Software Engineering
G22.2440-001

Session 1 – Sub-Topic 1 Presentation
Rational Tools Overview

Dr. Jean-Claude Franchitti

New York University
Computer Science Department
Courant Institute of Mathematical Sciences

Objectives

- Describe the Rational Tool Suite
- Provide an overview of each tool
- Discuss each one of the tools and how it fits into the overall software development process…
What is Software Development?

- In today’s business environment, almost every company produces some type of software. This can include:
  - Building new applications
  - Deploying applications in new production environments
  - Upgrading existing applications
  - Integrating with other software, partners and customers
  - Providing custom changes to packaged software

Why are Tools Needed?

- Software development requires teams to communicate and work together
- Changes need to be documented and tracked for:
  - Prioritization
  - Avoiding duplication of effort
  - Analyzing impact
  - Historical records
  - Legal Requirements
IBM/Rational Solution

- Supports every member of the development team
- Provides an end-to-end solution
- Can grow with your organization and projects
- Modular so appropriate tools can be mixed and matched and integrated
- Based on industry standards and best practices

Benefits the Entire Team

- Project Managers: Manage the project process, assignment scheduling and reporting. Efficiently integrate changes into production systems and easily gather necessary project data
- Business Analysts: Gather and store requirements and business terms, document use cases
- Technical Leads: Define the structure and design so that it integrates with new and existing systems
- Developers: Easy access to needed information to quickly build and test new systems
- Testers: Automated testing tools to ensure functionality, reliability and performance
A Sample Tool Set

- TUP [Team Unifying Platform]
  - Requisite Pro
  - ClearQuest
  - ClearCase
  - Project Console
- SoDA [Software Documentation Automation]
- Test Manager
- RUP [Rational Unified Process]
- XDE Developer for Java
- XDE Tester

Sample Tool Deployment Platform
Roadmap: Requirements Management

In this section:

- What is RequisitePro?
- What are requirements and why manage them?
- The Requirements Management Process
- Features and functions of RequisitePro

Why Requirements Management?

- Successful requirements management leads to a quality product:
  - that meets customer needs
  - is completed on time
  - is within budget
What is RequisitePro? (cont.)

- IBM Rational RequisitePro is an easy-to-use *requirements management tool*.
- It helps teams to:
  - Manage project requirements clearly
  - Promote communication and collaboration among team members
  - Reduce project risk
  - Keep stakeholders informed about the most current requirements information
  - Understand the impact of change which enables team members to manage project efficiently
  - Identifying the problem correctly to deliver the right product

What is RequisitePro? (cont.)

- RequisitePro
  - Improves and enhances the software development process.
  - Helps to prioritize requirements, to trace relationships between them, and mostly importantly track changes that affect them.
  - Captures the change history for each requirement.
  - Captures requirement traceability
What is a requirement?

- A requirement should be:
  - verifiable
  - necessary
  - complete
  - attainable
  - unambiguous
  - implementation-independent
  - concise
  - consistent
  - traceable

- Documenting requirements helps ensure the above.

Analyzing & Understanding Stakeholder Needs

- A **stakeholder** is someone who has a vested interest in the project/system/contract.
- **Stakeholder needs** can be obtained using various methods, such as:
  - interviews
  - questionnaires
  - requirement workshops
  - brainstorming sessions
  - story boards
  - prototypes

- The result of these interactions can easily be recorded in Rational RequisitePro.
- These documents become the **Functional Specification**
- A SoDA Report can be used to generate a custom Functional Specification from the Requisite Pro data
Examples of Requisite Pro documents that make up the Functional Specification may include:

- **Functional Framework Document**
  - Introduction (purpose, scope, overview)
  - Current and Proposed processes
- **Use-Case Documents** - designed to identify the functional behavior of the system
- **Supplementary Specification** – define non-functional requirements of the system
- **Glossary**
  - Common business terms and definitions
- **Requirements Management Plan** – deals with the definition of project requirements
- **Actor Specification** – define actors who use the system

**Features of RequisitePro**

- **Ease of Use**
  - Simple user interface and tight integration with Microsoft Word allows users to quickly & effectively document their project requirements.
- **Commercially Supported Databases** help to organize, track and communicate requirements.
  - Microsoft SQL Server 7.0 and 2000
  - Oracle versions 8. and 9
  - Access 2000 databases
- **Integrated with other lifecycle tools**
  - Integrated with Rational Rose®, Rational® XDE®, Rational® ClearCase®, Rational® ClearQuest®, Rational® SoDA®, Rational Unified Process, Rational® TestManager, and other products
Features of RequisitePro (cont.)

- **Traceability**
  - You can easily set up and track relationships
    - Between requirements
    - Between requirements and XDE models

- **RequisiteWeb**
  - Anyone who has Web access can view and modify requirements quickly and efficiently without having Rational RequisitePro loaded on their machine.

The Requirements Management Process

- The purpose of Requirements Management is to:
  - Understand the goals of the organization and its customers
  - Transform of these goals into potential functions and constraints applicable to the development and evolution of products and services.
  - Establish and maintain agreement with the client organizations and other stakeholders on what the system should do.
  - Provide system developers with a better understanding of the system requirements and tool integration.
  - Define the boundaries of the system.
  - Define a user-interface for the system, focusing on the needs and goals of the users.
1. Prepare Requirements Repository
   1.1 Set Up Functional
   1.2 Refine Project Scope

2. Define Requirements
   2.1 Define Current System Process
   2.2 Define High Level Requirements
   2.3 Develop Use Case Model
   2.4 Develop Management Overview
   2.5 Obtain Informal Client Approval

3. Prepare for Approval
   3.1 Finalize Functional Specification
   3.2 Complete IT Walkthrough
   3.3 Complete Functional PRB
   3.4 Incorporate PRB Feedback
   3.5 Baseline Requirements

To Change Management Process

Sample Requirements Management Process

Refer Requirements Management Process Definition v1.0

Level 1: Sub-processes
Level 2: Tasks

1. Prepare Requirements Repository
   1.1 Set Up Functional
   1.2 Refine Project Scope

2. Define Requirements
   2.1 Define Current System Process
   2.2 Define High Level Requirements
   2.3 Develop Use Case Model
   2.4 Develop Management Overview
   2.5 Obtain Informal Client Approval

3. Prepare for Approval
   3.1 Finalize Functional Specification
   3.2 Complete IT Walkthrough
   3.3 Complete Functional PRB
   3.4 Incorporate PRB Feedback
   3.5 Baseline Requirements

Basic tool navigation
RequisitePro Project Templates

- Creating a project
  - RequisitePro provides project templates to help you get started.

RequisitePro Project Templates (cont.)

- Use Case Template
  - Document types
    - Vision Document
    - Glossary
    - Use Case Specification
    - Supplementary Requirements Specification
    - Requirements Management Plan
    - Stakeholder Request Document
  - Requirement types
    - Feature
    - Glossary
    - Use Case
    - Supplementary Requirement
    - Requirements Management Plan Requirement
    - Stakeholder Request
RequisitePro Requirement Types

- RequisitePro provides the following requirements types:
  - Feature (FEAT)
  - Stakeholder Request (STRQ)
  - Supplementary (SUPL)
  - Glossary Item (TERM)
  - Use Case (UC)
  - Default for documents without requirements (NONE)

RequisitePro Requirement Types (cont.)

- This is an example of what a RequisitePro requirement looks like in a document:

  [UC1 The customer selects the withdraw money option from the menu.]
  [UC3 The ATM prompts the customer to enter the amount to be withdrawn.]
  [UC3.1 The ATM prompts the customer to operate in one of the two languages: English or Spanish]

  [FEAT1 Customer can also view account information via Internet.]
  [FEAT2 Customer can withdraw money.]
RequisitePro Attributes

- You can qualify requirement types using attributes.
- Attributes are data fields associated with each requirement.
- Each new RequisitePro requirement type is assigned default attributes by the system.
- You can delete, modify, or add attributes which are appropriate to your project. For examples: status, priority, risk, difficulty, cost to implement, and completion date are requirement attributes.

RequisitePro Attributes (cont.)
A view is a window that displays requirements, the attributes assigned to requirements, or the relationships between requirements.

A view displays information in spreadsheet-like tables or in outline trees.

You can create, organize, and track requirements in three kinds of views:

- Attribute Matrix
- Traceability Matrix
- Traceability Tree
After creating a view, you can *query* (filter and sort) its information in various manners.

**Filtering** is a way by which you can specify the amount of information you would like to view. You can specify the criteria on which to “filter” information.

For example: View only those requirements which have the priority set to “High”.

---

Working with Filters and Sorts (cont.)
Importing documents from external sources

- In RequisitePro, you can import documents from Microsoft Word and from CSV files.
- RequisitePro allows importing from
  - a document
  - a database
- RequisitePro provides an Import wizard under File > Import, to perform this functionality.
- For example: If you have already created a Vision document, you can simply import it and begin working on it.

Project Version Control

- RequisitePro maintains version information.
- Version information is unique, identifying information related to a project, document or requirement whenever it is modified.
- Version information constitutes the internally generated revision number, combined with information about the author, date, time, and reason for modification.
Project Version Control (cont.)

- RequisitePro uses revision numbers and revision labels for:
  - each requirement
  - document within a project
  - a project

- RequisitePro uses the following labels to identify versions of requirements, documents, and projects.
  - **Requirement revision label**: It displays the label for the last revision of the requirement.
  - **Document revision label**: It displays the label for the last revision of the document.
  - **Project revision label**: It displays the label for the last revision of the project.

Hierarchical relationships

- Hierarchical relationships can be used to subdivide a general requirement into more explicit requirements.

- A **hierarchical requirement** is a requirement which is included within an ordered requirement of the same type.

- A **hierarchical requirement** consists of
  - **parent requirements**
    - e.g., “The system shall print the Withdraw Money report”
  - **child requirements**
    - E.g., “The report will display the customer ID”

- Setting hierarchical relationships helps organize requirements.
Establishing Traceability

- What is traceability?
  - Traceability is a directional relationship between two requirements
  - With traceability you can define the dependencies between the requirements of the same type or different types.

- As requirements keep changing, they affect other requirements as well.
  - Rational RequisitePro allows you to create traceability relationships to effectively check these changes.

- Traceability feature ensures that the developed product is a quality product and meets all the requirements as specified by Stakeholders.
  - Abstract requirements such as features specified by Stakeholders can be linked to actual features in the software specifications.
  - Use Cases can be traced to features as required by the Stakeholders.

What is traceability?
Traceability is a directional relationship between two requirements.
With traceability you can define the dependencies between the requirements of the same type or different types.

As requirements keep changing, they affect other requirements as well.
Rational RequisitePro allows you to create traceability relationships to effectively check these changes.

Traceability feature ensures that the developed product is a quality product and meets all the requirements as specified by Stakeholders.
Abstract requirements such as features specified by Stakeholders can be linked to actual features in the software specifications.
Use Cases can be traced to features as required by the Stakeholders.
Establishing Traceability (cont.)

RequisitePro allows you to create two kinds of traceability views:

- **Traceability Matrix:** It shows relationships between requirements of the same or different types.
- **Traceability Tree:** It displays all internal and external requirements traced to or from a requirement

Establishing Traceability (cont.)

- There are two types of traceability relationships:
  - Direct relationship
  - Indirect relationship
- If you modify a requirement (change the name, text or attribute) which traces to or from another requirement, the relationship between the two is marked as **Suspect**.
- **Suspect** implies that because of modification to a requirement, the relationship may need modifications as well.
Establishing Traceability (cont.)

Setting up security

- An Administrator can assign requirement type, attribute, and traceability permissions.
- Groups can have the following permissions for the *selected requirement type*:
  - **Read**: Members of this group can only view requirements of this type.
  - **Update**: Members can modify requirements of this type.
  - **Create**: Members can create requirements of this type.
  - **Delete**: Members of this group can delete requirements of this type.
Roadmap: Change Management

In this section:
- Explain the need for change tracking
- Understand the change tracking process
- Introduce the ClearQuest architecture
- Define schemas and databases
- Explain the change request lifecycle
- Explain the ClearQuest process model

The Need for Change Tracking

- Avoid duplication of effort
- Track and report on status
- Prioritize work
- Historical records
- Legal requirements
- Improve communications and teamwork
- Others?...
Types of Change Requests

- Defects – changes to an existing system
- Enhancement requests – A request for new functionality to an existing system
- Feature requests – The ability to track a new feature in a new release
- Documentation – Changes to the project documentation (requirements, design, testing)

Change Request Tracking Process

- Change requests are created in a central repository
- New requests are assigned by a review committee
- Change requests are worked on by engineers and submitted
- Requests are tested and approved and eventually closed
Lifecycle of a Change Request

- Typically, a change request will be a state-based activity
- It will have a lifecycle that will transition it through several states
- The state a change request is in can be queried and tracked
- In ClearQuest, the process of taking a change request from one state to another state is called an action

Schemas and Databases

- A schema defines the structure of a ClearQuest database.
- There may be one schema for your installation or many different ones representing different database structures.
- There may be one or more user databases associated with a schema.
Sample Custom Use of ClearQuest

PM enters change request in ClearQuest
Is request valid?
Yes
No
Track to resolution in ClearQuest
Create and process QPATTS change request
Is subject of change request in full production?
Yes
Submit as ER / Project via QPATTS
No
Service Desk staffer enters defect request in HP OpenView
HP OpenView
Create PAR
Impact? no yes
Budget? yes
Close

Default Process Model

Submitted → Assigned → Opened → Resolved → Closed

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate

Submitted
Assigned
Opened
Resolved
Closed

postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
postpone
assign
reopen
validate
validate
revert
duplicate
duplicate
Sample Change Management Process

1. Identify and Record Change Request
   1.1 Receive and Record Change Request
   1.2 Assess Impact Assessment

2. Perform Impact Analysis
   2.1 Schedule Change Request for Assessment and Review
   2.2 Assess Impact
   2.3 Record Impact Assessment

3. Determine Action
   3.1 Schedule Change Request for Action Review
   3.2 Determine Action Disposition
   3.3 Review Priorities and Create Action Schedule

4. Implement Change
   4.1 Produce Detailed Action Work Plan
   4.2 Carry Out the Action Work Plan
   4.3 Review Implemented Changes
   4.4 Implement Changes in Appropriate Environment

5. Close Out Change Request
   5.1 Update Change Request

Sample Custom Process Model
ClearQuest Change Tracking

- Client access from Windows, UNIX, Web and E-mail
- Ability to create project specific databases
- Process model is customizable for each type of change request
- Customizable GUI
- Custom queries, charts and reports
- Integrates with other Rational Products

Starting the Client

- The main tool for users of ClearQuest is the ClearQuest Explorer
- When starting the Explorer you must choose a schema repository
- You must then log in to the appropriate database for your project
Choosing a Schema

- The first screen asks you to choose the appropriate schema
- If you only have one schema repository, this screen will not appear

Login

- Next you are required to provide a valid login and password for the database that you wish to attach.
Submitting a Change Request

- One of the most common tasks in ClearQuest is to submit a new change request
- There may be many different types of Change Requests (records) to choose
- The examples in this course will use the “defect” record type
Submitting a Defect

- From ClearQuest Explorer, click the down arrow next to “New Defect” and choose the defect record type.

- The main area of Explorer will now contain the new Defect screen.
- Fields in Red are mandatory.
- When complete, select OK.
Running Queries

- The folders pane in ClearQuest Explorer contains all of the public Queries available as well as any private queries you create.
- Double clicking an existing query will run the query.

A common query is the “My To Do List Query.” If finds any records assigned to whoever logged in to ClearQuest.
Drilling Down – cont’d

ClearQuest Charts

- ClearQuest provides a very robust charting and reporting mechanism
- Charts are accessed through the Public Folders section of the Explorer
- Private charts can also be created
Query Wizard Results

Executing a Chart
Chart Types

There are 3 main types of charts in ClearQuest

- Distribution Chart – Shows the distribution of defects based on the criteria specified
- Aging Chart – Shows the age of defects based on the criteria specified
- Trend Chart - Shows the numbers of defects over a period of time based on the criteria specified

Drilling Down

- Distribution Charts have a drill down feature
- Right click on a bar and Select Drill Down
- Drilling Down will give the result set of the bar selected
- This is only available on Distribution charts
Drilling Down – cont’d
Reports

- ClearQuest integrates with several report formatting tools
- The two most common integrations are Crystal Reports and SoDA for Word.
- These tools are used to create report formats that can be tied to ClearQuest queries to produce useful reports
- A report is a Report Format tied to a Query

ClearQuest Web

- ClearQuest provides a Web Interface that does not require the Client software to be loaded on a user’s machine
- Most ClearQuest functions are available via the Web interface
- This can be useful for infrequent users and remote users
Web Limitations

- Labels of mandatory fields remain red, even after you supply values
- There is no **Recently Submitted** query function
- Tab names appear in brackets
- You cannot drill down on charts
- Reports can be generated with existing report formats only

Roadmap: Configuration Management

In this section:
- Define Configuration Management
- Explore the reasons behind using CM
- Explain how ClearCase implements CM
- Learn the basic terminology used by ClearCase
- Explain the ClearCase View mechanism
- Distinguish between dynamic and static views
- Discuss typical use of Configuration Management
The Need for CM

- Software is easy to change……too easy!
- Configuration Management provides:
  - Structure for identifying and controlling change
  - Supporting a chosen methodology (e.g., legacy company Product Lifecycle methodology)
  - Determining project/product status

Scope

- ClearCase can be used to store and version:
  - Models
  - Diagrams
  - Test plans, scripts, and results
  - Source code
  - Components, packages, and executables
  - Any files or directories that need versioning…possible snapshots of Functional Specifications, Technical Conceptual Designs and or Technical Detail Designs
CM Roles

- The following roles have been identified as part of the Configuration Management process.
  - Configuration Manager
  - Configuration Administrator
  - Migration Manager
  - Systems Administrator
  - Developers
  - Testers
  - Technical Leads

Sample Custom CM Process Flow
ClearCase and CM

- Version Control
- Workspace Management
- Build Management
- Process Control

Versioned Object Base

- The ClearCase repository that stores files and directories
- Presents its contents as a file system viewable through the ClearCase Explorer or Windows Explorer
- Each change or set of changes to something in a VOB creates a new version
More About VOBs

- A ClearCase installation may have one or more VOBs [Versioned Object Base]
- VOBs may be shared across projects
- VOBs may be dedicated to specific projects
- Usually set up by the ClearCase Administrator or Configuration Manager

Elements

- An Element is a file or directory that is under source control (i.e. it resides in a VOB)
- This can be:
  - Source files and directories
  - Documentation
  - Executables
  - Tools
  - Anything…
Element Versions

- Each Element in a VOB will have a set of versions
- Versions represent revisions on the element

Views

- A view is the mechanism that ClearCase uses to “look into” the VOB
- All work must be done through a view
- This can be an isolated workspace for a user or a shared workspace for a group
- Views allow parallel development
- Views select the versions that the developer sees
Configuration Specifications

- A Configuration Specification is the set of rules that determines which version of an element is presented
- Every view has a configuration specification
- There is a default configuration specification that can be modified to suit the needs of the view

Types of Views

- Dynamic View
  - Uses a view storage directory on a network share
  - Utilizes Multi-Version File System (MVFS) which is a virtual look into a VOB or VOBs
  - See changes immediately
- Snapshot View
  - Can work disconnected
  - Element versions copied to local client machine
  - View must be updated periodically to see changes
Dynamic View

Snapshot View Basics

- A dynamic view is a look into a VOB at a specific version of an Element
- A snapshot view is a copy of that Element to a disk location
- Snapshot views provide a local copy of VOB information
- Check-out/check-in can still be done
- Updates are not seen immediately
Check-out/Check-in

- Elements in a VOB cannot be modified directly
- Elements must be checked-out in order to make modifications
- When a file is checked-out, a read/write copy is placed in the view storage directory
- A new version is reserved for the changes that are made
- The element must be checked back in and a new version is added to the version tree
Versioning Directories

- In ClearCase, directories are also under source control
- Any file additions, deletions or name changes will result in a new directory version reflecting the new namespace
- This happens automatically in CC Explorer

Branching

- Branching allows parallel development
- Two (or more) branches can be spawned from one version of an element
- Work can continue separately until it needs to be combined (e.g. Feature Integration)
- When work is combined, it is called merging.
Version Compares

- It is possible to compare a version of an element to another version graphically
- This is done via the Version Tree Browser

Overview of Merging

- If all work is done in a serial fashion, merging will not be necessary
- However, most projects will require additional branches to facilitate parallel development
- At some point, these separate changes will need to come together
- This is the work of merging!
Developers and ClearCase

- The process flow for a developer making changes to files stored in ClearCase is:
  - Create a view appropriate for the project, feature or release being worked on.
  - Check-out files, make modifications and check-in files
  - Build and test the work in isolated developer workspace
  - When directed by integration manager, merge the changes into the integration branch for that project
  - Changes will then be built and tested with changes from other developers
  - Baseline versions can then be set from the integration branch for testing or release

Sample Custom Change Management

- Company XYZ is currently using Visual Source Safe to store versioned code and files
- Late this year, the company will begin to transition away from VSS and start to store versioned code and objects in Clear Case
- Clear Case can be set up in the future to allow offshore resources to access the versioned objects so code can be added, changed and maintained
Roadmap: Design and Testing

In this section:

- Use Case Based Modeling and UML
- What is XDE Developer for Java
- What is Test Manager
- What is XDE Tester

Use Case Based Modeling

- Links stakeholder needs to software requirements.
- Defines clear boundaries of a system.
- Captures and communicates the desired behavior of the system.
- Identifies who or what interacts with the system.
- Validates/verifies requirements.
- Is a planning instrument.
Use Cases Relate to Requirements

- Each use case
  - Describes actions the system takes to deliver something of value to an actor.
  - Shows the system functionality an actor uses.
  - Models a dialog between the system and actors.
  - Is a complete and meaningful flow of events from the perspective of a particular actor.

Unified Modeling Language

- The **UML** is the standard language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system.
- The UML combines the best from
  - Data modeling
  - Business modeling
  - Object modeling
  - Component modeling
XDE Developer for Java

- Integrated with Requisite Pro
  - Use-case models in XDE are traced (linked) to their respective use case requirements in Requisite Pro
  - Allows for integration of use cases with all other requirements information (documents, attributes, traceability relationships, etc.)
  - Allows for the assignment of priorities, difficulties, dependencies and other project specific attributes

XDE/ClearCase Integration

- XDE and ClearCase can be integrated to provide:
  - A robust modeling environment in XDE with the version control and workspace management benefits of ClearCase. This ensures:
    - Security: Artifacts are secure and cannot be inadvertently overwritten
    - Versioning: As changes are made, ClearCase provides the tracking support
    - Organization: Artifacts can be organized and retrieved based on project standards
Test Management

- Test management involves storing test artifacts for future use, building traceability into test cases, and ultimately reducing the time it takes for test reviews
- E.g., Managing tests using **IBM Rational Test Manager**
  - Set up your requirements in RequisitePro
  - Set up the test inputs for the project
  - Create a test plan
  - Create folders to organize your test cases
  - Create your test cases
  - Establish traceability from your test inputs to your test cases
  - Set up your test cases so they actually test something

The Challenge of Manual Testing
Hours for Manual vs. Automated Testing

<table>
<thead>
<tr>
<th>Test Steps</th>
<th>Manual Testing</th>
<th>Automated Testing</th>
<th>Percent Improvement with Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Plan Development</td>
<td>32</td>
<td>40</td>
<td>25%</td>
</tr>
<tr>
<td>Test Case Development</td>
<td>262</td>
<td>117</td>
<td>55%</td>
</tr>
<tr>
<td>Test Execution</td>
<td>466</td>
<td>23</td>
<td>95%</td>
</tr>
<tr>
<td>Test Result Analysis</td>
<td>117</td>
<td>58</td>
<td>50%</td>
</tr>
<tr>
<td>Error Status/Correction Monitoring</td>
<td>117</td>
<td>23</td>
<td>80%</td>
</tr>
<tr>
<td>Report Creation</td>
<td>96</td>
<td>16</td>
<td>83%</td>
</tr>
<tr>
<td>Total Duration (Hours)</td>
<td>1090</td>
<td>277</td>
<td>75%</td>
</tr>
</tbody>
</table>

Source: QA Quest Newsletter from the Quality Assurance Institute (November 95)

Automated Testing Roadmap

- Use the right (combination of) tools
  - e.g., Test Manager and XDE Tester / Robot
- Have at least one real programmer in your testing-automation group
- Develop standards for your team.
- Figure out and document what you're testing, and keep it simple
- Use record-and-playback feature to baseline your scripts
Automated Testing Roadmap

- Modularize and build reusability into your scripts
  - Write wrappers around most functions and put them in libraries. Call wrapped functions whenever possible
- Document everything you're doing to the greatest detail as time allows
- Use a data-driven testing technique
  - When you use Rational tools this means creating datapools using Test Manager and your Robot/XDE Tester scripts will then contain links to these datapools.

Standards for Automated Testing Teams

- **Naming standards** for scripts, test logs, directory structures, datapools, and verification points help to keep everyone on the same page
- **Coding standards** should also be developed and enforced
- **Environment standards** should ensure that the computers you use all have the same operating system, RAM, hard drive space, and installed software configurations
- **Procedures for error and defect tracking** should describe how to log errors in scripts, submit defects (e.g., via ClearQuest), code workarounds into scripts, and remove it all after a bug is resolved
IBM Rational XDE Tester

- Automated testing tool for functional and regression testing of Java and Web applications
- Now referred to as “Rational Functional Tester for Java and Web”
- Enables resilient, reusable test scripting in Java with ScriptAssure
- Validates the interactive data generated by the application, minimizing script maintenance.
- Maps all test objects to a central repository, and automatically transmits changes to the objects to all the appropriate test scripts
- Tests applications built with J2EE, J2SE, HTML, DHTML, XML, JavaScript, and Java Applets
- Enables tests from a perspective of IBM WebSphere Studio development environments and tests the entire SWT library.

Object-Oriented Testing Technology

- XDE Tester uses object-oriented testing technology.
- XDE Tester lets you navigate through your application using
  - Record
  - Play back scripts
Using XDE Tester

- Rational XDE Tester is an object-oriented automated testing tool that lets you test Java and HTML applications running on the Windows NT, Windows 2000, Windows XP, and Linux platforms

- XDE Tester can be used to
  - Perform full functional testing by recording and playing back scripts that navigate through your application and test the state of objects through verification points
  - Create and edit simple and easy-to-read, object-oriented test scripts

XDE Tester and WSAD
ScriptAssure™

- The ScriptAssure™ feature is XDE Tester’s object recognition technology
- Enables you to successfully play back scripts even when the application-under-test has been updated
- Gives you *platform-independent and browser-independent* test playback.

XDE Tester & Test Manager

- XDE Tester is integrated with Rational TestManager
- TestManager is included in
  - Rational Suites Enterprise Studio
  - Rational Suites TestStudio
  - Rational TeamTest
XDE Tester & ClearCase

- XDE Tester is integrated with ClearCase and ClearCase LT
- Using ClearCase, you can compare the current version of a script to a previous version
- Using ClearCase, you can display the history of an element under source control
- Integration requires one of the ClearCase products for this integration to work in Rational Suites and TeamTest

Roadmap: Reporting and Statistics

In this section:
- What is Software Documentation
- What is Rational SoDA
- What is Rational Project Console
Software Project Documentation

- **What is software documentation?**
  - Software documentation is written text that accompanies the company’s software.
  - A software document explains how a software operates or how to use it.

- **Why use Software Project Documentation tools?**
  - Most software development organizations spend a substantial amount of time on documentation.
  - 20 to 30 percent (or more) of all software development effort is focused on documentation.
  - For this reason, documentation tools help improve productivity.

Rational SoDA product capabilities

- SoDA is an acronym for *Software Documentation Automation*.
- SoDA is a report generation tool that supports documentation requirements with an easy-to-use interface for defining custom reports and documents.
- SoDA automates the production of software documentation.
- Rational SoDA retrieves information from various sources and uses it to generate a document or report according to a template.
- SoDA comes with several predefined templates, or you can build custom templates for your specific needs.
Rational SoDA product capabilities (cont.)

- Rational SoDA is integrated with Word.
- SoDA for Word:
  - Document generation is added to the capabilities of Microsoft Word.
  - Performs incremental document regeneration.
  - Extracts data from multiple information sources, such as Rational Rose and Rational RequisitePro and creates a single document.
  - Maintains consistency between documents and information sources.

Rational SoDA product capabilities (cont.)

- SoDA generates complete documentation by automatically extracting data from various project tool databases.
- Rational SoDA is tightly integrated with Rational’s market leading development tools.
- SoDA gives users a single interface for reporting on requirements, design, test and defect status.
- SoDA automatically captures change, the user always has updated information.
- SoDA provides flexible report writer which enables all team members to create custom reports.
SoDA Key concepts & Terminology (cont.)

- Document Generation
  - **Intelligent Document Merging**: This is a process by which SoDA maintains consistency between the document and its source.
  - SoDA lets you add descriptive text, formatting and drawings.
  - SoDA regenerates a portion of a document.

- Report Generation
  - Reports can also be generated instead of a document.
  - Generating reports is a faster option.

- Template Customization
  - SoDA comes with many predefined templates
  - You can also build new templates from scratch.

What is ProjectConsole?

- IBM Rational ProjectConsole is an easy-to-use *Project Management and Metrics tool*.

ProjectConsole:

- Provides portal, web reporting and metric capabilities
- Quantifies the current project status
- Helps in assessing development trends of a project with up-to-date metrics
- Metrics results are presented visually in graphs, charts, and gauges.
What is ProjectConsole? (cont.)

ProjectConsole **charts** and **indicators** allow a project team to analyze:

- Low-level details
- Planned-versus-actual metrics
- Historical data
- Trend charts

The above analysis helps to:

- Set realistic project expectations
- More realistically assess potential risks
- Identify bottlenecks
- Realize the cause for late deliverables
- Take prompt corrective action
- Forecast future

Example of ProjectConsole Metrics Display
Metrics and Display Capabilities

- **ProjectConsole Dashboard** is used to display metrics
- Dashboard
  - Is used for project data analysis
  - Can be used by anyone involved with analyzing metrics
  - Gives access to meaningful and current measures for monitoring project progress
  - Displays measurements graphically in charts and other indicators
  - Can be used to track trends in defects, use cases, requirements, dependencies and more

Metrics and Display Capabilities (cont.)

- Data can be displayed in various display formats
  - Traffic lights
  - Gauges
  - Charts - line, bar, scatter plot, stacked bar chart formats, pie charts and area charts.
Metrics and Display Capabilities (cont.)

Dashboard showing overall status of all projects

Dashboard showing overall status of one project

Traffic Lights

Gauge

Pie Chart
ProjectConsole Summary

ProjectConsole

- With the information presented by ProjectConsole teams can set realistic project expectations, assess potential risks, identify bottlenecks, and realize the cause for late deliverables
  - Allows a team to automatically quantify current project status
  - Helps in assessing the development trends of project with up-to-date metrics
  - Collects metrics data on specified schedule or on demand from Rational tools and third party tools
- Results are presented visually in graphs, charts and gauges
- With charts and indicators teams can analyze low-level details, planned versus actual metrics, historical data and much more

Next Steps

- TUP [Team unifying platform] comes with RUP [Rational Unified Process]
- RUP is a software development methodology that is often customized by companies in various industries
- Companies typically follow a subset of the RUP principles
- Utilize the RUP link to gather facts about Object Oriented analysis and design
RUP [Rational Unified Process]

RUP Best Practices Implementation

- Practical Best Practices Process
  - Develop Iteratively
    - Guidance for activities and work products (artifacts)
  - Manage Requirements
  - Use Component Architectures
    - Process focus on architecture
  - Model Visually (UML)
    - Use cases which drive design and implementation
    - Models which abstract the system
  - Continuously Verify Quality
  - Manage Change
The Iterative Approach

In an iteration, you walk through all workflows

Questions?

“Everything should be made as simple as possible, but not simpler.”
- Albert Einstein