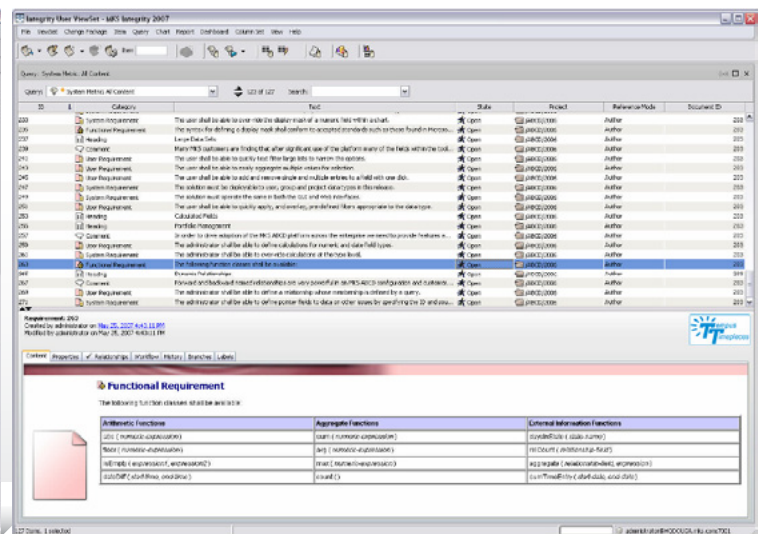
You can enter your requirements information directly in Integrity through its Document view. This feature provides analysts with a familiar format for prescribing and authoring requirements. Users can employ rich text mark ups (bold face, underline, etc.), create tables and embed images and other objects directly into their requirements documents.

### List View

For some users dealing with requirements in a flat list is a more productive method of working. Using the list view is an easy way to spot unwanted duplication in your requirements set or to perform batch operations. For example, you can easily assign work to users, do quick edits and create traces on multiple items in this view.



Document view supports structure, context based authoring, rich text, tables, inline images



List view supports filtering, sorting and inline editing

### Hierarchical Tree View

When structure and relationships are the primary area of interest, rather than the complete content of the item, the hierarchical tree view enables users to view, navigate and create links between requirements and from requirements to any other object in the system. This view is also a good way to see the rollup of metrics and status at each level in the hierarchy in order to quickly spot and correct items that need attention.

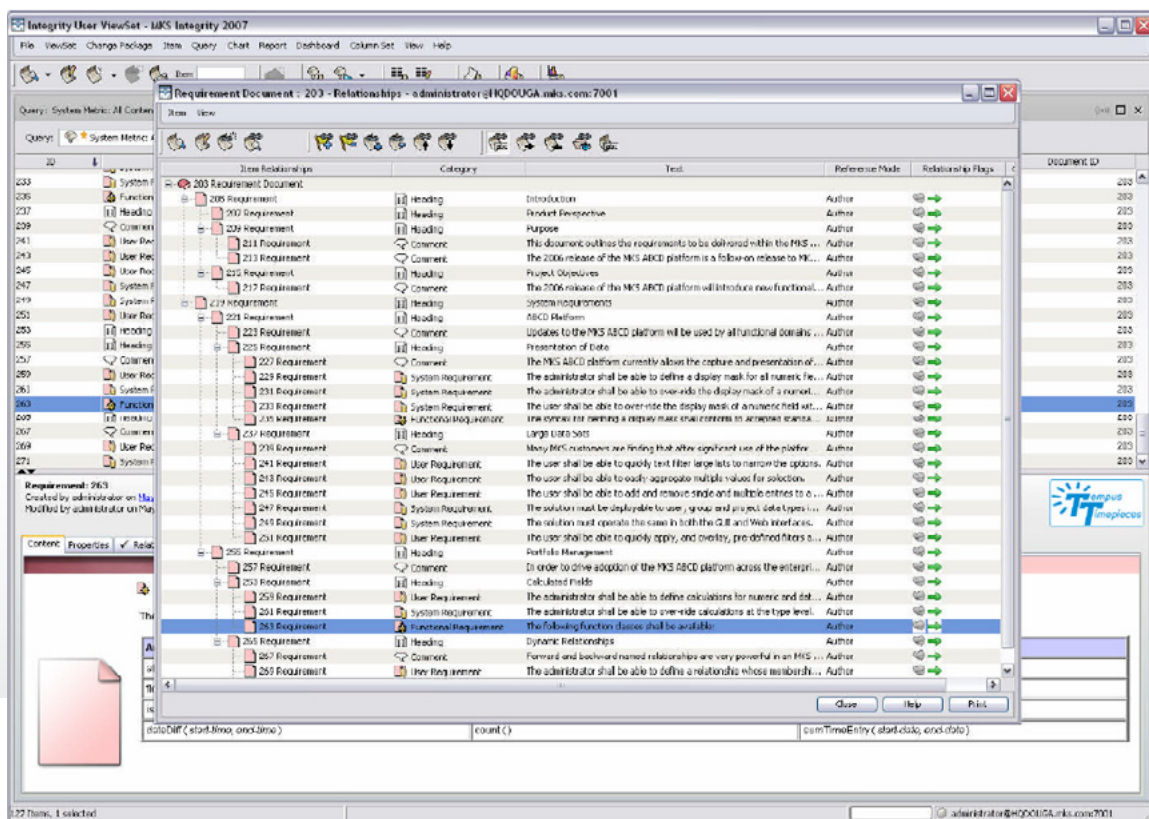
### MS Word, Excel and Project Integrations

For organizations that have used conventional methods such as Microsoft Word or Excel to document requirements, Integrity's integration with these tools allows them to leverage those assets and make use of their familiar authoring tool to build requirements. Completed requirements are imported into the system and Integrity then reproduces the document structure through a linked set of items. Using Integrity's reporting engine, requirements documents can be recreated when

necessary, or, with the Microsoft Excel and Microsoft Project integrations, bidirectional synchronization between Integrity and the external tool can be sustained.

### Other Requirements System Import

The integration of Integrity with requirements tools such as Blueprint Requirements Center, DOORS or Mercury Quality Center enables organizations with existing investments in those products to connect front-end requirements directly to related development activities. For organizations with existing RM investments, the integration with Integrity provides engineers, developers and QA representatives with a cost effective way to bridge the requirements and development phases of the lifecycle through an integrated solution. While not as effective as the single system approach, Integrity acknowledges that this may be a viable transitional approach in some cases.



Relationship view supports hierarchical decomposition

### Traceability of Requirements through the Downstream Lifecycle

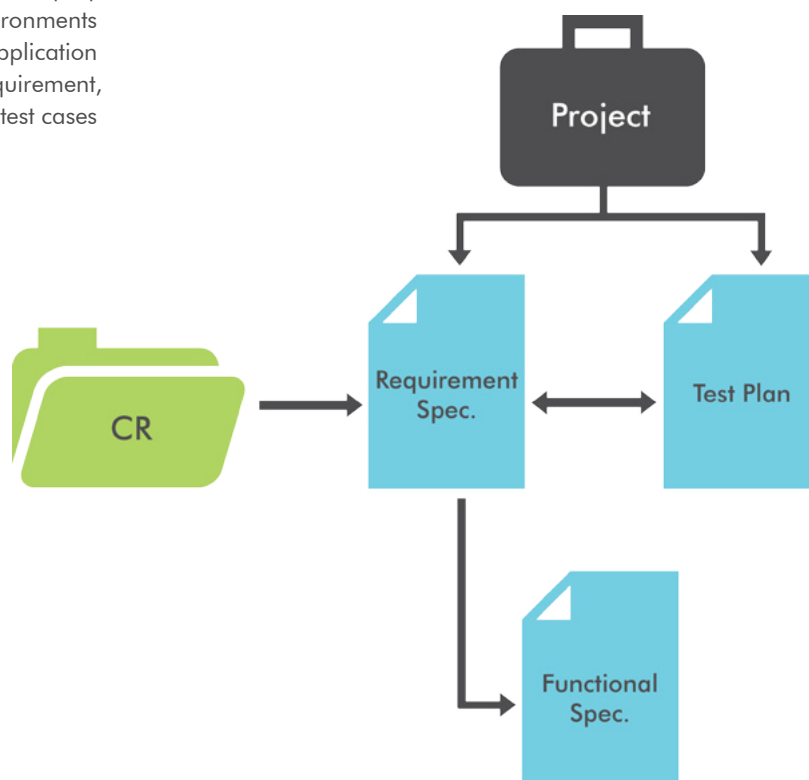
Integrity can seamlessly trace and navigate interactively from projects through various levels of requirements to design features and specifications, assigned tasks, testing and deployment activities, and view activity in context with associated source code changes -- all within in a single system and user interface.

This navigation of linked items provides traceability important for compliance, accountability and audits. It enables you to root out individual components that consistently cause problems. Also, you can confirm that all requirements have functional specifications and test cases so that they are developed and tested before the product is released or project milestones are reached. In a complex parallel development environment each Software Requirements Specification (SRS) document is linked to a test plan as well as a functional or design specification. Individual requirements within the SRS are linked to other requirements in the same document and to individual functional specifications and test cases.

During the lifecycle of a development project, changes may occur that have a potential impact on other aspects of the project. Business climates change, priorities shift and environments evolve; these fluctuations are felt throughout the application lifecycle. For example, if a change is made to a requirement, it could affect the functional specifications, tasks and test cases associated with it; if coding is running late, it could affect the functional specification it is associated with, which in turn could affect other tasks and tests associated with the specification. Ideally, you need to find out what the overall impact of a change will be before it occurs, but definitely as soon as it comes to light.

The Integrity solution for Requirements Management contains a number of item types to capture information across the application lifecycle. A Project item represents a software project and tracks the progress of it. A Requirement document contains requirements, business, user, functional, non-functional, and system needs content. A Functional Specification document tracks the specification and design of features which will implement the requirements. Process items like Tasks, not shown in the picture above, allow for the assignment of specific development work and track any defects that are uncovered in testing that need to be fixed. A Test Plan document contains the test cases and links to items, which tracks testing work and test results that need to be performed to validate a given requirement.

You can customize Integrity's item types by adding fields or field values, adding different types of relationships or removing those which do not pertain to your business. You can also create your own item types with their own metadata, workflow and behavior to augment the out-of-the-box configuration.



## Integrating Your Processes

Integrity workflows define and control software lifecycle processes –inclusive of the requirements phase. Each requirements item type in the system has its own workflow. These separate processes are integrated using rules so that they constrain and/or update each other. By unifying the requirements process with development and test processes, users capture and enforce process controls and interactions throughout the development lifecycle, ensuring a coordinated effort across the development, QA and deployment phases. Integrated workflows connect distinct cultural and organizational silos and prevent them from working in isolation. By unifying and coordinating processes across every stage of the lifecycle, teams are aware of their dependencies.

The following is a sample of some simple cross-workflow rules that could be used in Integrity configurations:

- Design work on a functional specification associated with a requirement cannot be initiated until the requirement is approved;
- Once work starts on a functional specification, the associated requirement is automatically moved to the “In Progress” state;
- A functional specification cannot be moved to the “Complete” state unless all related Task items are in the “Complete” state;
- When all related functional specifications move to the “Complete” state, the parent requirement is automatically moved to the “Implemented” state; and/or,
- A requirement cannot be moved to the “Satisfied” state unless all related functional specifications are in the “Complete” state.

Integrity’s workflow capability goes far beyond a simple level of association. In spite of all the similarities between different kinds of objects stored in an ALM repository, there is one area where you may want them to have significant differences - the workflow.

Within Integrity, workflow can also be attached to tasks or change requests related to the artifact, rather than being part of the artifact itself. Individual requirements are managed artifacts and hence are treated more like source code files than process items and so it makes sense that change requests are used to control their evolution as well. This integrated change management paradigm allows for better control of requirements churn, maintains a handle on project scope, and enables more effective delegation, authorization, and assignment of work across team members.

This model of association allows Integrity to accommodate multiple assigned users who must coordinate activities around a single workflow. There may be multiple workflows and a change request attached to a particular artifact in the system applies to that artifact and to all artifacts underneath it in the hierarchy.

## Analyzing the Impact of Changes to Requirements

To ensure that your final deliverables match your final specifications, a coordinated effort is required across all areas of the development organization and tight control over change must be achieved. Changes that have the potential to affect other activities must be carefully evaluated, managed and the impacts resolved to maintain the project’s integrity after the change.

Understanding the impact of change, then effectively managing it, is critical for the successful delivery of software applications. Changes to existing requirements, or the addition of new requirements, can severely impact the total project’s delivery schedule. For example, if a change is made to a requirement, it could affect the design specifications, tasks, and test cases associated with it; if coding is running late, it could affect the requirement it is associated with, which in turn could affect other tasks and tests associated with the feature. Project teams must be able to easily assess the resulting impact of these changes.

Traditional requirements management systems rely on a traceability matrix. Maintaining the links and dependencies between the items in the matrix can be a very manual and time consuming process; interpreting the matrix in projects of realistic size can be a nightmare. Because Integrity integrates requirements with all downstream development activities, you can navigate the potential impact of a change across all aspects of the application lifecycle, from requirements to coding to testing. By navigating up or down the relationships hierarchy in the Relationships view, you can evaluate the impact of a potential change before making it. This type of analysis is available to anyone in the organization and the information you see is always current and automatically kept up to date.

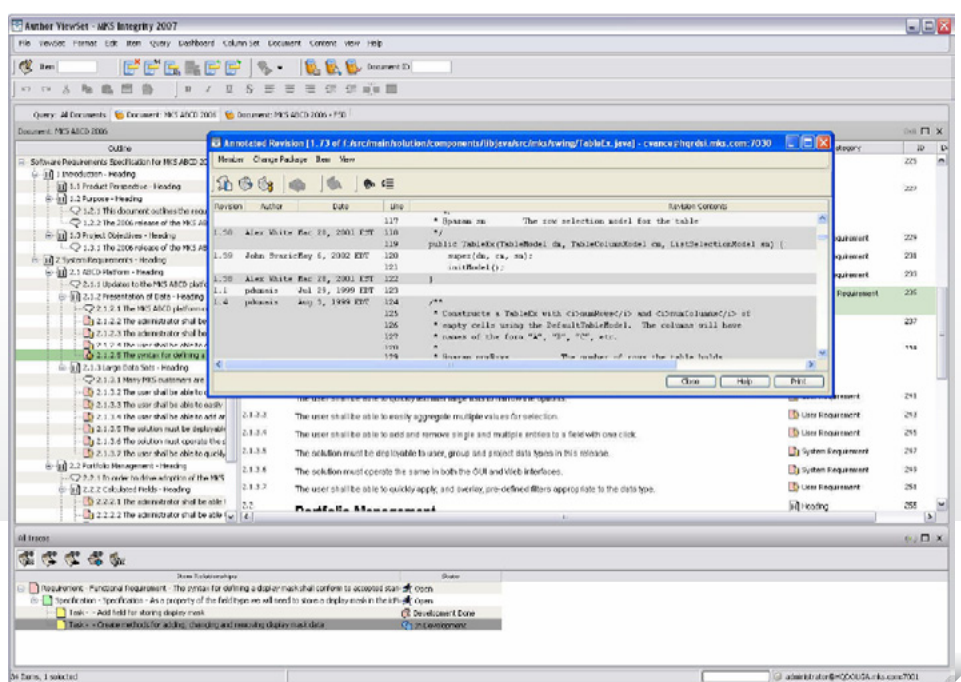
Each requirement should be traceable to a specific project objective. This traceability ensures that the software product satisfies all strategic goals and that individual requirements do not include inappropriate or extraneous functionality. It is important to know the source of all requirements and functionality so it can be verified as necessary, accurate and complete. Using the Relationships view, you can navigate through the network of related items, tracing each requirement to its originator upstream and to each downstream design specification and feature.

This traceability also enables you to review all aspects of your development process to satisfy an auditor, show compliance

with government and safety regulations, or analyze your own processes. The Relationships view enables you to find out information such as why a particular piece of code was changed and thereby trace the relationship. For example, you could trace backward from a source code change to its associated requirement; or you could show the code changes made to implement a requirement by tracing forward from the requirement to the code change. In addition to navigating the Relationships view, you can also print various types of tracability reports.

Integrity supports change management through integrated workflow processes, automatic flagging of suspect relationships and managed suspect relationship resolution. Whenever you revise a Requirement item, all directly linked items are automatically flagged as suspect. When you review your suspect items, if you make changes to key fields, all items that are directly related to the item you have changed are flagged as suspect. At each stage of change, the impact of the change can be analyzed, providing a high level of control.

The automatic flagging of suspect links is controlled through Integrity Requirements triggers and field rules. You can revise the existing rules or add your own. In addition, you can easily configure email notifications so that you are automatically notified whenever any of the items you are responsible for are flagged as suspect.



Tracing relationships from requirement to source code

## Requirements Reuse

Requirements reuse provides users with the unique ability to share a requirement across projects without absorbing unnecessary duplication of artifacts within a repository. Shared requirements can track with the ongoing change by the author or remain at a static point in time as the needs of the project dictate. Further, change to a shared requirement can be made by anyone and the system handles the branching and evolution of that requirement appropriately.

The concept of re-use is a familiar notion within the software development realm, but there are various definitions and use cases which must be taken into consideration when implementing a solution to address requirements re-use. Let's first look at the various parts of a requirement: data, metadata and relationships.

### Data

Describes an object, and is relevant to the object itself. An example of data may be a summary or description of a requirement.

### Metadata

This is data about the data, which aids in organizing or using the object within a process. It typically describes the current state of the object, and has the same scope as the data itself. For instance, metadata may describe the State/Stage within a requirement workflow (i.e. Approved, Rejected, Satisfied, 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); and/or,
- references (i.e. Defined By, Decomposes To).

Any given requirement can have information in each of the data, metadata and relationships categories and when requirements are shared, any or all of the information can also be shared.

Leveraging the powerful data model within Integrity, users may reuse individual or groups of requirements.

Reuse can occur within a number of scenarios leveraging the various parts of a requirement listed above.

## Reuse with Change Notification

In this situation, a requirement and all related information (data, metadata and relationships) is reused in its entirety. Project state determines the state of the requirements at 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.

### Reuse with Change Control

Reuse with Change Control is similar to Reuse with Change Notification in that data, metadata and relationships are reused in their entirety. The difference is that two projects sharing the same requirement only share it until the point in time where one project needs to change it. When the information changes a new version is created and only items referencing that new version are declared suspect. All other projects or documents are unaffected.

### Reuse with Annotations

In the two previously discussed reuse paradigms the requirements and related information (data, metadata, and relationships) 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 metadata 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 suspect other references, 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.

There are other models of reuse that can be described here, but your business will determine which model is most effective for you. The requirements management tool should, and in the case of Integrity, does allow you to implement the model that is most effective for your business challenges.

The Integrity Requirements solution template demonstrates a model with the benefits of all of the above described models and can be used out of the box or configured to address the specific needs of the business.

## Requirements History, Versions and Baselines

When you implement a complex reuse scenario, or even a system where requirements persist release over release, you must version your requirements much like the development organization versions source code. The term “version” may mean different things to different people, so let us start by defining the term and showing how it relates to similar terms like 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. All changes made to the item, whether it is to data, metadata or relationships, are captured in its history. From here you can discover answers to the who, when and what questions with respect to change on that item.

**Version** represents a meaningful point in an individual item’s history. Not all change to an artifact is significant and warrants a new version of the requirement. For example, if you reassign a given requirement from Nigel to Julia, is that a change that needs a specific version identifier? Likely not. The change is recorded to the item’s history but a new version is not created. Within the Integrity solution what is and is not considered a significant edit is configured by the administrator.

**Baseline** is a very similar concept to version but has a much different scope. Individual requirements are often organized into groups or sets. In Integrity, 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, it is a goal to work towards. For the purposes of this discussion we will call the goal-oriented baseline a **milestone** in order to distinguish between the two.

Tools on the market today for requirements management mention that they allow for the versioning of individual requirements. Many of these tools support versioning by way of cloning or copying the entire requirement and fewer go so far as to relate the copy to the original. The Integrity solution offers true versioning of the requirements artifacts with support for branches (or the ability to begin new work from a historical version of a requirement or document) as well as capturing the complete genealogy or version history of that artifact.

This ability provides an additional dimension to the traceability question –not only can you trace a requirement through implementation and deployment within the application lifecycle, but you can also trace that individual requirement’s evolution over time and across projects, getting a true picture of its use within your organization.

Shared Requirement: 146 - administrator@HQAKERS.mks.com:7001

Change Package Item View

Shared Requirement: 146  
Created by John.Doe (jdoe) on 22-Apr-2007 11:03:41 AM  
Modified by Julia.Goodwin (jgoodwin) on 22-Apr-2007 11:16:36 AM

Content Properties Workflow History Branches

History Order: Most recent first

Modified by Julia.Goodwin (jgoodwin) on 22-Apr-2007 11:16:17 AM

Modified Fields New Value  
Input Revision Date 22-Apr-2007 11:16:17 AM  
Shared Text The following function classes shall be available:

Arithmetic Functions	Aggregate Functions	External Information Functions
abs ( numeric-expression )	sum ( numeric-expression )	daysInState ( state-name )
floor ( numeric-expression )	avg ( numeric-expression )	relCount ( relationship-field )
isEmpty ( expression1, expression2 )	max ( numeric-expression )	aggregate ( relationship-field, expression )
dateDiff ( start-time, end-time )	count ( )	sumTimeEntry ( start-date, end-date )

Modified by Nigel.Smith (nsmith) on 22-Apr-2007 11:15:28 AM

Modified Fields New Value  
Input Revision Date 22-Apr-2007 11:15:28 AM  
Shared Text The following function classes shall be available:

Arithmetic Functions	Aggregate Functions	External Information Functions
abs ( numeric-expression )	sum ( numeric-expression )	daysInState ( state-name )
floor ( numeric-expression )	avg ( numeric-expression )	relCount ( relationship-field )
isEmpty ( expression1, expression2 )	max ( numeric-expression )	aggregate ( relationship-field, expression )
dateDiff ( start-time, end-time )	count ( )	sumTimeEntry ( start-date, end-date )

Shared Category Functional Requirement

Modified by Bart.Simpson (bsimpson) on 22-Apr-2007 11:03:41 AM

Modified Fields New Value  
Input Revision Date 22-Apr-2007 11:03:41 AM  
Shared Category Comment  
Referenced By 147  
Created by John.Doe (jdoe) on 22-Apr-2007 11:03:41 AM

Close Help Print

Every change to every object is captured and viewable as part of the audit history.

## Viewing Historical Items

As mentioned above, Integrity maintains the history for all requirements and represents an audit trail of all changes applied over time. You can use this information to either view and navigate through the item or produce historical reports that reflect the state of the requirement, or requirements documents, at any point in time whether denoted by a date, version identifier or baseline.

For example, if you wanted to see the SRS for Integrity 10.0 as of its Project Plan baseline, you could view it in the document view or run a historical report based on that baseline.

Historical reports allow for baselines to be compared. For example, you could use the Project Acceptance and Project Completion milestone dates and run a historical report to see how the content of a development project changed during its lifecycle.

This portion of a comparison report shows the following changes to the document: a change to the text of one requirement (section 2.2.1.1) and to the table within another (section 2.2.1.5), a dropped requirement (below section 2.2.1.5), and two added requirements (sections 2.2.1.4 and 2.2.2.3).

This historical report is just one such example and all defined reports, either by Integrity, administrators or end users, can be run in a historical context as shown.

Navigating the system as of a particular point in time -- be it for metrics analysis or reuse -- and being able to report on change across historical items, is invaluable to controlling churn and scope creep within a project and leveraging prior work across projects. Further, being able to reuse or branch those historical requirements to begin new work enables you to leverage your requirements assets across multiple projects.

The screenshot shows the Author ViewSet - MRS Integrity 2007 interface. The main window displays a document content report for Microsoft Internet Explorer. The report is titled "Document Content Report - Microsoft Internet Explorer" and shows a list of requirements on the left side. The right side of the report displays the content of requirement 2.2.1.5, which includes a table of functions and a section on Dynamic Relationships.

The table of functions is as follows:

Arithmetic Functions	Aggregate Functions	External Information Functions
abs ( numeric-expression )	sum ( numeric-expression )	getState ( state-name )
floor ( numeric-expression )	avg ( numeric-expression )	calcCount ( relationship-field )
isEmpty ( expression1, expression2 )	max ( numeric-expression )	aggregate ( relationship-field, expression )
isMatch ( expression, anytime )	min ( numeric-expression )	isMembership ( state-name, anytime )
timestamp ( quoted-string )	emptyList ()	SLIPCount ( /update .../ )

Below the table, there is a red text block: "The user shall be able to define ad-hoc calculations based on current item data."

The section on Dynamic Relationships includes the following text:

**2.2.2 Dynamic Relationships**

2.2.2.1 Forward and backward named relationships are very powerful in an MRS Integrity configuration and customers are asking for additional relationship types to round out our support for linking issues together. Of particular interest are relationships defined by a query rather than manually by an end user as well as relationships that allow you to access and present data on one issue even though it is housed in another.

2.2.2.2 The administrator shall be able to define a relationship whose membership is defined by a query.

2.2.2.3 The administrator shall be able to define a pick list which is populated by items from the database. Chosen items would conform to a defined type and state.

2.2.2.4 The administrator shall be able to define pointer fields to data on other issues by specifying the ID and source field.

Historical reporting allows for comparison of requirements baselines over timeframes

## Summary

Requirements management is an integral part of developing successful software products. Effective requirement management ensures that product and business objectives are met by providing approved and up-to-date requirements accessible to all members of the project team. It ensures that projects are delivered on schedule by providing a means to analyze and communicate changes to requirements.

Integrity is the only ALM platform available today to provide a single unified solution for managing requirements in direct relationship to all downstream development, QA and deployment phases of the lifecycle. Analysts, developers, testers, and release managers collaborate via a single platform, unified interface and common process. This facilitates a seamless, real-time flow of requirements data between requirement producers and consumers, and allows traceability through every stage of the development process.

For organizations with requirements management needs beyond the capabilities of first generation requirements management tools, Integrity's advanced architecture and single platform design supports sophisticated usage; including requirements versioning, baselining and reuse which enables you to perform complex reuse scenarios, parallel development and requirements configuration management tasks.

Integrity for Requirements Management is the solution for organizations seeking the powerful combination of requirements and process management. Whether you are a business analyst looking for downstream traceability, a development project lead looking to stay in lockstep with business need, or a tester or release manager seeking to ensure the final product is exactly what was ordered, PTC has a solution to satisfy your needs.

## 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](mailto:integrityinfo@ptc.com)

For more information visit: [PTC.com/products/integrity](https://www.ptc.com/products/integrity)

© 2011, 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, Mathcad, Creo, Elements/Pro, 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.