
# Requirements Management

*Enterprise Architect is an intuitive, flexible and powerful UML analysis and design tool for building robust and maintainable software.*

*This booklet explains the Requirements Management facilities of Enterprise Architect.*

# Enterprise Architect - Requirements Management



Printed: May 2010

**Publisher**

*Sparx Systems*

**Managing Editor**

*Geoffrey Sparks*

**Technical Editor**

*Brad Maxwell*

**Special thanks to:**

*All the people who have contributed suggestions, examples, bug reports and assistance in the development of Enterprise Architect. The task of developing and maintaining this tool has been greatly enhanced by their contribution.*

# Table of Contents

**Foreword** 1

## Requirements Management 2

### Model Requirements 4

**Internal Requirements** 6

Make Internal Requirement External 7

### Create Requirements 9

**Requirement Properties** 9

Color Code External Requirements 10

**Extend Requirement Properties** 11

Display Tagged Values On Diagrams 12

**Connect Requirements** 12

**Import Requirements and Hierarchies in CSV** 13

### Manage Requirements 14

**View Requirements** 14

**Trace Use of Requirements** 15

**Manage Requirement Changes** 15

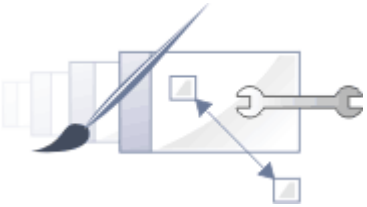
**Report on Requirements** 16

## Index 18

# Foreword

This user guide provides an introduction to the Requirements Management facilities of Enterprise Architect.

# Requirements Management



## Introduction

This section describes the Enterprise Architect *Requirements Management* facilities, and discusses:

- What requirements are
- How requirements are generated and organized
- How Enterprise Architect supports and simplifies Requirements Management.

Additional discussion of the management of requirements in Enterprise Architect can be found in the *Requirements Management with Enterprise Architect* white paper on the Sparx Systems website.

### What is a Requirement?

Requirements are essentially what the system, application or business process *is required* to do. A requirement can be broad and high level, defining - for example - a need for a process to update a particular database. A requirement can also be more specialized and detailed, recording the expectation that - for example - a system call must always be acknowledged by return. Detailed requirements can be organized into a hierarchy culminating in a high-level requirement, so that satisfying each of the detailed requirements results in meeting the higher-level requirements and ultimately the top-level requirement. This hierarchical structure helps manage the complexity of large systems with thousands of requirements and many processes being developed to implement the requirements.

### Gathering Requirements

Gathering requirements is typically the first step in developing a solution, be it for developing a system or a process. Requirements are gathered from all parties expected to use, maintain or benefit from the solution, and are organized into groups, functional areas and hierarchies as necessary. They can be transcribed into a spreadsheet or a requirements gathering or management tool, or they can be created in an integrated modeling tool such as Enterprise Architect.

### Requirements Management and Enterprise Architect

The management of requirements is one of the more problematic disciplines in software development, for reasons such as:

- Diverse group input into the requirements
- Organizational boundary divisions
- Tool boundary divisions
- Volatility of requirements
- Imprecision and ambiguity in natural languages.

These can cause issues with:

- Traceability and
- Integration with change and configuration management systems.

Enterprise Architect can reduce or eliminate these problems in Requirements Management; it is one of the few UML tools that integrate Requirements Management with other software development disciplines in the core product, by defining requirements within the model. Within Enterprise Architect, you can:

- Create 9 and view 14 requirements as entities and properties directly in the model
- Collate the requirements in an external CSV file and then import 13 them into your model
- Detail use cases 5 and scenarios directly in the model
- Enter standard attributes 9 (properties) for each requirement, such as difficulty, status and type, and define your own attributes 11 (properties)
- Trace 15 requirements to Use Cases, business rules, test cases and analysis artifacts (using, for example,

the Relationship Matrix - see *UML Modeling With Enterprise Architect - UML Modeling Tool)*

- Trace and view the impact of changes 15 on requirements (through, for example, the Traceability window - see *Using Enterprise Architect - UML Modeling Tool*) and review the changes 15 themselves
- Create customer-quality MS Word and HTML reports 16 on requirements.

**Note:**

All of these features are illustrated by examples in the *EAExample.eap* model, provided as part of your Enterprise Architect installation in the Enterprise Architect Program Files directory:

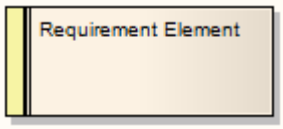
*..\Program Files\Sparx Systems\EA*

# 1 Model Requirements

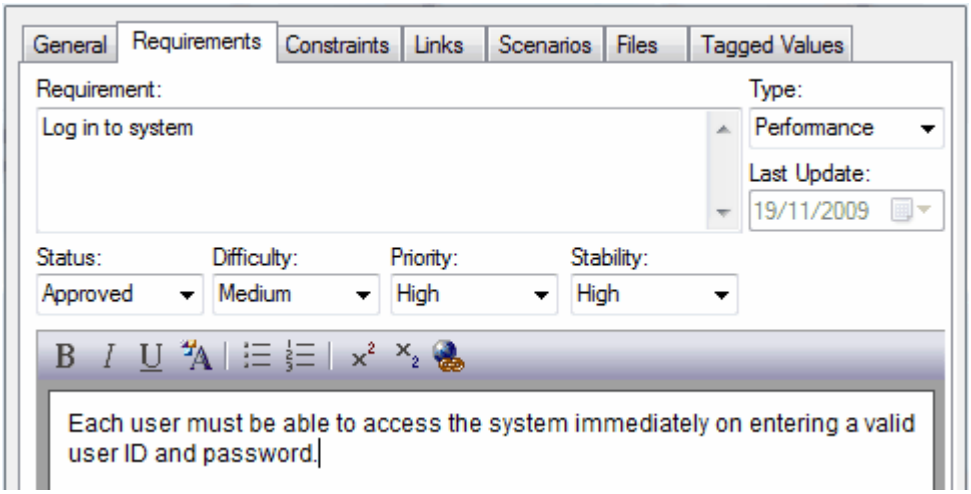
## Represent Requirements

In Enterprise Architect, a requirement can be modeled as an:

- External Requirement - an expectation of the system or process, what the system or process must provide, modeled as an *element* (see the *UML Dictionary*); for example, a business requirement or a stakeholder request - Requirements at this level have their own properties and are reported on separately in RTF reports



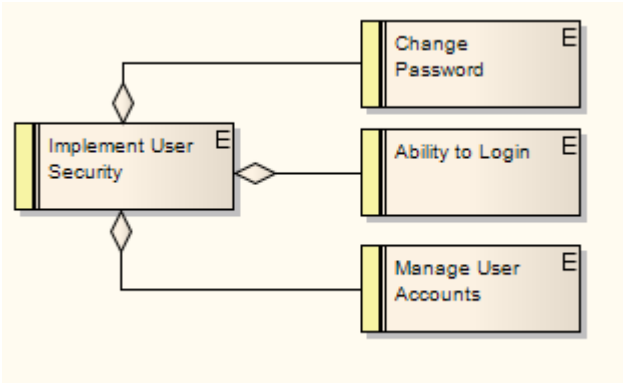

- Internal requirement 6 – a responsibility of an existing element, what the element must do or accomplish, defined as a *property* of the element.




Requirements Management in Enterprise Architect is primarily concerned with *external* Requirement elements and the elements that implement or *realize* them.

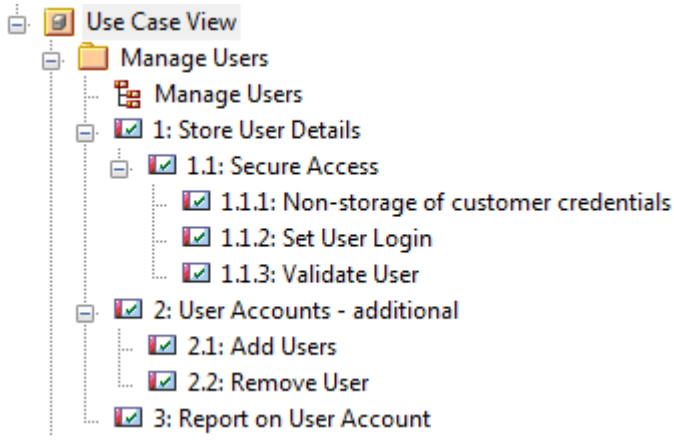
## Requirements in the Model

Requirement elements can be grouped and organized within Requirement diagrams (see the *UML Dictionary*). The Requirement elements are connected to each other by *Aggregation* relationships to form a hierarchy:




It is quite usual to develop a package of many hundreds of Requirement elements, arranged individually and in hierarchies of varying complexity. In the Project Browser you can use the **Show Level Numbering** facility to

highlight the order and arrangement of the Requirements quickly and easily (see *Using Enterprise Architect - UML Modeling Tool)*. The following illustration shows a number of Requirements in a package, where Level Numbering makes the order and arrangement clear:



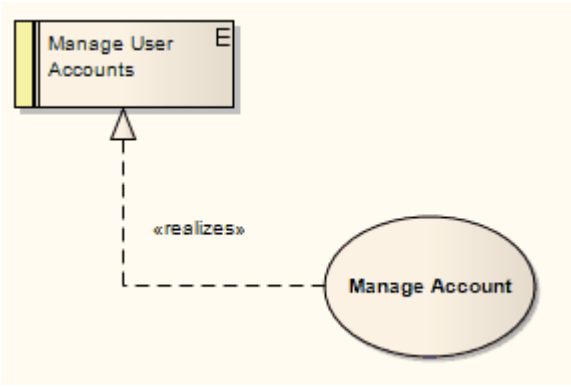

If elements are added, moved or deleted from the package, the numbering automatically adjusts.

**Note:**

This numbering can also be applied in the RTF report generator using the **LevelNumber** field in the **Element** section – *{Element.LevelNumber}.* (See *Report Creation in UML Models.*)

## Use Cases

Requirements are implemented (realized) by model elements such as Use Cases, Classes, Interfaces and Components. There are many ways to *trace* either the Requirement for the feature or service modeled by the elements, or the elements that develop the requirement, most visibly in Traceability diagrams that depict the Requirements and the model elements connected by *Realize* relationships (see *UML Model Management*). The Realize connector enables members of the project team to keep design objectives and development in tandem, and the development path and purpose clear.



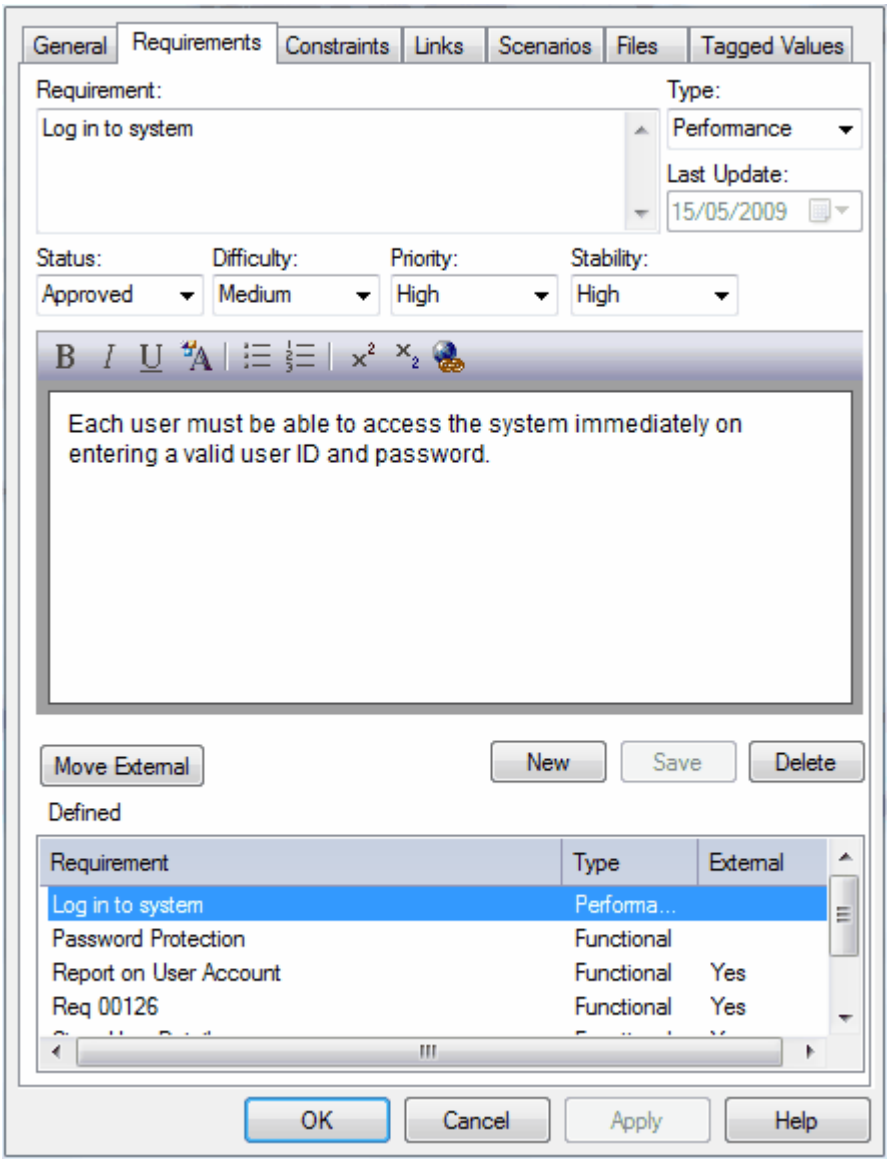

The more usual realization relationship is between a Requirement and a Use Case. A Requirement can be realized by one or more Use Cases, and a Use Case can realize one or more Requirements.

Whilst a Requirement defines a condition that must be met, the Use Case is the key to defining and visualizing *how* that condition is met. A Use Case diagram depicts the logical grouping of actions, processes and components to achieve a required result, and through the use of Actor elements also defines the user and/or system roles participating in the process. (See the *UML Dictionary.*)

Each Use Case (as a composite element) can contain a combination of child diagrams that define in greater detail how a particular activity or facility might be implemented - such diagrams include Sequence, Communication, Activity, State Machine and Business Rule Flow diagrams (see the *UML Dictionary*). The actual implementation of each Use Case is realized by Class, Component and Interface elements organized in their own diagrams. These realizations can also be captured and viewed in Traceability diagrams, depicting the full development pathway from initial requirement through to testing and production.

## 1.1 Internal Requirements

Internal requirements in Enterprise Architect are element responsibilities. They are defined on the Requirements tab of the element Properties dialog (see *UML Modeling With Enterprise Architect - UML Modeling Tool*).




Internal requirements form the functional requirements of the system to be built. The meaning of the requirement can vary depending on which element is the host; for example, a business process requirement might mean something different to a Use Case requirement, which again might mean something different to a Class requirement. In the example above, an internal responsibility to enable the user to login to the system has been defined for the *Login* Use Case. This is a responsibility of the Use Case - an action it is responsible for carrying out - and it applies only to this Use Case.

The significant parameters (or, in Requirement Management terms, *attributes*) are the Type, Status, Difficulty and Priority. Whilst you can provide a detailed description of the responsibility in the **Notes** field, there is more scope in the name (**Requirement** field) to define the nature of the responsibility. An additional field, **Stability**, indicates the probability of the requirement changing; high stability means a low probability of change.

The example Use Case above also has connections to two external requirements, which are system functions that the Use Case implements either in full or in part. You can convert 7 an internal responsibility into an external requirement.

You can also create internal responsibilities for an element using the Scenarios & Requirements window (see *Using Enterprise Architect - UML Modeling Tool*). A responsibility created in the window displays in the element Properties dialog, and vice versa.

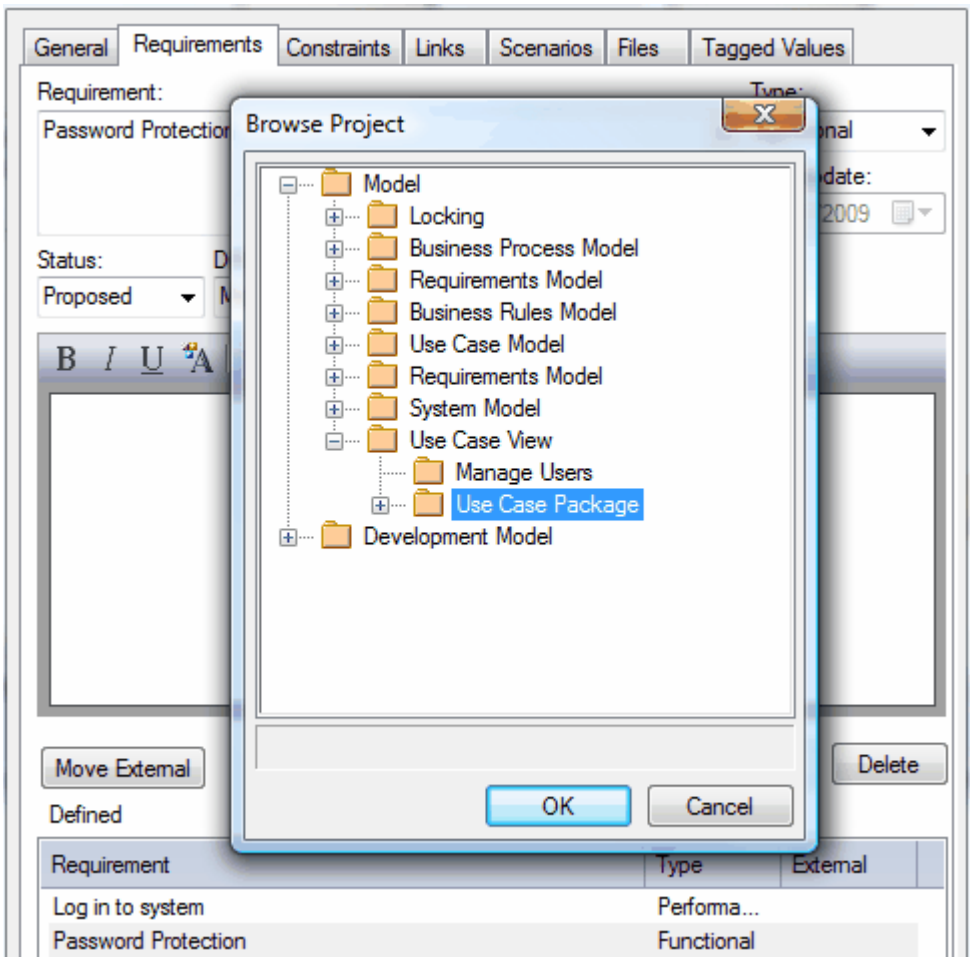
### 1.1.1 Make Internal Requirement External

Elements in Enterprise Architect have internal requirements, or responsibilities (what they must do or accomplish). These often overlap or duplicate more formal requirements that the system in general must meet. You can move internal requirements to external requirements in one go using the **Move External** function.

#### Procedure

If you have defined an internal requirement for an element and want to move it out (where it can perhaps be implemented by multiple elements), follow the steps below:

1. Double-click on the element in a diagram or in the Project Browser to open the element Properties dialog.
2. Click on the Requirements tab.
3. Locate and highlight the requirement.
4. Click on the **Move External** button. The Browse Project dialog displays.




5. Select the package to place the new requirement in.
6. Click on the **OK** button.

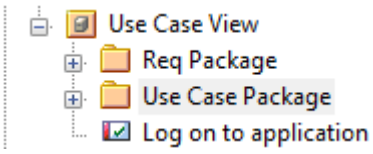
Enterprise Architect creates a new Requirement element in the target package and automatically generates a Realization connector from the element to the Requirement.

Defined

| Requirement | Type | External |
|---|---|---|
| Log on to application | Performance | Yes |

Notice the requirement is now marked external and the dialog fields grayed out. To edit its details 9, double-click on the requirement.

Also notice that a Requirement element has been created in the target package.



# 2 Create Requirements

Within Enterprise Architect you can create external Requirement elements in a number of ways, such as:

- Dragging a *Requirement* icon from the UML Toolbox into a specific diagram 9 (which also adds the Requirement to the diagram's parent package)
- Generating an element within a specific package 9 in the Project Browser
- Importing requirements from a spreadsheet application such as Excel, via CSV 13
- Creating Requirement elements on the Element List for the selected package or diagram (see *Using Enterprise Architect - UML Modeling Tool*)
- Converting an internal responsibility 7 into an external element, in a selected target package
- Importing requirements from another requirements management tool, such as Telelogic DOORS (in this case via the Sparx Systems MDG Link For DOORS integration tool).

**Notes:**

- The Requirement element name can be simply descriptive text, with or without a manually-typed reference number. However, as requirements often have to have a unique reference for external checking, you can use the Enterprise Architect auto-numbering facility to automatically apply a numbering system with or without prefixes and suffixes (see *UML Modeling With Enterprise Architect - UML Modeling Tool*). Set the element type to **Requirement**.
- External Requirement elements can be displayed with or without an identifying **E** in the top right corner. To toggle display of this letter, select or deselect the **Show stereotype icon for requirements** checkbox on the Options dialog, Objects page (**Tools | Options | Objects**).
- Requirement elements can be color coded to indicate their status; see the *Color Code External Requirements* 10 topic.

## Create Requirement Elements Within a Diagram

To create Requirement elements in a diagram, follow the steps below:

1. Open the Custom pages on the Enterprise Architect UML Toolbox (see *Using Enterprise Architect - UML Modeling Tool*).
2. Drag the *Requirement* element onto the current diagram.
3. Enter the **Name** and other details 9 for the requirement.

Enterprise Architect creates a Requirement element in the current diagram and in the diagram's parent package.

## Create Requirement Elements in Project Browser

To create a requirement directly in the Project Browser, follow the steps below:

1. Right-click on the required parent package to open the context menu.
2. Select the **Insert | New Element** menu option. Alternatively, press **[Ctrl]+[M]**.
3. In the New Element dialog select the **Requirement** type.
4. Enter the **Name** (or select **Auto**) and click on the **OK** button.

Enterprise Architect creates a requirement in the current package.

## 2.1 Requirement Properties

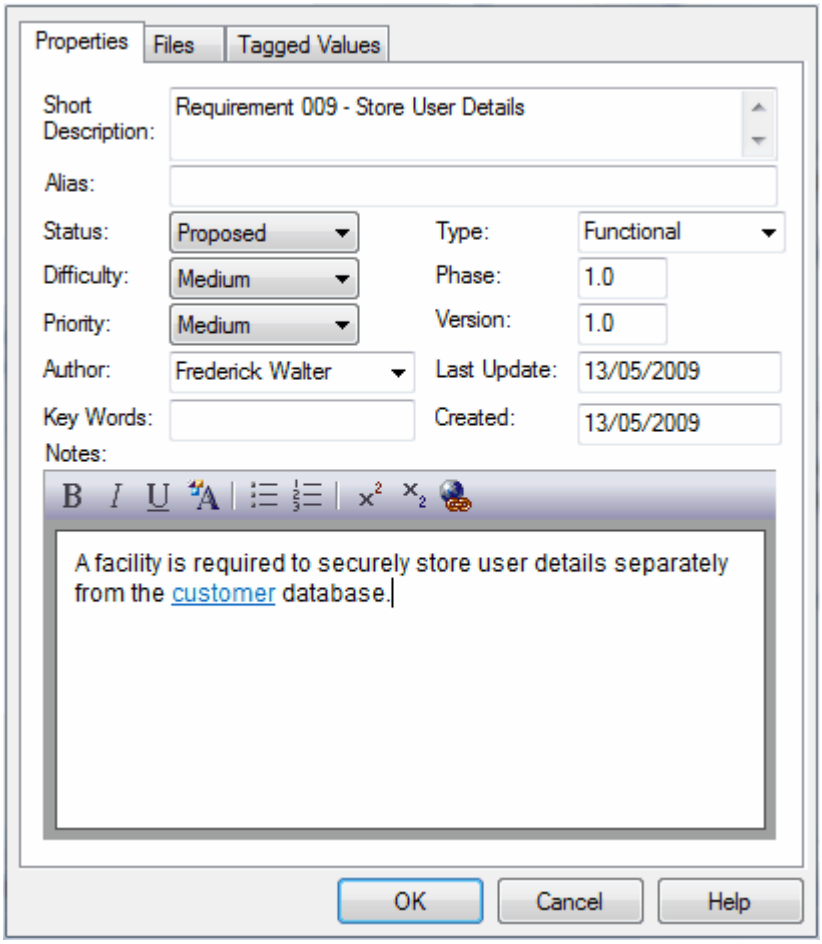
**Note:**

In Requirement Management tools and texts, the characteristics of a requirement are commonly called *attributes*. However, in UML the term *attribute* refers to a different type of feature (see *UML Modeling With Enterprise Architect - UML Modeling Tool*), and the requirement characteristics are defined as *properties*. In this Enterprise Architect documentation, the term *properties* is used.

Requirement properties differ slightly from the properties of other elements; they are mainly focused on the type, status, difficulty and priority of the Requirement. The **Notes** field is also important, as it describes

precisely what requirement the element represents. (See *UML Modeling With Enterprise Architect - UML Modeling Tool.*)

To edit the properties of a Requirement , double-click on the element in a diagram or right-click on it in the Project Browser and select the **Properties** context menu option. The Requirement Properties dialog displays.




If necessary, edit the name (**Short Description**) of the Requirement element, then type a detailed explanation of the requirement in the **Notes** field at the bottom of the dialog. Set the Status, Difficulty, Priority and Type, and other parameters as required.

You can add and delete **Status** field values, and assign a color to each value to indicate the status of the Requirement on a diagram; for more information, see the *Color Code External Requirements* 10 topic.

When you have finished entering the Requirement properties, click on the **OK** button.

In a project, it might be necessary to define more information in a requirement than is provided by the standard properties. For more information on extending the requirement properties, see the *Extend Requirement Properties* 11 topic.

### 2.1.1 Color Code External Requirements

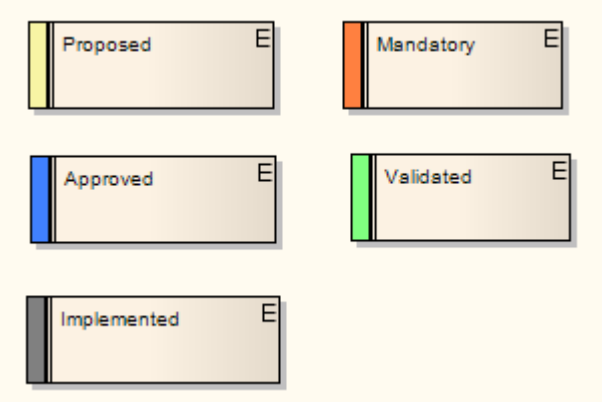
External requirements can be color coded to provide quick visual cues indicating the status of a requirement. To enable color coded external requirements follow the steps below:

1. Select the **Tools | Options** menu option. The Options dialog displays.
2. From the hierarchical tree select **Objects**, and select the **Show status colors on diagrams** checkbox to enable the status of external requirements to be represented by color coding.

The color code requirements use the following default conventions:

- Yellow for Proposed

- Blue for Approved
- Green for Validated
- Orange for Mandatory
- Black for Implemented.




You can change these colors, and add or remove status types, using the Status Types dialog (see the *Reference Data* topic in *UML Model Management*).

## 2.2 Extend Requirement Properties

A project might apply further properties to a requirement, such as cost, lateness penalty or risk to the business if not met. You can add these properties to specific Requirement elements, or configure them to be automatically available in all Requirement elements on creation, using *Tagged Values*. (These are sometimes referred to as *User-defined attributes*.) For an introduction to Tagged Values and how to use them, see *Using Enterprise Architect - UML Modeling Tool*.

Extended element properties are not visible unless you open the Tagged Values window for the element. Alternatively, you can display the additional properties on the element image on its diagrams 11.

### Add Tagged Values to Existing Requirements

To add a property to a single Requirement as a Tagged Value, simply click on the Requirement, display the Tagged Values window (**[Ctrl]**+**[Shift]**+**[6]**), and enter the name of the property as the tag name and the value of the property as the tag value.

It is likely that any property you add to one Requirement would also apply to others. You might therefore use a *predefined Tagged Value Type* to identify your Requirement property, so that you can select it whenever required. The predefined Tagged Value Type also enables you to define specific *values* for the Tagged Value. If the appropriate predefined Tagged Value Type does not exist, a Technology Developer can create it to add to the structured tags, reference tags, or customized tags collections. (See *SDK for Enterprise Architect*.)

### Configure Requirements to be Created With Extended Properties

If it is necessary to create all Requirements with the same extended set of properties, you can create a *Requirement Template* diagram and either create a special Requirement that defines those properties (as Tagged Values), or drag an existing Requirement with those properties onto the diagram. You then *set the Requirement Template diagram* as the template for all new Requirement elements, so that those new Requirements automatically have all of the properties you need. (See *UML Modeling With Enterprise Architect - UML Modeling Tool*.)

However, this then excludes other Requirement element formats, including the standard Requirement format. If you want to use another Requirement format, you have to replace or cancel the current Template. Alternatively, you can create a *Profile*.

A Profile also defines exactly what a new Requirement element should contain, and how it should display in diagrams. However, a Profile is a collection of *alternative* element definitions, so it does not override the default Requirement format, nor does it prevent you from defining several different types of Requirement element. You can therefore have separate and parallel definitions of elements for business requirements, system requirements, project requirements, or any other category of requirement you decide to work with.

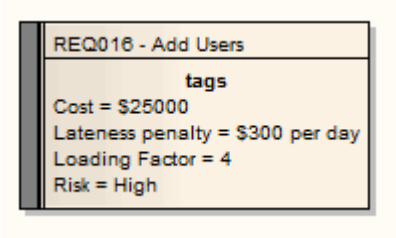
For information on importing and using existing Profile files, see the *UML Profiles* topic in *Extending UML Using Enterprise Architect*. For information on creating new Profiles, see the *Developing Profiles* topic in *SDK For Enterprise Architect*.

### 2.2.1 Display Tagged Values On Diagrams

If you have extended the properties of a Requirement, you might want to make those properties visible in the Requirement elements in your diagrams, by switching on display of the element *tags* compartment. You can do this in one of two ways:

- To display the *tags* compartment on all elements on a diagram, double-click on the diagram background and select the Elements tab of the Diagram Properties dialog; select the **Tags** checkbox and click on the **OK** button.
- To display the *tags* compartment on a specific element on a diagram, right-click on the element and select the **Feature Visibility** context menu option; select the **Tags** checkbox in the Show Element Compartments panel of the Feature Visibility dialog, and click on the **OK** button.

The Tagged Values are then displayed in the Requirement element on the diagram. (Both methods are documented in *UML Modeling With Enterprise Architect - UML Modeling Tool.*)




## 2.3 Connect Requirements

A Requirement element can be connected to other Requirements, most commonly using Aggregation relationships to form a hierarchy of requirements.

Requirements are also connected to other types of element, most commonly Use Cases, by Realize or Implements relationships. (See *The UML Dictionary* for details of these three relationships.)

These relationships are very important, both in identifying how the Requirements are organized and used in the model, and in tracing 15 the development from the Requirements throughout the model. Both of these tasks are very simple in Enterprise Architect, because once a connector on a Requirement exists, Enterprise Architect automatically lists the Requirement in the:

- Requirements Traceability window (an important tool in examining the role of Requirements in the model) (see *Using Enterprise Architect - UML Modeling Tool*)
- Require tab of the target element Properties dialog (see *UML Modeling with Enterprise Architect - UML Modeling Tool*)
- Scenarios & Requirements window
- Relationships Window (see *Using Enterprise Architect - UML Modeling Tool*)
- Dependency and Implementation reports (see *Report Creation In UML Models*)
- Standard RTF output. (See *Report Creation In UML Models.*)

The connector itself is also listed in the Links tab of the target element Properties dialog (see *UML Modeling with Enterprise Architect - UML Modeling Tool*), and in the Relationship Matrix (see *Using Enterprise Architect - UML Modeling Tool*). There are, therefore, many ways to locate, view and track Requirement relationships.

### Connect On Diagram

Relationships can be created on a diagram by clicking on the appropriate connector icon from the Requirement and Common pages of the UML Toolbox (see *Using Enterprise Architect - UML Modeling Tool*), clicking on the source (originating) element, and dragging to the target element.

If you are connecting elements in different packages, you can drag elements from the Project Browser onto a common diagram and set up the relationships there. Alternatively, you can quickly generate a *Realize*

connector by dragging an existing Requirement element from the Project Browser *over* the element that implements the Requirement. Enterprise Architect interprets this as a request to create the Realize connector and does so automatically. The Requirement element is not added to the diagram. However, if you subsequently drag the Requirement onto the diagram the connector is already in place.

### Connect Off Diagram

You can also connect a Requirement element to other elements without necessarily having the elements on the same diagram, or having a diagram open.

Use the Relationship Matrix to create relationships for requirements (see *UML Modeling with Enterprise Architect - UML Modeling Tool*); this is a convenient way of quickly building up complex relationships and hierarchies.

## 2.4 Import Requirements and Hierarchies in CSV

You can import Requirements from a spreadsheet application in CSV format. Before doing this you must create a CSV import file specification that:

- In the **Default Types** field has the value **requirement,package** to import requirements and a package structure to contain them
- Has the **Preserve Hierarchy** checkbox selected
- Identifies the data fields on the spreadsheet that are to be translated into Enterprise Architect, in the order in which they are plotted across the spreadsheet
- Is to operate on a spreadsheet containing the **CSV_KEY** and **CSV_PARENT_KEY** fields (which, if not generated by a CSV export from Enterprise Architect, you have added and populated yourself).

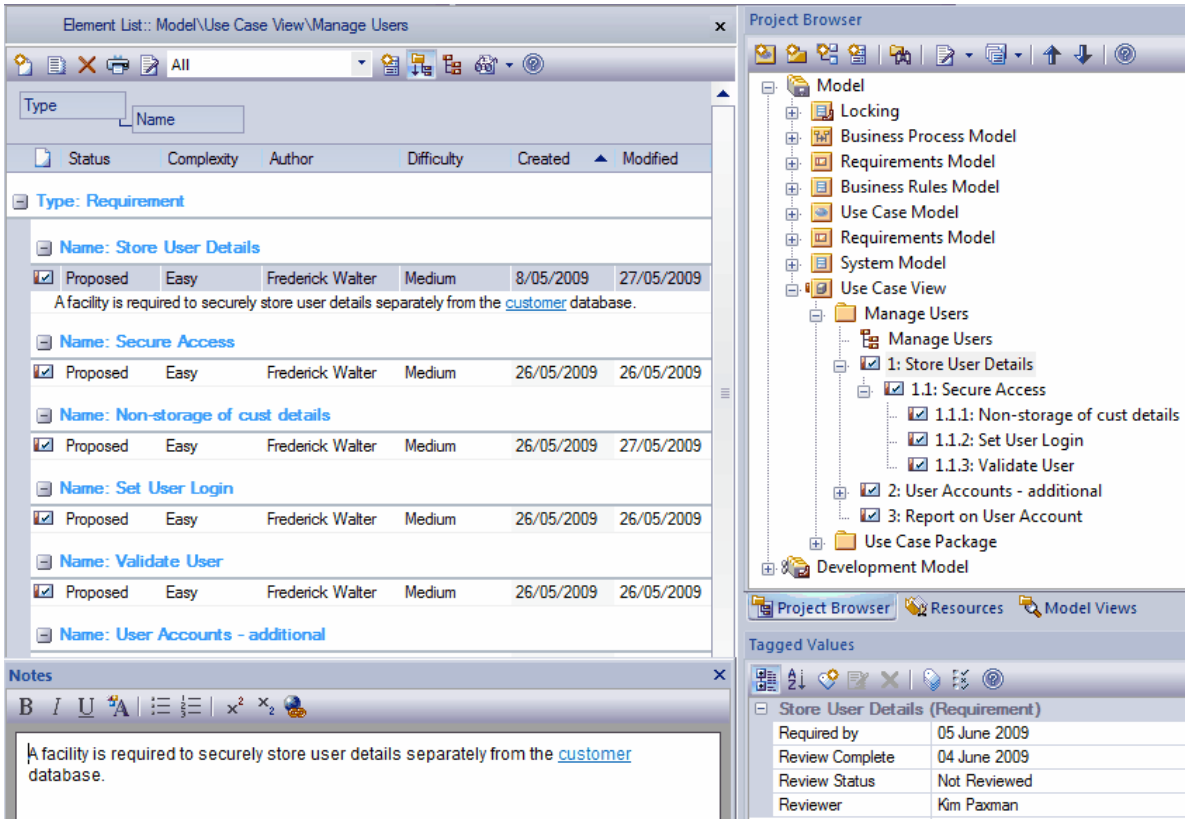
This enables you to import the individual and grouped requirements from the spreadsheet into Enterprise Architect, and to reconstruct the hierarchies of Requirements in the target package in the Project Browser.

For more information on importing in CSV format, see *UML Model Management*).

# 3 Manage Requirements

One of the main advantages of managing Requirements in Enterprise Architect is that you can display many different aspects of information on the Requirements, from overall organization and location through lists, current status, general properties, detailed individual notes and specific properties, to relationships and dependencies with other elements in the model.

As most of these aspects are displayed in dockable windows, you can review the Requirements from several different perspectives simultaneously in the Enterprise Architect work area, as shown below:




This display shows the position of the *Store User Details* Requirement element in the model, and how it relates to other Requirements (Project Browser); the default characteristics of the Requirement (Element List) and the extended characteristics (Tagged Values window), and a detailed description of the Requirement ( Notes window). You can configure some of these windows to display more information, and/or use other windows and facilities.

Brief descriptions of appropriate screens and their use are provided in the following topics:

- View Requirements 14
- Trace Use of Requirements 15
- Manage Requirement Changes 15
- Report on Requirements 16

## 3.1 View Requirements

Use the following windows and facilities to locate and list Requirement elements in the model; add, move and delete the elements; display and edit the properties and characteristics of individual elements, and generate reports on packages or specific elements.

- Project Browser - shows the content and structure of your model.
- Element List - lists the elements in a diagram or package, filtered and sorted according to the settings you define; shows all or selected default properties of each element.
- (Requirements) Diagram - shows the arrangement of a group of Requirements, and can show whether the

elements are in the same package or different packages.

- Model Search - enables you to locate Requirements in general in the model, or specific Requirement elements, according to the search criteria you use.
- Model Views - enables you to maintain links to commonly-used elements, and to rapidly show developments and changes in (Requirement) package contents through either reports or slide shows of selected diagrams.
- Properties - shows every *standard* property of a selected element, whether updated by the user or maintained automatically by the system.
- Tagged Values - shows *extended* properties of a selected Requirement element.
- Element Browser - shows every *added-on* property, such as attributes, operations, Tagged Values and constraints.
- Notes - displays the detailed description of a requirement, and any other additional information recorded on the requirement.

For more information on the Enterprise Architect windows, see *Using Enterprise Architect - UML Modeling Tool*).

## 3.2 Trace Use of Requirements

Having investigated the representation of requirements in your model, you might review either how they have been used to direct development through the model, or how a particular development was initiated. This is discussed in greater detail in the Traceability topics (see *UML Model Management*), but the windows and facilities you might use to follow development from Requirements are briefly described below. The significant feature in tracing Requirements and development is the connectors 12 between the elements.

- Relationships window - quickly identifies every relationship of which a selected Requirement element is a member, and the partner element in that relationship, whether or not the relationship is visible in the current diagram. If the partner element is not in the diagram, you have the option of adding it.
- Traceability window - a very useful tool in showing chains of relationships that include the selected element. The window can show, for example, that a Requirement A is realized by a Use Case X, but Use Case X *also* realizes Requirement B, which in turn is *also realized by* Use Case Y. You can control the type and extent of these relationship chains, but as the system checks the connectors and partner elements of every relationship within the limits you impose, the system can take some time to produce the final results.
- Relationship Matrix - a significant tool in mapping the relationships between the Requirements elements in a package and other elements in either that package or a different package. Where a relationship is missing, you can add it; if an existing relationship is misplaced, you can delete it.
- Properties dialog, Require tab - for elements *other than* Requirements (particularly Use Cases), shows all internal responsibilities and external requirement elements attached to the element.
- Scenarios & Requirements window - as for the Properties dialog, shows the Requirements and responsibilities of the selected element, and the scenarios and constraints under which the Requirements are being realized.

For further information on the Enterprise Architect windows, see *Using Enterprise Architect - UML Modeling Tool*. For information on the Relationship Matrix and element Properties dialog, see *UML Modeling with Enterprise Architect - UML Modeling Tool.*

## 3.3 Manage Requirement Changes

Because requirements are statements of what a system or process must do or provide, they have a great impact on the modeling and development of the system. A new requirement might initiate an extensive program of work, and changes to or removal of that requirement can therefore have a major effect on the model. Issues concerning requirements, and changes to Requirements, must both be carefully managed.

The first steps in managing changes to requirements would be to raise specific *Issue and Change* 16 request items against the Requirement element. You could monitor the appearance of these items using the filtered searches of the *Model Views* 16. You might then review 14 the Requirement properties and/or its relationship hierarchies. During model development, you might capture periodic *Baselines* 16 and use these to review the changes and, if necessary, roll them back to a previous point. You might also use the *Auditing* 16 facility to monitor changes as they are made, and to ensure that no unauthorized or potentially risky changes are being made in the model.

These facilities are briefly discussed in the following sections.

### Changes and Issues

A change is, very broadly, an item defining an addition or alteration to a requirement. An issue identifies either a failure to meet a requirement, or a risk in meeting the requirement. (See *Project Management with Enterprise Architect*.)

Changes and issues can arise in development at a number of levels, being raised for problems that apply system-wide down to within a specific element. There are two mechanisms that can be used to identify a change or issue, and the work required to resolve it:

- Change and Issue (or Defect) elements - structured comments that identify a problem at system-level, although they can also be attached to a specific element from which a problem arises. Both types of element resemble the Requirement element, and can be linked to one or more other elements that have to be reviewed, with relationships such as Association, Dependency and Realize. The two types of element can also form hierarchies or groups, where complex problems arise.
- Maintenance items raised against a specific element, and recorded for that element in the Maintenance window. Maintenance items enable the distinction between *Defects* (a failure to meet a requirement) and *Issues* (a risk factor that might affect satisfying the requirement). They also include *Tasks*, which record work items associated with the element.

Maintenance items are very specific, but if there is a possibility of an item having a wider impact on other elements or the system in general, you can translate the item into a Change or Issue element, or any other type of element that best identifies the problem and its solution.

### Model Views

Model Views are very useful for trapping changes and issues in the model, especially on Requirements. You can set up searches to identify the appearance of new Change or Issue elements, or to detect changes in the properties of the Requirement elements themselves. (See *Using Enterprise Architect - UML Modeling Tool*.)

### Baselines

A Baseline is a snapshot of a package or a model branch at a particular point in time, which you determine. You can use the Baseline as a distribution mechanism for changes to the model, but the main use is to enable you to compare the current model with a previous stage, and detect what changes have been made since the Baseline was captured. If you do not want a change to remain in the model, you can roll the affected elements back to the state they had in the Baseline. Therefore, if you maintain your requirements in a specific package or branch, you can capture Baselines of the package and ensure that changes conform to your change management process or, if not, can be reversed. (See *Baseline UML Models*.)

### Auditing

The Auditing facility enables you to capture any changes made to your model within the selection criteria that you define. You can, for example, configure the Auditing facility to specifically record changes to Requirement elements. As auditing is continuously monitoring, you can detect changes as they are made, and verify that they are acceptable. You can also store the log of changes, and review it later on. Note that you cannot reverse the changes automatically, as you can with Baselines. You might therefore use Auditing to identify changes that you will investigate more fully and - if necessary - reverse in a Baseline comparison. (See *Auditing UML Models*.)

## *3.4 Report on Requirements*

Enterprise Architect provides two report generation facilities that enable you to output RTF reports and HTML reports on your model structure and components. The RTF reporting facility is especially comprehensive, and contains a number of features that provide particular support to Requirements Management:

- A requirements report template that extracts details of external requirements in the model; you can copy and tailor this template for your particular needs.
- Options in the Element List and Model Search to generate RTF reports on selected (Requirement) items from the collected information.
- The Implementation Report, which lists for a selected package the elements that require implementers, together with any source elements in Realize (Implements) relationships with those elements.
- The Dependency Report, which lists for a selected package any elements that are dependent on another element for their specification. For example, a Use Case derives its specification from the Requirement that it realizes.

For further information, see *Report Creation in Enterprise Architect*.

# Index


## - A -

Audit
  Requirements 15
Autonumbering
  And Requirements 9

## - B -

Baseline
  Requirements 15

## - C -

Change
  Elements And Requirements 15
Change Management
  And Requirements 15
Color Code External Requirements 10
Connector
  List For Requirements 12
  Realization, Common Diagram 12
  Realization, Quick Generation Of 12
Create
  Requirements 9
CSV
  Import Requirement Hierarchies 13
  Import Requirements 13

## - D -

Docked Windows
  For Requirements 14

## - E -

Element
  Requirement 4
External
  Requirements 4
External Requirements
  Color Coded 10

## - H -

Hierarchy
  Of Requirements 4
  Requirement, Import Via CSV 13

## - I -

Import
  Requirements Via CSV 13
Internal Requirement
  Define 6
  Properties 6
Issue
  Elements And Requirements 15

## - L -

Level Numbering
  For Requirements 4

## - M -

Manage
  Requirements 14
Model
  Requirements 4
Model Views
  For Requirement Change Management 15
Move
  Internal Responsibility To External Requirement 7

## - P -

Properties
  Extend For Requirements 11
  Of Requirements, Extended 11
  Of Requirements, Standard 9
  Requirement, Display On Diagram 11, 12

## - R -

Realization
  Connector, Quick Generation Of 12
Reporting
  Dependency 16
  Implementation 16
  Requirements 16

Requirement
Aggregation 12
And Level Numbering 4
And Use Cases 4
Auditing 15
Autonumbering 9
Baselines 15
Change Management 15
Changes 15
Color Code Status 10
Connect On Diagram 12
Connect Through Relationship Matrix 12
Connectors 12
Convert From Responsibility 7
Create In Diagram 9
Create In Project Browser 9
Dependency Report 16
Docked Windows 14
Element 4
Element Template 11
Extend Properties, Default Format 11
External 4
Fast Generate Realization Connector 12
Gather 2
Hide Stereotype Letter 9
Hierarchies 4
Implementation Report 16
Import Via CSV 13
Internal 6
Issues 15
Manage 14
Model 4
Model Views 15
Profile 11
Properties, Display On Diagram 12
Properties, Extended 11
Properties, Standard 9
Realization 12
Report Template 16
Reporting 16
Review 14
Show Stereotype Letter 9
Status, Color Coded 10
Tagged Values 11
Tagged Values, Display 12
Trace Through Connectors 12
Trace Use Of 15
User-Defined Attributes 11
View 14
What Is A? 2
Windows For Tracing Use 15
Windows For Viewing 14
Requirements Management
And Enterprise Architect 2
In Example Model 2
Overview 2
Responsibility
Define 6
Move To External Requirement 7
Review
Requirements 14

# - S -

Show Status Colors on Diagrams
Menu Option 10

# - U -

Use Case
And Requirements 4

# - V -

View
Requirements 14

# - W -

What Is
A Requirement? 2

Requirements Management

www.sparxsystems.com