The Rap on RUP™ : An Introduction to the Rational Unified Process™

Jeff Jacobs
Jeffrey Jacobs & Associates
phone: 650.571.7092
email: jeff@jeffreyjacobs.com
http://www.jeffreyjacobs.com

Survey

- Does your organization have a well defined methodology/process?
- Does your organization use OOA/OOD?
- Does your organization use UML?

Agenda

- What is RUP?
- RUP Fundamentals
- Phases
- Product “features”
- Summary
- Questions

Process/Methodology Product Presentation

- No UML bashing
- No rhyming

Why RUP™ at SSQA?

- Most organizations have no well defined software development process
- Software quality depends on the full SDLC, not just testing
- *The (RUP™) Knowledge base allows development teams to gain the full benefits of the industry standard UML*
- RUP™ covers all UML models
- RUP™ is hot; the latest silver bullet…

What is RUP™?

- A “software engineering process” (methodology)
- A knowledge base “process product”
  - CD to create web site
- UML model focused, not “paper documents” (but...)

What is RUP™?

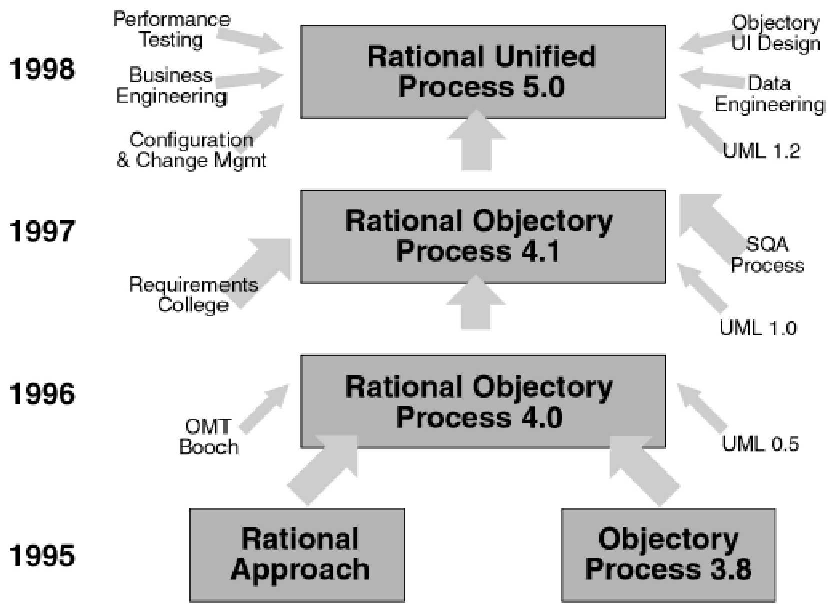
- Configurable process/product
  - Recognizes and supports variety of different project types
  - Support for tailoring and configuring project web sites
- Project oriented

3 Flavors of RUP

- Generic - not dependant on specific technology
- Microsoft Web Solution Technology
  - Additional templates, guidelines etc
- IBM Websphere™ Technology

History

- Methodology by Merger & Acquisition
- Objectory Process created in ‘80s
- Rational Approach created in ‘80s
- Acquisition of RequisitePro
- IBM acquires Rational



Genealogy of the Rational Unified Process

Why Should You Care About RUP™?

- Your organization is at SEI/CMM Level 1 "Ad Hoc"
  - Provides an excellent path to CMM Levels 2 and 3
- You need the basics for a "green field" effort
- You need to add OOA/OOD to your current process/methodology
- Management wants to know when you're going to use the latest silver bullet

RUP™ Fundamentals

- RUP is object and process oriented
  - Data takes a back seat
- Architecture is “key to success”
  - Emphasizes need for prototyping of core functionality, not just UI
- Iterative development
- Use Cases are primary requirements specification

4 Phases

- Inception
- Elaboration
- Construction
- Transition

Inception

- Establish business case and business models
- Establishes initial “vision”, high level requirements via “business” use cases
- Create stakeholder “buy in”
- Evaluate risks and return

Elaboration

- Detailed requirements
- Architecture and prototype
- Design

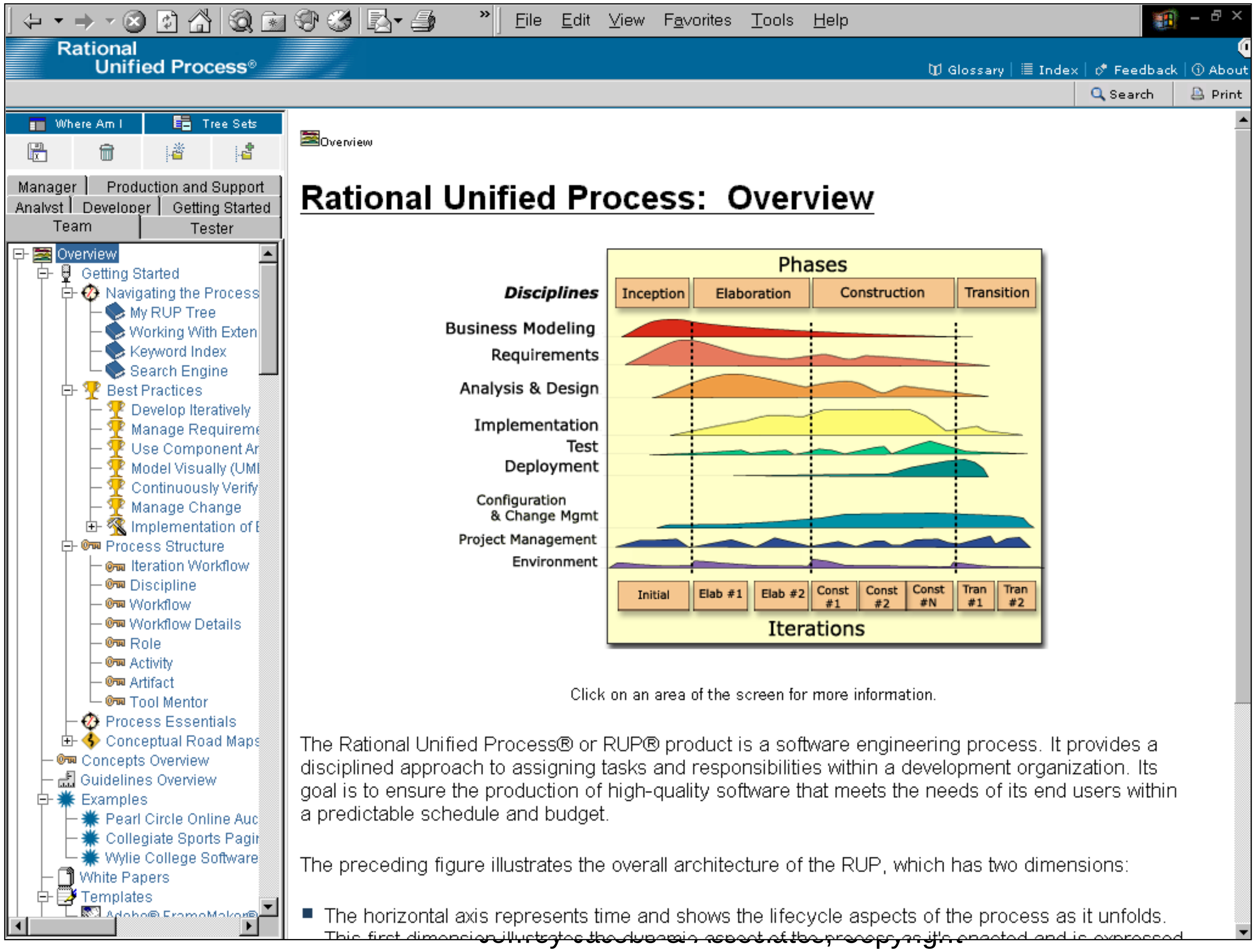
Construction

- Coding and testing

Transition

- Putting the product in the user's hands
- Highly variable, depending on project/product
  - Data migration
  - Training
  - Parallel Operations
  - Beta testing
  - Etc.

Overview of RUP™ (Organization)



Rational Unified Process: Overview

Click on an area of the screen for more information.

The Rational Unified Process® or RUP® product is a software engineering process. It provides a disciplined approach to assigning tasks and responsibilities within a development organization. Its goal is to ensure the production of high-quality software that meets the needs of its end users within a predictable schedule and budget.

The preceding figure illustrates the overall architecture of the RUP, which has two dimensions:

- The horizontal axis represents time and shows the lifecycle aspects of the process as it unfolds. This first dimension illustrates the dynamic aspect of the process as it's enacted and is expressed

Best Practices

- Develop software iteratively
- Manage requirements
- Use component-based architectures
- Model visually
- Continuously verify quality
- Control changes

Key Concepts of RUP™

- Organized by *discipline*
- *Workflow* - model of process for a discipline
- *Workflow Details* - 2nd level detail of workflow, detailing activities, roles and artifacts
- *Role* - who performs an activity
- *Activity* - defined piece of work that results in an artifact

More Key Concepts

- *Artifact* - a deliverable, may be document, model, code, etc
  - Templates and examples for many artifacts
- *Tool Mentor* - guide on using Rational Tools for RUP™

Analyst Roles

- Business-Model Reviewer
- Business Designer
- Business-Process Analyst
- System Analyst
- Requirements Specifier
- Test Analyst
- User-Interface Designer

Developer Roles

- Software Architect
- Designer
- User Interface Designer
- Capsule Designer
- Database Designer
- Implementer
- Integrator

Managers

- Project Manager
- Change Control Manager
- Configuration Manager
- Test Manager
- Deployment Manager
- Process Engineer
- Management Reviewer

Production and Support

- Technical Writer
- System Administrator
- Tool Specialist
- Course Developer
- Graphic Artist

Testers

- Tester
- Test Analyst
- Test Designer

Additional Roles

- Review Coordinator
- Technical Reviewer
- Stakeholder

Disciplines

- A collection of related activities that are related to a major *area of concern* within the overall project
- Disciplines span phases

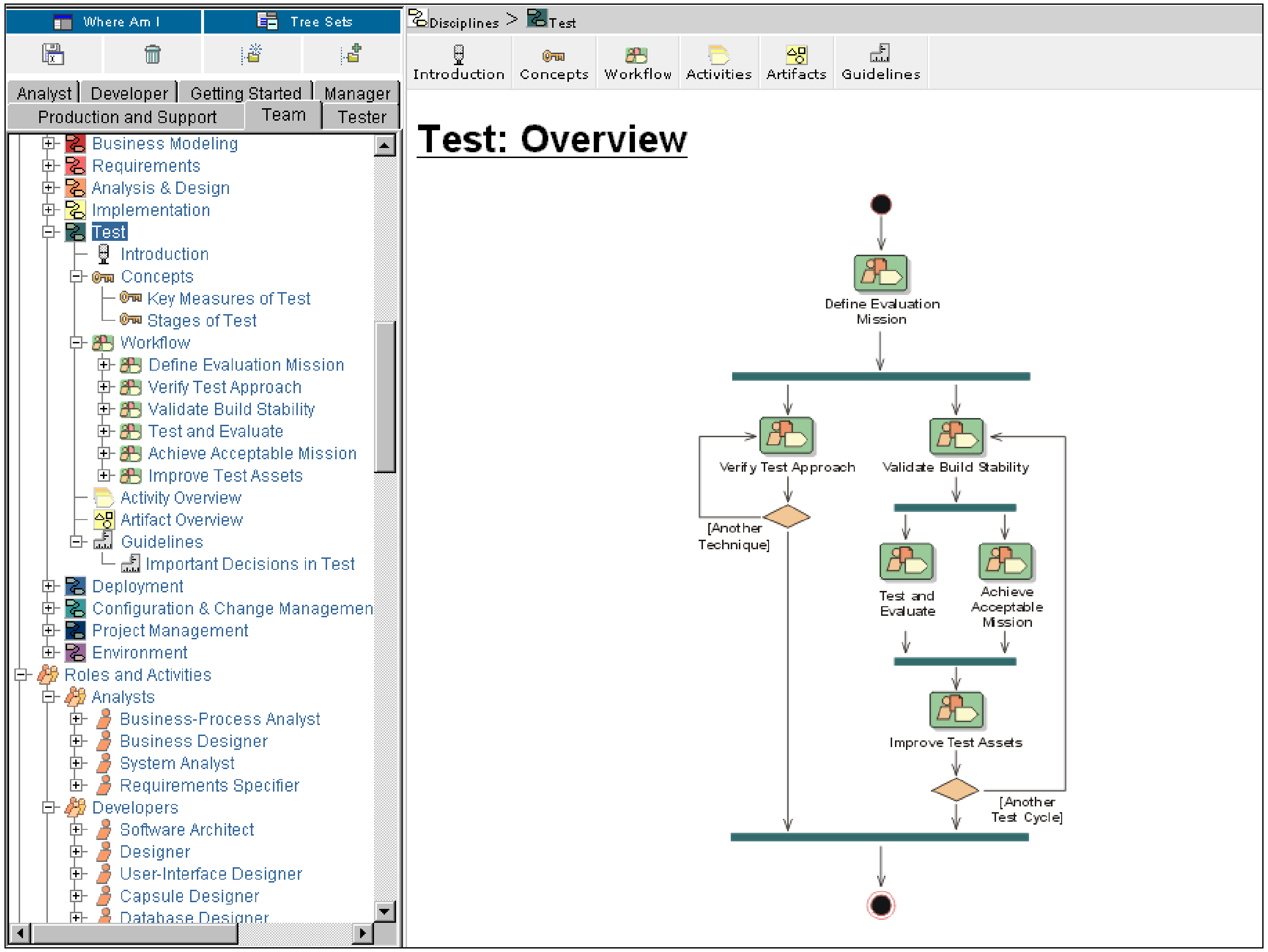
RUP™'s Disciplines

- Business Modeling
- Requirements
- Analysis and Design (Analysis <> Requirements)
- Implementation
- Test
- Deployment
- Configuration and Change Management
- Project Management
- Environment

Each Discipline is Composed of:

- Overview
- Introduction
- Concepts
- Workflow - the high level activity diagram (process flow)
- Workflow detail - second level process
- Activities - actions performed by roles
- Artifacts - deliverables
- Guidelines - tutorials, checklists, etc

Discipline Overview

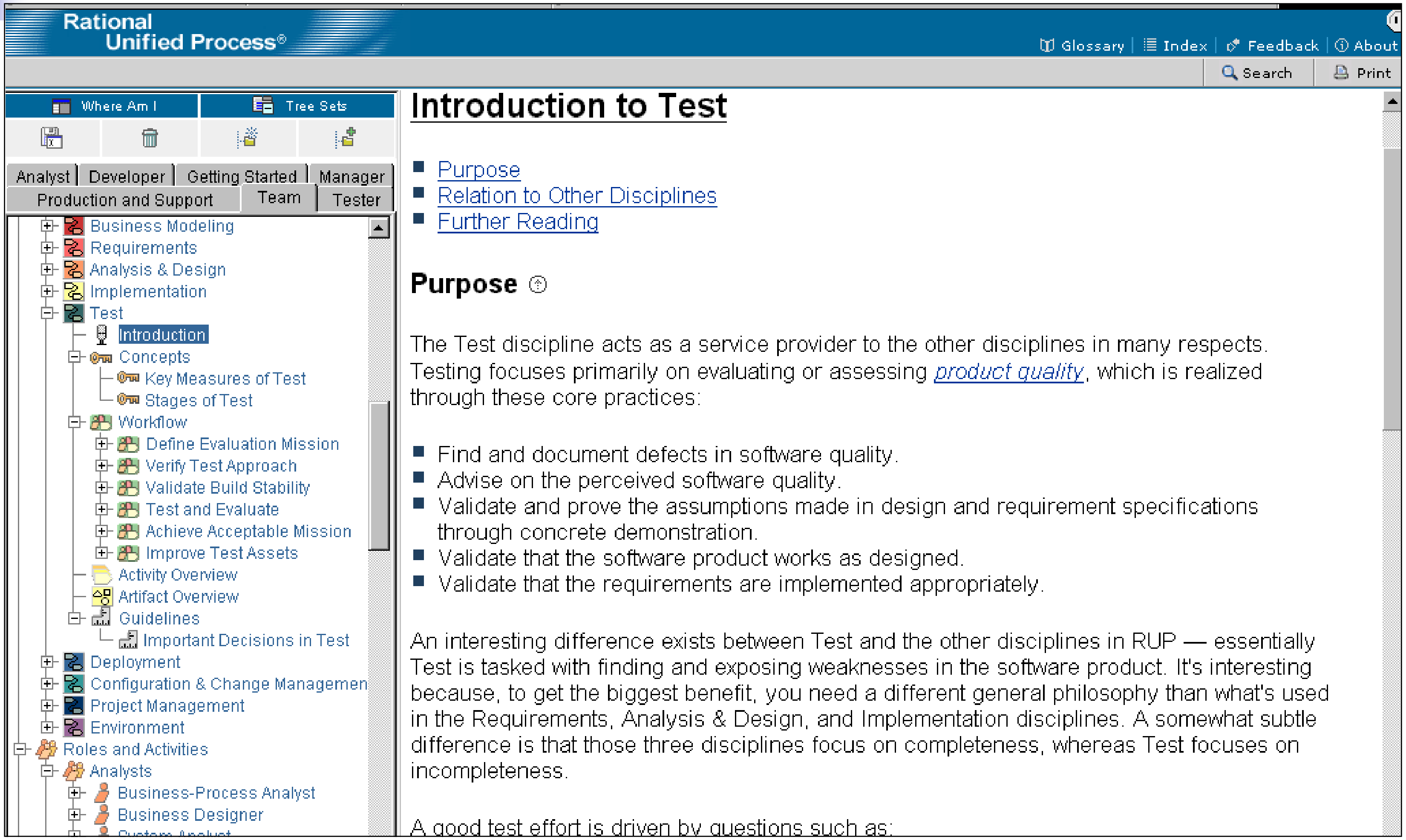


Business Modeling
Requirements
Analysis & Design
Implementation
Test
- Introduction
- Concepts
  - Key Measures of Test
  - Stages of Test
- Workflow
  - Define Evaluation Mission
  - Verify Test Approach
  - Validate Build Stability
  - Test and Evaluate
  - Achieve Acceptable Mission
  - Improve Test Assets
- Activity Overview
- Artifact Overview
- Guidelines
  - Important Decisions in Test

Deployment
Configuration & Change Managemen
Project Management
Environment
Roles and Activities
- Analysts
  - Business-Process Analyst
  - Business Designer
  - System Analyst
  - Requirements Specifier
- Developers
  - Software Architect
  - Designer
  - User-Interface Designer
  - Capsule Designer
  - Database Designer

Test: Overview

Introduction



Introduction to Test

- Purpose
- Relation to Other Disciplines
- Further Reading

Purpose

The Test discipline acts as a service provider to the other disciplines in many respects. Testing focuses primarily on evaluating or assessing *product quality*, which is realized through these core practices:

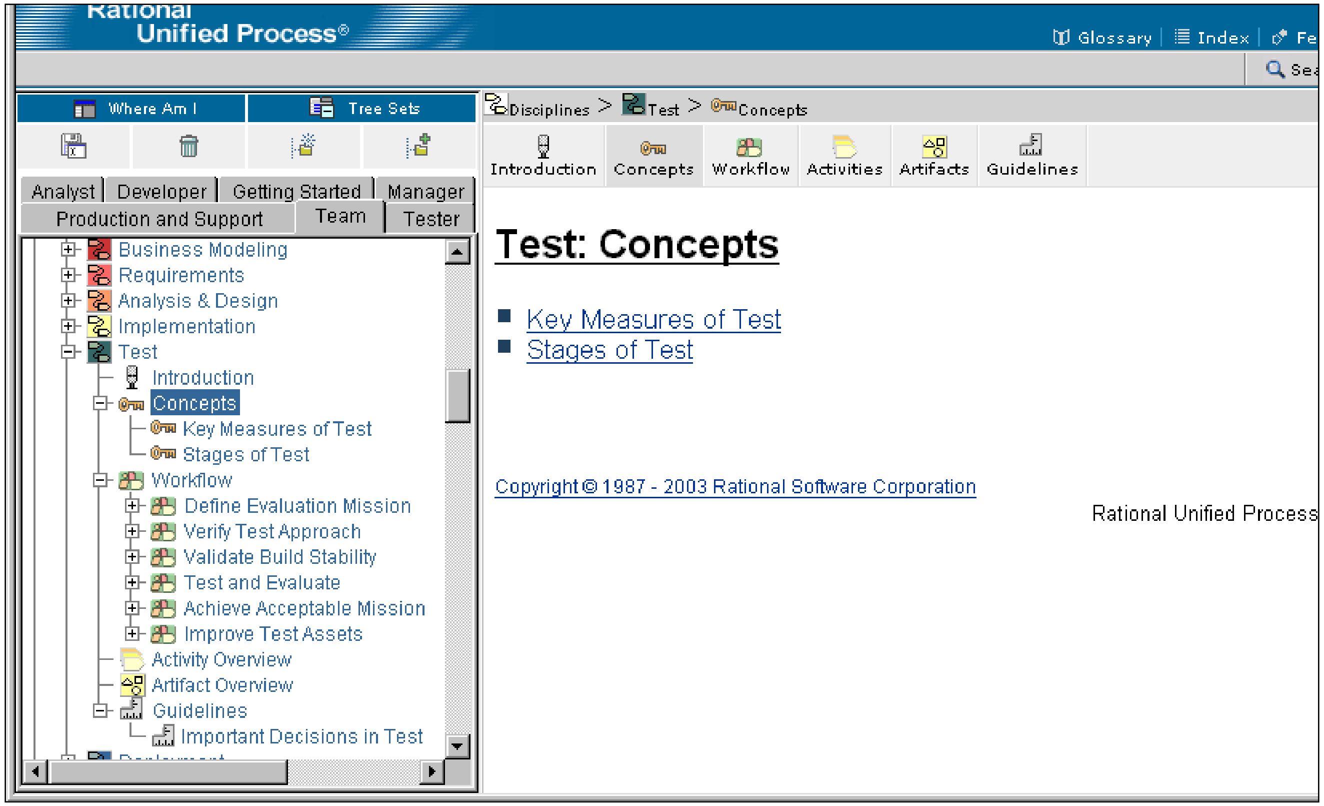
- Find and document defects in software quality.
- Advise on the perceived software quality.
- Validate and prove the assumptions made in design and requirement specifications through concrete demonstration.
- Validate that the software product works as designed.
- Validate that the requirements are implemented appropriately.

An interesting difference exists between Test and the other disciplines in RUP — essentially Test is tasked with finding and exposing weaknesses in the software product. It's interesting because, to get the biggest benefit, you need a different general philosophy than what's used in the Requirements, Analysis & Design, and Implementation disciplines. A somewhat subtle difference is that those three disciplines focus on completeness, whereas Test focuses on incompleteness.

A good test effort is driven by questions such as:

Concepts

- Links to tutorials, explanation, white papers, etc.



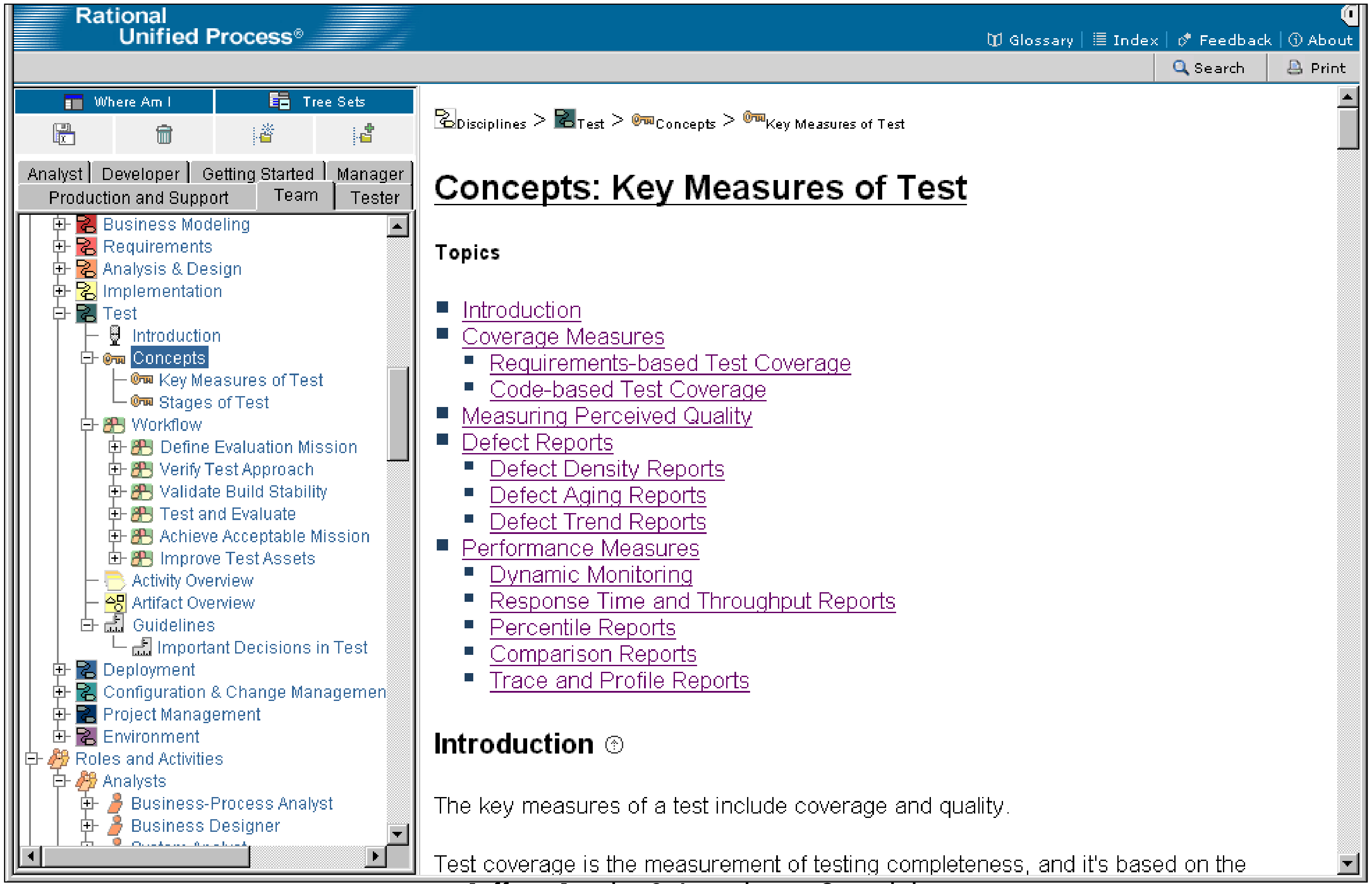
Test: Concepts

- Key Measures of Test
- Stages of Test

Copyright © 1987 - 2003 Rational Software Corporation

Rational Unified Process

Concept Example


Rational Unified Process®

Glossary | Index | Feedback | About

Search | Print

Where Am I | Tree Sets

Analyst | Developer | Getting Started | Manager | Production and Support | Team | Tester

- Business Modeling
- Requirements
- Analysis & Design
- Implementation
- Test
  - Introduction
  - Concepts
    - Key Measures of Test
    - Stages of Test
  - Workflow
    - Define Evaluation Mission
    - Verify Test Approach
    - Validate Build Stability
    - Test and Evaluate
    - Achieve Acceptable Mission
    - Improve Test Assets
  - Activity Overview
  - Artifact Overview
  - Guidelines
    - Important Decisions in Test
- Deployment
- Configuration & Change Managemen
- Project Management
- Environment
- Roles and Activities
  - Analysts
    - Business-Process Analyst
    - Business Designer

Disciplines > Test > Concepts > Key Measures of Test

Concepts: Key Measures of Test

Topics

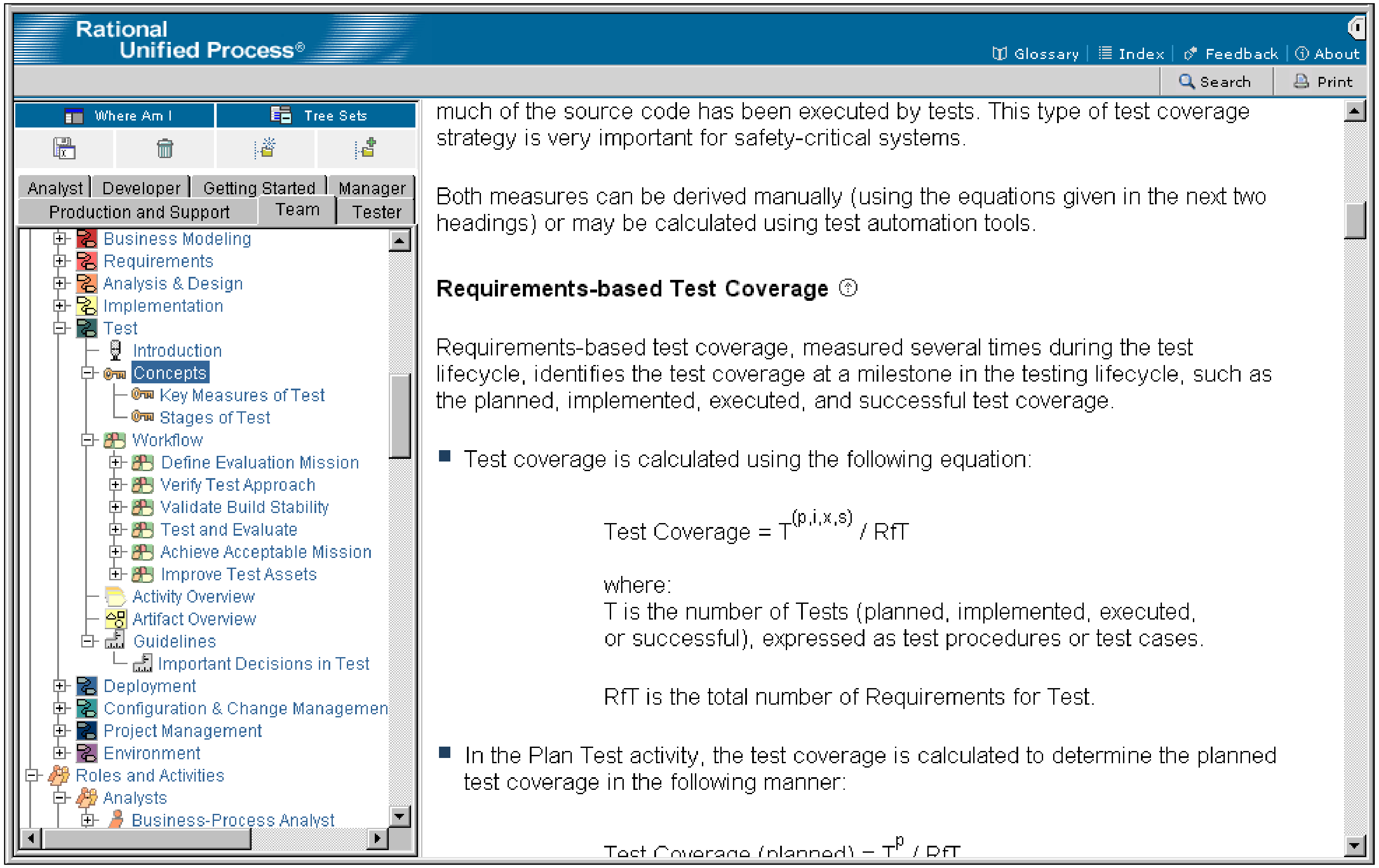
- Introduction
- Coverage Measures
  - Requirements-based Test Coverage
  - Code-based Test Coverage
- Measuring Perceived Quality
- Defect Reports
  - Defect Density Reports
  - Defect Aging Reports
  - Defect Trend Reports
- Performance Measures
  - Dynamic Monitoring
  - Response Time and Throughput Reports
  - Percentile Reports
  - Comparison Reports
  - Trace and Profile Reports

Introduction

The key measures of a test include coverage and quality.

Test coverage is the measurement of testing completeness, and it's based on the

More Concept


Rational Unified Process®

Glossary | Index | Feedback | About

Search | Print

Where Am I | Tree Sets

Analyst | Developer | Getting Started | Manager

Production and Support | Team | Tester

- Business Modeling
- Requirements
- Analysis & Design
- Implementation
- Test
  - Introduction
  - Concepts
    - Key Measures of Test
    - Stages of Test
  - Workflow
    - Define Evaluation Mission
    - Verify Test Approach
    - Validate Build Stability
    - Test and Evaluate
    - Achieve Acceptable Mission
    - Improve Test Assets
  - Activity Overview
  - Artifact Overview
  - Guidelines
    - Important Decisions in Test
- Deployment
- Configuration & Change Managemen
- Project Management
- Environment
- Roles and Activities
  - Analysts
    - Business-Process Analyst

much of the source code has been executed by tests. This type of test coverage strategy is very important for safety-critical systems.

Both measures can be derived manually (using the equations given in the next two headings) or may be calculated using test automation tools.

Requirements-based Test Coverage

Requirements-based test coverage, measured several times during the test lifecycle, identifies the test coverage at a milestone in the testing lifecycle, such as the planned, implemented, executed, and successful test coverage.

- Test coverage is calculated using the following equation:

$$\text{Test Coverage} = T^{(p,i,x,s)} / RfT$$

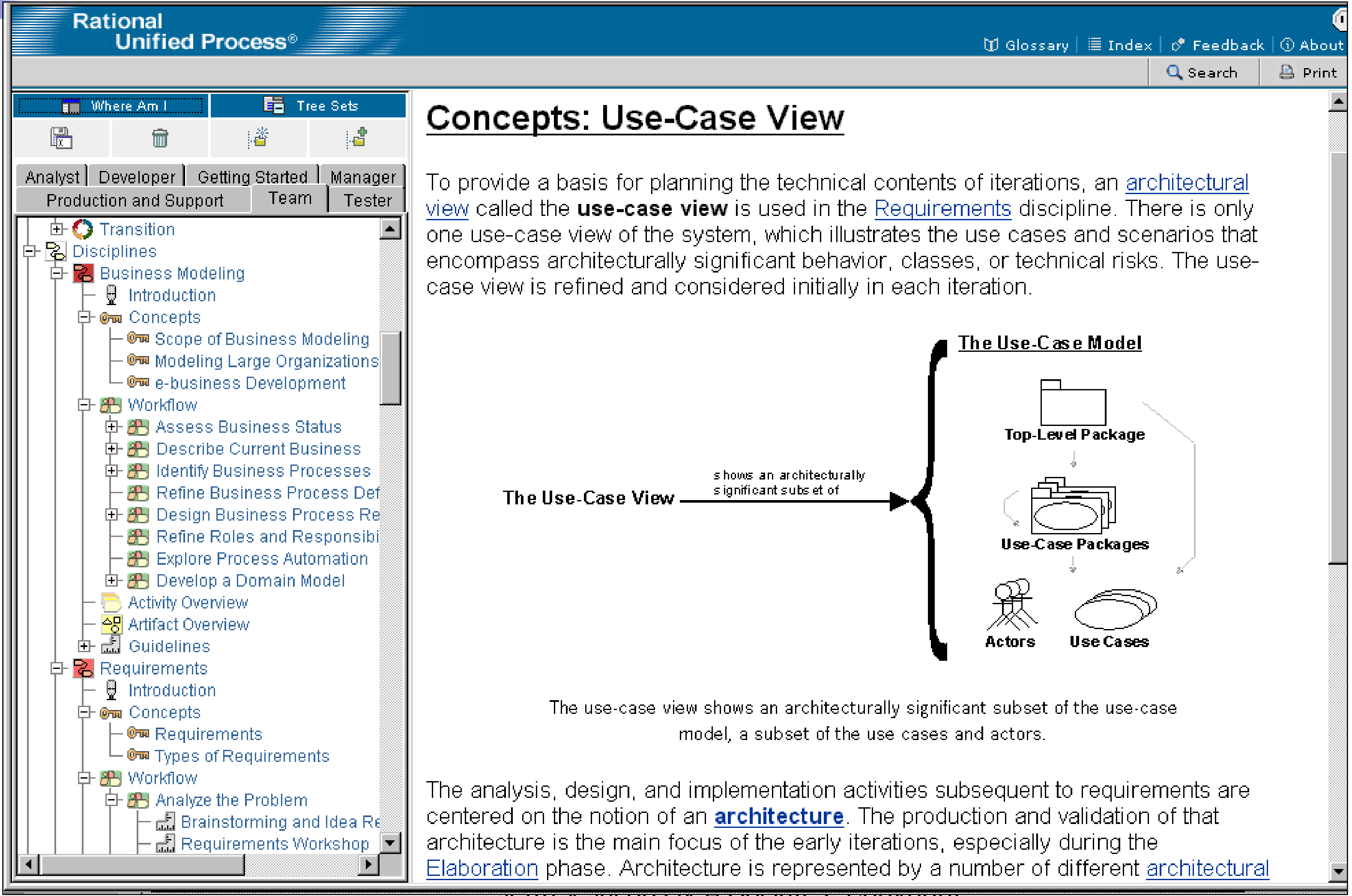
where:
T is the number of Tests (planned, implemented, executed, or successful), expressed as test procedures or test cases.

RfT is the total number of Requirements for Test.

- In the Plan Test activity, the test coverage is calculated to determine the planned test coverage in the following manner:

$$\text{Test Coverage (planned)} = T^{p} / RfT$$

Everything is Use-Case Driven



Rational Unified Process®
Glossary | Index | Feedback | About
Search | Print
Where Am I | Tree Sets
Analyst | Developer | Getting Started | Manager
Production and Support | Team | Tester
Transition
Disciplines
Business Modeling
Introduction
Concepts
Scope of Business Modeling
Modeling Large Organizations
e-business Development
Workflow
Assess Business Status
Describe Current Business
Identify Business Processes
Refine Business Process Def
Design Business Process Re
Refine Roles and Responsibi
Explore Process Automation
Develop a Domain Model
Activity Overview
Artifact Overview
Guidelines
Requirements
Introduction
Concepts
Requirements
Types of Requirements
Workflow
Analyze the Problem
Brainstorming and Idea Re
Requirements Workshop

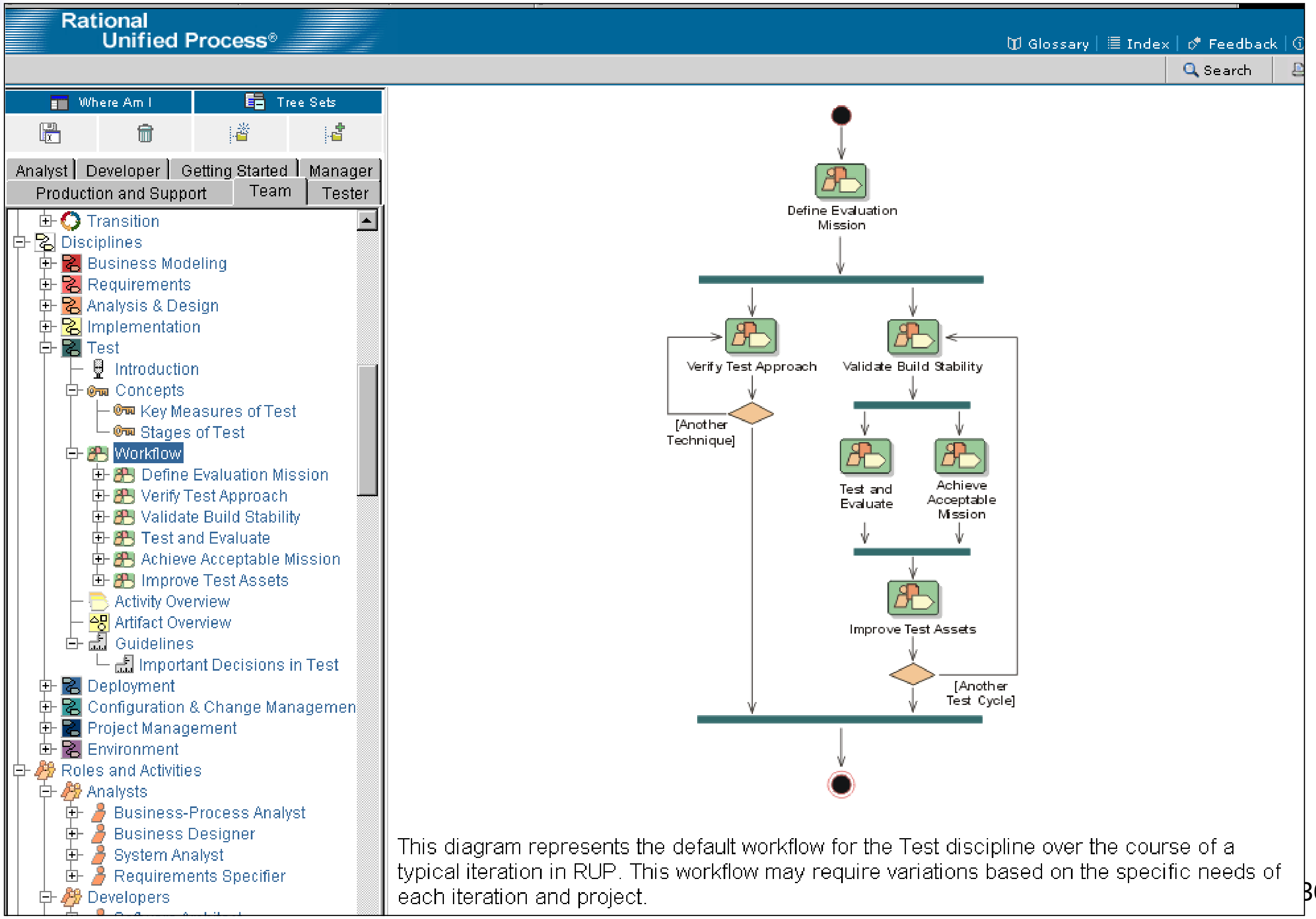
Concepts: Use-Case View

To provide a basis for planning the technical contents of iterations, an architectural view called the **use-case view** is used in the Requirements discipline. There is only one use-case view of the system, which illustrates the use cases and scenarios that encompass architecturally significant behavior, classes, or technical risks. The use-case view is refined and considered initially in each iteration.

The use-case view shows an architecturally significant subset of the use-case model, a subset of the use cases and actors.

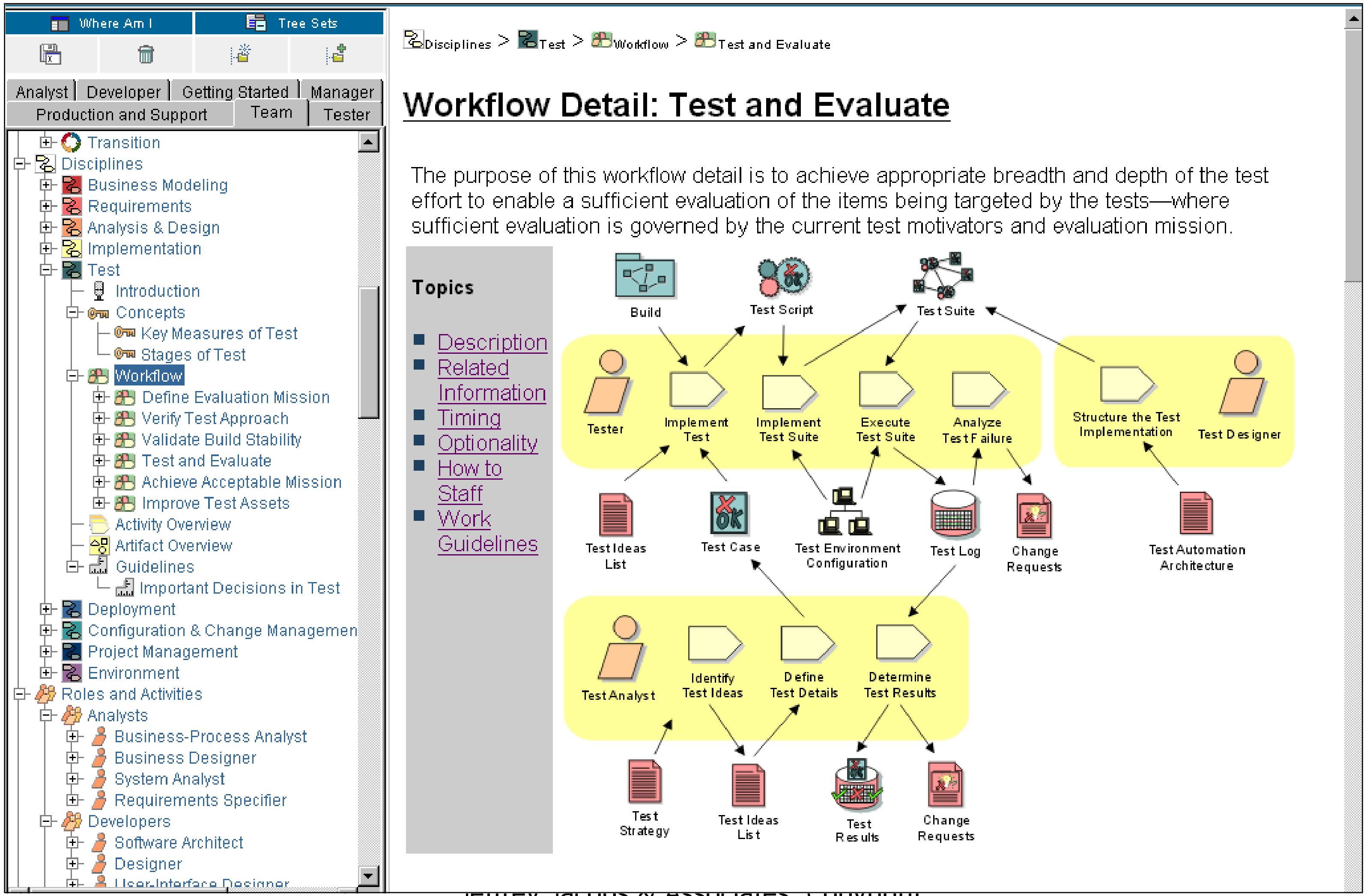
The analysis, design, and implementation activities subsequent to requirements are centered on the notion of an **architecture**. The production and validation of that architecture is the main focus of the early iterations, especially during the Elaboration phase. Architecture is represented by a number of different architectural

Workflow



This diagram represents the default workflow for the Test discipline over the course of a typical iteration in RUP. This workflow may require variations based on the specific needs of each iteration and project.

Workflow Detail



Where Am I | Tree Sets

Analyst | Developer | Getting Started | Manager | Production and Support | Team | Tester

- Transition
- Disciplines
  - Business Modeling
  - Requirements
  - Analysis & Design
  - Implementation
  - Test
    - Introduction
    - Concepts
      - Key Measures of Test
      - Stages of Test
    - Workflow
      - Define Evaluation Mission
      - Verify Test Approach
      - Validate Build Stability
      - Test and Evaluate
      - Achieve Acceptable Mission
      - Improve Test Assets
    - Activity Overview
    - Artifact Overview
    - Guidelines
      - Important Decisions in Test
  - Deployment
  - Configuration & Change Managemen
  - Project Management
  - Environment
- Roles and Activities
  - Analysts
    - Business-Process Analyst
    - Business Designer
    - System Analyst
    - Requirements Specifier
  - Developers
    - Software Architect
    - Designer
    - User-Interface Designer

Disciplines > Test > Workflow > Test and Evaluate

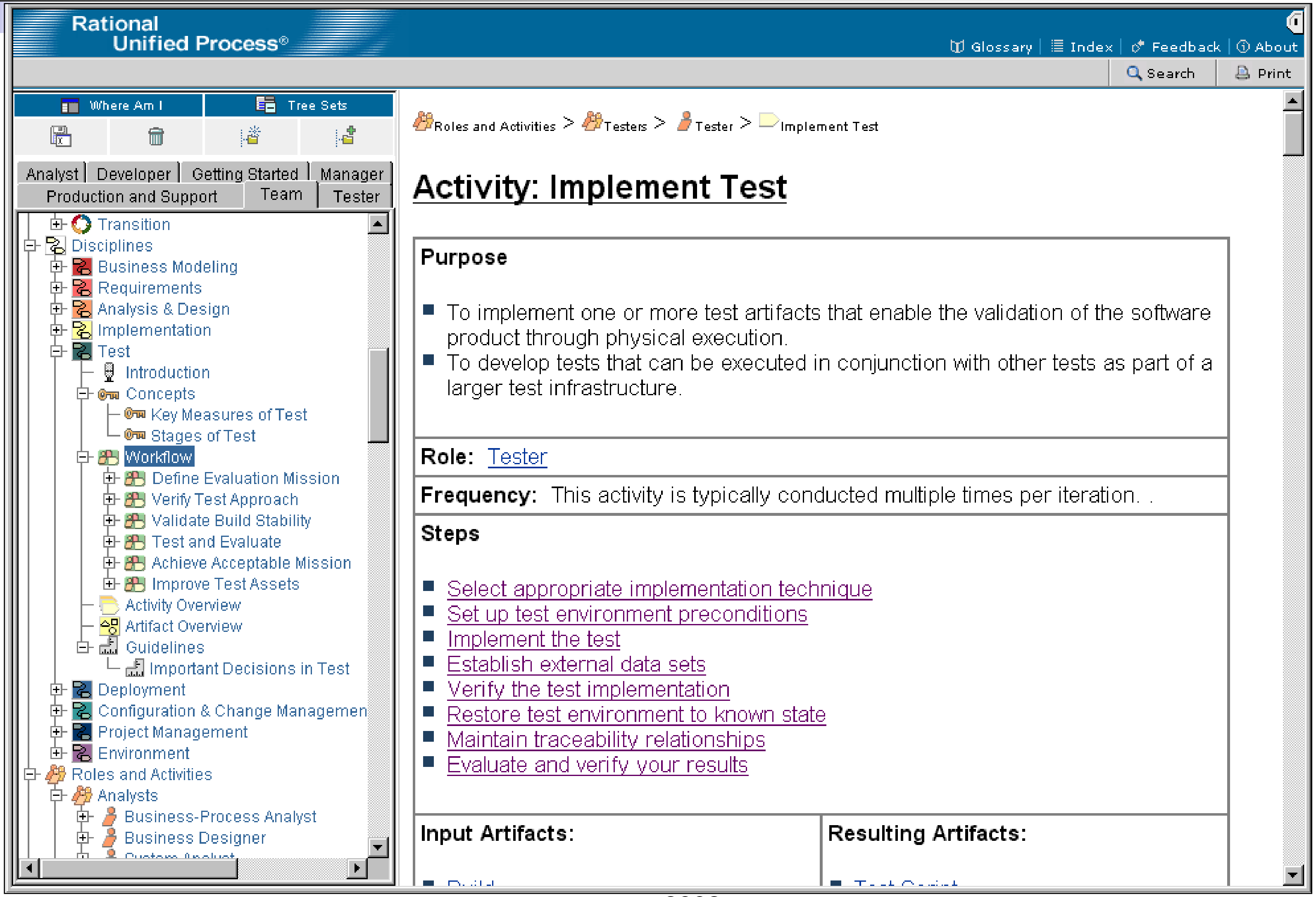
Workflow Detail: Test and Evaluate

The purpose of this workflow detail is to achieve appropriate breadth and depth of the test effort to enable a sufficient evaluation of the items being targeted by the tests—where sufficient evaluation is governed by the current test motivators and evaluation mission.

Topics

- Description
- Related Information
- Timing
- Optionality
- How to Staff
- Work Guidelines

Activity



Activity: Implement Test

Purpose

- To implement one or more test artifacts that enable the validation of the software product through physical execution.
- To develop tests that can be executed in conjunction with other tests as part of a larger test infrastructure.

Role: Tester

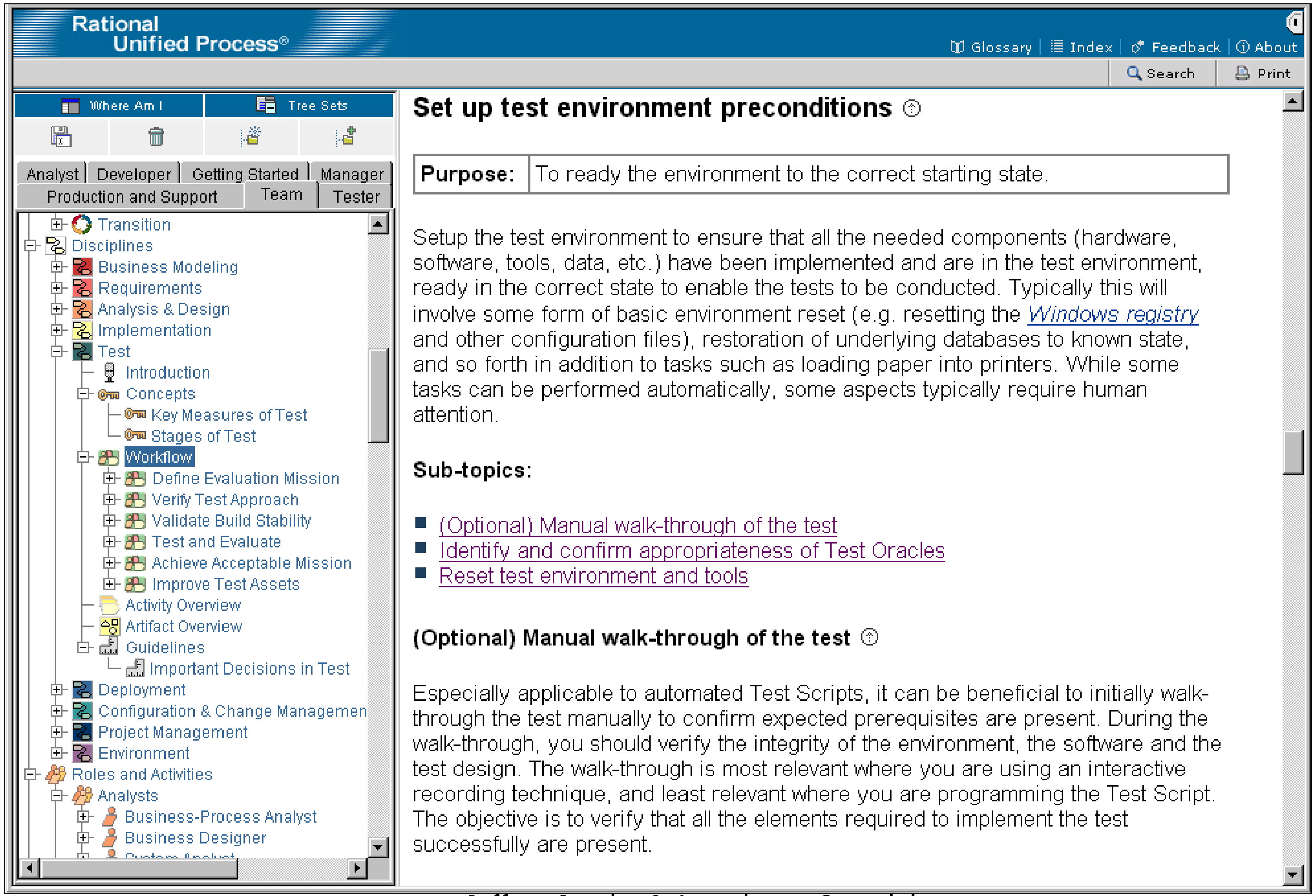
Frequency: This activity is typically conducted multiple times per iteration. .

Steps

- Select appropriate implementation technique
- Set up test environment preconditions
- Implement the test
- Establish external data sets
- Verify the test implementation
- Restore test environment to known state
- Maintain traceability relationships
- Evaluate and verify your results

Input Artifacts:	Resulting Artifacts:

Activity Step


Rational Unified Process®
Glossary | Index | Feedback | About
Search
Print
Where Am I
Tree Sets
Analyst | Developer | Getting Started | Manager
Production and Support | Team | Tester

- Transition
- Disciplines
  - Business Modeling
  - Requirements
  - Analysis & Design
  - Implementation
  - Test
    - Introduction
    - Concepts
      - Key Measures of Test
      - Stages of Test
    - Workflow
      - Define Evaluation Mission
      - Verify Test Approach
      - Validate Build Stability
      - Test and Evaluate
      - Achieve Acceptable Mission
      - Improve Test Assets
    - Activity Overview
    - Artifact Overview
    - Guidelines
      - Important Decisions in Test
  - Deployment
  - Configuration & Change Managemen
  - Project Management
  - Environment
- Roles and Activities
  - Analysts
    - Business-Process Analyst
    - Business Designer

Set up test environment preconditions

Purpose:	To ready the environment to the correct starting state.

Setup the test environment to ensure that all the needed components (hardware, software, tools, data, etc.) have been implemented and are in the test environment, ready in the correct state to enable the tests to be conducted. Typically this will involve some form of basic environment reset (e.g. resetting the *Windows registry* and other configuration files), restoration of underlying databases to known state, and so forth in addition to tasks such as loading paper into printers. While some tasks can be performed automatically, some aspects typically require human attention.

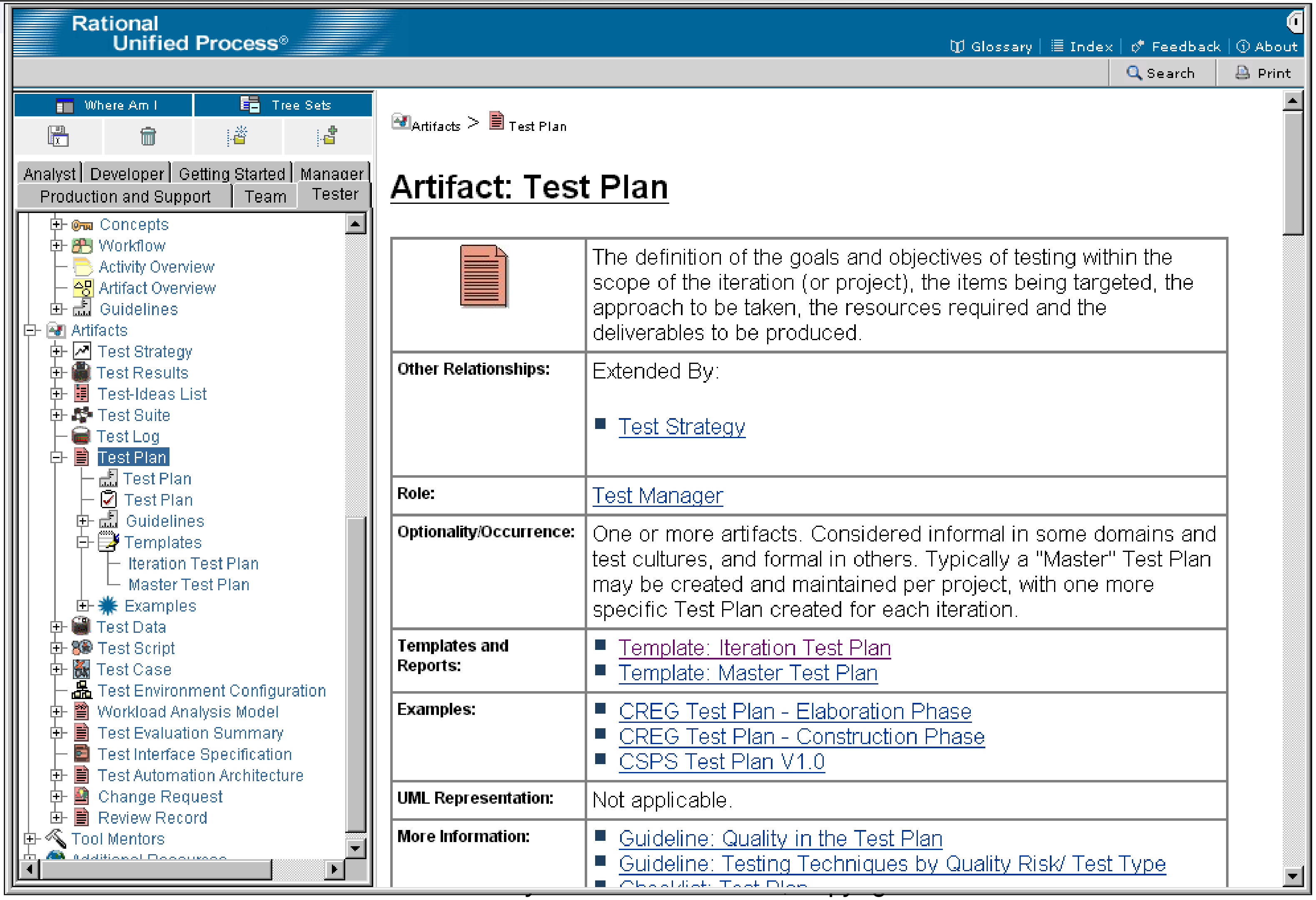
Sub-topics:

- (Optional) Manual walk-through of the test
- Identify and confirm appropriateness of Test Oracles
- Reset test environment and tools

(Optional) Manual walk-through of the test

Especially applicable to automated Test Scripts, it can be beneficial to initially walk-through the test manually to confirm expected prerequisites are present. During the walk-through, you should verify the integrity of the environment, the software and the test design. The walk-through is most relevant where you are using an interactive recording technique, and least relevant where you are programming the Test Script. The objective is to verify that all the elements required to implement the test successfully are present.

Artifacts May Be Documents, Models, Code, Etc. (Note change in tree)



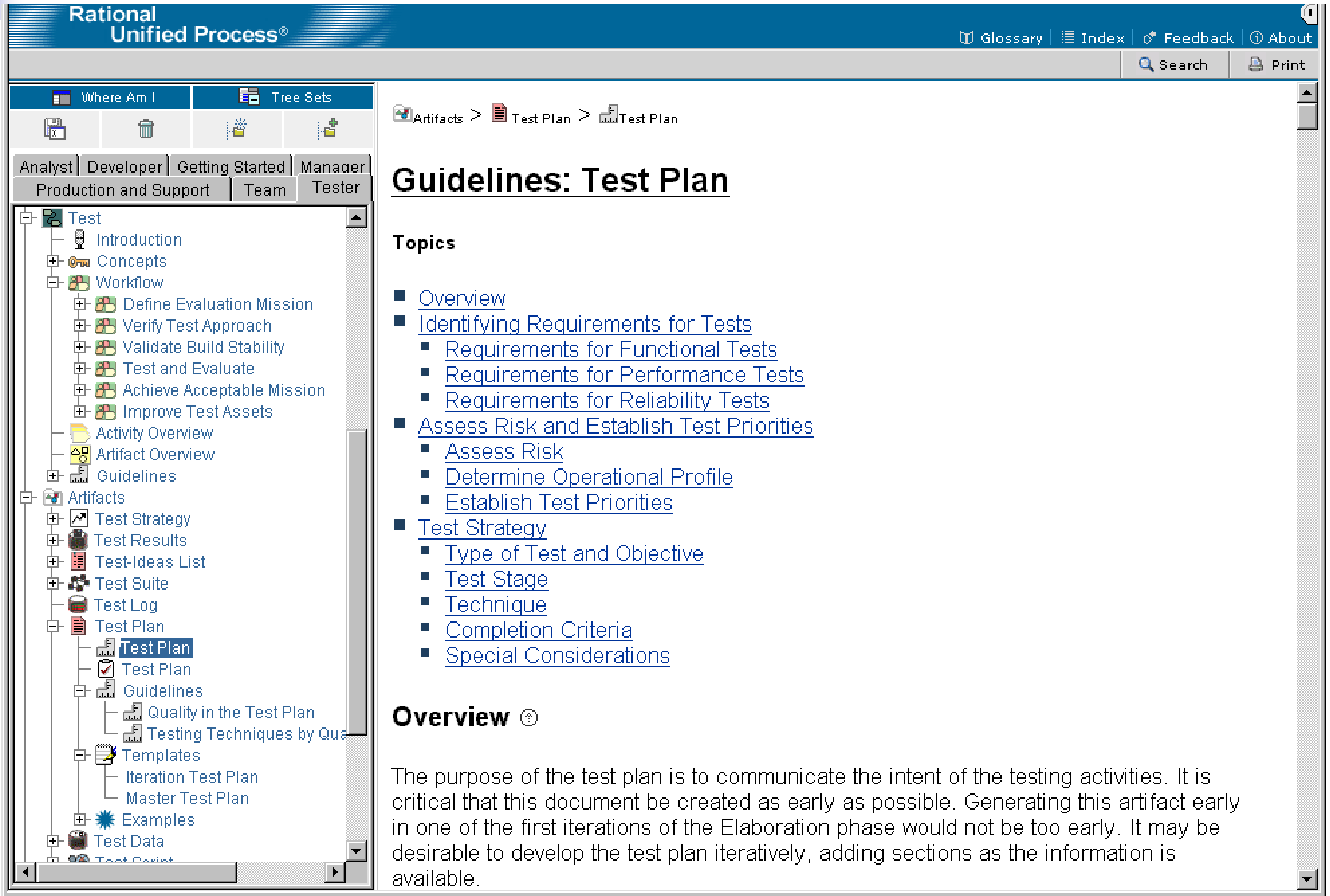
Artifact: Test Plan

	The definition of the goals and objectives of testing within the scope of the iteration (or project), the items being targeted, the approach to be taken, the resources required and the deliverables to be produced.
Other Relationships:	Extended By: ■ Test Strategy
Role:	Test Manager
Optionality/Occurrence:	One or more artifacts. Considered informal in some domains and test cultures, and formal in others. Typically a "Master" Test Plan may be created and maintained per project, with one more specific Test Plan created for each iteration.
Templates and Reports:	■ Template: Iteration Test Plan ■ Template: Master Test Plan
Examples:	■ CREG Test Plan - Elaboration Phase ■ CREG Test Plan - Construction Phase ■ CSPS Test Plan V1.0
UML Representation:	Not applicable.
More Information:	■ Guideline: Quality in the Test Plan ■ Guideline: Testing Techniques by Quality Risk/ Test Type

Document Templates

- Templates for document artifacts available in a variety of formats
  - Microsoft Word
  - HTML
  - Framemaker
  - Rational SODA
- Iteration Test Plan Template

Guidelines



Guidelines: Test Plan

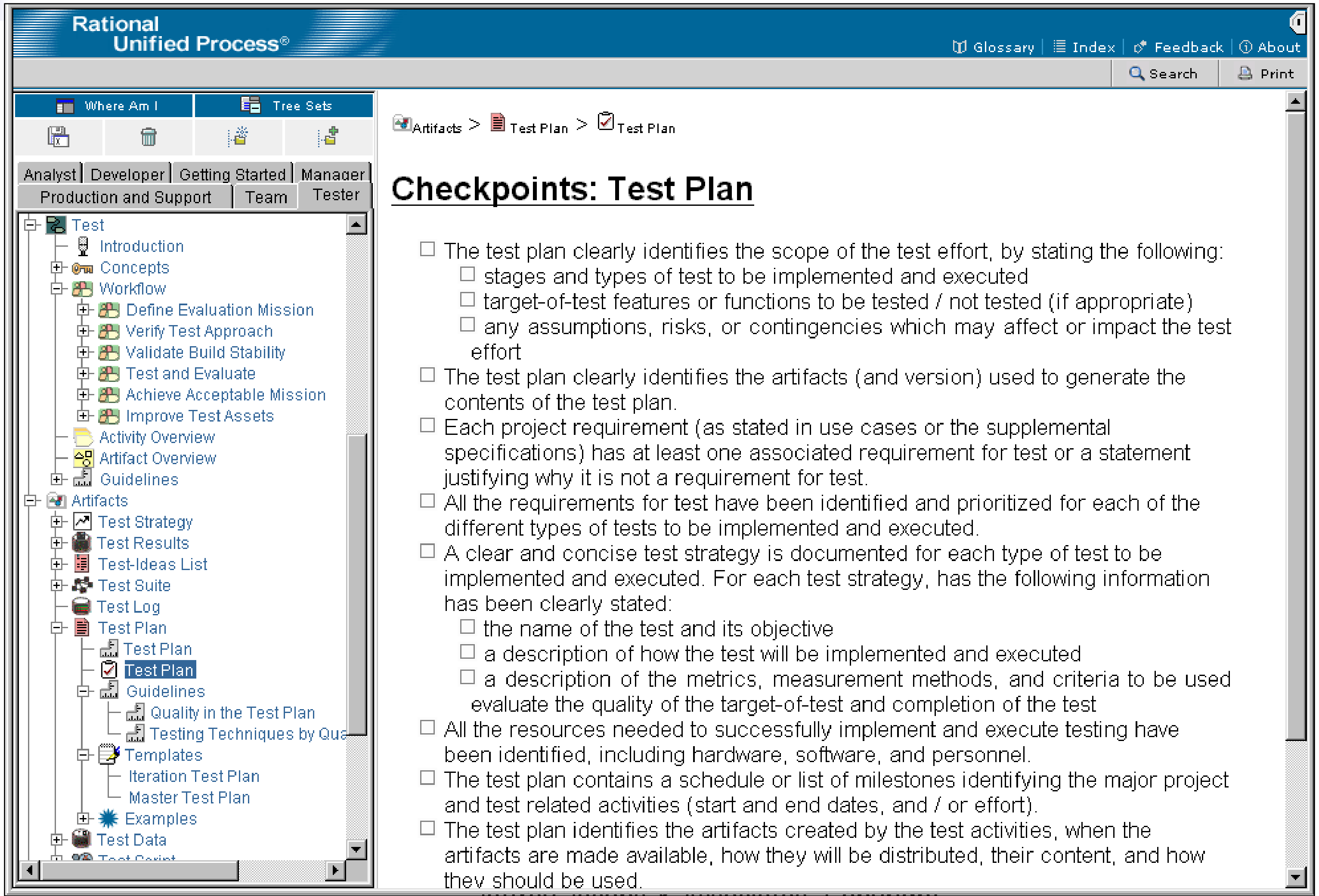
Topics

- Overview
- Identifying Requirements for Tests
  - Requirements for Functional Tests
  - Requirements for Performance Tests
  - Requirements for Reliability Tests
- Assess Risk and Establish Test Priorities
  - Assess Risk
  - Determine Operational Profile
  - Establish Test Priorities
- Test Strategy
  - Type of Test and Objective
  - Test Stage
  - Technique
  - Completion Criteria
  - Special Considerations

Overview

The purpose of the test plan is to communicate the intent of the testing activities. It is critical that this document be created as early as possible. Generating this artifact early in one of the first iterations of the Elaboration phase would not be too early. It may be desirable to develop the test plan iteratively, adding sections as the information is available.

Checkpoints for Quality Reviews



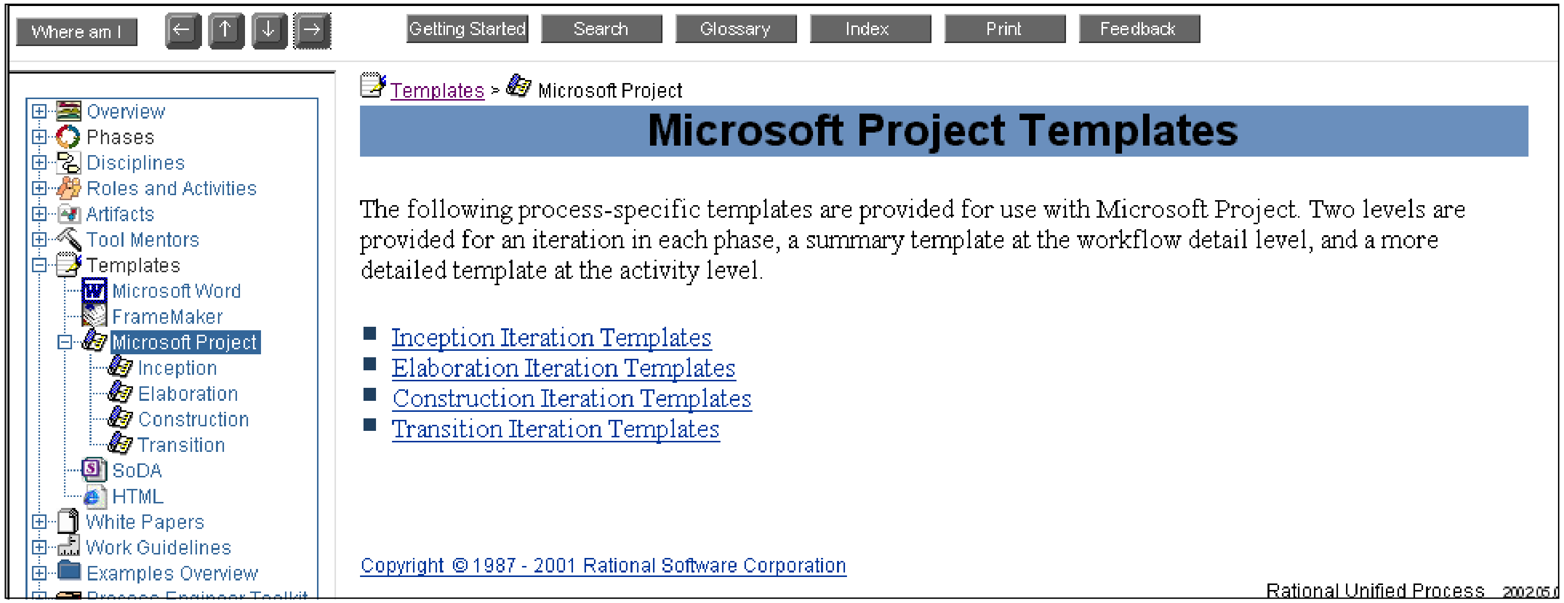
Checkpoints: Test Plan

- ☐ The test plan clearly identifies the scope of the test effort, by stating the following:
  - ☐ stages and types of test to be implemented and executed
  - ☐ target-of-test features or functions to be tested / not tested (if appropriate)
  - ☐ any assumptions, risks, or contingencies which may affect or impact the test effort
- ☐ The test plan clearly identifies the artifacts (and version) used to generate the contents of the test plan.
- ☐ Each project requirement (as stated in use cases or the supplemental specifications) has at least one associated requirement for test or a statement justifying why it is not a requirement for test.
- ☐ All the requirements for test have been identified and prioritized for each of the different types of tests to be implemented and executed.
- ☐ A clear and concise test strategy is documented for each type of test to be implemented and executed. For each test strategy, has the following information has been clearly stated:
  - ☐ the name of the test and its objective
  - ☐ a description of how the test will be implemented and executed
  - ☐ a description of the metrics, measurement methods, and criteria to be used evaluate the quality of the target-of-test and completion of the test
- ☐ All the resources needed to successfully implement and execute testing have been identified, including hardware, software, and personnel.
- ☐ The test plan contains a schedule or list of milestones identifying the major project and test related activities (start and end dates, and / or effort).
- ☐ The test plan identifies the artifacts created by the test activities, when the artifacts are made available, how they will be distributed, their content, and how they should be used.

More Stuff

- Project Management Templates
- Tool Mentors
- Sample Projects
- Reference/bibliography

Project Management Templates



Templates > Microsoft Project

Microsoft Project Templates

The following process-specific templates are provided for use with Microsoft Project. Two levels are provided for an iteration in each phase, a summary template at the workflow detail level, and a more detailed template at the activity level.

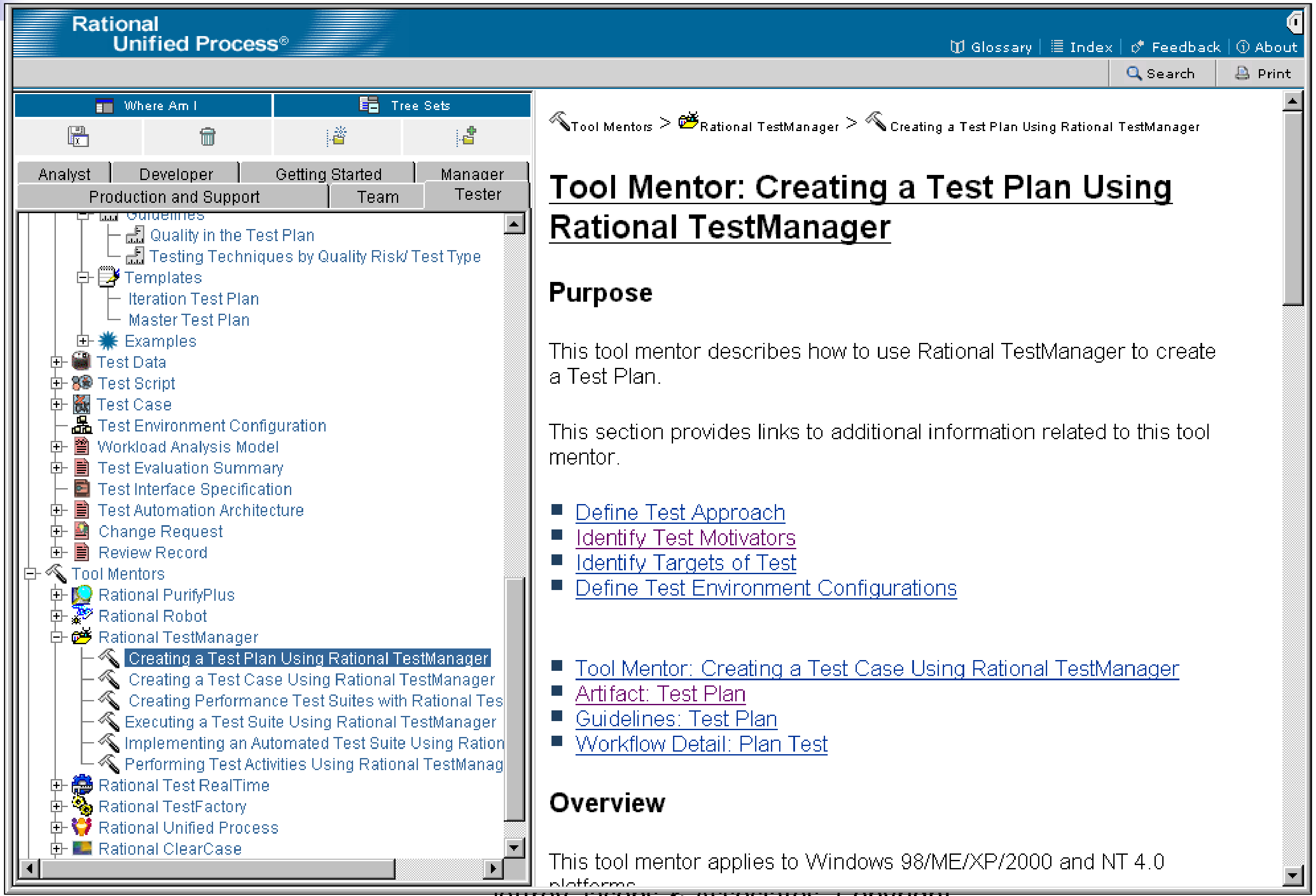
- Inception Iteration Templates
- Elaboration Iteration Templates
- Construction Iteration Templates
- Transition Iteration Templates

Copyright © 1987 - 2001 Rational Software Corporation

Rational Unified Process

- Summary MS Project Example
- Detail MS Project Example

Tool Mentors - How to Use Rational Tools in RUP™


Rational Unified Process®
Glossary
Index
Feedback
About
Search
Print
Where Am I
Tree Sets
Analyst
Developer
Getting Started
Manager
Production and Support
Team
Tester
Guidelines
Quality in the Test Plan
Testing Techniques by Quality Risk/ Test Type
Templates
Iteration Test Plan
Master Test Plan
Examples
Test Data
Test Script
Test Case
Test Environment Configuration
Workload Analysis Model
Test Evaluation Summary
Test Interface Specification
Test Automation Architecture
Change Request
Review Record
Tool Mentors
Rational PurifyPlus
Rational Robot
Rational TestManager
Creating a Test Plan Using Rational TestManager
Creating a Test Case Using Rational TestManager
Creating Performance Test Suites with Rational Tes
Executing a Test Suite Using Rational TestManager
Implementing an Automated Test Suite Using Ration
Performing Test Activities Using Rational TestManag
Rational Test RealTime
Rational TestFactory
Rational Unified Process
Rational ClearCase
Tool Mentors > Rational TestManager > Creating a Test Plan Using Rational TestManager

Tool Mentor: Creating a Test Plan Using Rational TestManager

Purpose

This tool mentor describes how to use Rational TestManager to create a Test Plan.

This section provides links to additional information related to this tool mentor.

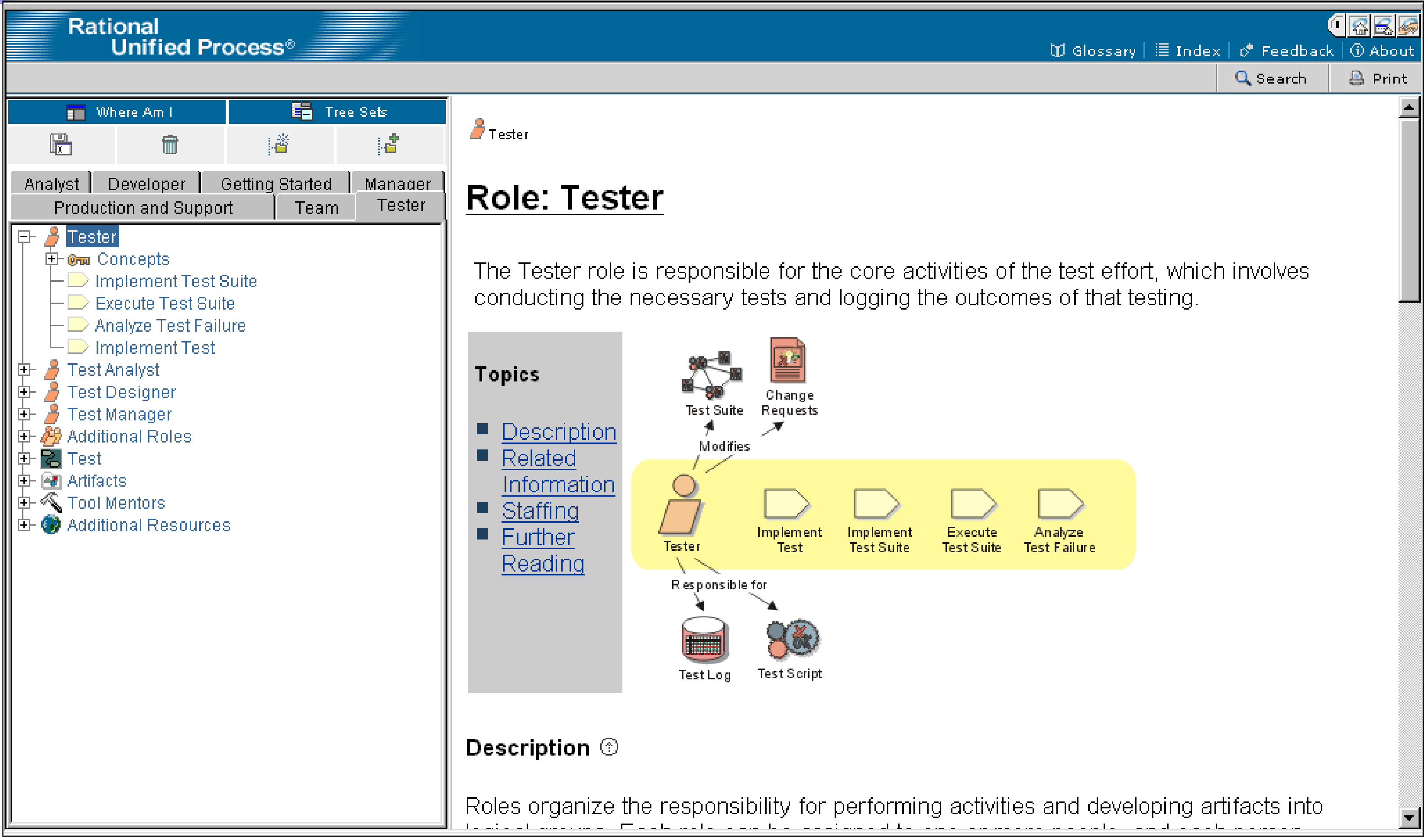
- Define Test Approach
- Identify Test Motivators
- Identify Targets of Test
- Define Test Environment Configurations

- Tool Mentor: Creating a Test Case Using Rational TestManager
- Artifact: Test Plan
- Guidelines: Test Plan
- Workflow Detail: Plan Test

Overview

This tool mentor applies to Windows 98/ME/XP/2000 and NT 4.0 platforms.

Tester Role



Rational Unified Process®

Glossary | Index | Feedback | About

Search | Print

Where Am I | Tree Sets

Analyst | Developer | Getting Started | Manager | Production and Support | Team | Tester

- Tester
  - Concepts
  - Implement Test Suite
  - Execute Test Suite
  - Analyze Test Failure
  - Implement Test
- Test Analyst
- Test Designer
- Test Manager
- Additional Roles
- Test
- Artifacts
- Tool Mentors
- Additional Resources

Tester

Role: Tester

The Tester role is responsible for the core activities of the test effort, which involves conducting the necessary tests and logging the outcomes of that testing.

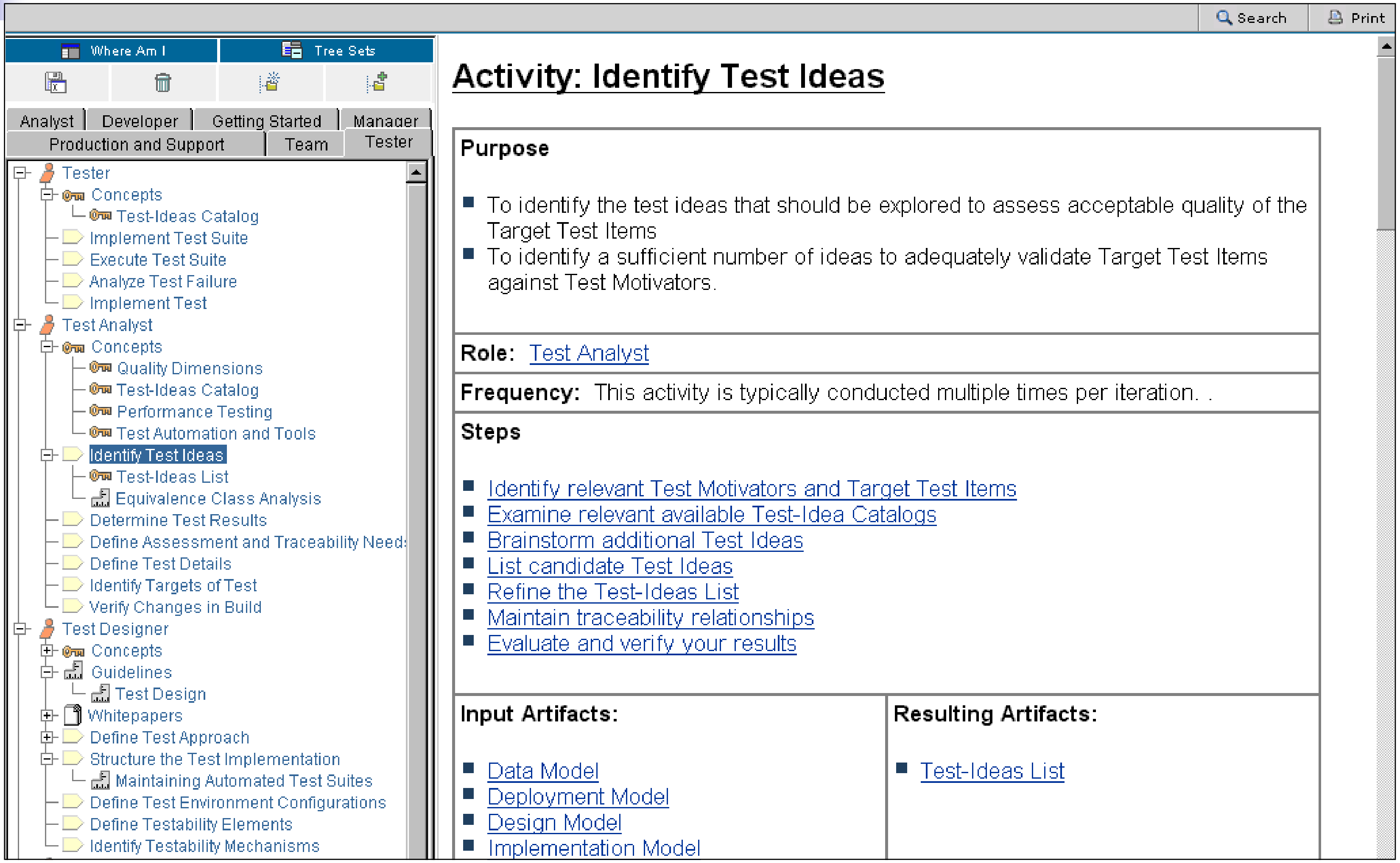
Topics

- Description
- Related Information
- Staffing
- Further Reading

Description

Roles organize the responsibility for performing activities and developing artifacts into

Tester Role Examples



- Tester
  - Concepts
    - Test-Ideas Catalog
  - Implement Test Suite
  - Execute Test Suite
  - Analyze Test Failure
  - Implement Test
- Test Analyst
  - Concepts
    - Quality Dimensions
    - Test-Ideas Catalog
    - Performance Testing
    - Test Automation and Tools
  - Identify Test Ideas
    - Test-Ideas List
    - Equivalence Class Analysis
  - Determine Test Results
  - Define Assessment and Traceability Needs
  - Define Test Details
  - Identify Targets of Test
  - Verify Changes in Build
- Test Designer
  - Concepts
  - Guidelines
    - Test Design
  - Whitepapers
  - Define Test Approach
  - Structure the Test Implementation
    - Maintaining Automated Test Suites
  - Define Test Environment Configurations
  - Define Testability Elements
  - Identify Testability Mechanisms

Activity: Identify Test Ideas

Purpose

- To identify the test ideas that should be explored to assess acceptable quality of the Target Test Items
- To identify a sufficient number of ideas to adequately validate Target Test Items against Test Motivators.

Role: Test Analyst

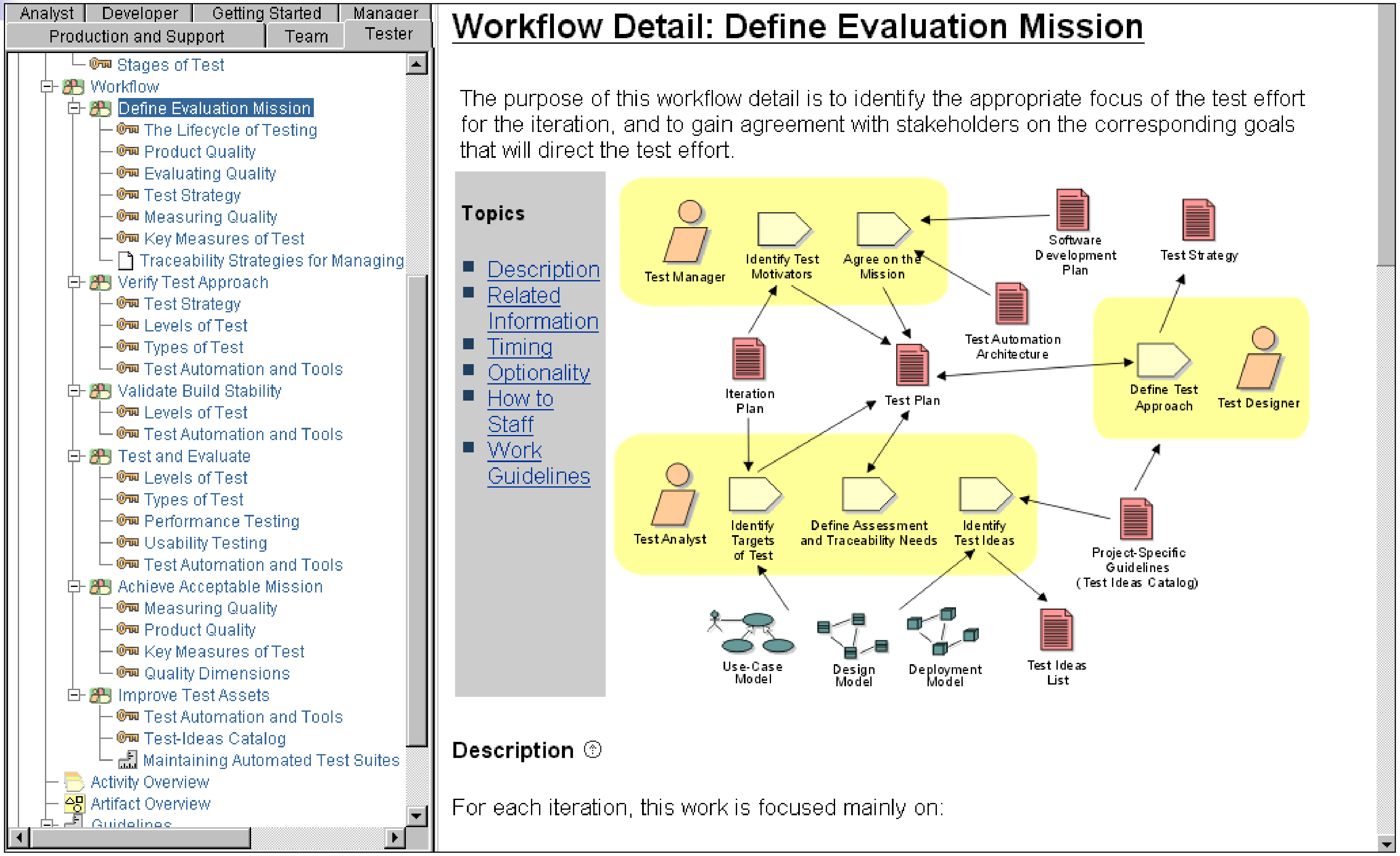
Frequency: This activity is typically conducted multiple times per iteration. .

Steps

- Identify relevant Test Motivators and Target Test Items
- Examine relevant available Test-Idea Catalogs
- Brainstorm additional Test Ideas
- List candidate Test Ideas
- Refine the Test-Ideas List
- Maintain traceability relationships
- Evaluate and verify your results

Input Artifacts:	Resulting Artifacts:
- Data Model - Deployment Model - Design Model - Implementation Model	- Test-Ideas List

Test Workflow Details



Analyst | Developer | Getting Started | Manager
Production and Support | Team | Tester

- Stages of Test
- Workflow
  - Define Evaluation Mission
    - The Lifecycle of Testing
    - Product Quality
    - Evaluating Quality
    - Test Strategy
    - Measuring Quality
    - Key Measures of Test
    - Traceability Strategies for Managing
  - Verify Test Approach
    - Test Strategy
    - Levels of Test
    - Types of Test
    - Test Automation and Tools
  - Validate Build Stability
    - Levels of Test
    - Test Automation and Tools
  - Test and Evaluate
    - Levels of Test
    - Types of Test
    - Performance Testing
    - Usability Testing
    - Test Automation and Tools
  - Achieve Acceptable Mission
    - Measuring Quality
    - Product Quality
    - Key Measures of Test
    - Quality Dimensions
  - Improve Test Assets
    - Test Automation and Tools
    - Test-Ideas Catalog
    - Maintaining Automated Test Suites
- Activity Overview
- Artifact Overview
- Guidelines

Workflow Detail: Define Evaluation Mission

The purpose of this workflow detail is to identify the appropriate focus of the test effort for the iteration, and to gain agreement with stakeholders on the corresponding goals that will direct the test effort.

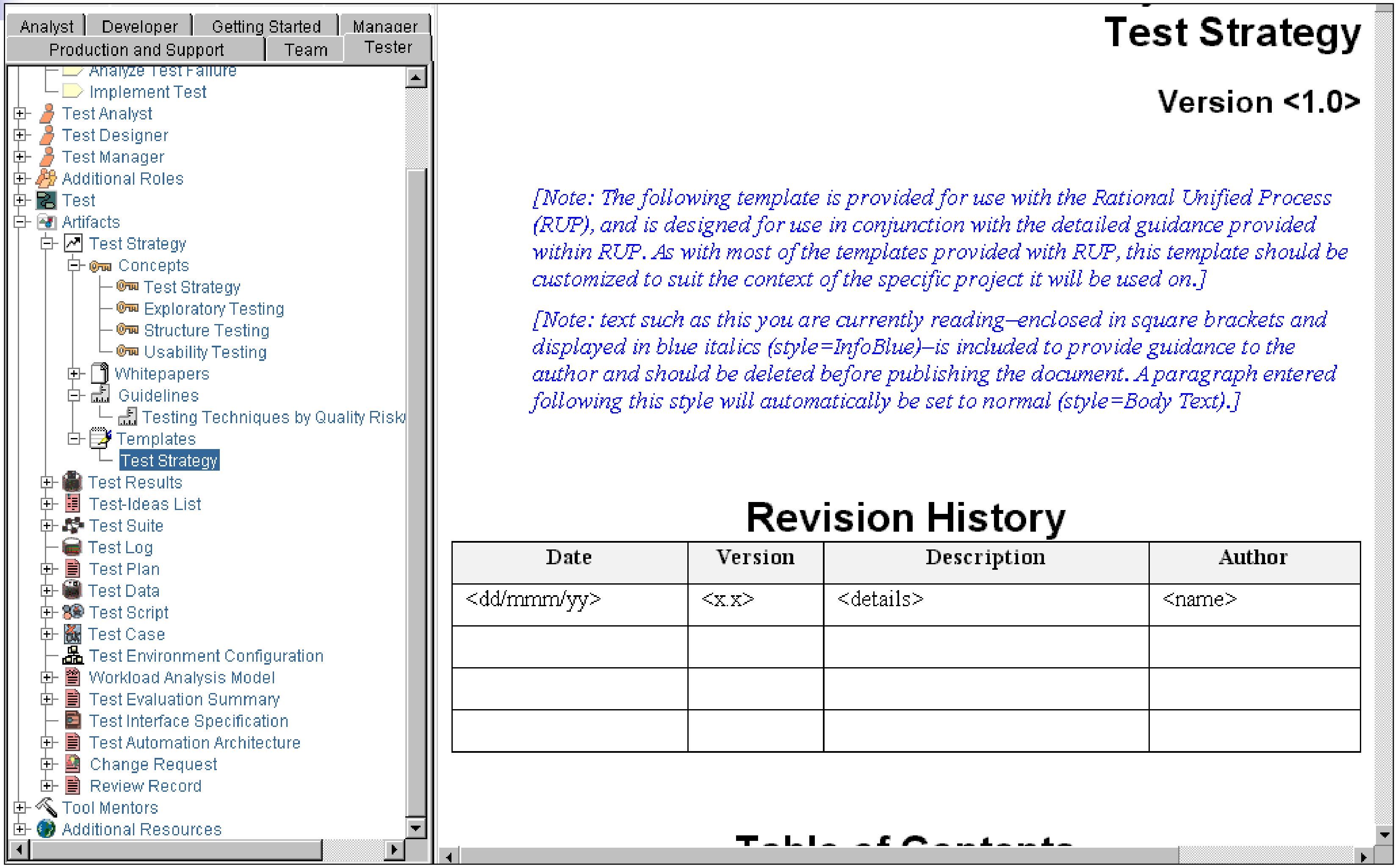
Topics

- Description
- Related Information
- Timing
- Optionality
- How to Staff
- Work Guidelines

Description

For each iteration, this work is focused mainly on:

Test Artifacts



Analyze Test Failure
Implement Test
Test Analyst
Test Designer
Test Manager
Additional Roles
Test
Artifacts
Test Strategy
Concepts
Test Strategy
Exploratory Testing
Structure Testing
Usability Testing
Whitepapers
Guidelines
Testing Techniques by Quality Risk
Templates
Test Strategy
Test Results
Test-Ideas List
Test Suite
Test Log
Test Plan
Test Data
Test Script
Test Case
Test Environment Configuration
Workload Analysis Model
Test Evaluation Summary
Test Interface Specification
Test Automation Architecture
Change Request
Review Record
Tool Mentors
Additional Resources

Test Strategy

Version <1.0>

[Note: The following template is provided for use with the Rational Unified Process (RUP), and is designed for use in conjunction with the detailed guidance provided within RUP. As with most of the templates provided with RUP, this template should be customized to suit the context of the specific project it will be used on.]

[Note: text such as this you are currently reading–enclosed in square brackets and displayed in blue italics (style=InfoBlue)–is included to provide guidance to the author and should be deleted before publishing the document. A paragraph entered following this style will automatically be set to normal (style=Body Text).]

Revision History

Date	Version	Description	Author
<dd/mmm/yy>	<x.x>	<details>	<name>

Summary

- Forms solid basis for improving software development process, particularly for ad-hoc, Level 1 organizations
- Provides basis for incorporating OOA/OOD/UML into current software development process
- Provides basis for development using IBM, Rational and Microsoft technologies
- 30 day on-line evaluation available, http://www.rational.com

Questions