Collaborative software development using IBM Rational Team Concert

Presentations
Collaborative software development using IBM Rational Team Concert

An IBM Proof of Technology

Welcome to the Technical Exploration Center

- Introductions
- Access restrictions
- Restrooms
- Emergency Exits
- Smoking Policy
- Breakfast/Lunch/Snacks – location and times
- Special meal requirements?
Introductions

- Please introduce yourself
- Name and organization
- Current integration technologies/tools in use

What do you want out of this Exploration session?

Agenda

- Introduction to Rational® Team Concert
- Lab Overview
- Module 1 Setting up the Team
- Module 2 Planning Your Work
- Module 3 Keeping Track of All Our Work
- Module 4 Performing and Sharing Your Work
- Module 5 Remembering Well Known SCM Configurations
- Module 6 User’s View of Build
- Module 7 Exploring Changes and Traceability
- Module 8 Endgame and a Tightened Process
- Module 9 Taking Control of Your Project
- Session Summary
Objectives

- Explore how IBM© Rational© Team Concert™ software can:
  - Enable development teams to collaborate in real time in the context of the work they are doing, especially in globally diverse environments
  - Enable projects to be managed more effectively by providing visibility into accurate project health information drawn directly from actual work
  - Automate traceability and auditability by managing artifacts and their inter-relationships across the lifecycle empowering teams to deliver more value
  - Provide customizable process design and enactment through rule-based process guidance, automation and definable checkpoints
- Provide a hands-on experience using Rational Team Concert to automate the software delivery process

Introduction to Rational Team Concert

An IBM Proof of Technology
What if your development tools knew…

… about your teams
… about your artifacts
… who is responsible for what
… about your process
  Code delivery rules, code quality, traceability, test runs, intellectual property
… how to bootstrap a project
… how to help new team members get started
… your favorite work item types and their state transitions
… when the build runs and what to do if it breaks

Collaborative Application Lifecycle Management

The challenge in enabling it

- Traditionally, each tool came with its own
  - User interface (UI) - Web and desktop presentations of views and tasks
  - Logic – Workflow, process, search, query, scale, security and collaboration
  - Storage – Availability, traceability
  - Privacy, backup/archive
- Resulting in...
  - Brittle integrations
  - Silos everywhere
  - High cost to maintain and administer
  - Proprietary APIs
IBM Rational Team Concert: An open, extensible architecture
Supporting a broad range of desktop clients, IDEs and languages

- **Eclipse Clients**
  - IBM Jazz™ Client
  - Eclipse Platform

- **Web Clients**
  - IBM Jazz™ Client
  - Web 2.0

- **Microsoft .NET Clients**
  - Visual Studio

- **Rational Desktop Clients**
  - Rational Software Architect
  - Rational Business Developer

- **Business Partner Extensions**

- **JAZZ SERVICES**
  - Presentation: Mashups
  - Discovery
  - Query
  - Storage
  - Administration: Users, projects, process

- **IBM Rational Extensions**

- **Your Extensions**

IBM Rational Team Concert
Software innovation through collaboration

- **Collaborate in context**
  - Integrated release planning and reporting, source control, document collaboration, work item, build management, chat and process guidance

- **Streamline agile development**
  - Out-of-the-box agile process configurations

- **Automate governance**
  - Assess project status and trends in real-time with web-based dashboards, metrics and reporting

- **Scale to the enterprise**
  - Supports teams ranging from a few to thousands of developers and stakeholders

- **Unify diverse teams**
  - Supports J2EE™, .NET, IBM i, IBM System z®, co-existence with popular toolsets

Open and extensible on Jazz
- Collaborate
- Automate
- Report
Pain points before Rational Team Concert

- joining a team
- get my environment configured to be productive
- what is happening in my team
- collecting progress status
- following the team’s process
- ad hoc collaboration/sharing of changes
- starting an ad hoc team
- is the fix in the build?
- what will be in the next build?
- tracking a broken build
- Avoid breaking a build/personal build
- why is this change in the build?
- reconstructing a context for a bug/build failure
- creating, tracking iteration plans
- interrupting development due to a high priority bug fix
- working on multiple releases concurrently
- tracking the code review of a fix
- referencing team artifacts in discussions
- how healthy is a component?
- collecting project data/metrics?

Team awareness

Build awareness

Project awareness

Boring and painful

Team Concert: An Overview

Planning
- Integrated release/iteration planning
- Effort estimation & progress tracking taskboards
- Out of the box process templates: formal or agile

Project Transparency
- Customizable web based dashboards
- Real time metrics and reports
- Project milestone tracking and status

SCM
- Component based SCM enables reuse across projects
- Change set based for easy addition or removal of features
- Server-based sandboxes
- Can also work with SVN, Git, ClearCase or Synergy

Work Items
- Defects, enhancements and conversations
- View and share query results
- Support for approvals and discussions
- Query editor interface
- ClearQuest or Synergy Bridge

Build
- Automated work item and change set traceability
- Build definitions for team and personal builds
- Local or remote build servers
- Multi-level continuous integration
- Integration with Build Forge

Jazz Team Server
- Single structure for project related artifacts
- World-class team on-boarding / offboarding including team membership, sub-teams and project inheritance
- Role-based operational control for flexible definition of process and capabilities
- Team advisor for defining / refining “rules” and enabling continuous improvement
- Process enactment and enforcement
- In-context collaboration enables team members to communicate in context of their work
Effectively using Rational Team Concert independently

Rational Team Concert Enterprise, Standard, Express, Express-C

- A stand-alone development environment optimized for small and mid-sized teams
- All the collaborative capabilities of the Jazz platform – plus integrated work items, SCM and build management
- Dashboards and real-time reports
- Team and Process-aware

Incremental Adoption by Subversion, ClearCase, ClearQuest and Git teams

- Manage planning and project status with work items and dashboards in Team Concert and develop with existing artifacts that reside in subversion, Git or IBM Rational ClearCase®
- Enables teams to reuse assets, process and investment in Subversion, ClearCase, IBM Rational ClearQuest®, Synergy or Git
- Third party connector to Jira
- Take advantage of new collaborative Application Lifecycle Management (ALM) in an evolutionary way with lower business risk
IBM’s Internal Deployment of Team Concert

- In Rational, we have over 3800 developers and testers as of October 2010
  - 265 development projects, seven application servers (hosting 25 RTC, 2 RQM and 1 RRC server instances), four DB2 servers
- Around 18000 unique developers by the same team managing jazz.net deployment
  - And this is NOT being mandated, but rather through viral adoption and real productivity gains

Rational Internal Deployment, June 2010
How we use Rational Team Concert

- Two-way Intel® Xeon® server running WebSphere® Application Server and another running IBM DB2®
- Jazz Project – Using bi-weekly iteration builds
  - Jazz and Rational Team Concert self hosting since 4Q06
  - Global team in seven locations in North America, Europe and India
  - ~150 developers, plus jazz.net webservice access
  - Repository >24G, >170K files, >60K work items
- Over 60 Rational development teams using Rational Team Concert
- More than 125 other teams around IBM

Customer feedback

"By helping us to make project deliveries more repeatable and predictable, we anticipate that Rational Team Concert will reduce project overrun costs by 20%."  
--Matt Pomroy - Executive, Software Engineering, Ascendant Technology

"Its automated project management dashboards are transparent to everyone – not just managers. This immediate and automated feedback helps keep teams on track and motivated to achieve project goals."  
--Han Jie - Senior Consultant, Siemens

"Where we previously used separate systems, with Rational Team Concert we now have well integrated functionality. Our developers are more efficient because they are better able to focus on important issues. Our project managers greatly value the ability to customize these dashboards and instantly provide status on their milestones!"  
--Mika Koivuluoma - Production Manager, TietoEnator

"Having a unified and extensible environment is very compelling for us. Rational Team Concert provides the team transparency and visibility needed to keep work progressing so everyone knows what’s going on without finger-pointing."  
--Carson Holmes - Unified ALM Services Manager, Noblestar

"With IBM Rational Team Concert we’ve seen a 30% productivity gain on our global projects. The ability to easily suspend and resume work, along with advanced SCM features, helps team members juggle multiple tasks and priorities without missing a beat."  
--Alain Bergeron - VP Consulting Services, CGI
Ecosystem of Rational Team Concert
Ready for Rational Business Partners Solutions

**Tasktop**
Integrate with Mylyn task focused interface

**BSD GROUP**
Synchronize defect and change workflows with HP Quality Center

**ARSIN Corporation**
Tools for validation of SAP business objects

**Prolifics**
Automated code review integrated with Rational Team Concert workflow

**Emlogic**
Enable bidirectional JIRA change management integration

**SOA software**
Deliver automated governance to insure compliance and minimize business risk

**blackduck**
Find and manage open source approvals

**dbMAESTRO**
Support for Database asset versioning from RTC

**GreenHAT**
Extends RTC collaboration to complex test environments

**aviarc**
supports RTC users through its comprehensive web development environment

**QSM**
Analyze project data and automate task-level work breakdown structures

**Maipsoft**
Collaborate with Microsoft® SharePoint® and Lotus® Quickr® libraries

**ITKO®LISA**
Creates RTC work items from LISA tests

Some of those integrated solutions might only be available with RTC 2.x or RTC 3.x

Questions
Rational Team Concert Client for Microsoft Visual Studio IDE

Collaboration, automation and reporting for heterogeneous development teams

An IBM Proof of Technology

Agenda

- Rational Team Concert client for Microsoft® Visual Studio® Integrated Design Environment (IDE)
  - Features and Benefits
- Summary: Supports all your Windows® development needs
  - Microsoft Windows 7, Vista and XP
  - Microsoft Visual Studio 2010, 2008 and 2005 IDE
  - Microsoft SQL Server® 2008 and 2005
Extend team collaboration to Visual Studio developers
Rational Team Concert client for Microsoft Visual Studio IDE

• **Unify Software teams**
  - Manage Change across development environments
  - Single repository for both development platforms (.NET and J2EE)
  - Cross platform team collaboration
    - Common Work items
    - Source Code Management
    - Builds
    - Plans
    - Reports
    - Feeds
IBM Software Group

Supports Visual Studio 2010, 2008 and 2005 Professional and Standard Editions

SCM, Work items & Build directly from VS.NET IDE

- **SCM**
  - Component based SCM enables reuse across projects
  - Change set based for easy addition or removal of features
  - Server-based sandboxes
  - Can also work with SVN, Git, ClearCase

- **Work Items**
  - Defects, enhancements and conversations
  - View and share query results
  - Support for approvals and discussions
  - Query editor interface
  - ClearQuest or Synergy connector

- **Build**
  - Work item and change set traceability
  - Request and monitor team and private builds
  - Local or remote build servers
  - Supports Ant and command line tools
  - Integration with Build Forge

- **Jazz Team Server**
  - Single structure for project related artifacts
  - World-class team on-boarding / off-boarding including team membership, sub-teams and project inheritance
  - Role-based operational control for flexible definition of process and capabilities
  - Team advisor for defining / refining “rules” and enabling continuous improvement
  - Process enactment and enforcement
  - In-context collaboration enables team members to communicate in context of their work

- **Iteration Planning**
  - Integrated iteration planning and execution
  - Task estimation linked to key milestones
  - Out of the box agile process templates

- **Feeds**
  - Users can subscribe to and receive Feeds
  - Feeds on work items, builds, queries, team events
  - Feed filters

- **Project Transparency**
  - Customizable web based dashboards
  - Real time metrics and reports
  - Project milestone tracking and status

Open and extensible on Jazz
Collaborate in context
Right-size governance
Day one productivity
A Rich native client in Visual Studio

Develop in Visual Studio

- Connect to repository
- Deliver changes
- Accept invite to join project
- View Pending Changes
- View Work Items from history
- Change Status of Work Items
- View Repository Files
- Search Repository
- Create/View Reports
- Author iteration plans
- Request a build
- Track MS-Build Progress/Result
- View Dashboards
- Subscribe to / Receive Feeds
- Change Delivery Flow
- Merge Changes
- View History
- Create Work items
- Associate Work items
- Run Queries
- Create Workspaces / Streams

Web/Eclipse UI

- Create Project Area*
- Customize Process*
- Create Team Area*
- Customize Work items*

Inside Visual Studio .NET Shell

- Join Development
- Or
- Accept invite to join project
- Connect to repository
- Or
- Accept invite to join project

Do Visual Studio builds integrate with Team Concert?

- Yes. They do.
  - MSbuild support
  - Request builds, view build results and compare builds from Visual Studio
  - Use the Eclipse client to control the build definition and any of the team concert clients to control and view the build results.
### Details of Team Concert views surfaced in Visual Studio

<table>
<thead>
<tr>
<th>View (Tool Window)</th>
<th>Feature</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solution Explorer</td>
<td>Share new or existing Visual Studio solutions with the Jazz repository. Control which file types in a solution are versioned. Glyphs/icons display files version control state. Rename and move files while retaining history. Display history of selected files.</td>
<td>Jazz is selected as the SCM provider for the Visual Studio solution. All updates to the Solution Explorer are then managed by Jazz and Rational Team Concert.</td>
</tr>
<tr>
<td>Work Items View</td>
<td>Result set from a work item query is displayed here. Can sort the view. Quick edit UI to update most work item fields. Bulk Edit several work items. Create work items and Queries.</td>
<td>Similar to Eclipse Work Items View.</td>
</tr>
<tr>
<td>Team Advisor View</td>
<td>Shows advice when certain operations fail. Includes an explanation and links that help resolve the problem. Non-modal, so you can navigate to other views to track down or correct the problem.</td>
<td>Similar to Eclipse Team Advisor View.</td>
</tr>
</tbody>
</table>
### Details of Team Concert views surfaced in Visual Studio

<table>
<thead>
<tr>
<th>View (Tool Window)</th>
<th>Feature</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pending Changes View</strong></td>
<td>Shows status of all changes made by the developer and provides access to advanced SCM features.</td>
<td>Similar to Eclipse Pending Changes View</td>
</tr>
<tr>
<td></td>
<td>Deliver change sets</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Attach and resolve work items</td>
<td></td>
</tr>
<tr>
<td></td>
<td>View and modify change flow of work to streams</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Create baseline, rollback to another baseline</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Create snapshots</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Merge, rollback a change set</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Compare / merge with previous versions via the default editor or via external Compare Merge tools</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Filter displayed change sets based on regular expressions</td>
<td></td>
</tr>
<tr>
<td><strong>Change Set Explorer</strong></td>
<td>Explore a change set and its contents</td>
<td>Same as Rational Team Concert’s Eclipse UI capabilities.</td>
</tr>
<tr>
<td></td>
<td>View before and after state of file in the change set</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Associate work items</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Remove work items</td>
<td></td>
</tr>
<tr>
<td><strong>Change Set Search View</strong></td>
<td>Developer can search for change sets using search criteria</td>
<td>Same as Rational Team Concert’s Eclipse UI capabilities.</td>
</tr>
<tr>
<td></td>
<td>Useful for looking at past deliveries of work</td>
<td></td>
</tr>
<tr>
<td><strong>History View</strong></td>
<td>Shows history of files and components.</td>
<td>Similar to Rational Team Concert Eclipse UI</td>
</tr>
<tr>
<td></td>
<td>Compare with previous version, local version or arbitrary version</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Can then open in Change Set Explorer view</td>
<td></td>
</tr>
<tr>
<td><strong>Repository File Browser</strong></td>
<td>Browse repository workspaces, components, files and folders without explicitly downloading them to the users local hard disk. Browse and view files selectively.</td>
<td>Same as Rational Team Concert’s Eclipse UI capabilities.</td>
</tr>
<tr>
<td><strong>Builds View</strong></td>
<td>Monitor builds</td>
<td>Similar to Rational Team Concert Eclipse UI</td>
</tr>
<tr>
<td></td>
<td>View build results</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cancel builds</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Associate Work Items to build</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Create a stream/repository workspace from build</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Compare builds</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Tag a build</td>
<td></td>
</tr>
<tr>
<td><strong>Feeds</strong></td>
<td>Browse feeds</td>
<td>The edit work items feature is there only in the Visual Studio Client</td>
</tr>
<tr>
<td></td>
<td>Edit work item comments in-line via feeds</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Edit Feeds filters</td>
<td></td>
</tr>
<tr>
<td><strong>Search Results View</strong></td>
<td>Displays your repository search results</td>
<td>Similar to Rational Team Concert Eclipse UI</td>
</tr>
<tr>
<td></td>
<td>Context specific menu item lets you work with your search results</td>
<td></td>
</tr>
</tbody>
</table>
What do I need to get started?
Rational Team Concert client for Visual Studio

- **Prerequisites**
  - Rational Team Concert v3.0
  - Visual Studio 2010, 2008 or 2005
  - Register at [www.jazz.net](http://www.jazz.net)
  - Download from [www.jazz.net](http://www.jazz.net)

- **Learn More**
  - Watch the video!
  - Heterogeneous development with Rational Team Concert

- **Technical Articles**
  - [Source Controlling Visual Studio Projects and Solutions in Team Concert](#)
  - Continuous Integration with Rational Team Concert and Microsoft Visual Studio
  - Mapping your Visual Studio Projects and Solutions to Jazz Components

- **Provide Community Feedback**
  - [jazz user forum](#)
  - Submit bugs or enhancement requests

IBM Rational Team Concert
transparent  integrated presence
wikis  OPEN real-time reporting
chat  automated hand-offs  Web 2.0
custom dashboards  automated data gathering
EXTENSIBILITY  Eclipse plug-ins  SERVICES
architecture  FREEDOM TO CREATE
JAZZ TEAM SERVER
Lab Overview

An IBM Proof of Technology

Scenario for PoT Labs

- You are joining a new project called Squawk that has recently been started in your company.
- You will be using Rational Team Concert as the project's collaborative development environment.
- You have joined the project at the start of Milestone 3. You and all your team mates will be contributing new content to the application.
Scenario for PoT Labs

- Squawk is a simple program that will print out different sounds depending on who “squawks”. For example, the Dog squawker goes “bark” and a Cat squawker goes “meow”. Your main task is to create a new squawker, along with tests and documentation.

- At the same time as creating new squawkers, you will get to participate in some planning activities, interact with your fellow team members, deliver your work to the project, trigger automated builds and various tasks typical for project teams everywhere.

- The project team structure mimics the four major components:
  - Core Library
  - Documentation
  - User Interface
  - Release Engineering (build)

- You are assigned to the Core Library and Documentation teams with a team leader (one of the instructors). Welcome to the team!

---

Eclipse Overview

- Menu
- Button bar
- A View
- All screen elements below the button bar are called Views
- Add new Views using the Window->Show View menu

- Change Perspective
- Current Perspective
- This View has different areas accessed via the View tabs
- View tabs
IBM Software Group

Rational Team Concert Terminology

- Jazz artifacts are stored in a repository.
- The repository contains project areas, which are the system's representation of a software projects.
- Each project area has an associated process, which governs how the project is run.
- Project Areas are decomposed into a set of team areas, which describe the teams that work on the project.
- Teams use a stream to store the master copy of project’s files.
- Team Members use a personal repository workspace to work on project files.
Sequence of events – Lab 1 and 2

Lab 1
- Accept Team Invite
- Configure Instant Messaging*
- Chat with the Team*

Lab 2
- Create New Plans
- Distribute Iteration Plan
- Create Work Items
- Use Work Item Tags**
- Examine Iteration Plan

* RTC for Visual Studio supports drag and drop to external instant messaging tools only.
**Tag Cloud view is only available in the RTC Eclipse client.

Sequence of events – Lab 3 and 4

Lab 3
- Create Query
- Explore Web UI
- Project Status with Team Central View
- Track your Work with My Work View*

Lab 4
- Create Workspaces
- Create and Deliver Work

* Not currently available in Visual Studio
Sequence of events – Lab 4 and 5

- Team Lead (instructor)
- Team Member (student)

1. Create Component Baselines
2. Create and Promote Workspace Snapshots
3. Accept New Baselines
4. Explore Snapshots
5. Accept all Changes

Sequence of events – Lab 6 and 7

- Team Lead (instructor)
- Team Member (student)

1. Request Integration Build
2. Request Personal Build
3. Explore Changes for Build
4. Explore Changes for a Work Item
5. Explore Changes for a File
6. Explore Build
Sequence of events – Lab 8 and 9

- Experience Process Enactment
- Create a Personal Dashboard
- Explore Reports Using Web UI
- Change to the Endgame Iteration

Lab 8

Lab 9

Lab Conventions

- The hostname used to connect to the Jazz Team Server is `jazz-server`
- Each student is assigned a unique user id of the form `student<N>` based on their student number, for example, `student1`
  - Examples in the lab workbooks use student1, you will need to adjust per your assigned id
- Every student creates his or her own unique Squawker.
  - Examples in the lab workbooks use `Lion`
- Students can choose any squawker they want but should include your student id in the name
  - `<squawker>_<student id>`, e.g. `Lion_student1`
- Work items created should include the full squawker name in the summary title
  - `<squawker name> Implementation` and `<squawker name> Documentation` e.g. `Lion_student1 Implementation`
- Optionally, adjust the language settings in the VM for international keyboards. Go to Control Panel -> Regional and Language Options. Select the Languages tab and then click Details in the Text Services and input languages section. Add your local keyboard and make it the default input language.
Jazz.net Registration

- Not a Member yet?
  - If you have web access to your email server
    - You will receive a confirmation and password resetting instructions
  - Go to [www.jazz.net](http://www.jazz.net) and register now.

- Creating a Jazz.net account allows you to:
  - take part in the Jazz community.
  - download product trials, betas, and other previews of Jazz technology.
  - have access to articles, tech notes, tutorials
  - interact directly with the development teams and other members of the Jazz community to ask questions, report bugs, provide feedback and help guide the evolution of Jazz technology.

Questions
Setting up the Team

An IBM Proof of Technology

Objectives

- In this lab, you will learn how ramp up projects quickly and dramatically improve on-boarding and off-boarding of team members.
- You will perform some initial setup of IBM Rational Team Concert™ to enable your machine to communicate with the server.
- You will enable instant messaging in Rational Team Concert.
Joining a project

- For most environments, joining a project can be complicated.
- Team Concert makes this as easy as possible.
- Adding a new team member to a project generates a Team Invitation email.
- Contents of the email can be used to set up the new team member’s access to the project resources in Team Concert.

Communication

- Users of Team Concert can use a variety of tools to communicate with team members.
  - Email
  - Instant Messaging/Chat *
  - Really simple syndication (RSS) feeds
  - Web UI
  - Team Concert client
- Team members can use all the typical communication mechanisms to keep working together as a team, regardless of where they are physically located. This collaboration allows for a single view of project data
  - Integrated instant messaging/chat for immediate feedback
  - RSS feeds to notify you of significant events on the project in real time
  - The web UI used for anyone on the team, or who has an interest in the project

* RTC for Visual Studio supports “drag and drop” to external instant messaging tools only.
Lab #1 Scenario

- You arrive at work on Day 1 and receive an email inviting you to the Squawk project.
- You start Team Concert and get connected to the project right away.
- You use instant messaging to chat with your colleagues on the project.

Lab #1 Overview

- Use the team invitation received by email to get connected to the Squawk project.
- Configure your Team Concert workspace for instant messaging.
- Explore the organization of your new team and start up a friendly chat to introduce yourself.
Lab #1 Concepts learned

- Team Invitations make it easier to get team members connected to your project resources managed in Rational Team Concert.
- Rational Team Concert has built-in instant messaging support that makes it easy to connect and collaborate with your teammates.

Questions
Planning your work

An IBM Proof of Technology

Objectives

- Understand the agile project planning capabilities of IBM Rational Team Concert™
- Learn about Work Items and how they are central to Rational Team Concert.
- In the lab, you will learn how to create and work with Work Items and Iteration Plans.
Project plans

- A plan revolves around the following elements
  - Teams
  - Time
  - Work

- Planning levels
  - Release
  - Iteration/Sprint
  - Your day-to-day work

Team

- The project team is divided into one or more teams.
  - A team will focus on one aspect of the project
- Each person can divide their time across many projects and between different teams.
  - Team Concert will take into account participation in different projects and teams.
Time

- A project has one or more schedules or **timelines**.
- Each timeline is broken down into a series of iterations.
- Each iteration can be broken into smaller iterations as required.
- A milestone marks the end of an **iteration**.
- At the end of any iteration, you may produce a **release**.

---

Work

- All work in your project is tracked as one or more work Items.
- Different kinds of work items are available including:
  - Plan work item types, for example:
    - Plan Item
    - Epic
    - Story
  - Execution work item types, for example:
    - Task
    - Defect
- Plan work item types are used to capture high-level plan elements.
- Execution work item types are used to capture the lower-level details and the work should be completed in a single iteration.
- Each kind of work item has its own life cycle.
Team, work items and plans

- Work items connect your team to the plan.
- A project can have an overall plan.
  - This shows all high-level plan work items for the project.
- Each team can have a plan for the project.
  - This shows all high-level plan work items for that team for the project.
- Each team can have a plan for an iteration.
  - This shows the work assigned to that team for that iteration.
- Schedule risk assessment plans can also be used to work out the level of uncertainty that work items will be completed on time.
- Developers can use developer taskboards to more easily visualize their work.

![Backlog](image)

Overall project plan
Team project plan

Module 2 - Planning Your Work

Team iteration plan

Module 2 - Planning Your Work
Schedule risk assessment plan

More detailed developer estimation: low, medium, high

Color codes high risk tasks for quick identification and action.

Automatically calculates probability of task fitting into the schedule.

Developer’s taskboard

See the work currently in progress.

Drag-and-drop work items to change their state.
Plan modes

- Plans can be displayed in different modes
- **View As: Ranked List** is ideal for managing SCRUM backlogs
  - Support coarse and fine grained prioritization
  - Ranking is reflected in all planning views, for instance, iteration plans and release plans

Plan snapshots - Compare plans over time

- Plan snapshots help you reflect on the quality of your planning.
- You can compare snapshots and see how your plan has changed over time.
Lab #2 Scenario

- You want to track all the work on your projects.
- All your work (for example: plan items, stories, tasks and defects) are based around the concept of work items.
- You see how work items are fundamental to Rational Team Concert and how you use these work items to track the work you do.
- You prioritize and link your work so that you can do the right things at the right time in the plan.

Lab #2 Overview

- The instructor will create new stories to create additional squawkers.
- You will create task work items for your squawkers.
- You will assign your work to the right team member (you!)
- You will set the priority for your task work items and estimate how long they will take to complete.
- You will link your new task work items to the stories created by the instructor. The relevant plans will be updated automatically.
- You will explore how tagging can make it easier to find work items.
- The instructor will send you the plans through chat and you will examine the plan. *

* RTC for Visual Studio supports drag-and-drop to external instant messaging tools only.
Lab #2 Concepts learned

- Work items in Rational Team Concert are a central team artifact in the development process.
- Everything gets tracked using work items so nothing gets lost which provides project transparency and real time data access.
- Work items are used to create the iteration plan linking project data to the overall plan.
- Plans are live, dynamic and visible to the entire team helping to create a collaborative project environment.
- Video overview available from the online Help under Tours
Optional slides

Rational Team Concert supports teams with a traditional planning approach.

Formal project management process template

- A single timeline with a release and backlog divided into more traditional project phases.
- New work item types, features, and views in the template such as risks and risk actions
New schedule view - Gantt charts

- Visualize a plan based on a timeline.
  - Gantt charts based on schedule constraints and dependencies
  - Shows work breakdown and schedule

Resource allocation and scheduling

- Search and add a resource to your plan based on availability.
- Allocate ranges of availability to a project or team.
Fine-grain time tracking

- Move a task between people.
- Track time towards different time codes.
- Extract time tracking data using the REST API.
Keep track of all our work

An IBM Proof of Technology

Objectives

- Explore IBM Rational Team Concert™ query capabilities.
- Create and run queries.
- Use and configure the Team Dashboard* and My Work* views to get a real time view of project, team and individual status.
- Use feeds to get real time view of project, team and individual status.

* Not currently available in Visual Studio
Real time collaboration

- The modules before showed how the project team plans the work for an iteration.
  - But how does the project keep track of all the planned work items?
  - How do I see who may help me with my actual problem?
  - How do I get the most recent status of the project?

- What if your tool knows the actual status of your team's work?
  - Rational Team Concert stores all artifacts for the development project in one repository and provides powerful query capabilities to retrieve and display data.

- Take advantage of Rational Team Concert’s extensive collaboration capabilities
  - Define queries on Work Items to find your work and the work of others.
  - See who is online and ready to collaborate with you. *
  - See the event log for build or work item events that are interesting to you and follow RSS feeds for News.
  - Generate, display and export reports on the status and health of the project.

- Rational Team Concert displays the information in automatically refreshed views that are configurable, so that you are up to date with the information you need in real time.

* Not currently available in Visual Studio
Work item queries

- Provides real-time project health information and transparency of status through automated data gathering.
- Rational Team Concert provides a query mechanism to find work items in a Project Area allowing for more project transparency.
  - The query scope for work items is the project area.
- The user interface includes
  - an editor for building structured work item queries
  - an end-user configurable work item view to browse the query results.

Feeds

- Provides a real-time view of the project’s health
- Rational Team Concert lets you configure a feed by subscribing to your team events.
  - Build events for My Teams feed delivers notification of build results.
  - My Work Item Changes feed delivers notification of changes made to work items that you own, created, modified, or subscribe to.
  - My Teams in Project Area feed delivers notification of all events in your team areas.
Lab #3 Scenario

- You recently joined the development staff of the Squawk project.
- Your environment was properly set up by accepting the invitation for the core project team.
- Now it is your task to become familiar with the work and the tasks to do.
- You are using the real time collaboration capabilities of Rational Team Concert to be up to date with the
  - Latest news feeds,
  - Status of the project, and
  - Work items assigned to you

Lab #3 Overview

- As a user you will
  - Write and run work item queries in the Eclipse client and in the web UI.
  - Use the capabilities of the Team Dashboard View*
  - Configure the My Work View*
  - Use the capabilities of the Feeds tool window**

* Not currently available in Visual Studio
** Performed only in Visual Studio
Lab #3 Concepts learned

- Rational Team Concert provides powerful query capabilities for work items creating real-time access to detailed project data.
- Create customized queries or use predefined queries to enable unique project views for a wide range of users.
- Rational Team Concert helps teams collaborate by creating an environment where real time project status and data are available.
- Easily customized views to fit your needs.

Questions
Performing and sharing your work

An IBM Proof of Technology

Objectives

- Understand Software Configuration Management (SCM) concepts in IBM Rational Team Concert™.
- Create and use a repository workspace for work assigned to you.
- Create or make changes to artifacts under source control.
- Associate changes with work items.
- Deliver changes from repository workspaces to streams.
- Accept changes from other members of your team.
- Understand conflict resolution.
Basic Jazz SCM anatomy

- Streams are for sharing.
- Repository workspaces are your personal space.
- The local sandbox is a folder on your local files system where you develop and test.
- Change-sets flow back and forth.

Components

- Repository workspaces
  - Partitioned into components
  - Jazz understands the structure of your components
  - Jazz directly supports component-based development
Components (continued)

- Components Track Changes
  - Configuration of resources builds from the change set flow
  - Each change set builds on what came before

- Components Change History
  - A time-ordered sequence of change sets
  - Describes how the component’s content was built from nothing
Change set details

- Composed from a collection of changes to one or more files and folders
  - A change set that affects multiple resources is committed as a single atomic unit
- Indicates the reason for the changes
  - Through a comment, or
  - By referencing the relevant work item
- Can be shared with another team member
  - Using a stream, or
  - From your repository workspace in a work item

Streams

- Stream
  - A place to share source with your team
Typical journey for a change set

1. Change

2a. Check-in

2b. Change set

3. Complete (usually combined with deliver)

4. Deliver

Change-set delivery is process enabled

- The deliver operation is process-enabled, allowing the team’s process to check and enforce delivery rules automatically.
Delivery notifications

- An incoming change-set is
  - In the change history of the stream, but
  - Not in the change history of your repository workspace
- Accept adds the change-set to your repository workspace’s change history
Suspending your changes

There are times when your development work can’t move forward…

- Interruption to focus on higher priority work
- Defer code conflicts
- Exploratory work that you may not commit
- Enter **Change Set Suspension**: Safely put your work on hold.
- The change is preserved in your repository workspace but hidden from other incoming or outgoing changes.
- **Very cool!!!**

Conflicts in Jazz SCM

- Component modified in multiple workspaces that have the same flow target, can lead to conflicts in the change sets that result.
- Structural conflicts
  - Incoming and outgoing change sets include changes to the same directory namespace, usually by moving, removing, or renaming files or folders.
- Content conflicts
  - Incoming and outgoing change sets include changes to the same file.
- Possibility to avoid conflicts by locking resources in a stream
  - Useful when modifying a file type that would be difficult to merge if someone else made concurrent changes such as images.
Resolving content conflicts

- Manually resolve
  - Resolve the conflicts in the Eclipse Compare Editor.
- Auto resolve
  - Attempts to automatically resolve by merging non-conflicting changes such as simple additions or removals.
- Resolve with proposed
  - Replaces the file in your workspace with the one that contains the conflicting changes.
- Resolve with mine
  - Replaces the file that contains the conflicting changes with the file that is currently in your workspace.
- Discard
  - Resolve a conflict by discarding the change set in your workspace that conflicts with the one you have accepted.

Resolving structural conflicts

- Auto resolve
  - Attempts to resolve the conflict by automatically merging the content of the incoming change set with the content of your workspace
- Resolve with proposed
  - Applies all of the conflicting structural changes to your workspace
- Move
  - Apply a subset of the conflicting structural changes to your workspace by moving or renaming individual conflicted items
- Resolve with mine
  - Remove the conflicting structural changes from the change set
Lab #4 Scenario

- You have been given the task of contributing your own squawker class along with its documentation and, optionally, its test case.

Lab #4 Overview

- You will spend a little time understanding the key concepts of the SCM system in Jazz.
- You will create your own squawker, basic documentation and optionally its test case against the work items you created in Module 2 Planning Your Work.
- You will deliver this work so that other people can use it.
- Finally, you will bring in changes from other members of your team so your code is up to date with everyone else.
Lab #4 Concepts learned

- Jazz Source Control provides private **repository workspaces** to track and back up your changes before you share them with the team using a **stream** for integration.

- A **change set** is the fundamental unit of change and collaboration in your team environment.

- A change set can be associated with a **work item**, which can then be **delivered** as a unit and provides traceability and transparency to the development life cycle.

- The **Pending Changes** view is central to these operations by enabling real time updates and efficiency.

- Video overview available from the online Help under **Tours**

---

**Questions**
Remembering well known SCM configurations

An IBM Proof of Technology

Objectives

- Understand how Component Baselines and Workspace Snapshots can be used.
- Create new repository workspace from a snapshot for maintenance purposes.
- Utilize the Pending Changes view to increase productivity.
What about these questions?

- How do I find a known good configuration of a component?
- How about a known good configuration of an entire stream?
- Hey, exactly what was in that milestone build a year ago?
- That is, what about fixed configurations that do not change anymore?

- Use baselines and snapshots…

Baseline

- An immutable copy of a component’s configuration
  - At a particular point in time, and
  - There can be multiple baselines of a component
- Serves as a fixed point of reference
  - For initializing streams and repository workspaces
  - For sharing source with people or processes
- Can be easily compared
  - With the current state of a stream or repository workspace
  - With another baseline
Snapshot

- A **collection of one baseline per component** in a repository workspace or stream
  - Captures an important repository workspace configuration for later re-creation
  - There can be multiple snapshots of a repository workspace or stream
  - Provides traceability to historical artifacts
- Like baselines, snapshots are used for sharing and collaborating with team members
  - Create a repository workspace or stream
  - Update the contents of a repository workspace
  - Recreate a prior build using a build-created snapshot

Answers to those tough questions

- How do I find a known good configuration of a component?
  - Use a baseline!
- How about a known good configuration of an entire stream?
  - Use a snapshot!
- Hey, exactly what was in that milestone build a year ago?
  - Use a snapshot or baseline!
Lab #5 Scenario

- You have contributed your own squawker class along with documentation and delivered your work.
- Your teammates have been creating and delivering their own squawkers and documentation, which you have accepted.
- These changes need to be captured so that they can be used for further work or returned to at some point in the future if necessary.

Lab #5 Overview

- The instructor will play the role of Team Lead, creating baselines and snapshots to capture all the work completed by the team.
- You will then explore the new baselines and snapshots by querying their contents.
- You will revert a component in your workspace to a previous baseline version with the replace operation which provides a convenient way to reconfigure your workspace.
Lab #5 Concepts learned

- **Baseline** and **snapshot** artifacts increase traceability and enable collaboration among teams and team members.
- **Baselines** are an efficient means to mark artifacts within a single component for later reference.
- **Snapshots** are an efficient means to mark artifacts across a set of related components for later reference.
- It is easy to create a new repository workspace or stream from a snapshot. This is useful for maintenance purposes, fixing builds or forking the code.
- The **Pending Changes** view is central to these operations by providing an easy to use interface to review changes and appropriately update your workspace.

Questions
User’s view of build

An IBM Proof of Technology

Objectives

- Understand the build functionality of IBM Rational Team Concert™
- Understand the flexibility of the build process and how it enables collaboration and teaming
- Observe policies and processes that relate to consistency and repeatability
- Explore build results and observe traceability to artifacts
- Perform a build or a rebuild
Rational Team Concert build

- Is an integral part of the project infrastructure
  - Consistent, repeatable process throughout the project
- Brings awareness of build progress and results to developers
  - Easy sharing of information
- Links build results to related Jazz artifacts
  - Integrated experience, traceability and tracking “baked in”
- Allow developers to have a private build area
  - Build and test code before delivering to the main branch
- Accommodates existing build technologies (Ant, CruiseControl, Build Forge, Maven, and others)
  - Leverages technology that fits your project best
Build is very visible to the user

Build in Visual Studio client
Support for MSBuild and Devenv

- Build template for Microsoft Visual Studio builds
- RTC build engine supports publishing compile and test results from different languages, such as NUnit, MSTest
- Possibility to add Microsoft Visual Studio specific configurations to the definition, such as which kind of tests are going to run

Personal builds

- Builds normally run from a dedicated repository workspace.
- Personal builds
  - Run from your repository workspace.
  - Allow you to build your changes before delivering them to the stream.
  - Provide you with some assurance that your changes will not disrupt the team builds when you deliver them.
Personal builds in Visual Studio and Web Client

Builds and snapshots

- A build can request a snapshot
  - If there are any changes in a component since the last build
    - A new baseline is created with the same name as the snapshot name
  - Convenient for reproducing build problems
Web-based build management

- From Rational Team Concert web UI:
  - Request new build
  - Request rebuild of existing build
- Exposes build facilities to wider community
- Provides access to build function from any desktop

Build email notification

- Email notification criteria is defined in project area configuration data section
Lab #6 Scenario

- You have recently joined your company’s exciting new project called “Squawk.”
- By now you have
  - Planned and tracked your work,
  - Developed a new squawker,
  - And created baselines and snapshots.
- You are now ready to build your application with help of the Team Concert build engine.

Lab #6 Overview

- The instructor will then demonstrate how a build engineer, team lead or other appropriate role, can request a build for use by the project team.
- You will explore the results of existing builds.
- You will request a private build to ensure that your changes won’t break the build.
Lab #6 Concepts learned

- In this module, you explored the build capabilities of Rational Team Concert. You have explored existing builds and learned how to request new builds or rebuilds.
- Treating the build as an integral part of the project infrastructure makes it easy to keep processes and policies consistent and repeatable.
- Every team member has access to build data which promotes communication and collaboration among the contributors – on local or remote sites.
- Linking build results directly to Jazz artifacts provides a high level of traceability.
- Using existing build technologies makes it easy to adapt to needs of different projects.

Questions
Exploring changes and traceability

An IBM Proof of Technology

Objectives

- This lab will demonstrate how information is linked within IBM Rational Team Concert™ to establish traceability.
- Determine what work items and files are included in a build.
- Determine change sets that are included in a build.
- Determine who changes files, when and why.
- Compare versions of a file.
- Observe specific changes to files.
Builds
Keep traceability with work items, change sets, repository workspace and more

Identify work items and change sets that went into the build.

Identify work items and change sets that went into the build.
Work items
Know what build it has been included in and its change sets

List associated work items.

Drill down into the details of a work item.

Work items in Visual Studio client
Know what build it has been implemented in, its change sets and other factors

List associated work items.

Drill down into the details of a work item.
Change sets

Easily allow users to understand to keep track of all related elements

Review change sets that make up the build.

Change Explorer lists files that were modified for a given change set.
Compare changes
Quickly provide users with the ability to identify differences in elements

Compare changes between versions of a file.
Visualize change history

Colors indicates when changes have been made. Hover the mouse over the change and get more details.

Use annotation to view specific changes.

Annotate support in Visual Studio client

- Right-click Annotate View of any file in the repository.
Lab #7 Scenario

- You have completed some builds for the Squawk project and are now ready to look at how Rational Team Concert links the software artifacts that make up the builds.
- You will investigate the build artifacts to see how Rational Team Concert automatically manages traceability.
- You will review the change sets (work items and associated changes under source control) that make up the build and explore the change history.

Lab #7 Overview

- You will experience how information is linked within Rational Team Concert.
- As a team member you will explore how traceability helps answer questions such as
  - What work items went into a build?
  - What changes were made for a work item?
  - What build did a work item get delivered in?
  - Who changed a file, and why?
  - What are the specific changes made on a resource?
  - How to visualize the change history for a resource?
Lab #7 Concepts learned

- Rational Team Concert maintains full traceability for changes contained in a build.
- Work items maintain a record of the changes made to resources maintaining consistency and transparency in the project.
- Changes are collected and managed as change sets and available for reporting purposes and analysis.
- Users can drill down into the detailed change history of every artifact, enhancing collaboration and quality.
Endgame and a tightened process

An IBM Proof of Technology

Objectives

- Understand how process is defined in Jazz and implemented by IBM Rational Team Concert™
- Understand how roles can be used to control process workflow
Motivation for the team process component

- Generally all software teams have some sort of process
  - May be formal or informal…
- Successful teams…
  - Believe their software process helps produce quality software
  - Own their process and accept accountability for it
  - Continually adapt their process to changing needs
- However, success depends on…
  - Common understanding by all team members
  - Consistent execution
- Many times…
  - Process relies on documents (or word of mouth) for understanding and human memory for execution and is otherwise very manual
  - Leads to inconsistent or erroneous execution

What if your tools understood how your team works?

In a basic process model…

- Teams work on projects
- Each project follows a process
- Each team is unique and thus can work differently
- Work inside the scope of a team follows the team’s process
- Cross-team work follows the process of the broader team
- Team members play roles defined by the process
- Process manifests itself through artifacts types, operations manipulating the artifacts, and artifact change events
Jazz process support

- Support different degrees of flexibility and formalism
- Allows for **predefined** processes
- Allows for **emerging** processes
- Allows for **variations**
- Allows for **exceptions**
- Allows for **consolidation**
- Allows for **evolution**
- Allows for **extensions**
- Put knowledgeable human in the center
- Comprises runtime, authoring, and inspection support

Project area iteration structure and terminology

**Line of development** \(\{1..n\}\)

**Iteration** \(\{1..n\}\)

**Process state**

**Timelines** are an element of a project area that own a set of deliverables and its production schedule (**maintenance**, **new release development**).
- Often represents parallel development
- A team area is associated with a timeline

**Iteration** represents some project work interval
- Any depth of nested iterations
- Process specification in any iteration
- May contain start and end dates

**Process state** is defined as the current iteration in a timeline
- Indicated by the blue arrow

Example:
*Main Development Line* process state:

2.0 M2

*Maintenance Development Line* process state:

1.0 Fix pack 1
Process is defined in one or more iterations

- Specified as a set of component operation rules
- Rules are assigned to user roles (default, contributor, team lead and others)
- You can have the general process defined for the project
- Override or augment the general process in planned iterations

Active process defined by process state: \( \{1.0, M1, \text{endgame}\} \) in this example

A team area can augment or override the process of any iteration

Who’s who on plans...

Timeline

Release N

- Project Release Plan (alias Product Backlog) = All top level WIs planned for an iteration and its sub-iterations
- Team Release Plan (alias Team Backlog) = All WIs planned for an iteration and its sub-iterations for a given team
- Iteration plan (alias Sprint Backlog) = All WIs planned for an iteration for a given team
- Another iteration plan

<table>
<thead>
<tr>
<th>Category</th>
<th>Team</th>
<th>Release</th>
<th>Work Item</th>
<th>Release</th>
<th>Work Item</th>
<th>Release</th>
<th>Work Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cat. A</td>
<td>Team T1</td>
<td>M 1 (Sprint)</td>
<td>WI</td>
<td>M 2 (Sprint)</td>
<td>WI</td>
<td>M 2 (Sprint)</td>
<td>WI</td>
</tr>
<tr>
<td>Cat. B</td>
<td>Team T2</td>
<td>M 1 (Sprint)</td>
<td>WI</td>
<td>M 2 (Sprint)</td>
<td>WI</td>
<td>M 2 (Sprint)</td>
<td>WI</td>
</tr>
<tr>
<td>Cat. C</td>
<td>Team T1</td>
<td>M 1 (Sprint)</td>
<td>WI</td>
<td>M 2 (Sprint)</td>
<td>WI</td>
<td>M 2 (Sprint)</td>
<td>WI</td>
</tr>
</tbody>
</table>

WI: Top level work item (Stories, epics)

WI: Other work item (Defects, tasks, enhancements...)
Lab #8 Scenario

- As you approach your final milestone, you have the chance to alter the process for the iteration so that your rules get stricter.
- For example, you might insist that all tests run to completion and without error before you are allowed to deliver any changes.

Lab #8 Overview

- In this lab, the team will move to the *Endgame* iteration of milestone 1.0 M3 and will experience a change in the process.
  - In the M3 Endgame iteration, the Core Library team has customized the process such that changes can be delivered only if the team lead has approved the work item associated with the delivery.
- The instructor will move the project to the 1.0 M3 Endgame iteration.
- As a user with contributor role on the Core Library teams, you will make a change to the squawker class.
- During the delivery, user will notice the change to the squawker class for the Core Library will not complete because the work item associated with the delivery does not have the approval of the team lead.
- The instructor (as team lead) will approve the work item.
- The user will now be able to deliver the work item.
- The instructor will move the project back to the 1.0 M3 Development iteration.
Lab #8 Concepts learned

- Jazz processes capture the idea and the notion of choreographies of collaboration.
- With Jazz collaboration, rules are your friend—not something you have to fight. Keep your processes as concrete as possible and as strict as necessary.
- Process sandboxes allow good things to happen on all levels.
- Process support in Jazz is an ongoing endeavor.

Questions
Taking control of your project

An IBM Proof of Technology

Objectives

- Learn about the Jazz dashboards and reports and how these powerful capabilities can assist your team in tracking project status and make informed-based decisions to keep it on track.
Dashboards

- Dashboards are a web UI component intended to provide information about the project status at a glance.
- It provides for easy drill-down capabilities to get more complete information.
- Dashboards are available in the IBM Rational Team Concert™ Standard and Enterprise editions.
- Customizable mini-dashboard gives quick access to frequently used information, such as “work item assigned to me.”

Dashboards for team unity

- Clearly understand team goals
- Risks, issues, challenges surfaced at both the team and project level
- Real time status
- Transparency and control using customizable dashboards
Dashboards and reporting for progress

Trending by project or by individual team

Current milestone status

Team member details

Display your choice of reports and queries in your own dashboard, for example, to control the flow of work items.
Roll up information and drill down for details

Dashboards – project status with burndown charts
Mini-Dashboard and Open Social Gadget

- Customizable mini-dashboard gives quick access to frequently used information, such as “work item assigned to me”.
- Integrate with any product that supports Open Social Gadgets

For a detailed report, open the Reports page and choose from a variety of available reports.
Reports – Displaying the project status

● Reports are available in the Rational Team Concert Standard and Enterprise edition.
● Rational Team Concert uses the BIRT* reporting engine
● A variety of out-of-the-box reports are available to display an actual overview of your projects:
  ‣ Reports for the health of your builds
  ‣ Reports for viewing the team load and the distribution of work items
  ‣ Reports for your code
  ‣ And others
● Reports can be arranged in the web UI Dashboards
● Reports can be exported to: .pdf, .xls, .doc and .ppt formats

*BIRT, or Business Intelligence and Reporting Tools, is an open source Eclipse-based reporting system that integrates with your Java™ or J2EE application to produce compelling reports.

Lab #9 Scenario

● As a member of the Squawk project, create and customize your own private dashboard and include content from another project area. You will also explore some out-of-box reports.
Lab #9 Overview

- The instructor will create a new project area which you will later include in a dashboard.
- Customize your personal dashboard for the Squawk project.
- Add content from the JUnit project on your Squawk project dashboard.
- Explore reports using the web UI.

Lab #9 Concepts learned

- Rational Team Concert provides **transparency and control**, using customizable dashboards
- Rational Team Concert automated project management dashboards are transparent to everyone—not just managers. This immediate and **automated feedback helps keep teams on track and motivated** to achieve project goals
- Rational Team Concert comes with a variety of out-of-the-box report formats to display and export the actual real time, in context project status.
Questions
Session summary

An IBM Proof of Technology

- We have described current collaboration challenges with distributed teams.
- We have explored how IBM Rational Team Concert™ can:
  - Enable development teams to **collaborate in real time in the context** of the work they are doing, especially in globally diverse environments
  - Enable projects to be managed more effectively by providing visibility into **accurate project health information** drawn directly from actual work
  - Automate traceability and auditability by **managing artifacts and their inter-relationships** across the lifecycle empowering teams to deliver more value
  - Provide **customizable process design and enactment** through rule-based process guidance, automation and definable checkpoints
- We have provided a hands-on experience using Rational Team Concert to automate the software delivery process.
Next steps

- Engage your local IBM Rational® software team.
  - Provide a customized demo for your team
  - Conduct a targeted proof of concept

- Register on jazz.net and explore learning tutorials and videos.
  - http://jazz.net

Rational Team Concert sandbox

- Create work items
- Deliver source code
- Customize dashboard
- Edit plans
- Invite others

Get up and running in seconds!  No waiting in line!  Evolving over time!
Additional resources

- Learn more about and download free trials of Rational Team Concert at http://ibm.com/rational/rtc
- Participate in the open commercial development of Jazz by joining the community http://jazz.net
- Learn more about the Jazz technology and the future IBM Rational product roadmap http://ibm.com/rational/jazz/roadmap

Questions
Thank You

We appreciate your feedback in order to improve this educational event. Please fill out the survey form.
© Copyright IBM Corporation 2008, 2011.

The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, these materials. Nothing contained in these materials is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software. References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. This information is based on current IBM product plans and strategy, which are subject to change by IBM without notice. Product release dates and/or capabilities referenced in these materials may change at any time at IBM’s sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way.

IBM, the IBM logo and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at ibm.com/legal/copytrade.shtml

Other company, product and service names may be trademarks or service marks of others.