



IBM® Rational® Rhapsody® Gateway Add On



Best Practices Guide

Rhapsody[®]

**IBM[®] Rational[®] Rhapsody[®]
Gateway Add On**

Best Practices Guide



License Agreement

No part of this publication may be reproduced, transmitted, stored in a retrieval system, nor translated into any human or computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual or otherwise, without the prior written permission of the copyright owner.

The information in this publication is subject to change without notice, and Dassault Systèmes and its affiliates assume no responsibility for any errors which may appear herein. No warranties, either expressed or implied, are made regarding Rhapsody software and its fitness for any particular purpose.

Trademarks

Reqtify is a registered trademark of Dassault Systèmes or its affiliates in the US and/or other countries.

Rhapsody Gateway, IBM, the IBM logo, DOORS and Rhapsody are trademarks or registered trademarks of IBM Corporation.

All other product or company names mentioned herein may be trademarks or registered trademarks of their respective owners.

© Copyright 2001-2011 Dassault Systèmes. All rights reserved.

Contents

Contents	5
Practical Considerations	7
Low-Level Requirement Derivation	7
Derivation from the User Requirements	7
Provided in an External Source	8
Single vs. Multiple Projects.....	8
Risk Assessment as a Driver for Requirement Development.....	8
Anchors vs. Dependencies	10
Persistent Change Recording.....	10
Analysis of a Particular Process.....	11
Overview	11
Assumptions	11
Introducing the Actual Project Structure.....	12
Managing the System Model.....	12
Initial Structure.....	13
Subsystem Specifications Model	13
Subsystem Model.....	14
System Model.....	15
System Model Reflexive Link	15
Rhapsody Gateway Project Structure Analysis	16
Complete Project Structure	18
Undefined Requirements	19

Practical Considerations

This section is intended to introduce guidelines that will help the user to work efficiently with Rhapsody Gateway.

Low-Level Requirement Derivation

If you are going to use Rhapsody Gateway, the first thing to identify is where the requirements are coming from. There are two possible ways:

- ◆ Are the system/low-level requirements derived from the user requirements, created using UML?
- ◆ Are the requirements provided in an external source?

Derivation from the User Requirements

If the low-level requirements are created using UML from the user requirements, the user requirements will be imported into Rhapsody via Rhapsody Gateway.

Rhapsody will then be used to derive such lower-level requirements and include any files or additional information (Use Cases for example) that may be needed by the subsystem teams/contractors, as a UML model.

In order to provide a hand-off UML model to the contractors, typically the original user requirements are not provided, and only the system/low-level requirements are given.

This can be achieved by providing a 'bridging model' or a 'Specifications model', which contains the system/low-level requirements and also any additional files needed (except the original user requirements).

Provided in an External Source

If the low-level requirements are not created using UML but, for instance, within DOORS then these requirements will be imported into a Rhapsody model via Rhapsody Gateway.

This can be a requirements model, where contractors can be supplied with their respective set of requirements. This may be achieved by either a sharing model or an archiving model. What is important to note, however, is that the impact analysis is maintained. Indeed the consequences of modifying or removing a covering element (such as a Use Case) can be viewed down the hierarchy.

Single vs. Multiple Projects

When users cover requirements using Rhapsody models, two basic configurations are available:

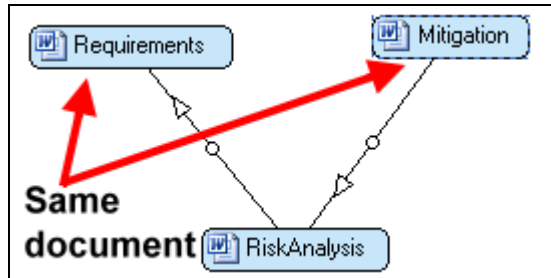
- ◆ Users work with a Single Project, the following elements have to be noted:
 - All Rhapsody Gateway information and customization are available in one place.
 - In the Rhapsody Gateway, only one person at a time can modify anything on the project that requires a restricted access. The appropriate files are locked during the user's work. He can use Marks to provide information to other users.
 - Multiple users can modify the Rhapsody model.
 - The Configuration Management is essential to manage the project.
- ◆ Users work with a Multi-Project:
 - Multiple users can add and customize their own Rhapsody Gateway information. The Configuration Management can also be used to restrict access, Diff Merge Types, etc.
 - A mechanism for collating models together must be devised if total coverage needs is necessary.

Risk Assessment as a Driver for Requirement Development

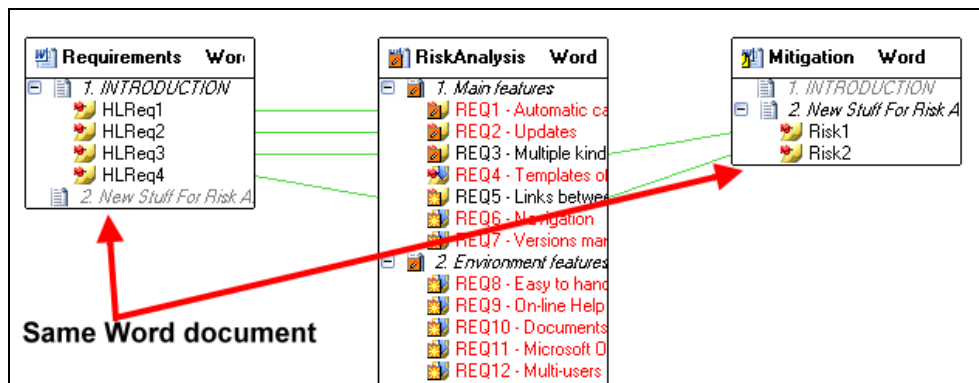
In this section, you are analyzing a Specification and a Risk Analysis documents. The Specification links all requirements to elements in Risk Analysis (possibly within DOORS). Risk Analysis elements that require mitigation point back to 'new' elements in the Specification document, created in the process. While attempting to capture the model of the requirements hierarchy in Rhapsody Gateway, the circularity will flag up a rule check error named '**Traceability Graph Violation**'.

To resolve this kind of error, two solutions are possible:

1. Create a mimic with bidirectional links by simply adding the same document more than once to your Rhapsody Gateway project, as shown in the following figure:



The same document can be effectively analyzed twice and some links can be added between them. However, this document should have different names and also needs to be analyzed using different type files in order to extract different information. See the following figure for more information.



2. Another option is to use requirements or entities. As a reminder, an entity is a non-requirement element that **MUST** cover a requirement. An entity can only reference requirements, derived requirements or macro-requirements and cannot reference itself.

Therefore, if you analyze everything in the Requirements document with one type and define specific requirements or entities in the RiskAnalysis document, anything in RiskAnalysis not pointing to anything in Requirements will be highlighted in the Rule check.

A slight complication is that the 'new' requirements have to be traced 'to' the Risk Analysis requirements/entities - as opposed to 'from'. You cannot trace 'to' an entity but only to a requirement.

Therefore, if you do not care about direction then you can use entities and of course you would not need to put the document on twice. Once will be enough.

If the direction is important, then it is a better solution to use requirements. We have the possibility to cover in both directions and we can define a rule for the RiskAnalysis document to treat the non-coverage of requirements.

Anchors vs. Dependencies

Requirements can be connected to design elements in Rhapsody in several ways; by Anchor or by dependency.

Anchors require no stereotype, have no 'features', are 'owned' by the Requirement element and may not be ideal since the Requirements may be read-only. For instance, in a lower-level project that has a Specification Package added by reference some requirements are in read-only mode.

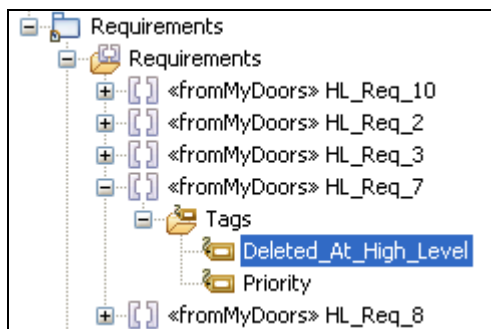
Dependencies require a stereotype such as <<trace>>, <<satisfy>>, they can have features (e.g. Tags), and are 'owned' by the 'other element'. In fact, dependencies are the recommended way of connecting requirements to design.

Persistent Change Recording

The orange markers added by Rhapsody Gateway on modified requirements are transient. A more persistent change record can be maintained through Snapshots and Marks.

Define and use a specific filter to show only requirements with such Marks in the main Rhapsody Gateway window.

By default, when an Add High Level Requirements operation is performed, only requirements that were deleted will have a tag 'Deleted_At_High_Level' added to the exported requirements in Rhapsody. (It will be up to the Rhapsody user to delete the model elements.)



In order to get more information in Rhapsody after the export, create new attributes in Rhapsody Gateway such as Modified or Under Revision. This will allow corresponding tags to appear in the exported requirements in Rhapsody.

Analysis of a Particular Process

Overview

This section is intended to explain Best Practice when:

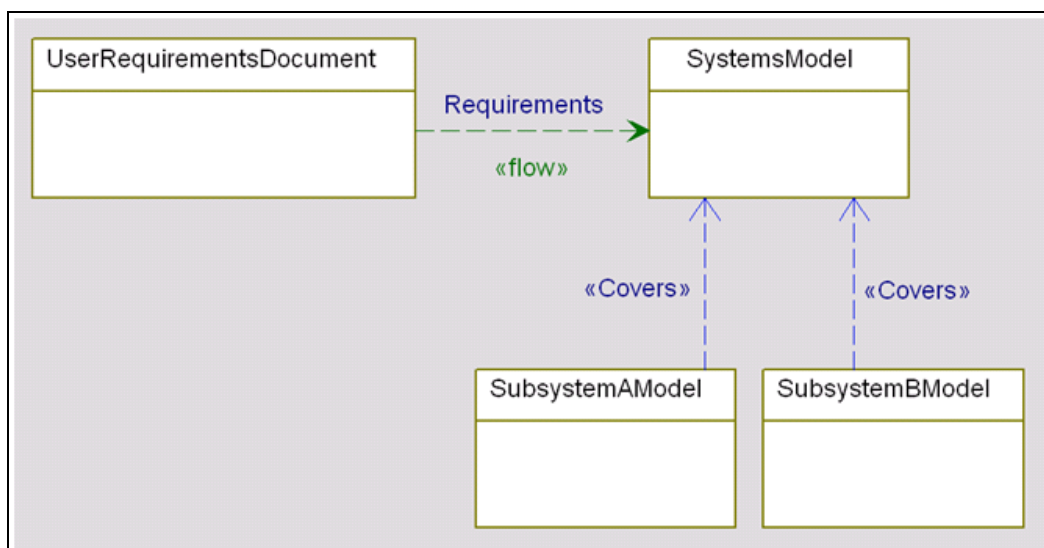
- ◆ Defining Requirements in a Rhapsody Model
- ◆ Covering these Requirements in downstream Models

Assumptions

This guide assumes a basic model structure of a Systems Model that is covered by two Subsystem Models.

<<trace>> is used in this guide as a generic reference type – actual user models will use <<verify>>, <<satisfy>>, <<refine>>, etc. This is a simple modification to the basic structure.

The following UML diagram corresponds to the Rhapsody Gateway project that will be created:



Introducing the Actual Project Structure

In reality, each Subsystem Model is isolated from the Systems Model by an intermediary 'Specifications' Model.

This allows the isolation of individual Subsystem Specifications from the System model as whole.

- ◆ Contractors see only their Specification Models.
- ◆ System Modelers can package individual Subsystem Specification Artifacts easily.
- ◆ This requires no Rhapsody Gateway customization. Rhapsody Gateway only 'sees' the specification package and as a result does not need to filter out requirements that are for 'this' Subsystem model.

The Requirements Manager is be able to analyze the full system hierarchy using Rhapsody Gateway, e.g. URD->SRD->Design. This is why requirements need to be added by Rhapsody Gateway and why the models are then fed back up to the chain.

The Systems Team is be able to perform system requirement definition in Rhapsody, i.e. Rhapsody Gateway needs to handle Rhapsody like any other requirements tool. This part concerns modeling systems requirements, such as using SysML.

One aspect of managing complexity in a system of systems, is the ability to allocate parts of the requirements to different Teams. This is called separation of concern.

Obviously, the different Teams need to be able to see traceability at their level. Therefore, once the Teams receive the systems model the requirements need to be added by the Rhapsody Gateway rather than added to the model.

Also, the different Teams need more than just requirements, such as use cases and system context. This is why the model is sent to the subsystem Team in addition to the requirements.

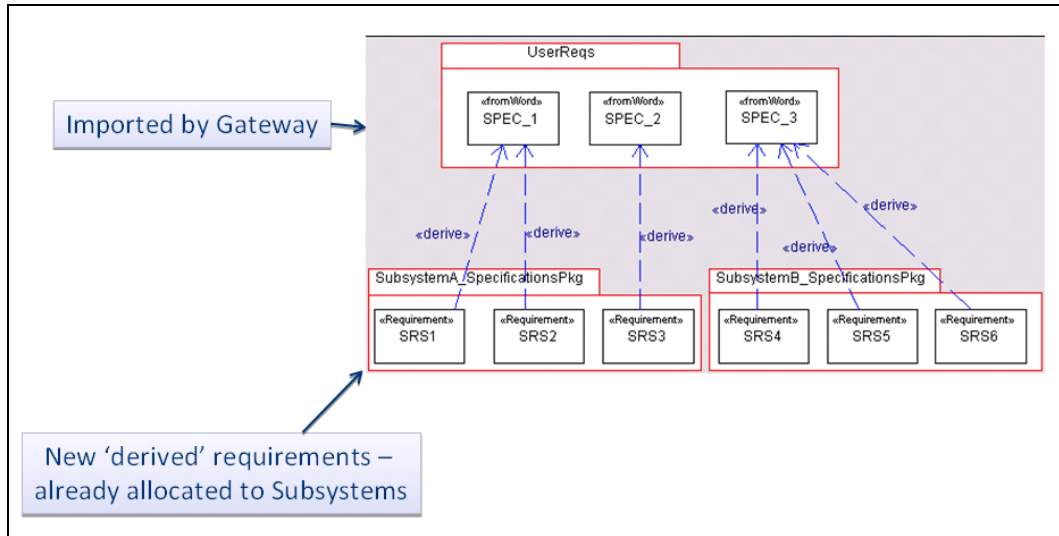
Managing the System Model

To define requirements in the Rhapsody Model, the external requirements should be imported using Rhapsody Gateway.

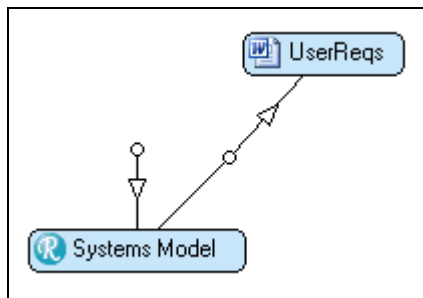
These external requirements are traced to 'derived' requirements already allocated to Subsystems Models.

Initial Structure

The following diagram shows the corresponding initial structure of the Rhapsody Model.

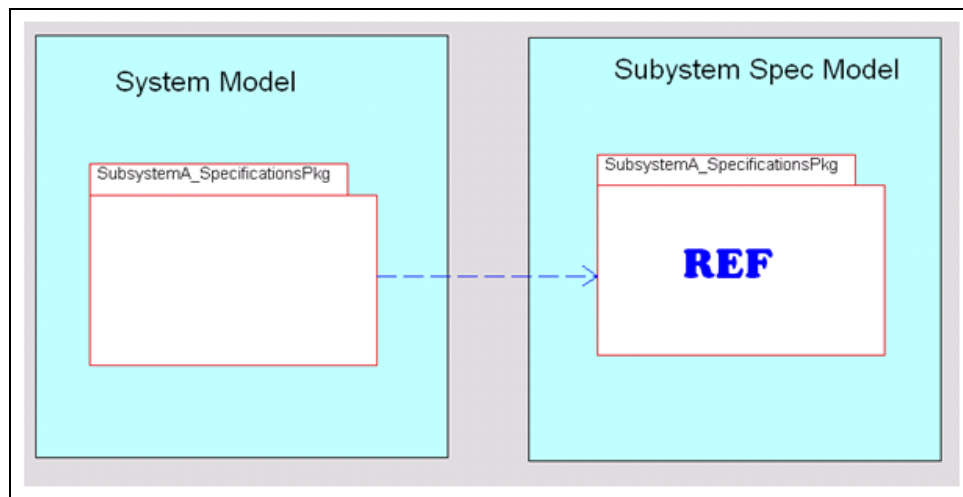


The Rhapsody Gateway project corresponding with the previous structure looks as follows:



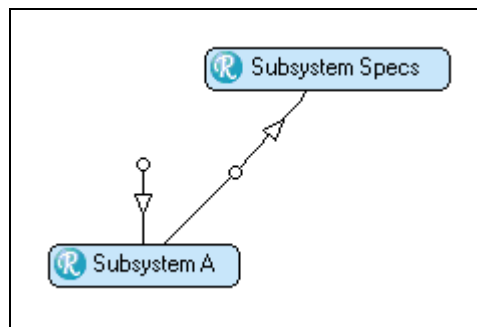
Subsystem Specifications Model

From the Rhapsody tool, create a reference of a System Model. The Subsystem Specification Package is added by reference from the System Model. Once the element has been referenced, you can also include requirements and other model artifacts.

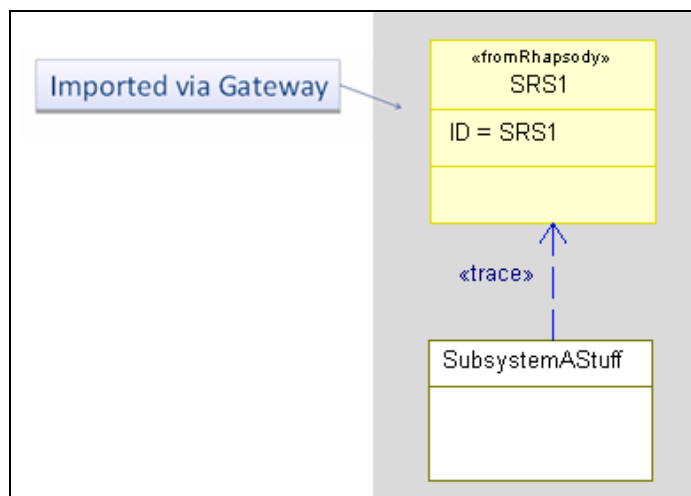


Subsystem Model

To express the covering of the System Spec Model by the Subsystem, create a Rhapsody Gateway project which corresponds to the following project.

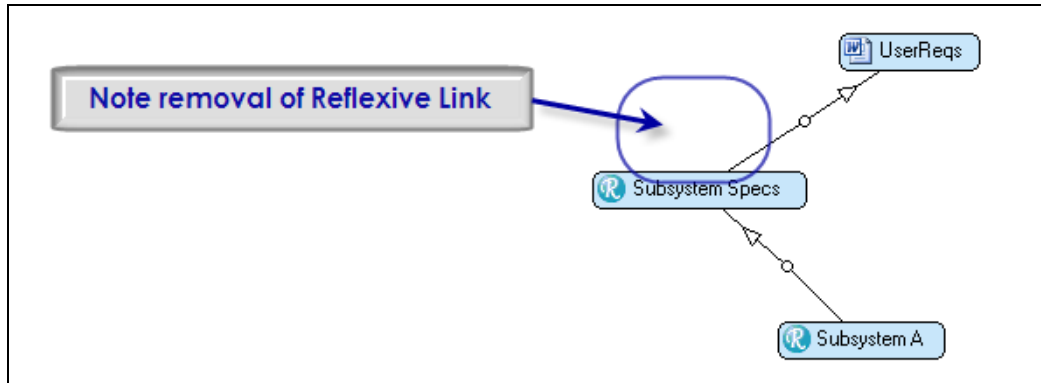


The requirements are imported from the Subsystem Specification Model using the Rhapsody Gateway tool. They are tagged "fromRhapsody".



System Model

To show the coverage of System Requirements by Subsystems the System Model is now modified from the Rhapsody Gateway project. The Subsystem A covers the Systems Model and this one covers the User Requirements Document.

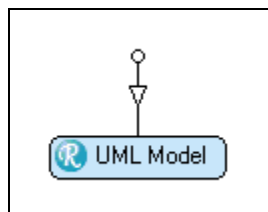


From the project view, remove the reflexive link on the Systems Model.

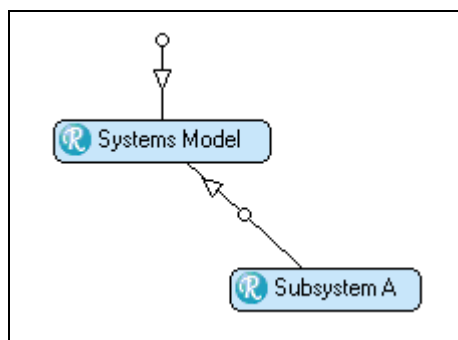
System Model Reflexive Link

The reflexive link can be construed in two ways according to the situation:

- ◆ As shown above, a single reflexive link applied on a model notices that all requirements need to be covered by the model itself:



- ◆ If the Systems Model is covered by another document and also by a reflexive link, this means that all the covering document must cover the Systems Model. The reflexive link is optional, and some requirements can recursively cover the Systems Model.

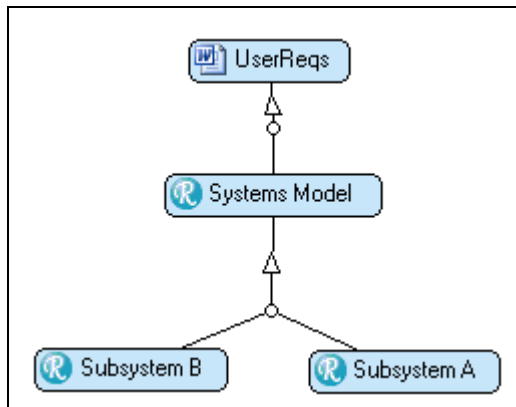


To differentiate between requirements which are only met in Subsystem models and those which are satisfied in the System Model, the better situation is to add multiple documents in Rhapsody Gateway to be covered.

Rhapsody Gateway Project Structure Analysis

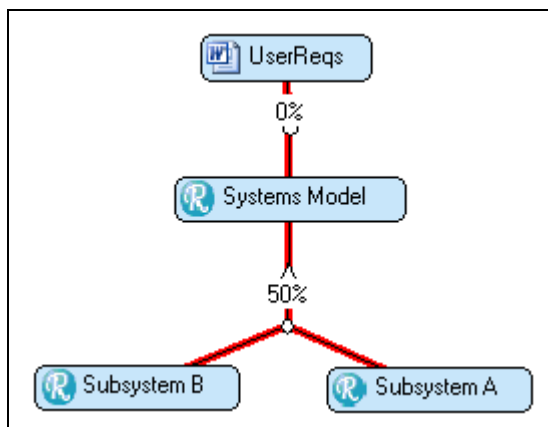
The following section presents the Rhapsody Gateway Project Structure (Systems Model) and its combined coverage.

The complete Rhapsody Gateway project corresponding with the system looks like the following figure:

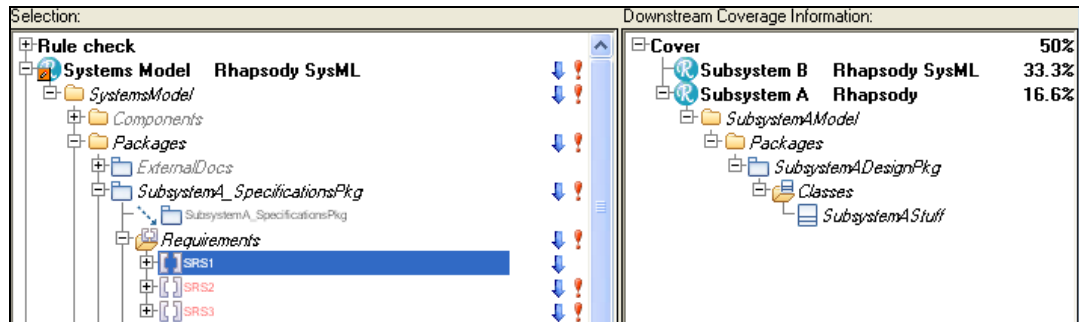


The analysis of this project enables the sighting of the coverage.

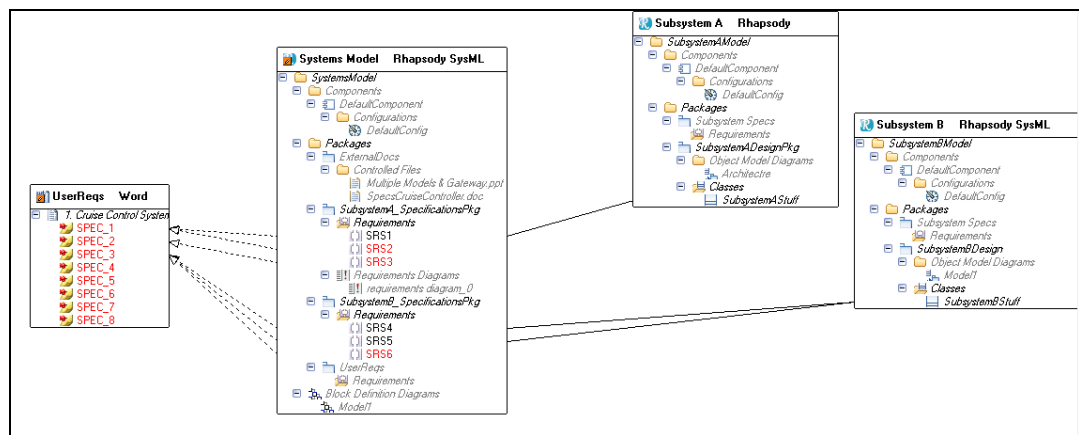
The Management View summarizes the coverage ratio.



The Coverage Analysis View shows the coverage in detail.

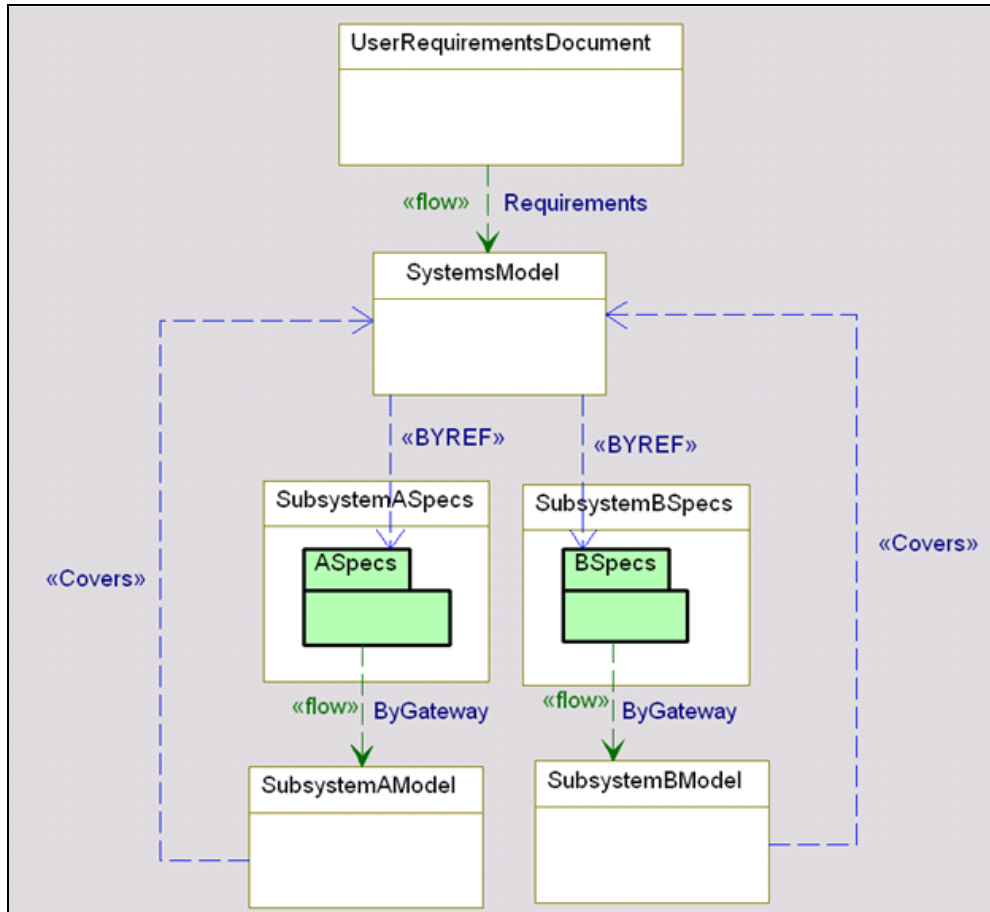


Open the Rhapsody Gateway Graphical View to visualize the entire project structure.



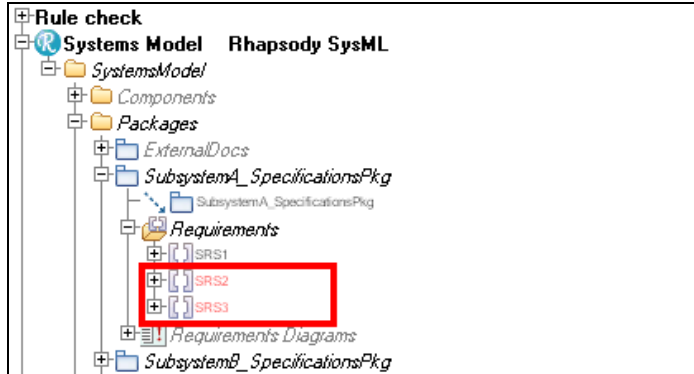
Complete Project Structure

The following diagram summarizes the entire structure of the Rhapsody Model. This diagram corresponds with the entire Rhapsody Gateway project previously created step by step.



Undefined Requirements

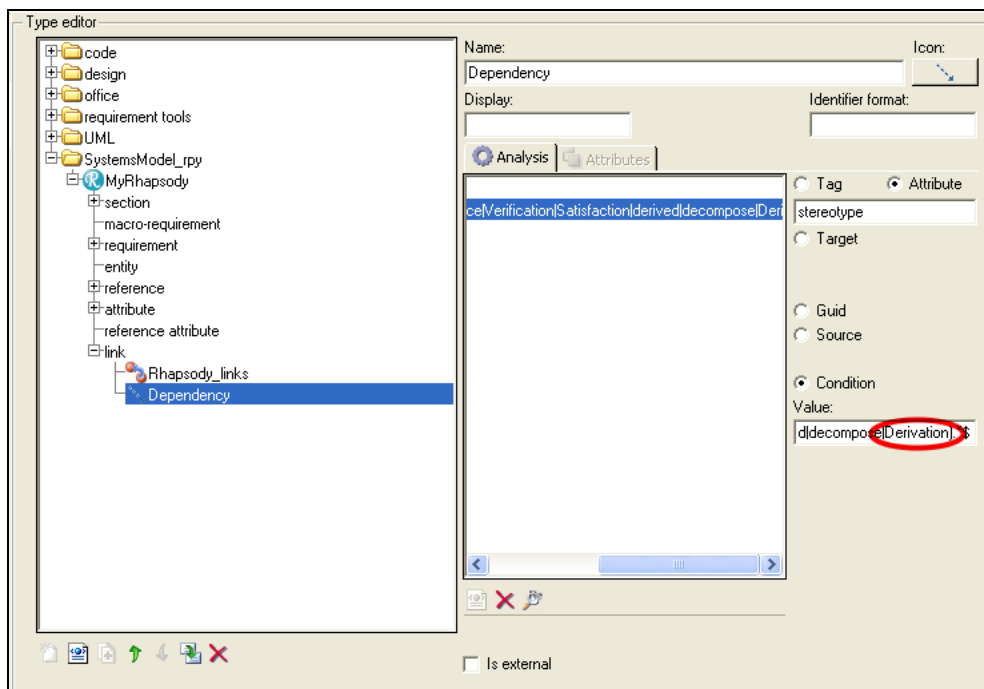
If you simply open the Rhapsody Gateway project corresponding with Subsystem A, some requirements are marked as Undefined, as shown below.



The Low Level requirements added by reference to the Specifications model contain Dependencies which are Derivations. These dependencies are created by default; they cannot be removed. These Derivations reference some User Requirements that are not added to the model. This implies that they are Unresolved. Rhapsody Gateway will flag their errors in Rule check list.

To avoid this, you can customize the UML type to ignore all links that are derivations. Thus, the Subsystem A model will analyze the Specifications with a customized UML type.

In other words, add the term 'Derivation' as a dependency that should not to be included as a link in the created customized UML type.



Once the new customized Rhapsody type is applied on the Specifications, the Undefined requirements disappear from the Rule check list. Requirements must be covered now.