



An IBM Proof of Technology

Collaborative software development using IBM Rational Team Concert

Presentations



PoT.Rational.07.2.038.04

© Copyright International Business Machines Corporation, 2008, 2009. All rights reserved.

US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Collaborative software development using IBM Rational Team Concert

An IBM Proof of Technology



© 2009 IBM Corporation

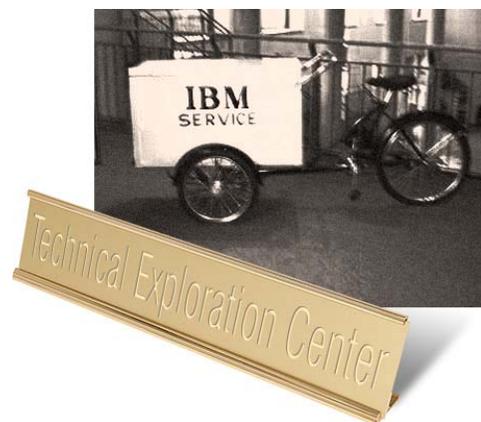
Welcome to the Technical Exploration Center

- 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 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
- 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...

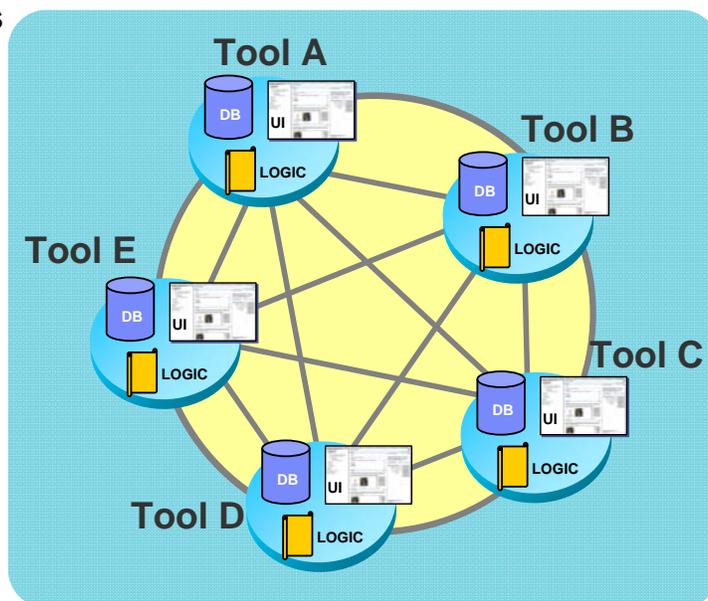
- ... about your teams
- ... about your artifacts
- ... who is responsible for what
- ... about your process
 - Code delivery rules, code quality, traceability, test runs, intellectual property
- ... how to bootstrap a project
- ... how to help new team members get started
- ... your favorite work item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



Collaborative Application Lifecycle Management

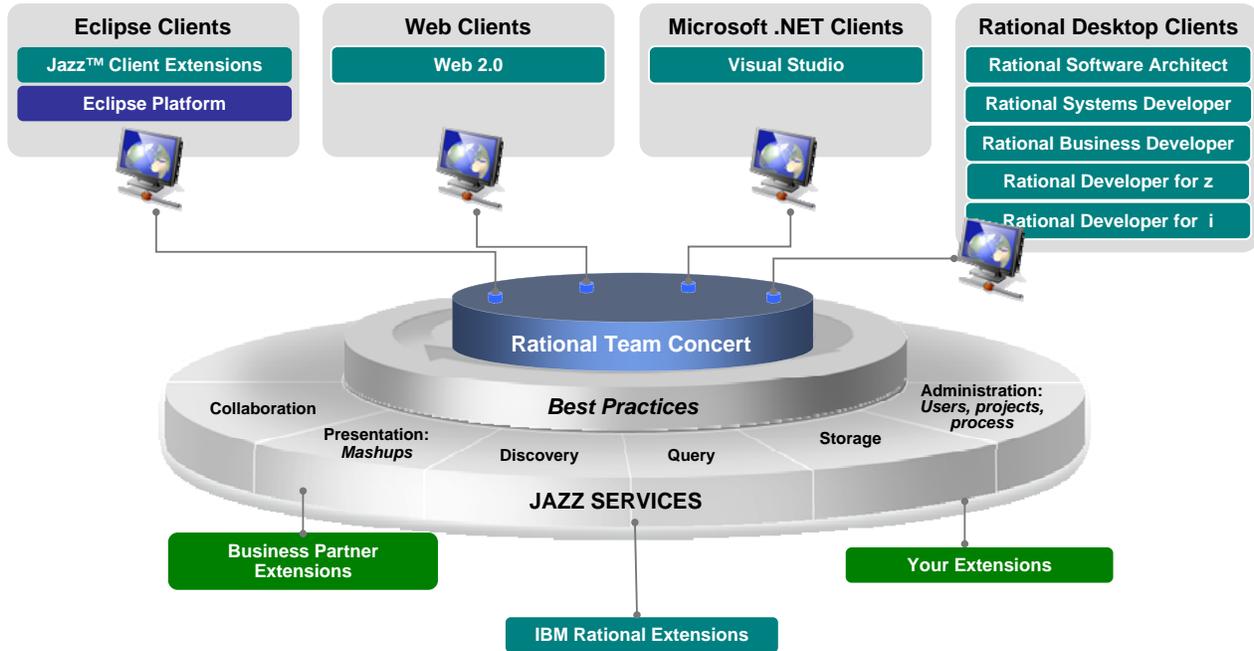
The challenge in enabling it

- Traditionally, each tool came with its own
 - ▶ 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 API's



Rational Team Concert: An open, extensible architecture

Supporting a broad range of desktop clients, IDE's and languages



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, System z, co-existence with popular toolsets



Open and extensible on

- ✓ 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

Agile Planning

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

Project Transparency

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

SCM

- Integrated stream management
- Component level baselines
- Server-based sandboxes
- Identifies component in streams and available baselines
- SVN, Git, CC bridge, connector

Work Items

- Defects, enhancements and conversations
- View and share query results
- Support for approvals and discussions
- Query editor interface
- ClearQuest® bridge, connector

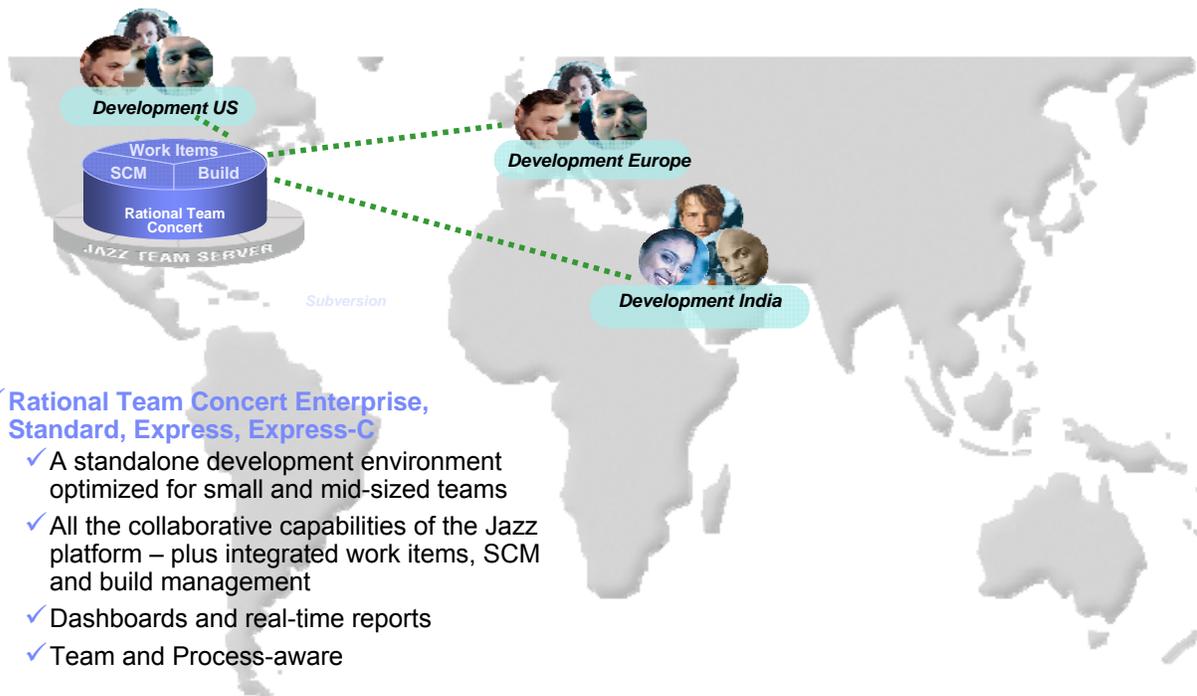
Build

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

Jazz Team Server

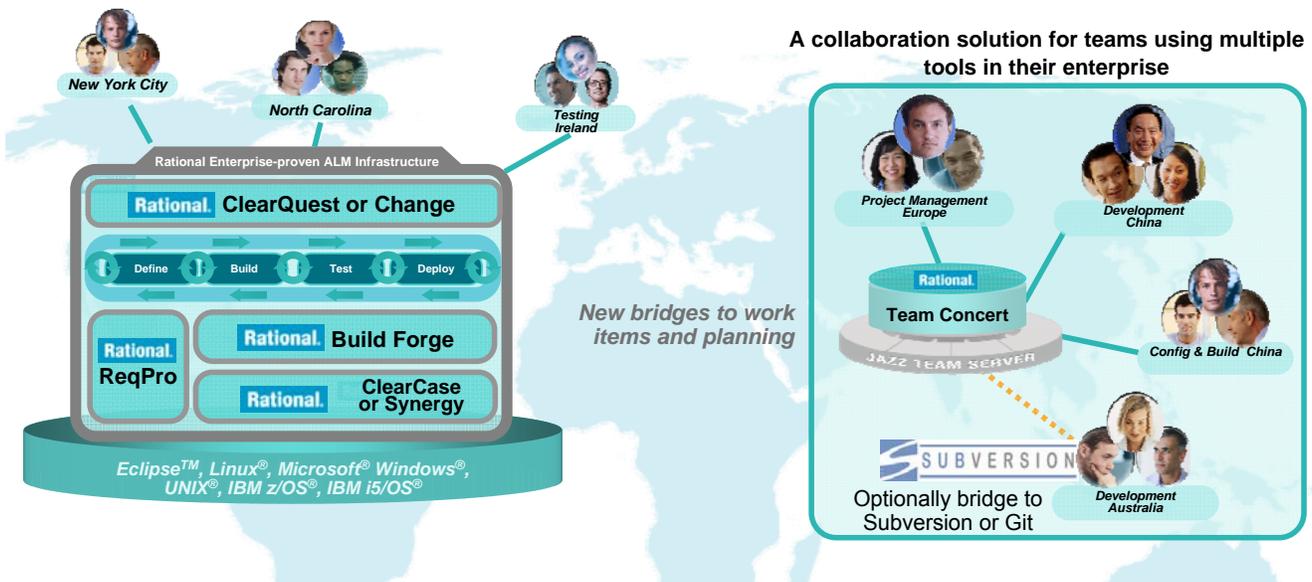
- Single structure for project related artifacts
- World-class team on-boarding / off-boarding including team membership, sub-teams and project inheritance
- Role-based operational control for flexible definition of process and capabilities
- Team advisor for defining / refining "rules" and enabling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

Leveraging Rational Team Concert independently



- ✓ Rational Team Concert Enterprise, Standard, Express, Express-C
 - ✓ A standalone 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



A collaboration solution for teams using multiple tools in their enterprise

New bridges to work items and planning

- Manage planning and project status with work items and dashboards in Team Concert and develop with existing artifacts that reside in subversion, Git or ClearCase® (new bridges in RTC 2.0)
- Enables teams to reuse assets, process and investment in Subversion, ClearCase/ClearQuest or Git
- Third party connector to Jira
- Take advantage of new collaborative ALM in an evolutionary way with lower business risk

Envisioning a platform that can transform software delivery

Jazz is a project and platform for *transforming how people work together* to deliver greater value and performance from their software investments.



- robust, extensible and scaleable
- globally distributed, fluid & dynamic
- community-based & open at Jazz.net



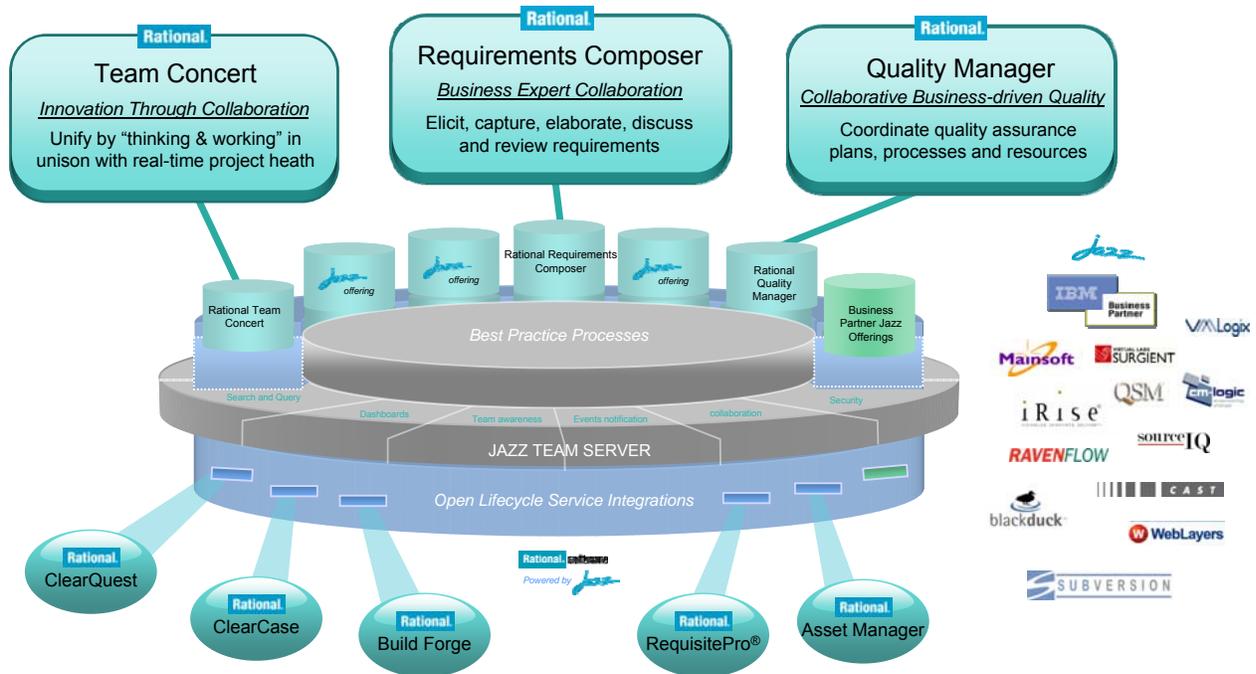
Open Commercial Development at jazz.net

Delivering greater openness and customer participation in the products they depend on for software delivery

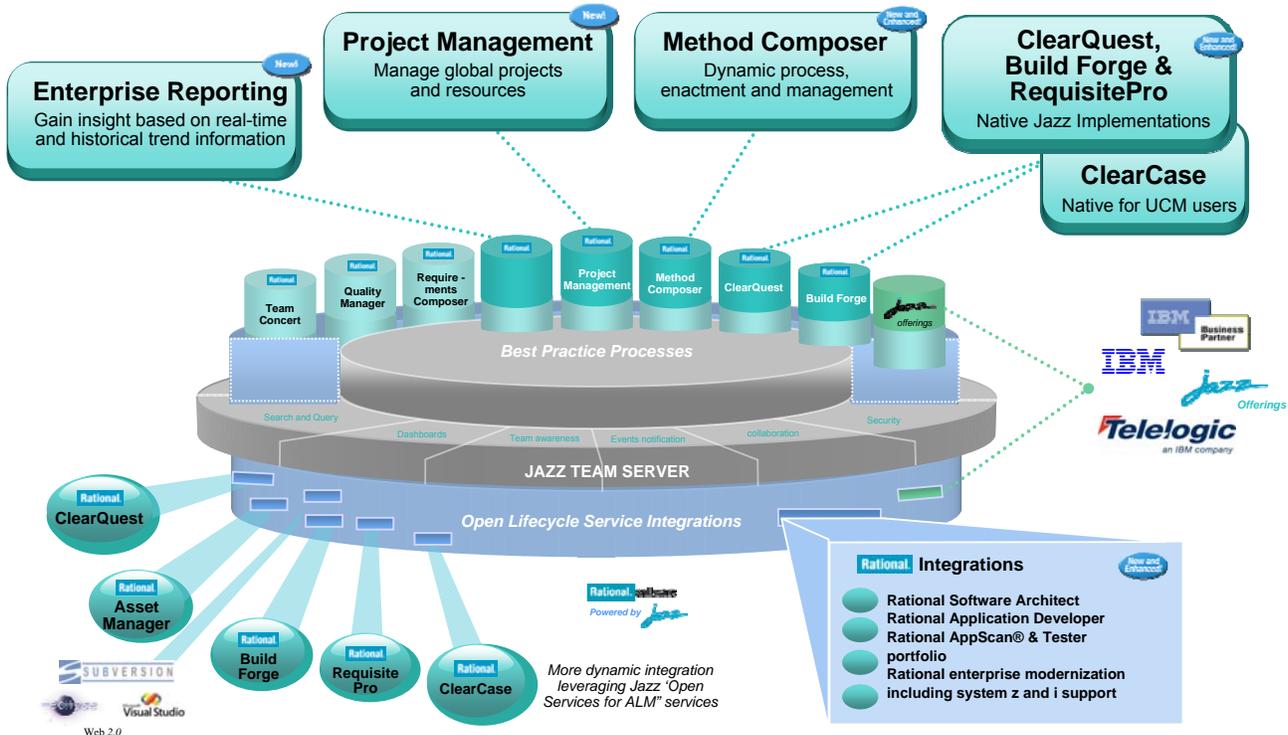
- IBM is opening up the **Rational Software Delivery Platform** for greater ease of consumption, extensibility and integration to meet the unique usage needs of our customers
- IBM is providing transparent, collaborative customer participation in the development of new Rational technologies through an open commercial community



Introducing the first wave of Jazz offerings

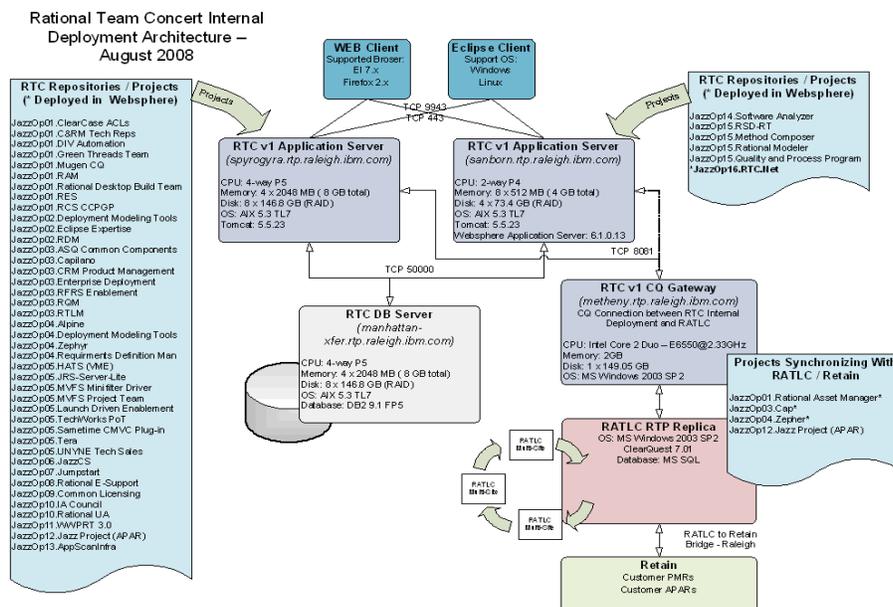


The road ahead: What to expect from Rational Jazz based offerings that will be delivered throughout 2009



IBM's Internal Deployment of Team Concert

- In Rational, today we have 2,016 developers and testers
 - ▶ 80 development projects, two application servers (hosting 20 RTC server instances), one DB2 server
- In IBM we have 40,000 developers - By end of 2009 1/3 of these will be using RTC
 - ▶ And this is NOT being mandated, but rather through viral adoption and real productivity gains



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 keeps 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."

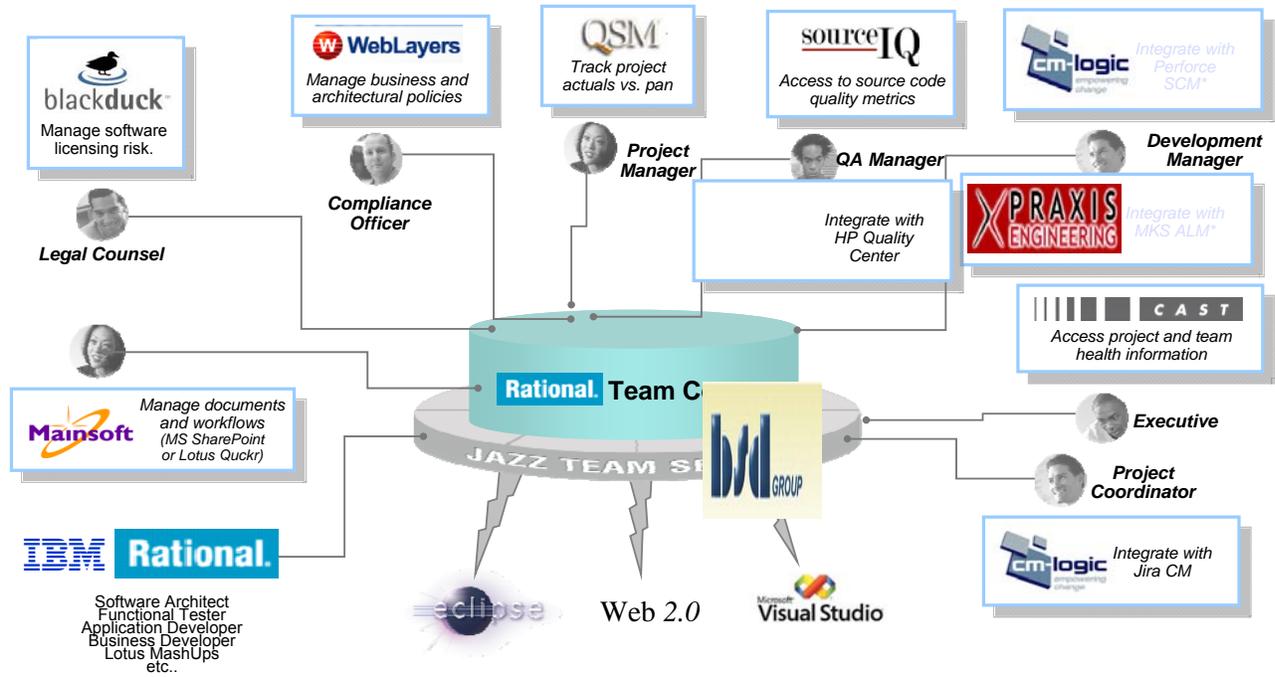
--Carson Holmes - Unified ALM Services Manager, Noblestar



"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

A Growing Ecosystem of Rational Team Concert Partners



*Under development still

Questions

Enterprise Scalability for Team Concert



Express-C

- Small Teams**
- Quick Start
 - Collaborative ALM
 - Open source servers

Express

- Mid Sized Teams**
- Project Dashboards
 - Collaborative ALM
 - Commercial Middleware

Standard

- Corporate Teams**
- Customizable workflow
 - Advanced Reporting
 - Departmental Scaling

Enterprise

- Enterprise Teams**
- Customizable Workflow
 - Advanced Reporting
 - High Availability
 - Unrestricted scaling based on hardware

	Express-C	Express	Standard	Enterprise
Maximum developers/contributors	10 / unrestricted	50 / unrestricted	250 / unrestricted	Unrestricted / unrestricted
Database	Derby Only	Derby, DB2, Oracle, SQLServer	Derby, DB2, Oracle, SQLServer	Derby, DB2, Oracle, SQLServer
Application Server	Tomcat Only	Tomcat, WebSphere	Tomcat, WebSphere	Tomcat, WebSphere
<ul style="list-style-type: none"> • Agile planning: daily, iteration, release • SCM -Stream and component based • Builds – w/ Continuous Integration • Work items w/ discussion, approvals • Customizable Process Templates • Project Milestone Tracking & Status • Subversion Integration Bridge • Cross project dashboards • Cross repository dashboards • Project level permissions 	✓	✓	✓	✓
<ul style="list-style-type: none"> • Work Item custom attributes & presentation • Role-based process permissions 	✓	✓	✓	✓
Customizable Dashboard Mashups	1 per project	1 per project	unlimited project, team, and personal dashboards	unlimited project, team, and personal dashboards
Advanced Reports and Customization			✓	✓
Customizable work item workflow			✓	✓
Plan risk assessment			✓	✓
CC/CQ Connectors and Bridges			✓	✓
Floating Licenses Available			✓	✓
LDAP import / synchronize			✓	✓
HTTP proxy support			✓	✓
High Availability				✓*

New in 2.0 for Express-C/Express



* See Jazz.net for prerequisites



IBM
Software
Group

Rational Team Concert Client for Microsoft Visual Studio IDE

Collaboration, automation and reporting for heterogeneous
development teams

An IBM Proof of Technology



© 2009 IBM Corporation

Agenda

- Rational Team Concert client for Microsoft® Visual Studio IDE
 - ▶ Features and Benefits
- Summary: Supports all your Windows® development needs
 - ▶ Windows Platforms
 - ▶ SQL Server Database
 - ▶ Visual Studio 2005 and 2008 IDE



Unify ALL your teams using Eclipse, the web or Visual Studio, with a single collaborative development platform.



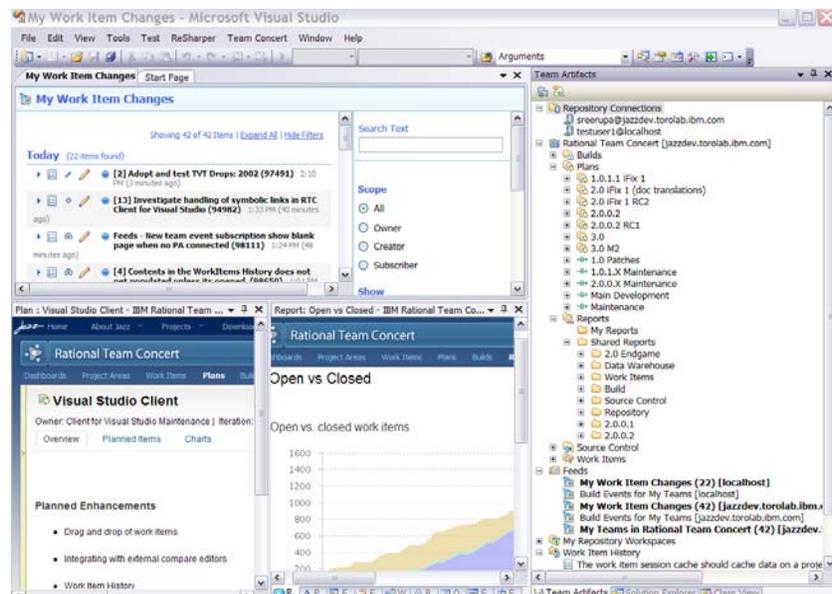
Rational. software

Extend team collaboration to Visual Studio developers

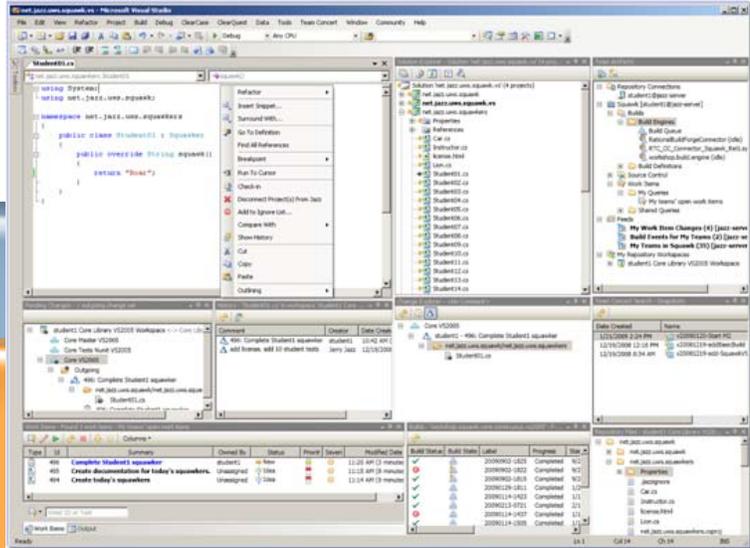
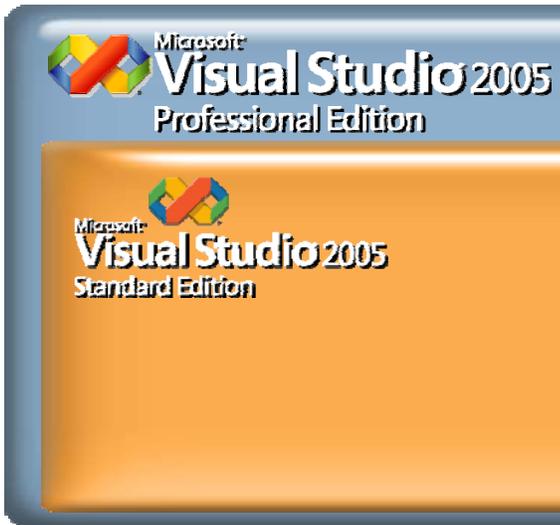
Rational Team Concert client for Microsoft Visual Studio IDE

• 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



Supports Both Visual Studio 2005 and 2008 Professional and Standard Editions



Open and extensible on *Jazz*

- Collaborate in context
- Right-size governance
- Day one productivity

SCM, Work items directly from VS.NET IDE – Build CLI

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

SCM

- Integrated stream management
- Component level baselines
- Server-based sandboxes
- Identifies component in streams and available baselines
- ClearCase connector
- Integrates with external compare-merge editors

Work Items

- Defects, enhancements and conversations
- View and share query results
- Support for approvals and discussions
- Query editor interface
- ClearQuest 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

Microsoft **SQL 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

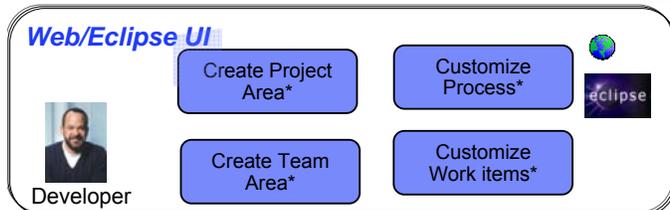
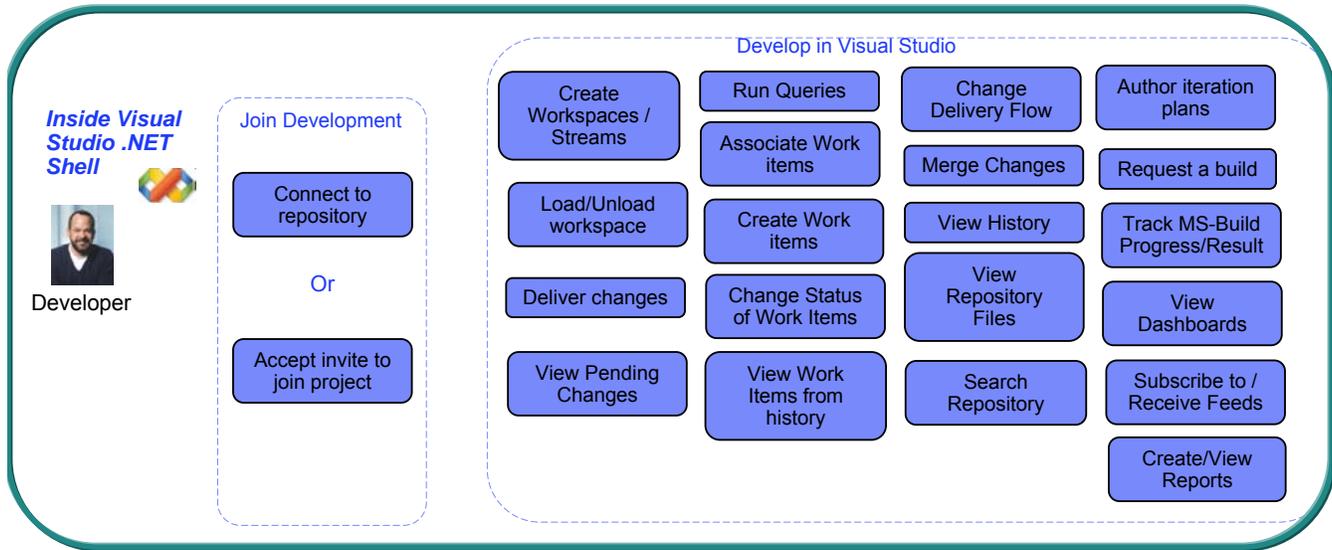
Jazz Team Server

- 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

IBM **DB2**

ORACLE

A Rich native client in Visual Studio



Do Visual Studio builds integrate with Team Concert?

• **Yes. They do.**

- ✓ Just use MS-Build command line builds with Rational Team Concert continuous Integration build engine
- ✓ Link command line MS-Builds to the Rational Team Concert build toolkit and store results in the repository
- ✓ Use the Eclipse client to control the build definition and any of the team concert clients to control and view the build results.

Build Definition

ID: workshop.squawk.core.continuous.vs2005 Project or Team Area: Core Library

Name	Value	Description
antLibrary	plugins\org.apache.ant_1.7.0.v200803061910\lib	Directory relative the build engine eclipse directory where the ANT libraries are found
buildEngineDirectory	C:\Jazz\TeamConcertBuild\buildsystem\buildengine	Absolute Directory where the build engine is located
buildType	Debug	Set to Debug or Release
jre	C:\Jazz\TeamConcert\client\ eclipse\jdk	Java Runtime environment containing a tools.jar lib
loadDirectory	C:\squawk.vs2005\build.input	Where the workspace is loaded
loadDirectoryIsAbsolute	true	loadDirectory is relative to the build engine's eclipse directory unless this is set to true

Build workshop.squawk.core.continuous.vs2005 20090902-1825

Status: Successful
 State: Completed
 Start time: Sep 2, 2009 6:25 p.m.
 Completed: Sep 2, 2009 6:25 p.m.
 Duration: 55 seconds

Build Status	Build State	Label	Progress	Start Time	Duration	Build	Completion Time	Build Engine	Tags	Requester
✓	Completed	20090902-1825	Completed	9/2/2009 6:25	54 sec.	workshop...	9/2/2009 6:26	workshop		Jerry Jazz
✓	Completed	20090902-1822	Completed	9/2/2009 6:22	47 sec.	workshop...	9/2/2009 6:22	workshop		Jerry Jazz
✓	Completed	20090902-1815	Completed	9/2/2009 6:15	54 sec.	workshop...	9/2/2009 6:16	workshop		Jerry Jazz
✓	Completed	20090129-1811	Completed	1/29/2009 6:11	3 minut.	workshop...	1/29/2009 6:1	workshop		Jerry Jazz
✓	Completed	20090114-1423	Completed	1/14/2009 2:23	2 minut.	workshop...	1/14/2009 2:2	workshop		Jerry Jazz
✓	Completed	20090213-0721	Completed	2/13/2009 7:21	32 sec.	workshop...	2/13/2009 7:2	workshop		Jerry Jazz
✓	Completed	20090114-1437	Completed	1/14/2009 2:37	3 sec.	workshop...	1/14/2009 2:3	workshop		Jerry Jazz
✓	Completed	20090114-1505	Completed	1/14/2009 3:05	2 minut.	workshop...	1/14/2009 3:0	workshop		Jerry Jazz
✓	Completed	20090114-1407	Completed	1/14/2009 2:07	2 minut.	workshop...	1/14/2009 2:1	workshop		Jerry Jazz

Details of Team Concert views surfaced in Visual Studio

View (Tool Window)	Feature	Comments
Solution Explorer	<ul style="list-style-type: none"> 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 	<ul style="list-style-type: none"> 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,
Team Artifacts View	<ul style="list-style-type: none"> Allows developers to explore multiple Team Concert repositories from here. Join multiple projects View streams View repository workspaces Load and unload workspaces and start work Create new workspace Create new stream Create new component View and run pre-defined queries Create work items and queries Create / View Plans Create / View Reports Subscribe to Feeds Request for Builds 	<ul style="list-style-type: none"> Similar to Eclipse Team Artifacts Navigator Has components, streams and baselines unlike VS Team System source control Does not copy workspace contents to perform branches (weakness of VS Team System which does not scale for large workspaces)
Work Items View	<ul style="list-style-type: none"> 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 	Similar to Eclipse Work Items View

Details of Team Concert views surfaced in Visual Studio

View (Tool Window)	Feature	Comments
Pending Changes View	<ul style="list-style-type: none"> Shows status of all changes made by the developer and provides access to advanced scm features. Deliver change sets Attach and resolve work items View and modify change flow of work to streams Create baseline, rollback to another baseline Create snapshots Merge, rollback a change set Compare / merge with previous versions via the default editor or via external Compare Merge tools Filter displayed change sets based on regular expressions 	Similar to Eclipse Pending Changes View
Change Set Explorer	<ul style="list-style-type: none"> Explore a change set and its contents View before and after state of file in the change set Associate work items Remove work items 	Same as Rational Team Concert's Eclipse UI capabilities.
Change Set Search View	<ul style="list-style-type: none"> Developer can search for change sets using search criteria Useful for looking at past deliveries of work 	Same as Rational Team Concert's Eclipse UI capabilities.

View (Tool Window)	Feature	Comments
History View	Shows history of files and components. Compare with previous version, local version or arbitrary version · Can then open in Change Set Explorer view	Similar to Rational Team Concert Eclipse UI
Repository File Browser	Browse repository workspaces, components, files and folders without explicitly downloading them to the users local hard disk Browse and view files selectively.	Same as Rational Team Concert's Eclipse UI capabilities
Builds View	Monitor builds View build results Cancel builds Associate Work Items to build Create a stream /repository workspace from build Compare builds Tag a build	Similar to Rational Team Concert Eclipse UI
Feeds	Browse feeds Edit work item comments in-line via feeds Edit Feeds filters	The edit work items feature is there only in the Visual Studio Client
Search Results View	Displays your repository search results Context specific menu item lets you work with your search results	Similar to Rational Team Concert Eclipse UI

What do I need to get started?

Rational Team Concert client for Visual Studio

• Prerequisites

- ✓ Rational Team Concert v2.0
- ✓ Visual Studio 2005 or 2008
- ✓ Register at www.jazz.net
- ✓ Download from 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](#)
- ▶ [Integrating Visual Studio builds with Team Concert](#)
- ▶ [Mapping your Visual Studio Projects and Solutions to Jazz Components](#)

• Provide Community Feedback

- ▶ [jazz.user forum](#)
- ▶ [Submit bugs or enhancement requests](#)

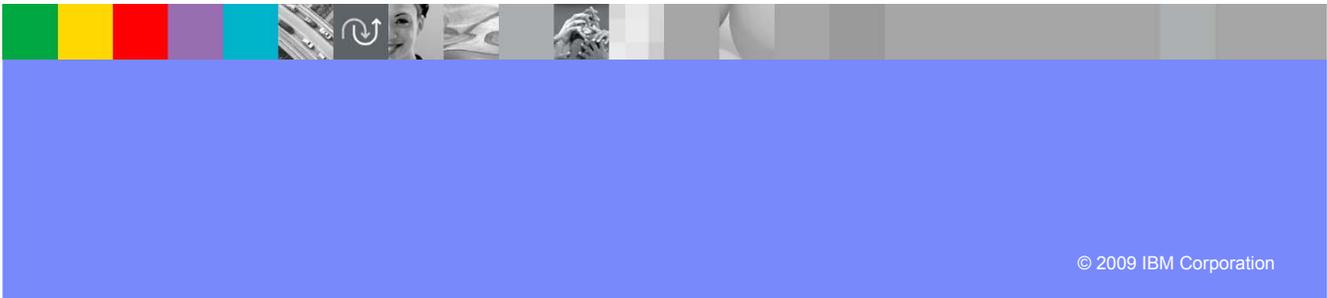




IBM
Software
Group

Lab Overview

An IBM Proof of Technology



© 2009 IBM Corporation

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”. The Dog squawker goes “bark”, the Cat squawker goes “meow”, etc. 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

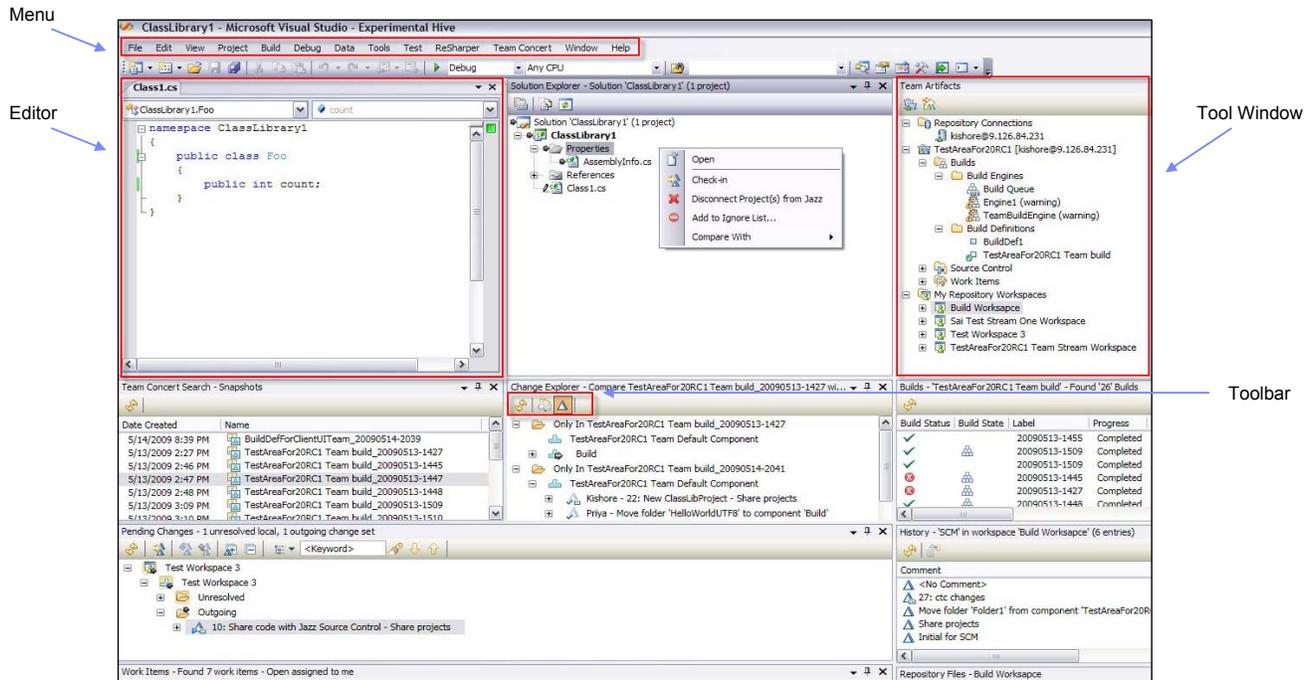
Change Perspective

Current Perspective

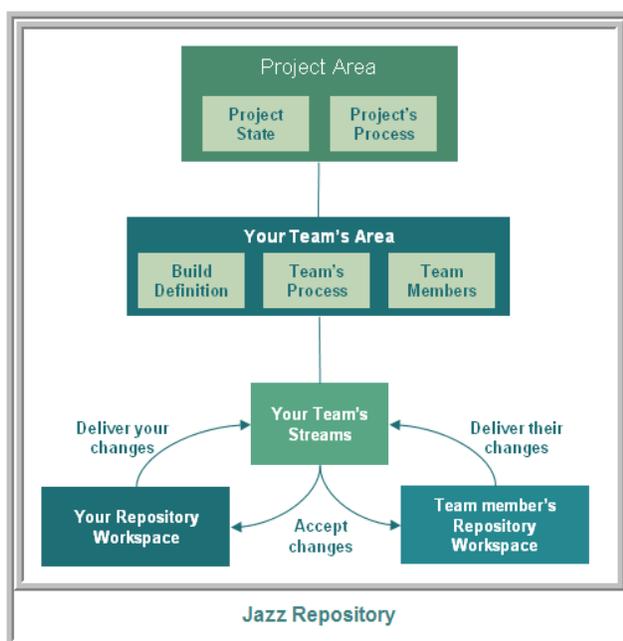
This View has different areas accessed via the View tabs

View tabs

Visual Studio Client



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.

Squawk Project in Team Concert

Project Area

Streams

Repository Workspaces

Team Areas

Process Definition

Squawk Project in Visual Studio Client

Project Area

Builds

Streams

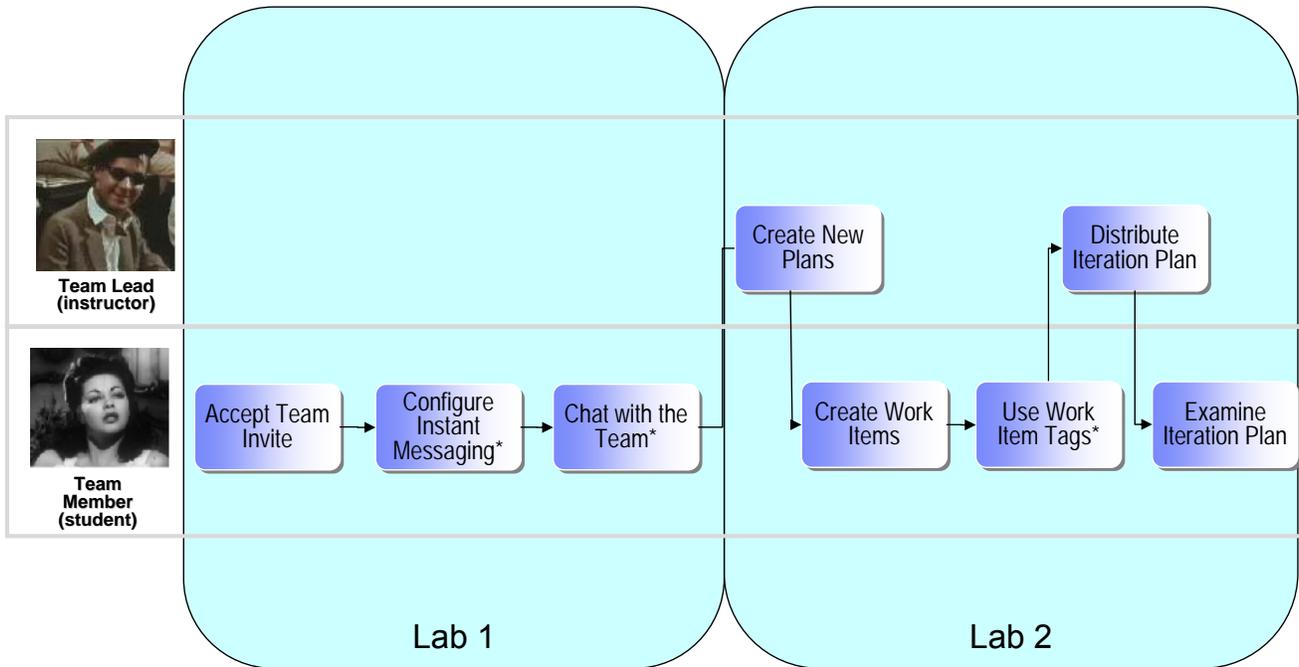
Queries

Repository Workspaces

Build Results

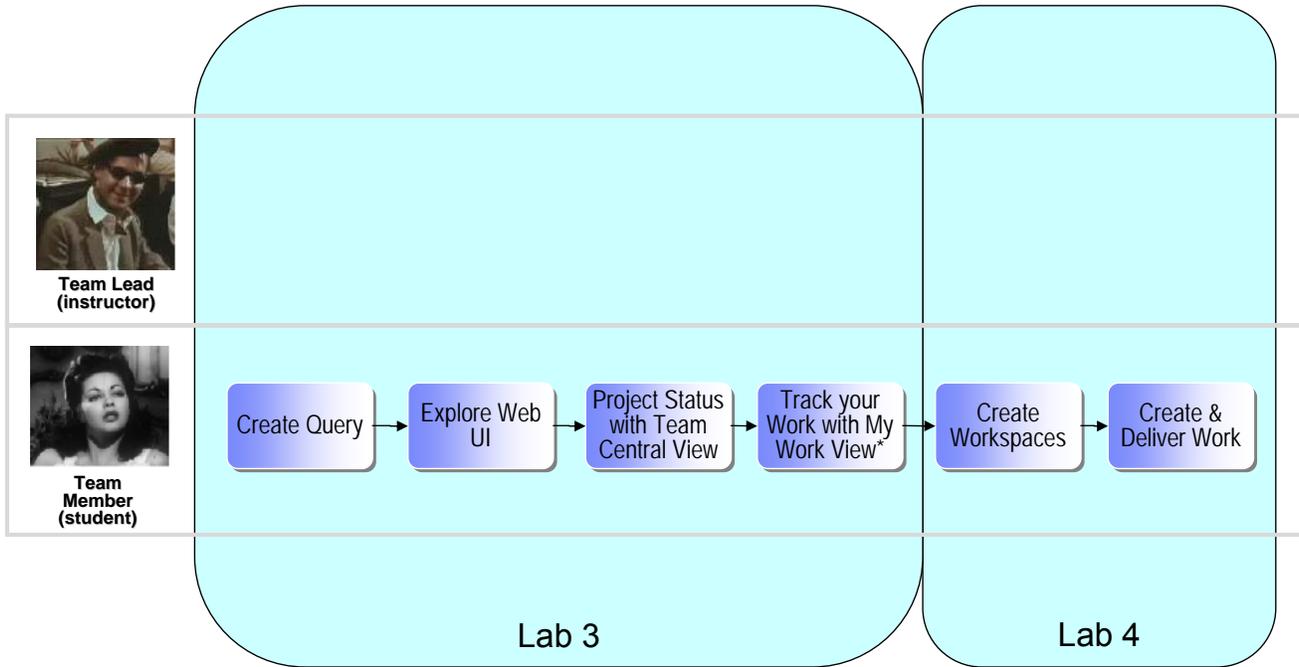
Work Items

Sequence of events – Lab 1 & 2



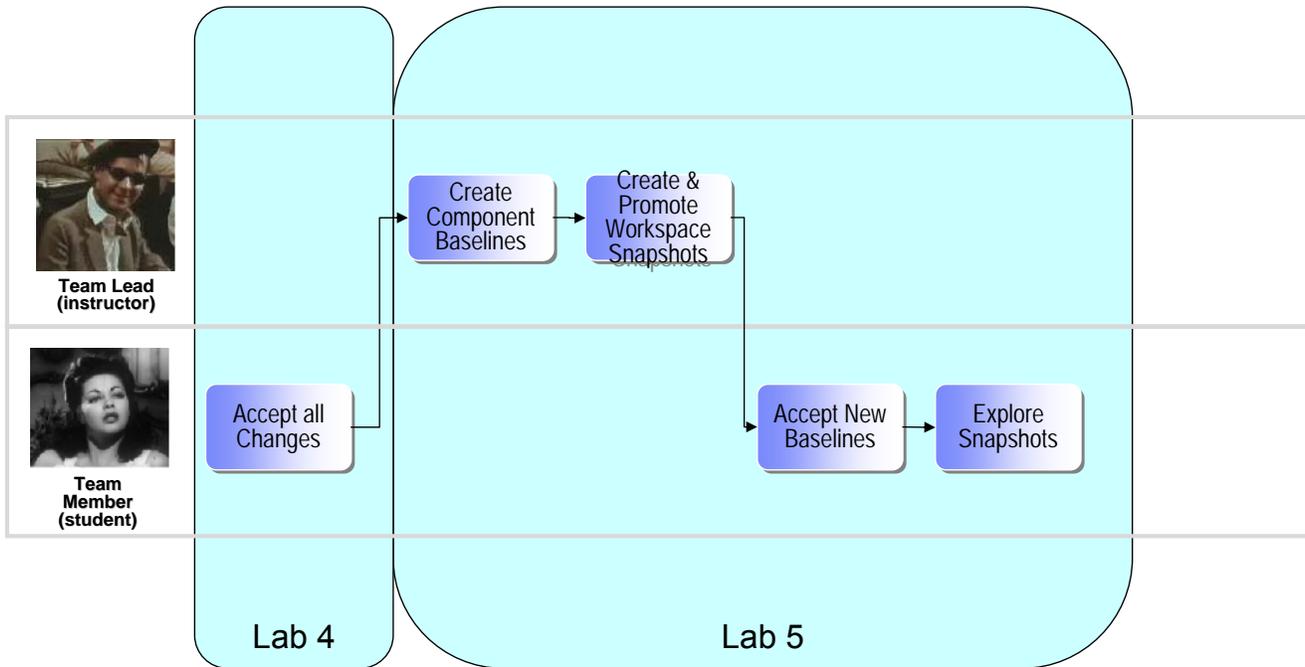
* Not currently available in Visual Studio

Sequence of events – Lab 3 & 4

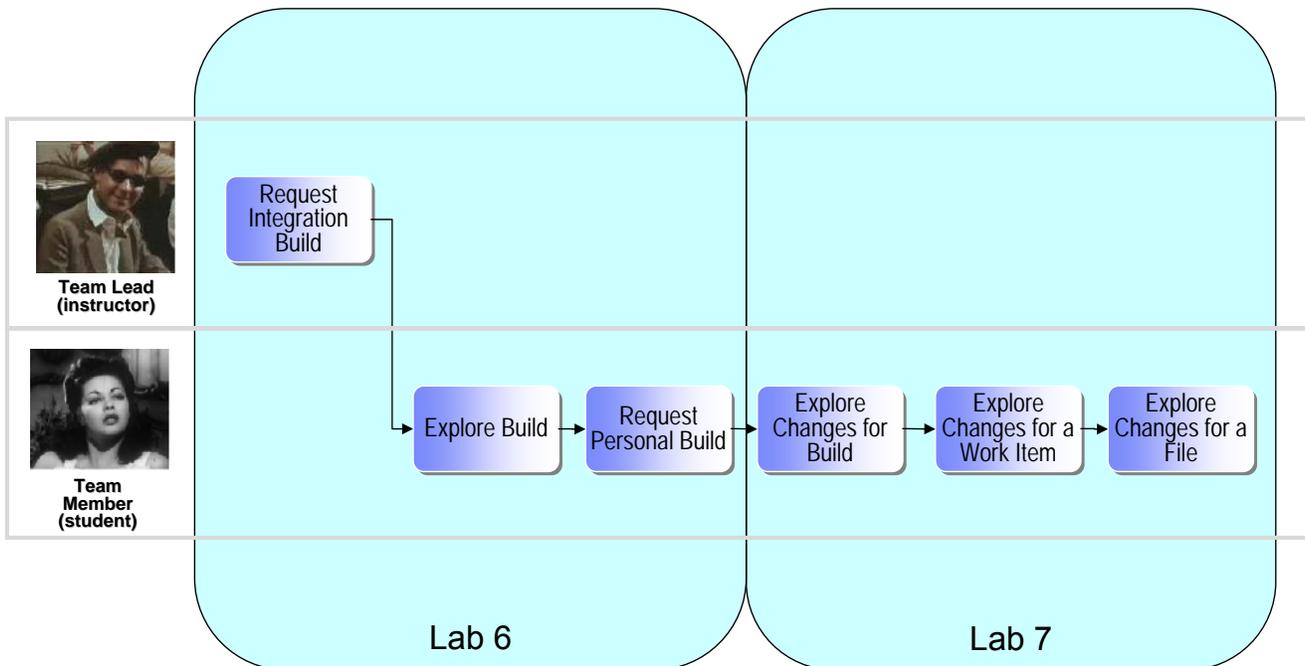


* Not currently available in Visual Studio

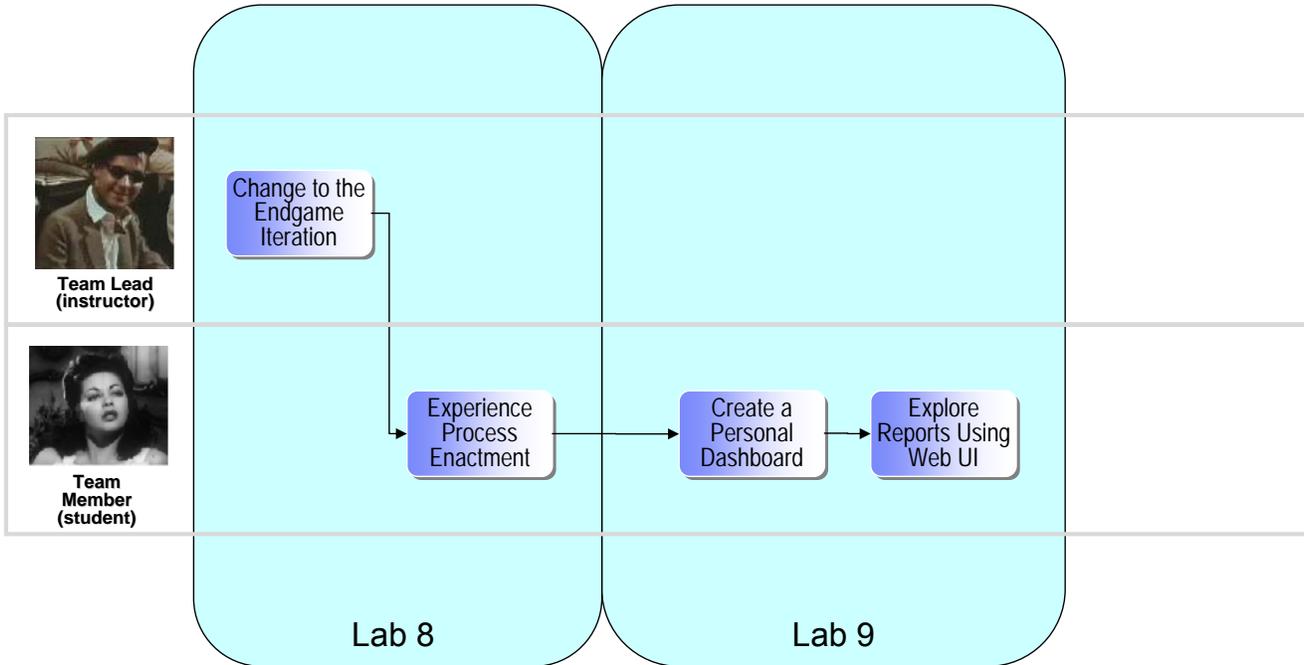
Sequence of events – Lab 4 & 5



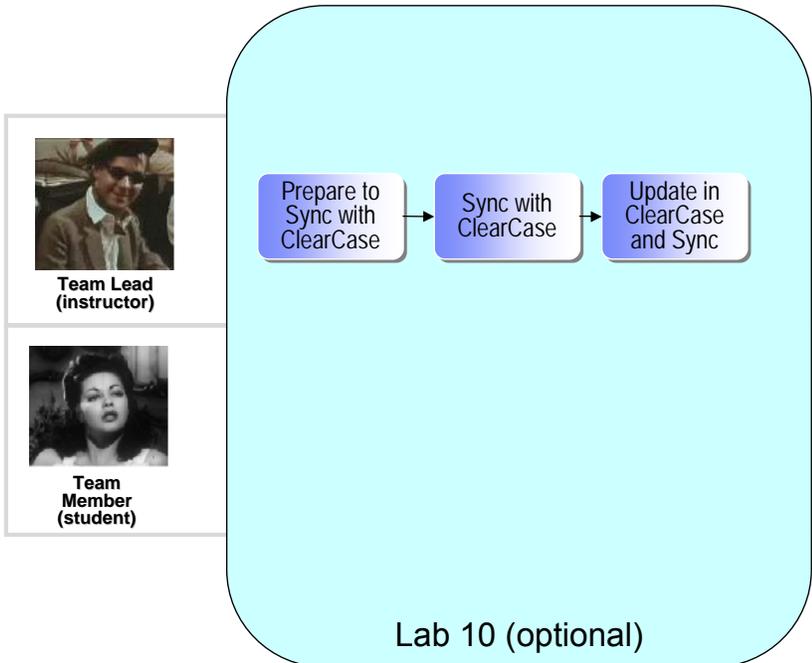
Sequence of events – Lab 6 & 7



Sequence of events – Lab 8 & 9



Sequence of events – Lab 10

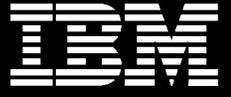


Lab Conventions

- 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, e.g. *student1*
 - ▶ Examples in the lab workbooks use *student1*, you will need to adjust per your assigned id
- Every student creates their 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 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.



IBM
Software
Group

Setting up the Team

An IBM Proof of Technology



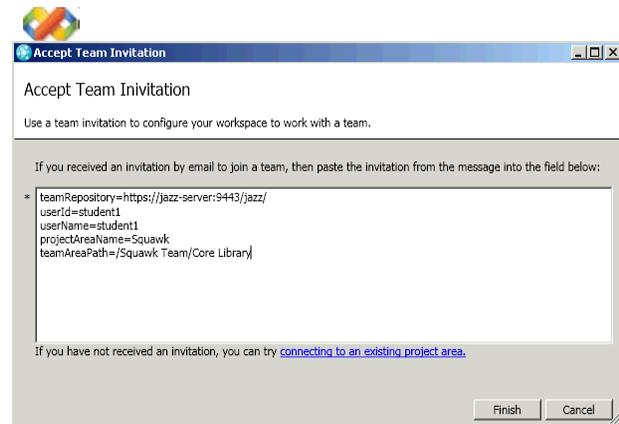
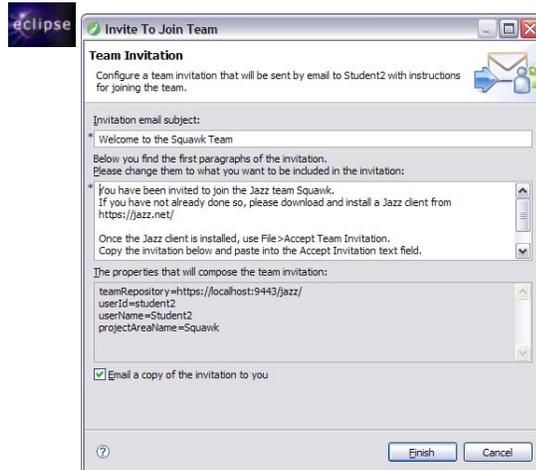
© 2009 IBM Corporation

Objectives

- In this lab you will learn how ramp up projects quickly and dramatically improve onboarding and offboarding of team members
- You will perform some initial setup of 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
 - ▶ E-mail
 - ▶ Instant Messaging/Chat *
 - ▶ 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

* Not currently available in Visual Studio

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 to your colleagues on the project



Lab #1 Overview

- Use the team invitation received via 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



* Not currently available in Visual Studio

Lab #1 Concepts Learned

- Team Invitations make it easier to get team member's 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 *

* Not currently available in Visual Studio



IBM
Software
Group

Planning Your Work

An IBM Proof of Technology



© 2009 IBM Corporation

Objectives

- Understand Rational Team Concert's agile project planning capabilities
- 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

Squawk Teams (7)

- ▼ Squawk Maintenance Team
 - Core Library M1 Maintenance
- ▼ Squawk Team
 - Core Library
 - Documentation
 - Release Engineering
 - User Interface

1.0 M3 3 Aug 2009 - 31 Dec 2009

493	Create futuristic squawkers
470	Support theme based squawk parties
471	Support other types of technologies like audio and video squawkers
484	Predefine template squawker classes
469	Allow new squawkers to be defined from a user interface instead of code

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 project and between different teams
 - ▶ Team Concert will take into account participation in different projects and teams

Squawk Teams (7)

- ▼ Squawk Maintenance Team
 - Core Library M1 Maintenance
- ▼ Squawk Team
 - Core Library
 - Documentation
 - Release Engineering
 - User Interface

Core Library Members (38)

April Blues	contributor
Jerry Jazz	teamlead, contributor
Zara Intern	contributor
student1	contributor
student10	contributor
student11	contributor
student12	contributor
student13	contributor
student14	contributor
student15	contributor

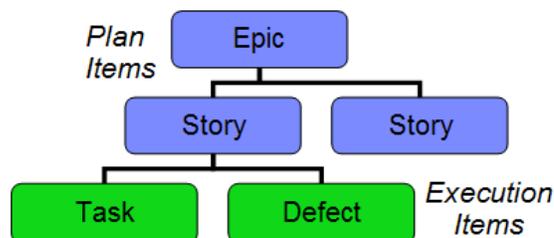
Time

- 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 lifecycle



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 visualise their work

Overall Project Plan

Backlog

Team Area: Squawk Team | Iteration: Release 1 (1/15/08 - 12/31/09) | 12 Closed | 9 Open

Overview | Planned Items | Charts

Progress: 80/141 Story Points | Estimated: 100%

View As: Backlog

	Create today's squawkers	High	5 pts	0/0	494
	Create documentation for today's squawkers.	High	1 pt	0/0	495
	Create futuristic squawkers	High	5 pts	0/0	493
	Create a broader collection of squawkers	High		0/11	466
	Predefine template squawker classes	Medium	5 pts	0/0	484
	Support other types of technologies like audio and video squawkers	Unassigned	20 pts	0/0	471
	Support theme based squawk parties	Unassigned	8 pts	0/0	470
	Allow new squawkers to be defined from a user interface instead of code	Unassigned	12 pts	0/0	469
	Set up a project status web page	Unassigned	5 pts	0/0	468

Team Project Plan

Team's Project Plan

Team Area: Core Library | Iteration: Release 1 (15/01/2008 - 31/12/2009) | 2 Closed | 0 Open

Overview | **Planned Items** | Charts

Progress: 2/2 Story Points | Estimated: --

View As: Iterations

Release 1 Closed Items: 2 Open Items: 0		Progress: 2/2 Story Points		Estimated: --	
1.0 M1 Closed Items: 0 Open Items: 0		No Work		Estimated: --	
1.0 M2 Closed Items: 2 Open Items: 0		Progress: 2/2 Story Points		Estimated: --	
<	▼ Create a .net implementation of the Squawker application	20 pts	2/2	Medium	482
0	▼ Define a Jazz build definition that runs MS Build	1 pt	0/0	Unassigned	488
0	▼ C# implementation	1 pt	0/0	Unassigned	487
1.0 M3 Closed Items: 0 Open Items: 0		No Work		Estimated: --	

Team Iteration Plan

Core for M1

Team Area: Core Library | Iteration: 1.0 M1 (15/01/2008 - 24/01/2008) | 5 Closed | 0 Open

Overview | **Planned Items** | Charts | Release Notes | Retrospective

Progress: 14/14 | 0 h | Estimated: --

View As: Work Breakdown

April Blues Closed Items: 2 Open Items: 0		Load: 0/0		No Work Time Left		Estimated: --	
<	▼ Define a handful of example squawkers			Medium	6/6 h		463
	▼ Create Car squawker	2 hours		Medium			43
	▼ Create Lion squawker	2 hours		Medium			42
Jerry Jazz Closed Items: 3 Open Items: 0		Load: 0/0		No Work Time Left		Estimated: --	
<	▼ Create core application based on initial prototype			High	8/8 h		464
	▼ Core Code	1 day		Medium			7
<	▼ Define a handful of example squawkers			Medium	6/6 h		463
	▼ Cat Implementation	1 hour		Medium			9
	▼ Dog Implementation	1 hour		Medium			8

Schedule Risk Assessment Plan

More detailed developer estimation.. low, nominal, high

Color codes high risk tasks for quick identification and action

Automatically calculates probability of task fitting into the schedule

Task Description	Risk Level	Duration	Estimate	Probability	
Link for enhanced hours in Work Item web	High	2 hours	1.0	71.777	
Area functionality on the service layer	High	3 days	1.0	77066	
of namespaces, no prefixes and media	High	2 hours	1.0	77921	
udent tests.URServiceTest.testCommentU	High	2 days	1.0	76899	
This custom attributes at the same time	High	2 hours	1.0	54771	
Include the of each attribute in the ResultRowDTO	High	2 hours	1.0	76854	
Workflow on Resource is not updated on project area	Medium	2 hours	1.0	77291	
Align ROA collection with OSLCs atom feed RDF Bags	Medium	2 days	1.0	69970	
Consider to surface links as attributes in the	High	4 hours	3 days	1.0	70731
WorkItem REST API shall also expose the 'Included in	High	1 day	2 days	0.995	73221
Preserve Mentioned By links when their target is protected	Medium	2 hours	0.994	69970	
OSLC REST cleanup	High	3 hours	3 days	0.746	77322
Check enforcement of permissions for Web UI bulk operation	Medium	4 hours	1 day	0.629	77096
Color migration problem	Medium	4 hours	1 hour	0.605	68972
PUT/POST	Medium	4 hours	0.486	64246	
reference	Medium	2 days	0.109	64247	
work item in it	Unassigned	4 hours	0.109	76150	
message returned to the Web UI	Medium	2 hours	4 hours	0.083	74800
latest search was added by	Unassigned	4 hours	0.057	74767	

Developer's Taskboard

See the work currently in progress

Drag and drop work items to change their state.

To Do	In Progress	Done
<ul style="list-style-type: none"> Improve documentation for 4.4 (55) Provide improved Assertion syntax (60) 	<ul style="list-style-type: none"> javadoc updates for @ignore in 4.3 (30) Based on the assertThat syntax we should provide assumptions and theories support (59) assertArrayEquals misses differences (7) testCount hard-coded to 1 for childless Description (27) Tests on protected methods fail (14) assertThat fails with Class tests (documentation problem) (10) 	<ul style="list-style-type: none"> [Docs] Cookbook TestRunner section incorrect (23) shows green bar while assert false (41) Should not call derived's afters if super's before failed (17) @After method not called after my test timeout in 4.3.1 (16)

Plan Modes

- Plans can be displayed in different modes
- **View As: Backlog** is ideal for managing SCRUM backlogs
 - ▶ Support coarse & fine grained prioritization
 - ▶ Ranking is reflected in all planning views, e.g. iteration plans and release plans

JUnit Product Backlog

Team Area: JUnit Team | Iteration: 4.4 | 0 Closed | 4 Open

Overview | Planned Items | Charts | Retrospective

Progress: 0/8 Story Points Estimated: 100%

View As: Backlog

As a user I want to re-run only failed tests	8 pts	0/0	High	62
Provide improved Assertion syntax	0 pts	0/0	High	60
Improve documentation for 4.4	0 pts	0/0	Medium	55
No download link to latest version			Medium	12

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

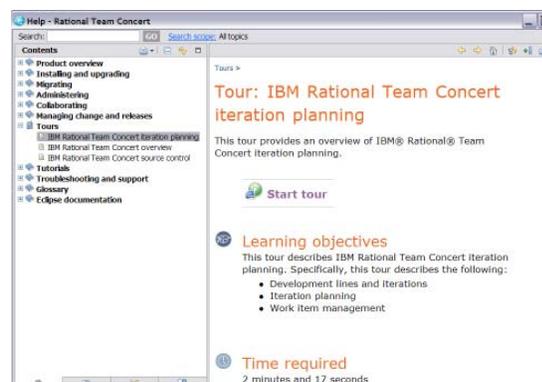
- The instructor will create new Stories to create additional squawkers.
- You will create Task work items for your squawkers
- You will assign your work to the right team member (you!)
- You will set the priority for your Task work items and estimate how long they will take to complete
- You will link 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 via Chat and you will examine the plan. *



* Not currently available in Visual Studio

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**

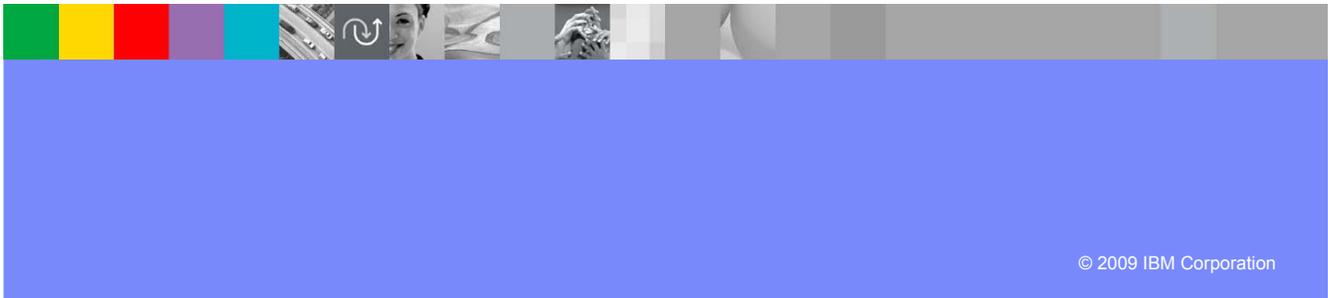




IBM
Software
Group

Keep Track of All Our Work

An IBM Proof of Technology



© 2009 IBM Corporation

Objectives

- Explore Rational Team Concert query capabilities
- Create and run queries
- Use and configure the Team Central* 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.

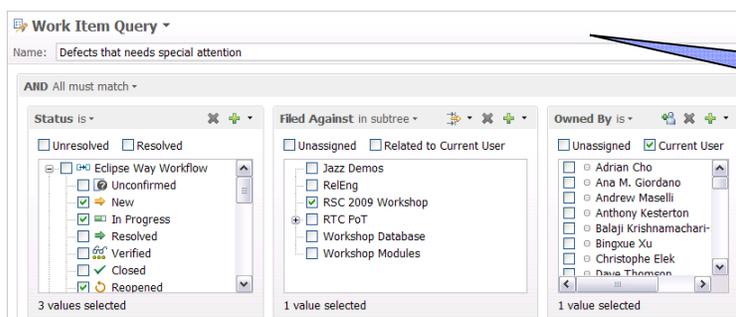
Real time collaboration

- Utilize 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

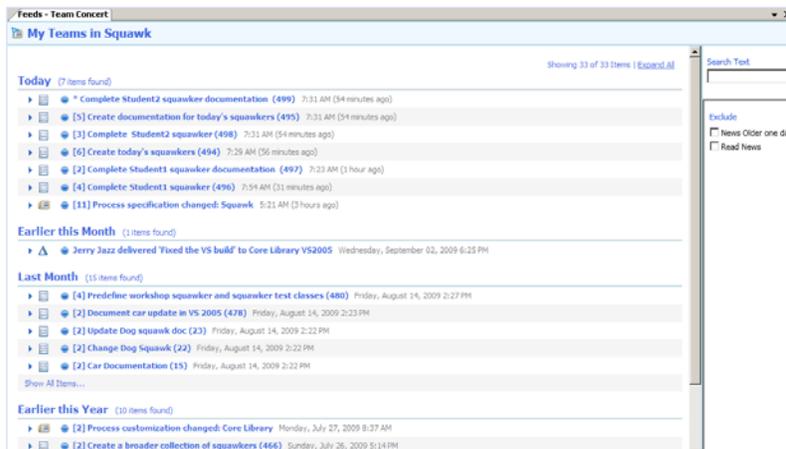
- Provides real-time project health information and transparency of status through automated data gathering.
- Rational Team Concert provides a query mechanism to find work items in a Project Area allowing for more project transparency.
 - ▶ The query scope for work items is the project area.
- The user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



Wizard to help you create powerful queries

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 Central 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



IBM
Software
Group

Performing and Sharing Your Work

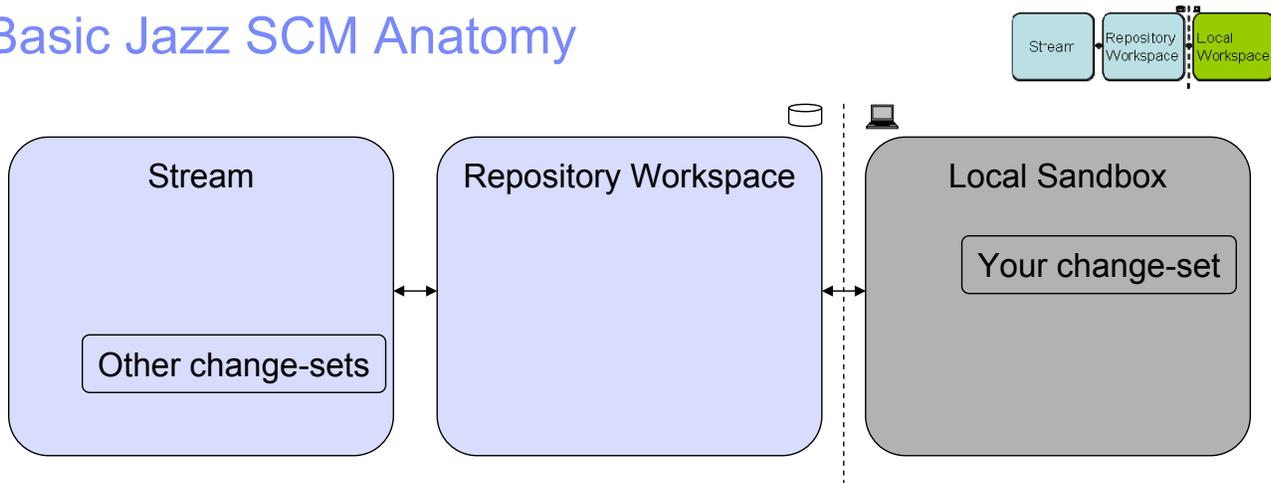
An IBM Proof of Technology



Objectives

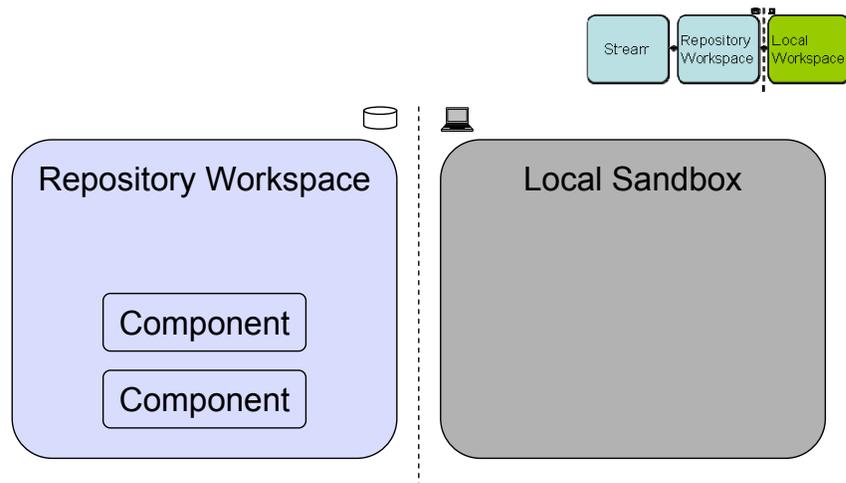
- Understand Software Configuration Management (SCM) concepts in Rational Team Concert
- Create and use a Repository Workspace for work assigned to you
- Create or make changes to artifacts under source control
- Associate changes with Work Items
- Deliver changes from Repository Workspaces to Streams
- Accept changes from other members of your team
- Understand conflict resolution

Basic Jazz SCM Anatomy



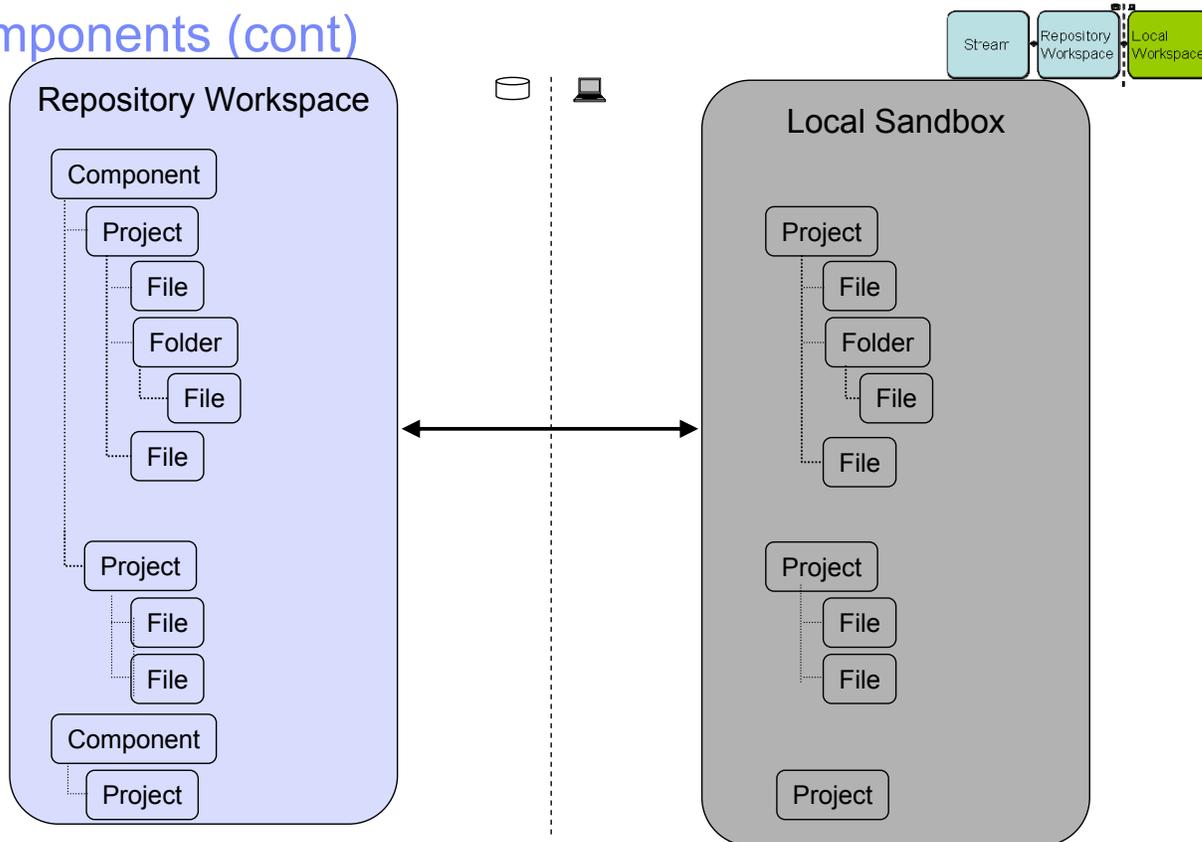
- Streams are for sharing
- Repository workspaces are your personal space
- The local sandbox is a folder on your local files system where you develop and test
- Change-sets flow back and forth

Components

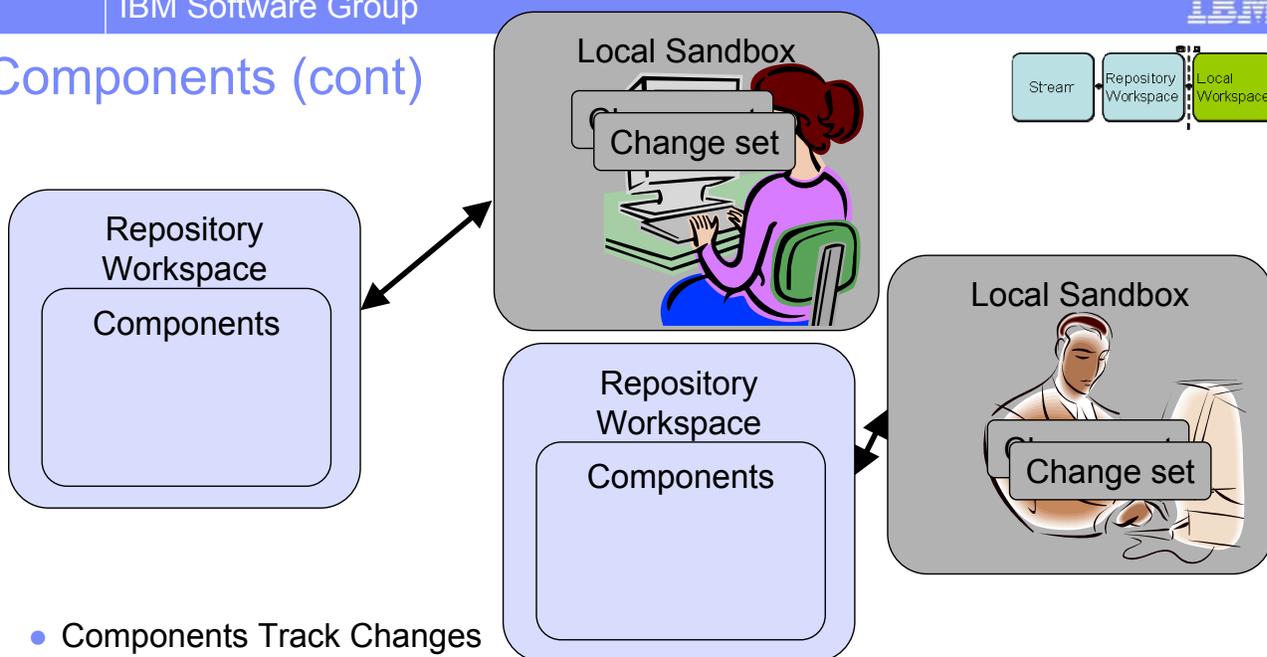


- Repository Workspaces
 - ▶ Partitioned into components
 - ▶ Jazz understands the structure of your components
 - ▶ Jazz directly supports component based development

Components (cont)



Components (cont)

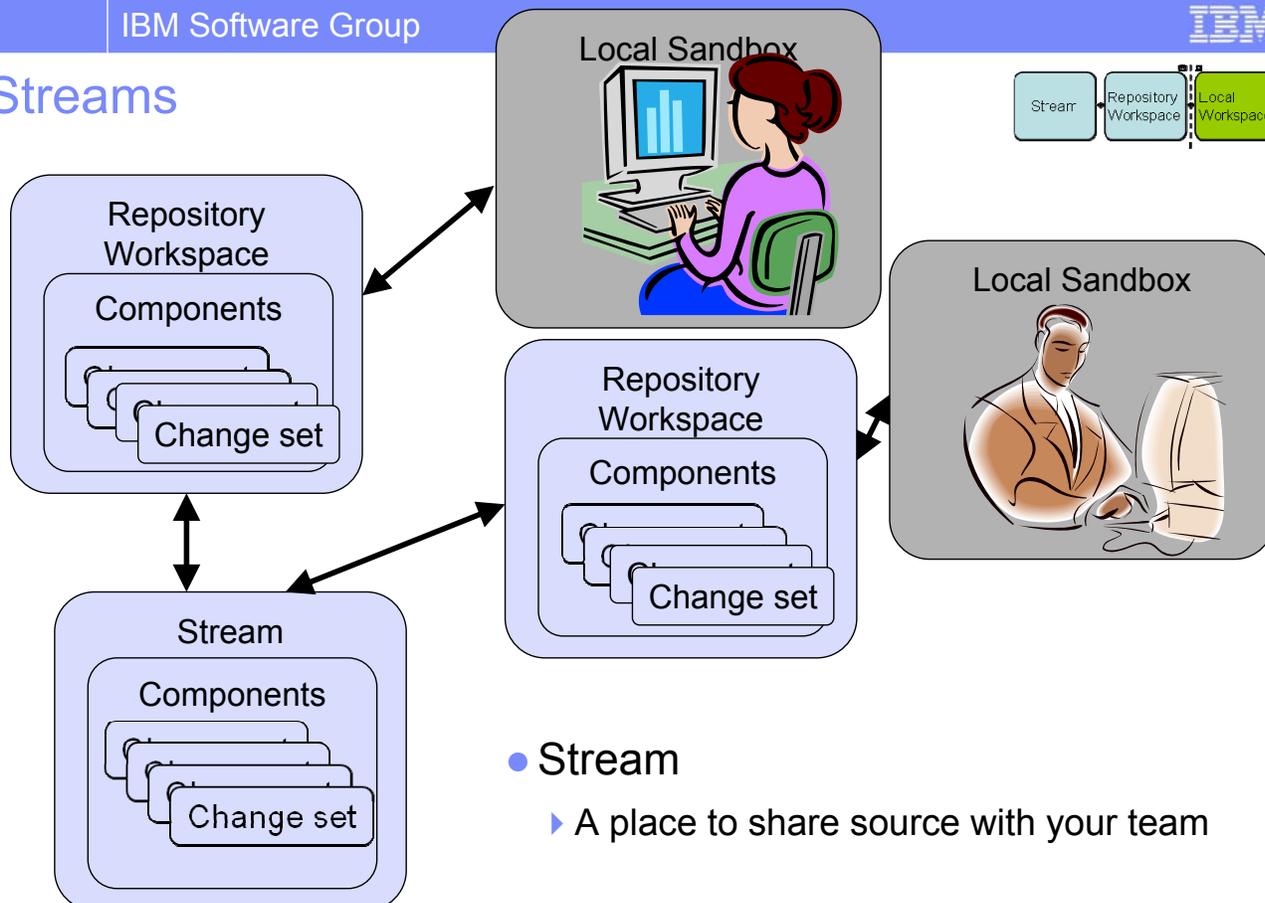


- **Components Track Changes**
 - ▶ Configuration of resources builds from the change set flow
 - ▶ Each change set builds on what came before
- **Component's Change History**
 - ▶ A time-ordered sequence of change sets
 - ▶ Describes how the component's content was built from nothing

Change set Details

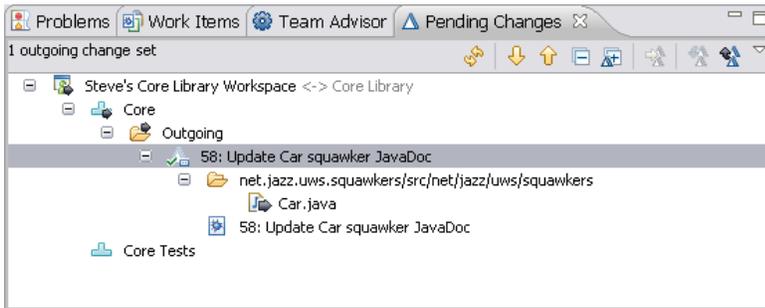
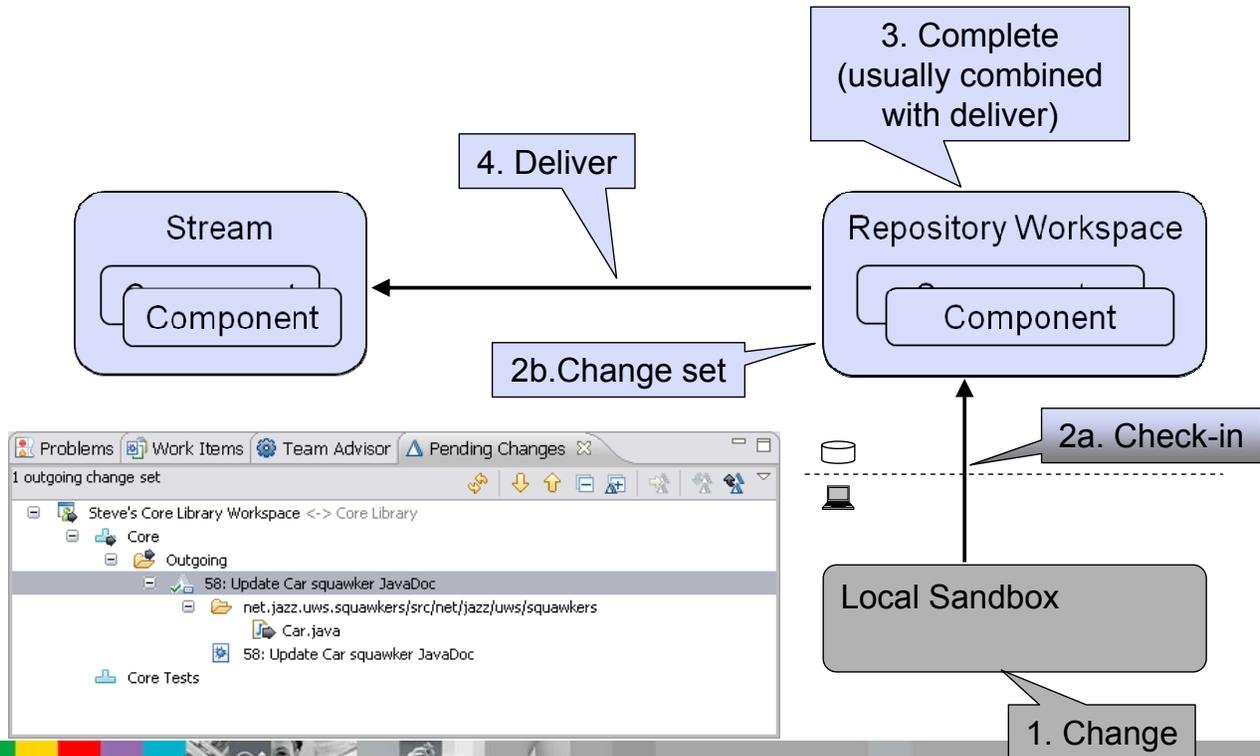
- Composed from a collection of changes to one or more files and folders
 - ▶ A change set that affects multiple resources is committed as a single atomic unit
- Indicates the reason for the changes
 - ▶ Via a comment, and/or
 - ▶ By referencing the relevant work item
- Can be shared with another team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

Streams



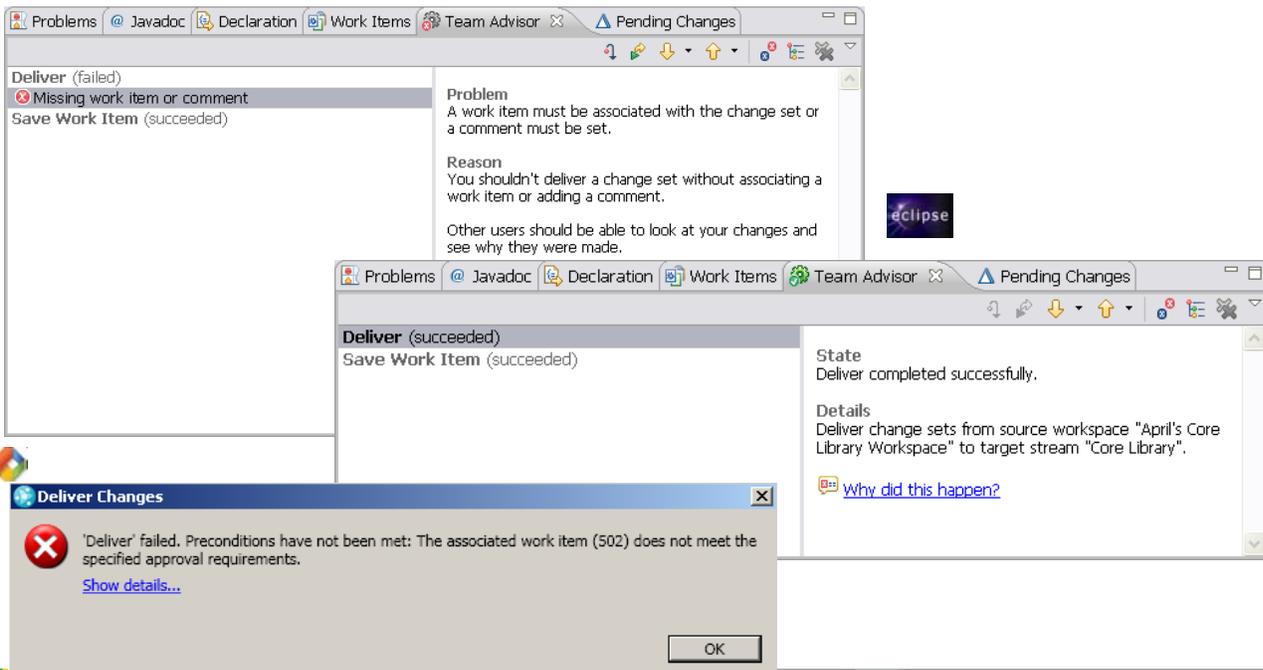
- Stream
 - ▶ A place to share source with your team

Typical Journey For A Change set

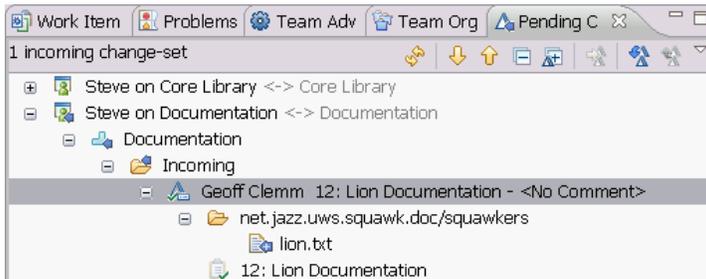
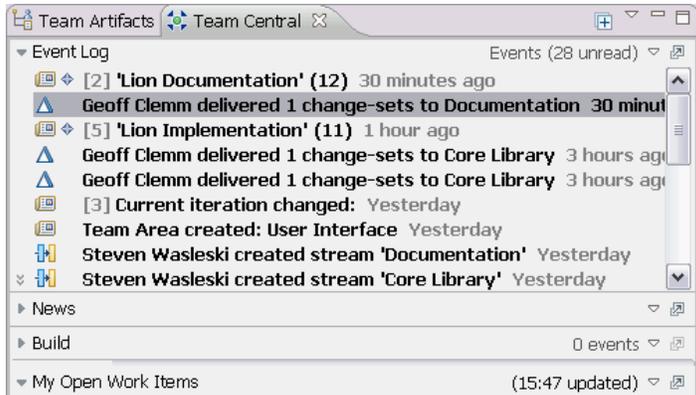


Change-set Delivery is Process Enabled

- The deliver operation is process-enabled, allowing the team's process to check and enforce delivery rules automatically



Delivery Notifications

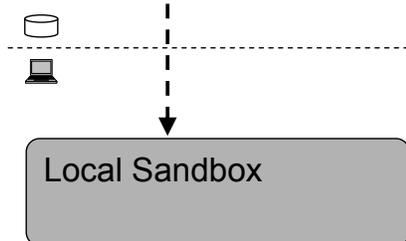
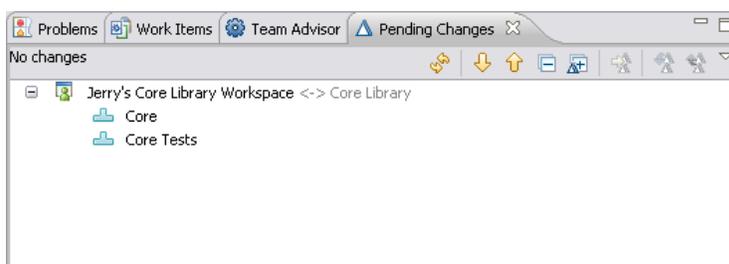
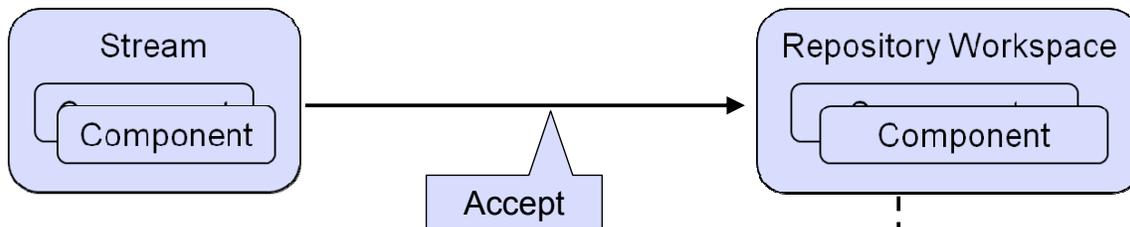


* Not currently available in Visual Studio

Getting Teammates' Delivered Work



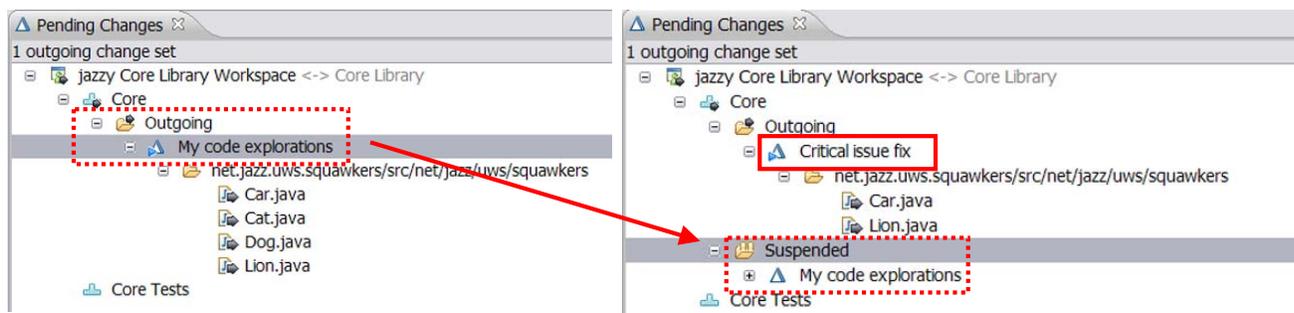
- An incoming change-set is
 - ▶ In the change history of the stream, but
 - ▶ Not in the change history of your repository workspace
- Accept adds the change-set to your repository workspace's change history



Suspending Your Changes

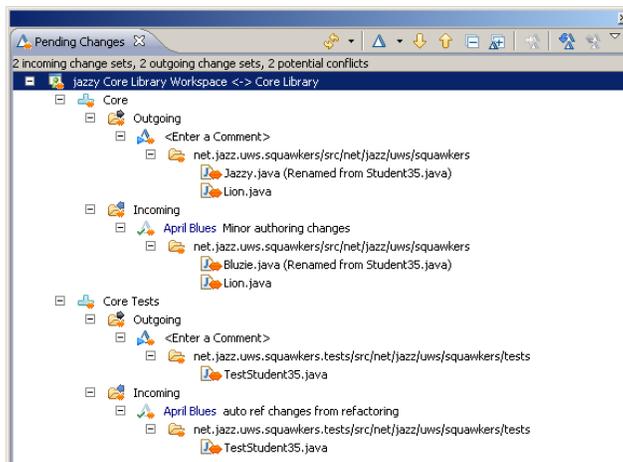
There are times when your development work can't move forward...

- 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!!!**



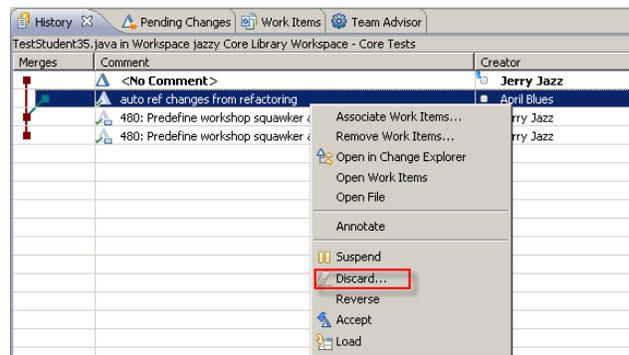
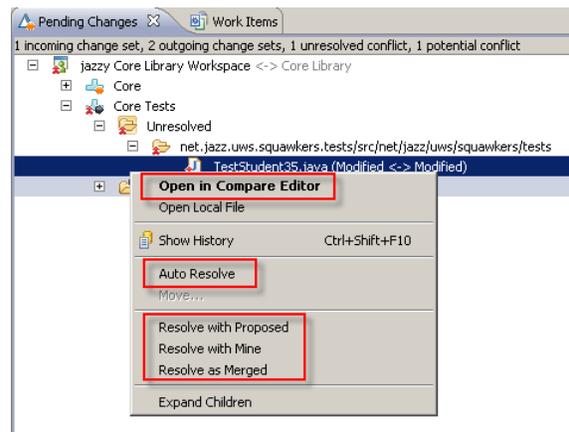
Conflicts in Jazz SCM

- 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



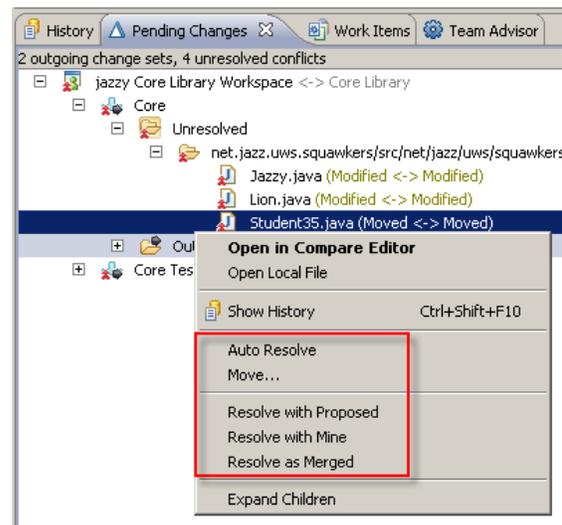
Resolving Content Conflicts

- **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

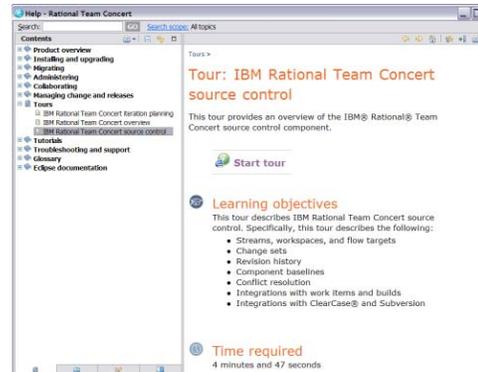
- You have been tasked with contributing your own squawker class along with its documentation and, optionally, its test case

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 lifecycle
- 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**



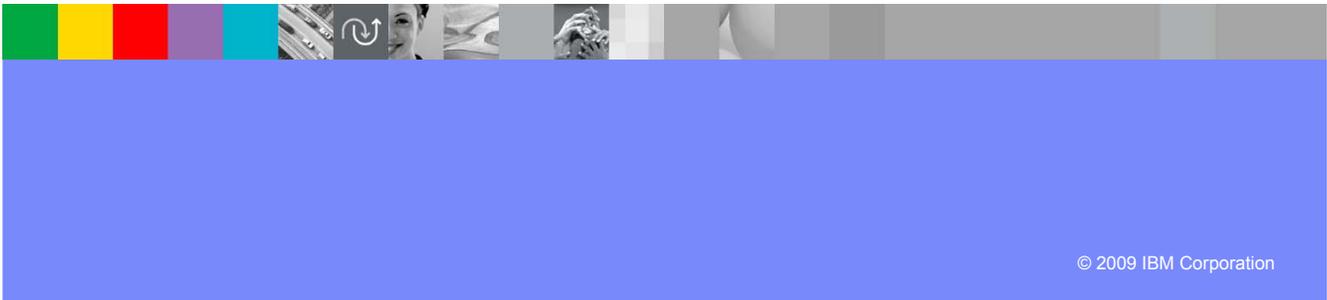
Questions



IBM
Software
Group

Remembering Well Known SCM Configurations

An IBM Proof of Technology



© 2009 IBM Corporation

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

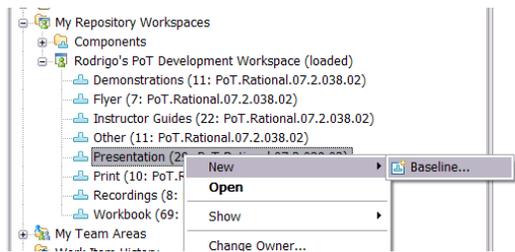
What About These Questions?

- 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?

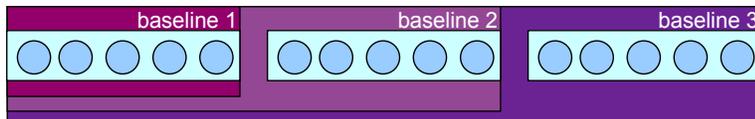
- Use baselines and snapshots...

A Baseline

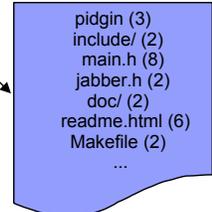
- Is an immutable copy of a component's configuration
 - ▶ At a particular point in time, and
 - ▶ There can be multiple baselines of a component
- Serves as a fixed point of reference
 - ▶ For initializing streams and repository workspaces
 - ▶ For sharing source with people or processes
- Can be easily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



Change history



Configuration



A Snapshot

- Is 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
 - ▶ Re-create a prior build via 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

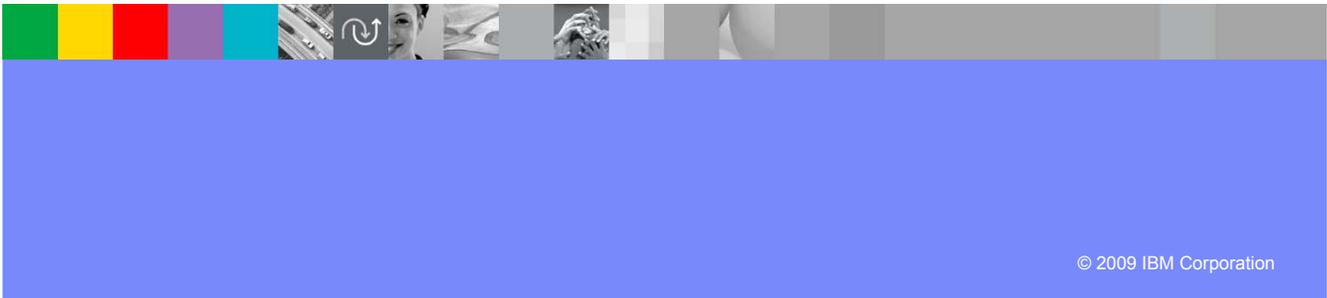
- **Baseline** and **snapshot** artifacts increase traceability and enable collaboration among teams and team members
- **Baselines** are an efficient means to mark artifacts within a single component for later reference
- **Snapshots** are an efficient means to mark artifacts across a set of related 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



IBM
Software
Group

User's View of Build

An IBM Proof of Technology

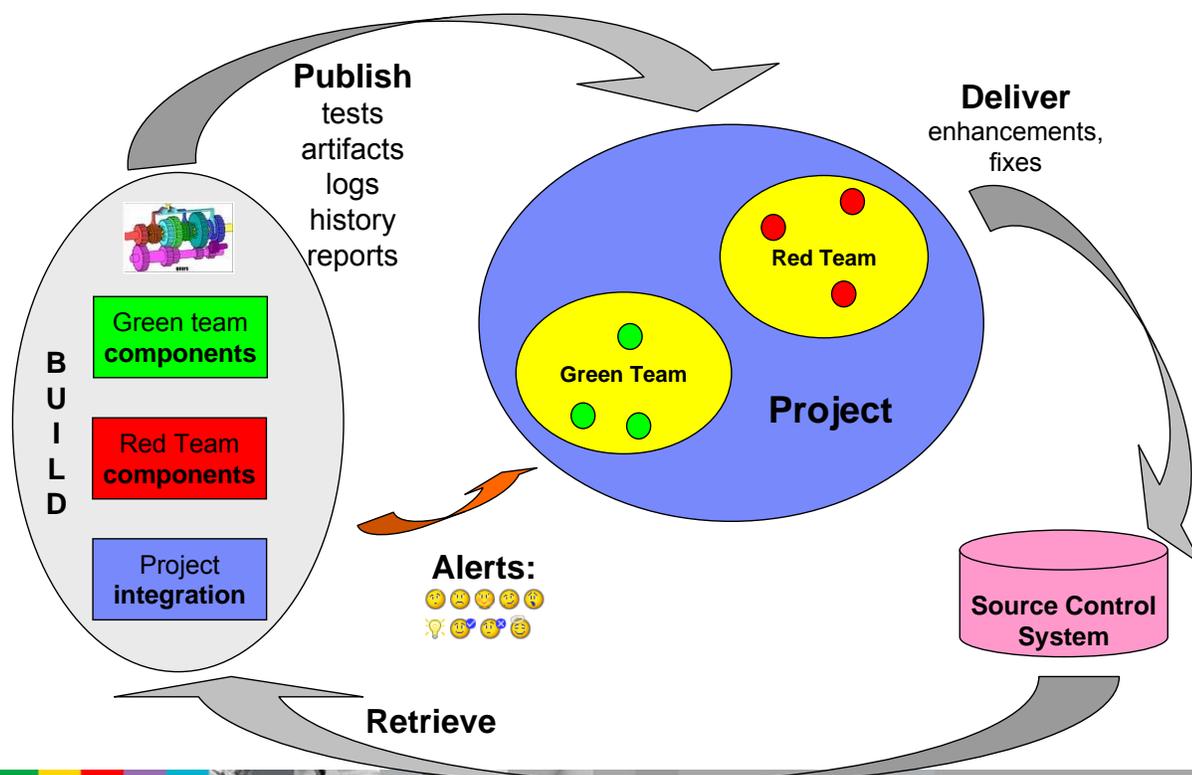


© 2009 IBM Corporation

Objectives

- Understand the build functionality of 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

Build in the World of Agile Team Development



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, ...)
- ▶ Leverages technology that fits your project best

Build is very visible to the user

The screenshot shows the IBM Team Central interface for a build. The main area displays details for a completed build: 'Build workshop.squawk.core.continuous.build B20080407-1554-workshop.squawk.core.continuous.build'. It shows a status of 'Completed' with a duration of 23 seconds, completed on April 7, 2008. A 'Status Trend' bar shows a series of green bars. Below this is a 'Contribution Summary' with links for Downloads, External Links, Logs, Repository Workspace, Snapshot, Compile (0 errors, 0 warnings), JUnit (4 tests, 0 failures, 0 errors), Work Items (1 included in build), and Changes (Show changes). A 'General Information' section lists the Requested by (Zach Builder), Build Definition, Build Engine, Build History (16 builds), and Tags. On the right, there are sections for 'Recent builds' and 'Alerts', with an alert for 'April Blues delivered'. At the bottom, a 'History' table lists several completed builds with their labels, progress, and completion times.

Build	Label	Progress	Time	Duration
workshop.squawk.core.continuous.b...	B20080407-155...	Completed	April 7, 2008 6:53:51...	23 sec
workshop.squawk.core.continuous.b...	B20080314-150...	Completed	March 14, 2008 3:07:...	28 sec
workshop.squawk.core.continuous.b...	B20080313-133...	Completed	March 13, 2008 4:37:...	12 sec
workshop.squawk.core.continuous.b...	B20080313-104...	Completed	March 13, 2008 1:45:...	38 sec
workshop.squawk.core.continuous.b...	B20080307-145...	Completed	March 7, 2008 5:59:1...	11 sec

Build in Visual Studio Client

The screenshot shows the Visual Studio Team Client interface. The main area displays details for a completed build: 'Build workshop.squawk.core.continuous.vs2005 20090902-1825'. It shows a status of 'Successful' with a duration of 55 seconds, completed on Sep 2, 2009. A 'Published build' callout points to the build status. Below this is a 'General Information' section with details like Requested by (Jerry Jazz), Build Definition, Build Engine, and Build History (13 builds). A 'Contribution Summary' shows 3 downloads, 1 external link, and 3 logs. On the right, there is a 'Team Artifacts' pane showing a tree view of build artifacts, with a 'My builds' callout. At the bottom, a 'Build Events for My Teams' section shows a list of events, with a 'Build events' callout. A 'History' table at the very bottom lists several completed builds with their labels, progress, and completion times.

Build Status	Build State	Label	Progress	Start Time	Completion Time	Build Engine	Tags	Requester
✓	Completed	20090902-1825	Completed	9/2/2009 6:26	9/2/2009 6:26	workshop...		Jerry Jazz
✓	Completed	20090902-1822	Completed	9/2/2009 6:22	9/2/2009 6:22	workshop...		Jerry Jazz
✓	Completed	20090902-1815	Completed	9/2/2009 6:15	9/2/2009 6:15	workshop...		Jerry Jazz
✓	Completed	2009129-1811	Completed	1/29/2009 6:11	1/29/2009 6:11	workshop...		Jerry Jazz

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.

Eclipse Client

The screenshot shows the Eclipse IDE interface. On the left, the 'Team Artifacts' view shows a tree structure with 'Builds' selected. A context menu is open over the 'Builds' folder, with 'Request Build...' highlighted. In the center, the 'Request Build' dialog is open. It contains the following fields and options:

- Build:** workshop.squawk.core.continuous.build
- Build Options:**
 - Personal Build
- Repository workspace:** April's Core Library Workspace
- Component load rules:** (empty)
- Build Properties:**
 - Show the Builds view after submitting the request

At the bottom of the dialog are 'Submit' and 'Cancel' buttons. On the right, the 'Builds' view is visible, showing a list of build entries with columns for Build, Label, Progress, Start Time, Duration, and Completion Time. A red circle highlights the 'Personal Build' checkbox in the dialog, and a red arrow points from it to the 'Builds' view.

Personal Builds in Visual Studio and Web Client

Visual Studio Client

The screenshot shows the Visual Studio IDE interface. On the left, the 'Team Artifacts' view shows a tree structure with 'Builds' selected. A context menu is open over the 'Builds' folder, with 'Request Build...' highlighted. In the center, the 'Request Build' dialog is open. It contains the following fields and options:

- Build:** workshop.squawk.core.continuous.vs2005
- Build Options:**
 - Personal Build
- Repository workspace:** student1 Core Library VS2005 Workspace

At the bottom of the dialog are 'Submit' and 'Cancel' buttons. On the right, the 'Builds' view is visible, showing a table of build entries:

Build Status	Build State	Label	Progress	Start Time	Duration	Build	Completion Time
✓	Completed	20090902-1825	Completed	9/2/2009 6:25	54 sec...	workshop...	9/2/2009 6:26
✓	Completed	20090902-1822	Completed	9/2/2009 6:22	47 sec...	workshop...	9/2/2009 6:22
✓	Completed	20090902-1815	Completed	9/2/2009 6:15	54 sec...	workshop...	9/2/2009 6:16
✓	Completed	20090129-1811	Completed	1/29/2009 6:11	3 minut...	workshop...	1/29/2009 6:1...
✓	Completed	20090114-1423	Completed	1/14/2009 2:23	2 minut...	workshop...	1/14/2009 2:2...
✓	Completed	20090213-0721	Completed	2/13/2009 7:21	32 sec...	workshop...	2/13/2009 7:2...
✓	Completed	20090114-1437	Completed	1/14/2009 2:37	3 secon...	workshop...	1/14/2009 2:3...
✓	Completed	20090114-1505	Completed	1/14/2009 3:05	2 minut...	workshop...	1/14/2009 3:0...
✓	Completed	20090114-1407	Completed	1/14/2009 2:07	2 minut...	workshop...	1/14/2009 2:1...
✓	Completed	20090115-1852	Completed	1/15/2009 6:52	3 minut...	workshop...	1/15/2009 6:5...
✓	Completed	20090114-1448	Completed	1/14/2008 2:48	2 minut...	workshop...	1/14/2008 2:5...
✓	Completed	20090114-1359	Completed	1/14/2009 1:59	3 minut...	workshop...	1/14/2009 2:0...
✓	Completed	20090114-1443	Completed	1/14/2009 2:43	3 secon...	workshop...	1/14/2009 2:4...

A red circle highlights the 'Personal Build' checkbox in the dialog, and a red arrow points from it to the 'Builds' view.

Web Client

The screenshot shows the Web Client interface. On the left, the 'Builds for workshop.squawk.core.continuous.vs2005' view shows a table of build entries. In the center, the 'Request Build for workshop.squawk.core.continuous.vs2005' dialog is open. It contains the following fields and options:

- Build Options:**
 - Personal Build
- Repository workspace:** student1 Core Library VS2005 Workspace

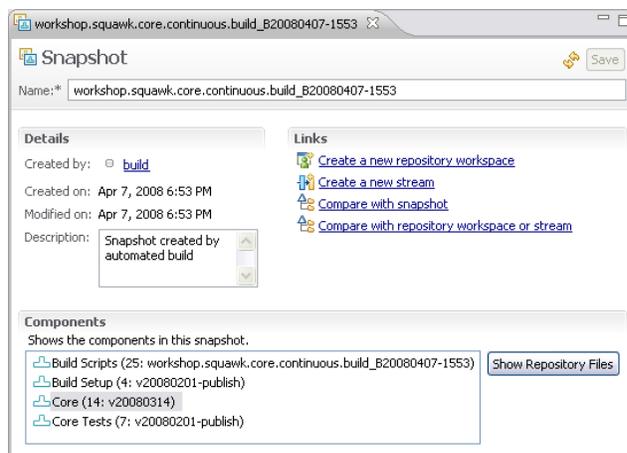
At the bottom of the dialog are 'Submit' and 'Cancel' buttons. On the right, the 'Builds for workshop.squawk.core.continuous.vs2005' view is visible, showing a table of build entries:

Label	Progress	Tags	Start Time	Duration	Estimated Completion
20091028-1321	Completed		Oct 28, 2009 1:31:25 p.m.	1 minute, 58 seconds	
20090902-1825	Completed		Sep 2, 2009 6:25:55 p.m.	55 seconds	
20090902-1822	Completed		Sep 2, 2009 6:22:11 p.m.	48 seconds	
20090902-1815	Completed		Sep 2, 2009 6:15:07 p.m.	54 seconds	
20090213-0721	Completed		Feb 13, 2009 7:21:16 a.m.	32 seconds	
20090129-1811	Completed		Jan 29, 2009 6:11:58 p.m.	3 minutes, 7 seconds	

A red circle highlights the 'Personal Build' checkbox in the dialog, and a red arrow points from it to the 'Builds' view.

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 Definitions > com.example.integration >

Submit

Request Build

Build: com.example.integration

Build Options

 Personal Build

Repository Workspace:* My workspace

Browse

Build Properties

Name: myProperty Value: myValue

Remove

Description:

Add

Submit

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 (Ant, CruiseControl , Build Forge, Maven, ...) makes it easy to adapt to needs of different projects.



IBM
Software
Group

Exploring Changes and Traceability

An IBM Proof of Technology



© 2009 IBM Corporation

Objectives

- This lab will demonstrate how information is linked within 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

Keeps traceability with work items, change set, repository workspace, etc.

ID	Status	P	S	Summary	Owned By	Created By
35	Resolved			Ignore	Zach Builder	Jerry Jazz
32	Resolved			Problem with build B20080306-1029-squawk.ml.maint...	Jerry Jazz	Jerry Jazz
31	Resolved			Clean up Dog squawker	Zara Intern	Jerry Jazz

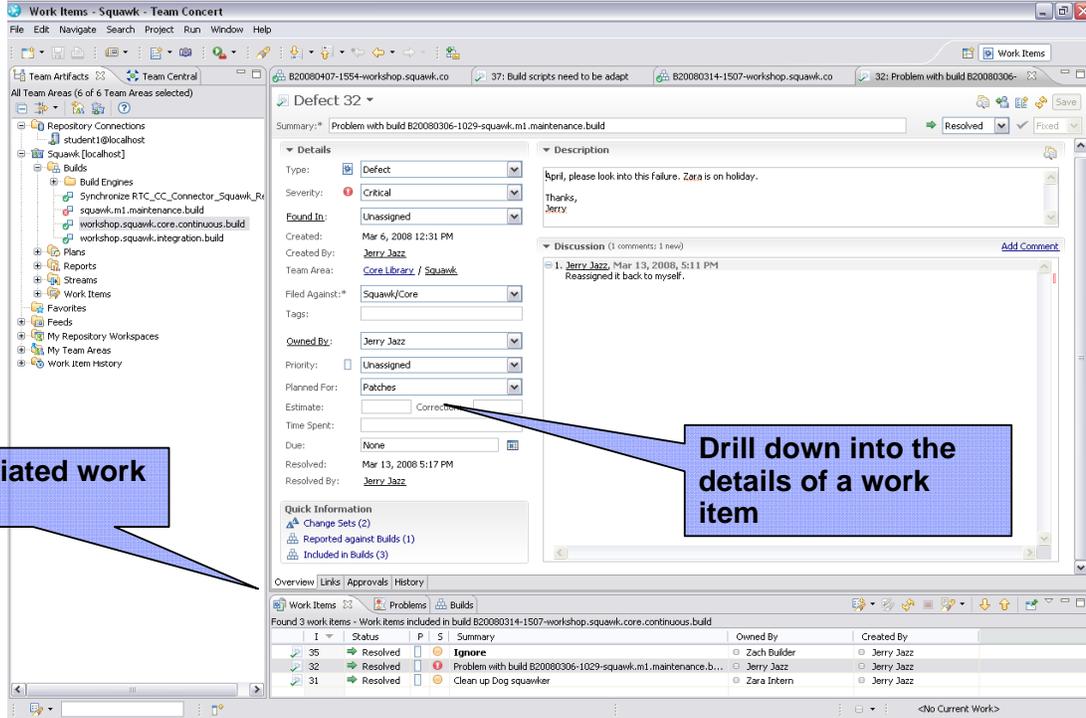
Builds in Visual Studio Client

Keeps traceability with work items, change set, etc.

Build Status	Build State	Label	Progress	Start Time	Duration	Build	Type	ID	Status	Priority	Severity	Summary
	Abandon	20091029-0336				shop.squawk		496	New			Complete Student1 squawker
	Cancel	20091029-1821				shop.squawk						
	Open	20091029-1822				shop.squawk						
	Open	20091029-1823				shop.squawk						
	Open	20091029-1811				shop.squawk						
	Open	20091014-1421				shop.squawk						
	Open	20091014-1422				shop.squawk						
	Open	20091014-1423				shop.squawk						
	Open	20091014-1424				shop.squawk						
	Open	20091014-1425				shop.squawk						
	Open	20091014-1426				shop.squawk						
	Open	20091014-1427				shop.squawk						
	Open	20091014-1428				shop.squawk						
	Open	20091014-1429				shop.squawk						
	Open	20091014-1430				shop.squawk						
	Open	20091014-1431				shop.squawk						
	Open	20091014-1432				shop.squawk						
	Open	20091014-1433				shop.squawk						
	Open	20091014-1434				shop.squawk						
	Open	20091014-1435				shop.squawk						
	Open	20091014-1436				shop.squawk						
	Open	20091014-1437				shop.squawk						
	Open	20091014-1438				shop.squawk						
	Open	20091014-1439				shop.squawk						
	Open	20091014-1440				shop.squawk						

Work Items

Know what build it has been implemented, its change sets, etc.

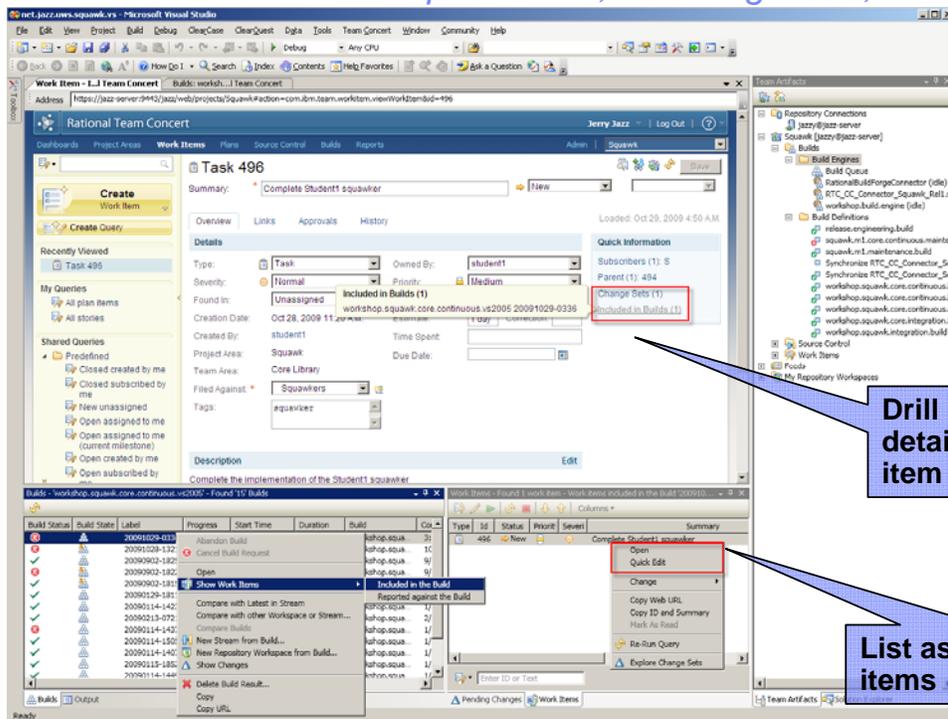


List associated work items

Drill down into the details of a work item

Work Items in Visual Studio Client

Know what build it has been implemented, its change sets, etc.



Drill down into the details of a work item

List associated work items

Change Sets

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

Defect 32
Summary: Problem with build 820080306-1029-squawk.m1.maintenance.build

Attachments: [Table with columns: Id, Name, Created, Created by, Size, Type]

Subscribers: [Add, Remove, Add, Open, Remove]

Links:

- Change-Sets
 - Changes in Core - Jerry Jazz 3/13/08 5:16 PM
 - Changes in Core Tests - Jerry Jazz 3/13/08 5:16 PM
 - Included in builds
 - squawk.m1.maintenance.build M20080313-1516
 - workshop.squawk.core.continuous.build 820080314-1507-workshop.squawk.core.continuous.build
 - workshop.squawk.integration.build I20080314-1507-workshop.squawk.integration.build
 - Reported against builds
 - squawk.m1.maintenance.build 820080306-1029-squawk.m1.maintenance.build

Overview Links Approvals History

Work Items Problems Builds Change Explorer

<No Comment>

net.jazz.uws.squawkers/src/net/jazz/uws/squawkers
Dog.java

Review change sets that make up the build

Change Explorer lists files that were modified for a given change set

Change Sets in Visual Studio Client

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

Task 496 Complete Student's squawker

Overview Links Approvals History

Attachments: Add File, Browse

Links:

- Change Sets
 - Changes in Core VS2005 - Jerry Jazz - Oct 29, 2009 3:39 AM

Change Explorer: net.jazz.uws.squawkers/src/net/jazz/uws/squawkers, Dog.java

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 the ability to identify differences in elements

Compare changes between versions of a file

The screenshot shows the Eclipse IDE with the 'Java Source Compare' window open. It compares two versions of 'Dog.java'. The left pane shows the current version with a change highlighted in red: `private static final String BOW_WOW = "Bow-wow!";`. The right pane shows the previous version with the same line: `private static final String BOW_WOW = "Bow-wow!";`. Below the code panes is a table of changes:

Comment	Creator	Date Created	Merges
32: Problem with build B20080306-1029-squawk.m1.mal...	Jerry Jazz	3/13/08 5:16 PM	
31: Clean up Dog squawker	Zera Intern	3/6/08 12:26 PM	
22: Change Dog Squawk	April Blues	1/23/08 6:20 PM	
Share projects	Jerry Jazz	1/23/08 2:05 PM	

Compare Changes in Visual Studio Client

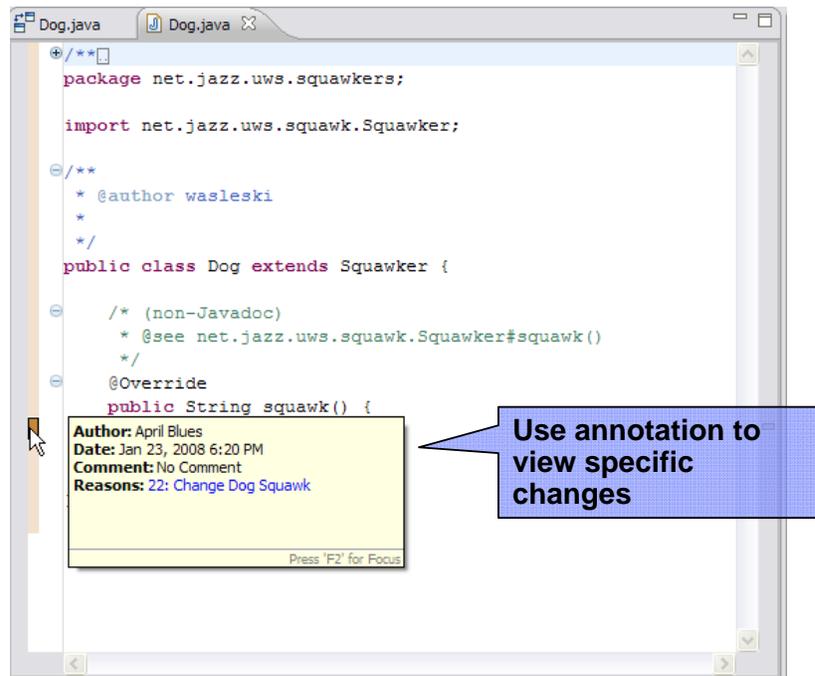
Quickly provide users the ability to identify differences in elements

Compare changes between versions of a file

The screenshot shows the Visual Studio Client interface. On the left, a context menu is open over the 'Change Explorer' window, with 'Compare With Previous' selected. A red arrow points from this menu item to the 'Rational Team Concert - Compare and Merge' window. This window displays a side-by-side comparison of 'Student01.cs'. The left pane shows the current version with a change highlighted in red: `return "Roar";`. The right pane shows the previous version with the same line: `return "student01";`.

Visualize Change History

Colors indicates when changes have been made, hover the mouse over the change and get more details



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



IBM
Software
Group

Endgame and a Tightened Process

An IBM Proof of Technology



© 2009 IBM Corporation

Objectives

- Understand how process is defined in Jazz and implemented by 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, 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...

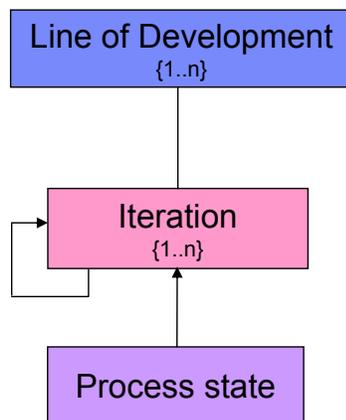
- Teams work on projects
- Each project follows a process
- Each team is unique and thus can work differently
- Work inside the scope of a team follows the team's process
- Cross-team work follows the process of the broader team
- Team members play roles defined by the process
- Process manifests itself 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 process **consolidation**
- Allows for process **evolution**
- Allows for **extensions**
- Put knowledgeable human in the center
- Comprises runtime, authoring, and inspection support

Project Area Iteration Structure and Terminology



Timelines are an element of a project area that own a set of deliverables and its production schedule (**maintenance, new release development**).

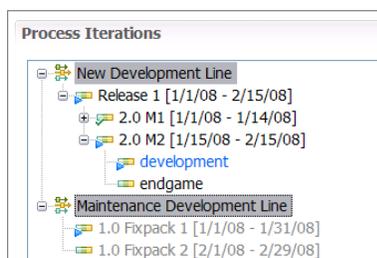
- Often represents parallel development
- A team area is associated with a timeline

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



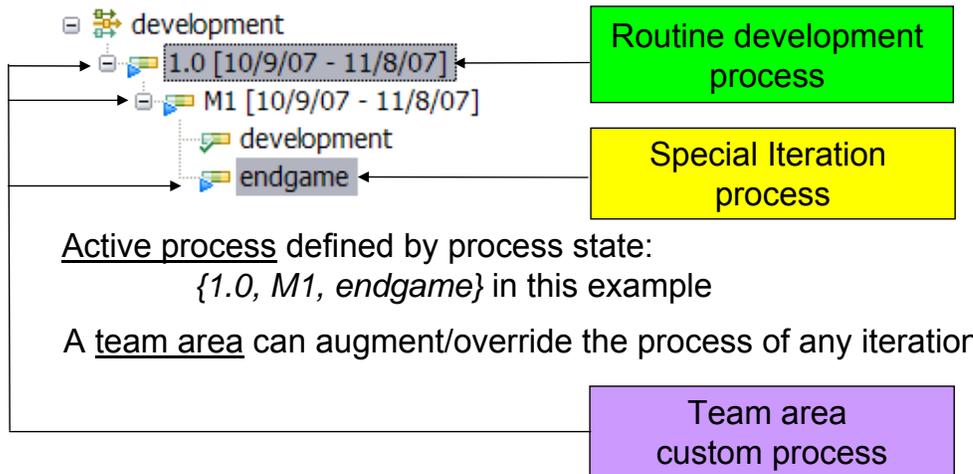
Example:

Main Development Line process state:
2.0 M2

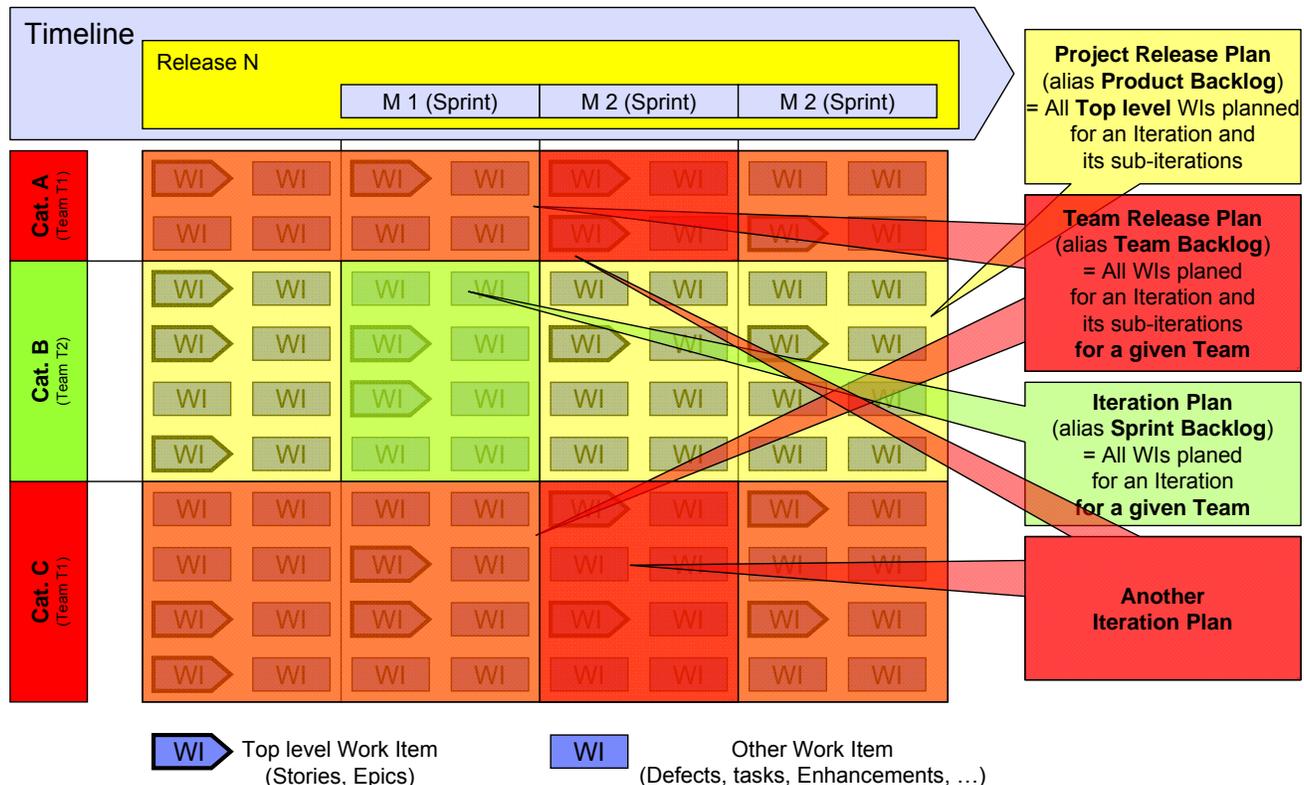
Maintenance Development Line process state:
1.0 Fix pack 1

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...)
- You can have the general process defined for the project
- Override/augment the general process in planned iterations



Who's who on Plans...



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



IBM
Software
Group

Taking Control of Your Project

An IBM Proof of Technology



© 2009 IBM Corporation

Objectives

- Learn about the Jazz Dashboards and Reports and how these powerful capabilities can assist your team to track 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 to get more complete information.
- Dashboards are available in the Rational Team Concert Standard and Enterprise editions.

Dashboards for team unity

Transparency and control via customizable dashboards

The screenshot displays the 'Development' dashboard with several key sections:

- General:** Includes tabs for 'Builds', 'Open Defects by Themes', 'Open Defects by Team', 'SVT Defects by Team', 'Defect Trends by Team', and 'Deferred Items by Team'.
- Endgame Has Started:** A warning icon and text indicating that a detailed plan is available for the 'End Game Plan'.
- Jazz Development Description:** A text box explaining the 'Umbrella area for all Jazz development work' and the role of the 'Development' team area.
- Risks (10):** A list of risk items with IDs and descriptions, such as '48863 [Risk] Migrating/importing SVN repositories'.
- RFS (28):** A list of Risk Finding Set (RFS) items, including '9619 [RFS] Monitoring a continuous build'.
- Track Build Items (1):** A section for tracking build items, currently showing 'RC1 status'.
- Development Plans (33):** A Gantt-style chart showing work items across various team areas like 'APT', 'CC Connector RC1', and 'Release Engineering'.
- Jazz Development Event Log (2 new):** A log of recent events, such as 'Test scenario: Process author creates/refines a Process Template'.

Clearly understand Team Goals

Risks, Issues, Challenges surfaced at both the Team and Project Level

Real time Status

Dashboards and Reporting for progress

The screenshot displays a comprehensive dashboard with several key components:

- Work Item Comparison (Cont'd):** Three line charts showing trends for different team areas over time.
- Work Item Members (16):** A list of team members with their roles and contributions, including André Weinand, Benjamin Pasaro, and others.
- Current Milestone status:** A callout box pointing to the 'Current Work Item Plans' section, which shows the current iteration (0.6 RC1) and a progress bar.
- Team Member Details:** A callout box pointing to the 'Work Item Members' list.
- Work Item Event Log:** A list of recent events such as 'WebUI - Hide comment is confusing - lost work' and 'Indexing slow on attachment with long lines'.

Trending by Project or by Individual Team

Current Milestone status

Team Member Details

Dashboards – personalized visibility

Display your choice of reports and queries in your own dashboard, e.g. to control the flow of work items.

The screenshot shows a personalized dashboard with the following features:

- Navigation:** Home, Project Areas, Dashboards, Work Items, Iteration Plans, Reports.
- Recently modified (95) Tags:** A list of tags like '1.candidate', 'adminui', 'burndown', etc.
- Open Work Items by Type:** A stacked area chart showing work items categorized by type (Defect, Task, etc.) across iterations.
- Open vs Closed Work Items:** A line chart comparing the number of open and closed work items over time.
- Open Work Items by Priority:** A stacked area chart showing work items by priority (Low, Medium, High) across iterations.
- Blocking Work Items:** A bar chart showing the number of blocking work items.
- New Work Items by Severity:** A bar chart showing the number of new work items by severity (Unknown, Blocker).
- Closed Work Items by Priority:** A bar chart showing closed work items by priority.

Roll up information and drill down for details

All Dashboards > Rational Team Concert > RTC Development >

Dashboard Viewlets

Home | Live Reports | Work Items | Trend Reports

Work on the RTC dashboard viewlets

The Dashboard Viewlets component provides the viewlets for the Jazz Foundation Dashboard component that are specific to Rational Team Concert. This is a virtual component - all the code is associated with other components naturally associated with these viewlets (e.g. Work Item viewlets with Work Items etc.).

Current Dashboard Viewlets Plans (0)

Current Iteration: **3.0 M1**

No results available.

Dashboard Viewlets Events (99 new)

- Curtis d'Entremont delivered 3506.v20090730 to 'SCM-Dashboard Viewlets' 5 hours ago
- Curtis d'Entremont delivered 1904.foundation.10x.integration_i20090729-1701 to 'RelEngBuilder-Dashboard Viewlets' 5 hours ago
- Curtis d'Entremont delivered 6330.foundation.10x.integration_i20090729-1701 to 'Repository-Dashboard Viewlets' 5 hours ago
- Curtis d'Entremont delivered 71.v20090730 to 'Reports Examples-Dashboard Viewlets' 5 hours ago
- Curtis d'Entremont delivered 2415.v20090730b to 'Reports-Dashboard Viewlets' 5 hours ago
- Curtis d'Entremont delivered 2547.F10xM3_20090729 to 'Process UI-Dashboard Viewlets' 5 hours ago
- Curtis d'Entremont delivered 2720.F10xM3_20090729 to 'Process-Dashboard Viewlets' 5 hours ago

Page 1 of 13

Dashboard Viewlets Members (8)

Adam Archer	buildmaster, contributor, integrationbuildmeister
Christophe Elek	parttime
Curtis	buildmaster, contributor, integrationbuildmeister
d'Entremont	
Dejan Glazic	componentlead, contributor, integrationstreamadmin, dashboardadmin
Mike Pawlowski	buildmaster, contributor, integrationbuildmeister
Rob Retchless	integrationbuildmeister, buildmaster, contributor
Susan F. McCourt	
Thomas Frauenhofer	

Work Item Queries

- Dashboard Resolved Fixed No Milestone (6)
- Open viewlet items (3)
- Open Dashboard items (92)
- Dashboard Viewlets Assigned No Milestone (38)
- Ready to be worked on (New->Triaged) (4)

Open Dashboard items (92) Tags

less | more

c_alm collaboration dashboard mp-note noteworthy plans queryeditor rcm-ux refresh size testcandidate usability_test ux viewlet

Open vs Cl

Show Parameters

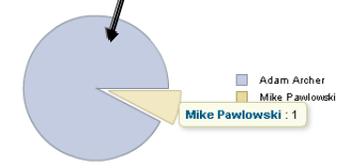
No results were found.

Open Dashboard Items (Current Milest... (13) Owned By

Adam Archer
Mike Pawlowski : 1

jazz

See Work Items and drill down to view them



Dashboards – project status with Burndown charts

All Dashboards > Rational Team Concert >

RTC Development

General | 2.0.0.1 | 1.0.1.1 iFix 1 | Backlog Cleanup | APARs | QCERT | SVT Hot | 2.0 Plan | RTC 2.0 Perf and Scalability | RFEs | 2.0 Polish | Open Defects by Themes | Builds

Open Defects by Team | 2.0 Endgame Fixes | SVT Defects by Team | 2.0 Endgame Defect Trend Comparison | Defect Trends by Priority 1.0 | Tests | 1.0.1 | 2.0 DCP | **Burndown M3** | Velocity

2.0 Endgame Defect Trends

M3 Burndown

Burndown during the M3D1 iteration. Build, APT, Work Items

Agile Planning

M3 Burndown

Reports, SCM

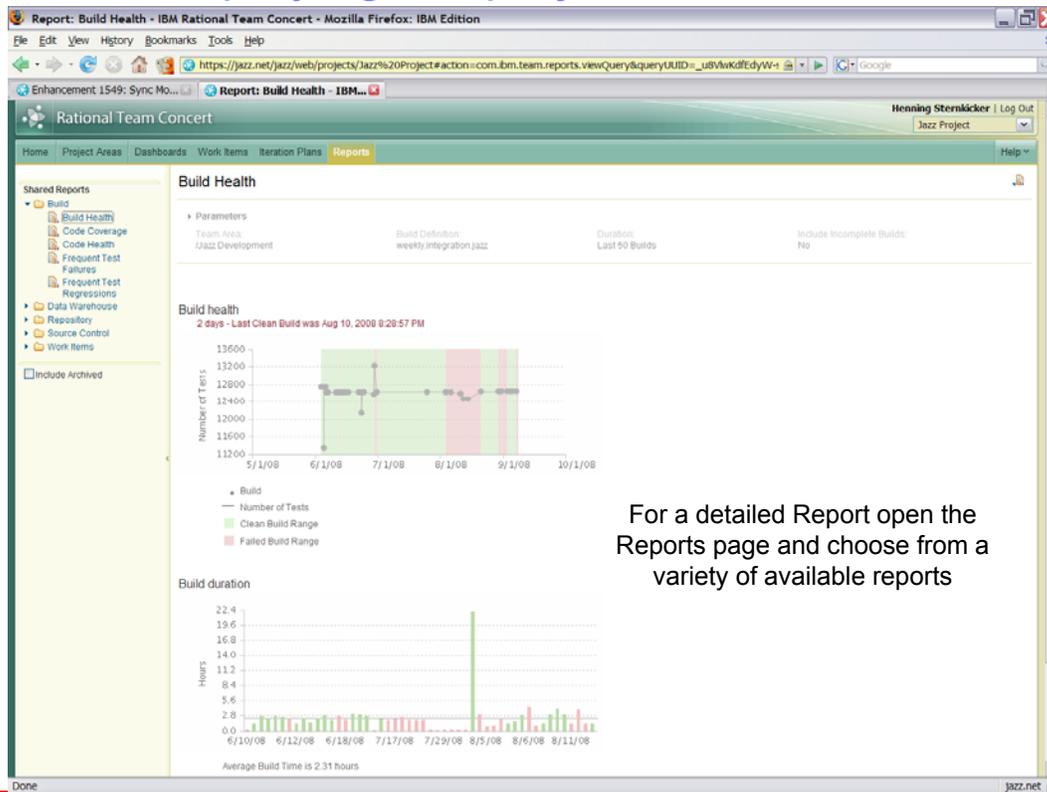
Source Control

M3 Burndown

CC Connector, CQ Connector

CQ Connector

Reports – Displaying the project status



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
 - ▶ Etc.
- Reports can be arranged in the Web UI Dashboards
- Reports can be exported to: .pdf, .xls, .doc, .ppt formats

*BIRT is an open source Eclipse-based reporting system that integrates with your Java™/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** via 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.



IBM
Software
Group

Integrating with Other SCM Systems

An IBM Proof of Technology

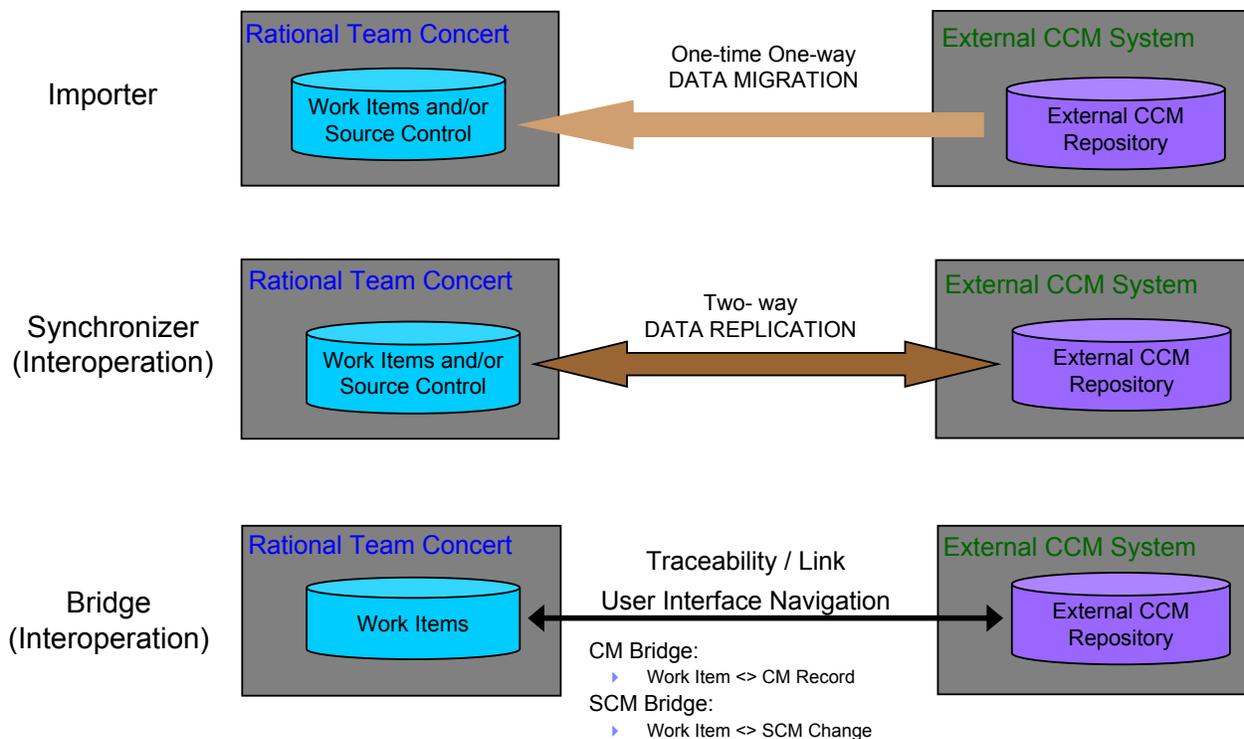


© 2009 IBM Corporation

Objectives

- Explore how Rational Team Concert integrates with other SCM systems
- See how Rational Team Concert synchronizes its repository with Rational ClearCase

Interoperation with other CCM Systems



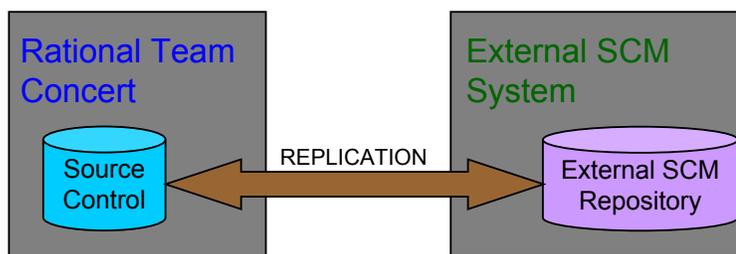
Interoperation with other Source Control systems

Synchronization between SCM systems and Rational Team Concert Source Control

Uses:

- Manage code changes made to same code base by teams using Rational Team Concert and another SCM system
- Gradual and low-risk migration from other SCM system to Rational Team Concert

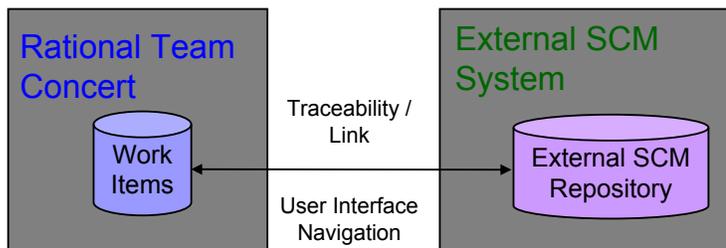
Example: ClearCase Synchronizer



Bridge between SCM Systems and RTC Work Items

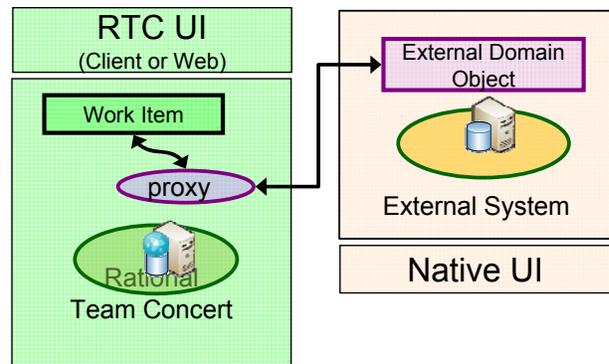
Uses:

- Introduce Rational Team Concert when not willing or ready to change SCM systems. Allows project to use their current SCM system along with Rational Team Concert Work Items, Agile Planning, and Build
- Example SCM Bridges: ClearCase, Subversion, Git (incubator)

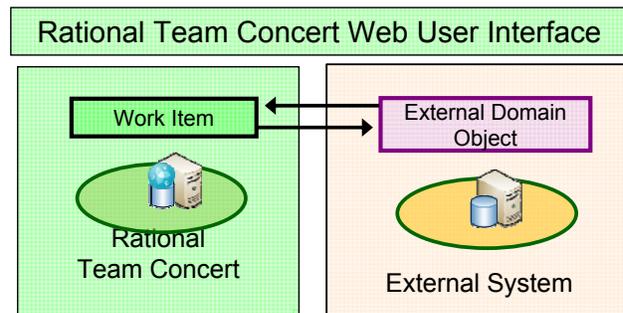


Work Items – Synchronizers (Connectors) and Bridges

- Synchronizers (Connectors)
 - ▶ Synchronize data between systems
 - ▶ Synchronization Rule controls mapping
 - ▶ Changes to data values and process state are synchronized with the other system
 - ▶ Native user interface for either system can be used to make changes
 - Rational Team Concert IDE Client or Web UI
 - Native UI for external system



- Bridges
 - ▶ Support relationships between Work Items and a similar external domain object (CM)
 - ▶ Data is not synchronized; references provide a loose coupling of data in each system
 - ▶ Changes must be made using the appropriate user interface for each record
- ClearQuest Bridge as an example:
- ▶ Relationships must be defined using the ClearQuest UI as integrated in the Rational Team Concert Web UI

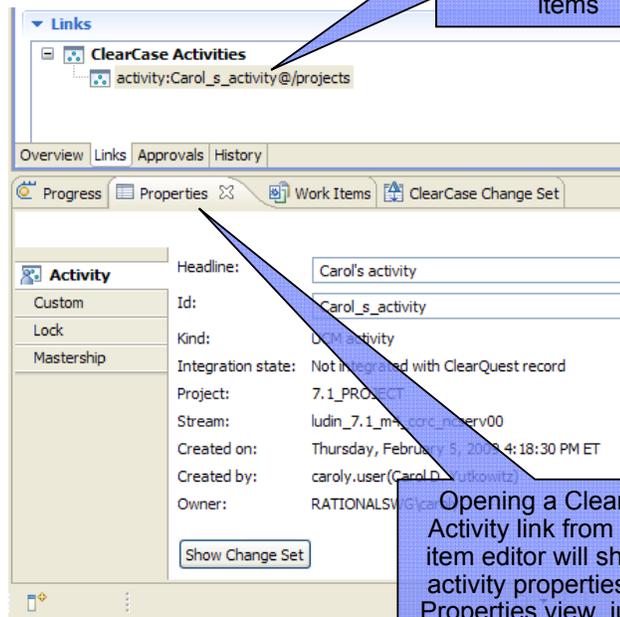


Integrating Other Repositories with Team Concert

- Importer – migrate to Rational Team Concert
 - ▶ Jazz SCM – CVS, Subversion, ClearCase
 - ▶ Jazz Work Items – JIRA, Bugzilla
- Bridge – Lifecycle integration
 - ▶ Jazz Work Items – ClearCase, Git, Subversion (SCM)
- Synchronizer – interoperation between repositories
 - ▶ Jazz SCM – ClearCase
 - ▶ Jazz Work Items - ClearQuest

ClearCase Bridge

- Now you can link Rational Team Concert work items with ClearCase UCM change sets
- If you use the ClearCase Remote Client and UCM you can associate a UCM change set with a Team Concert work item.
- Then you can use agile planning, taskboards and dashboards to show project status of work done in ClearCase!



References to ClearCase Activities will also show up on the Links tab of their associated work items

Opening a ClearCase Activity link from a work item editor will show the activity properties in the Properties view, just as if the properties were requested from CCRC directly.

ClearCase Importer Wizard

- Imports from ClearCase base or UCM with history
- Choose all baselines or particular baselines or labels
- ClearCase 7.0, 7.0.1 and 7.1.x supported

Import from an existing Base ClearCase branch

ClearCase Branch Type Selector:

Format: <branch-type-name>@<VOB-selector>

Locked ClearCase Label Type Selector:

Format: <label-type-name>@<VOB-selector>

Import with history from ClearCase

Choose Baselines to Import

Import all baselines

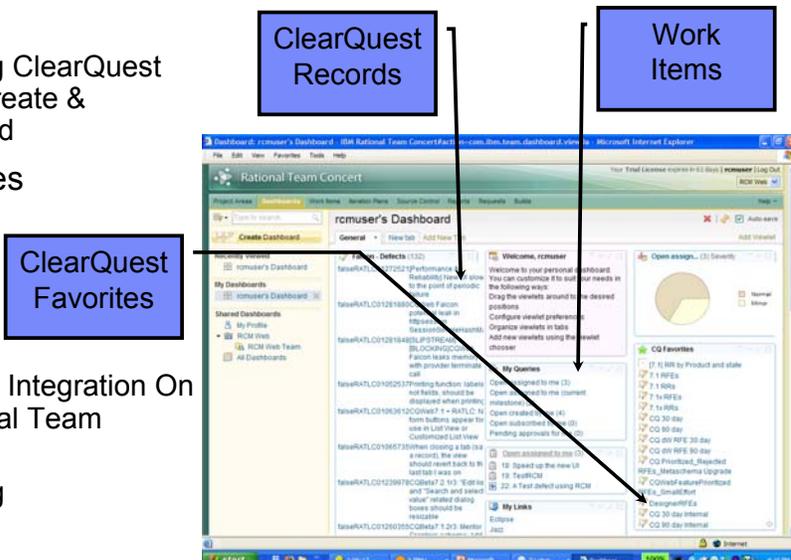
Import all baselines which have the following attribute type:

Format: <attribute-type-name>@<VOB-selector>

ClearQuest Bridge

Lower Total Cost of Ownership

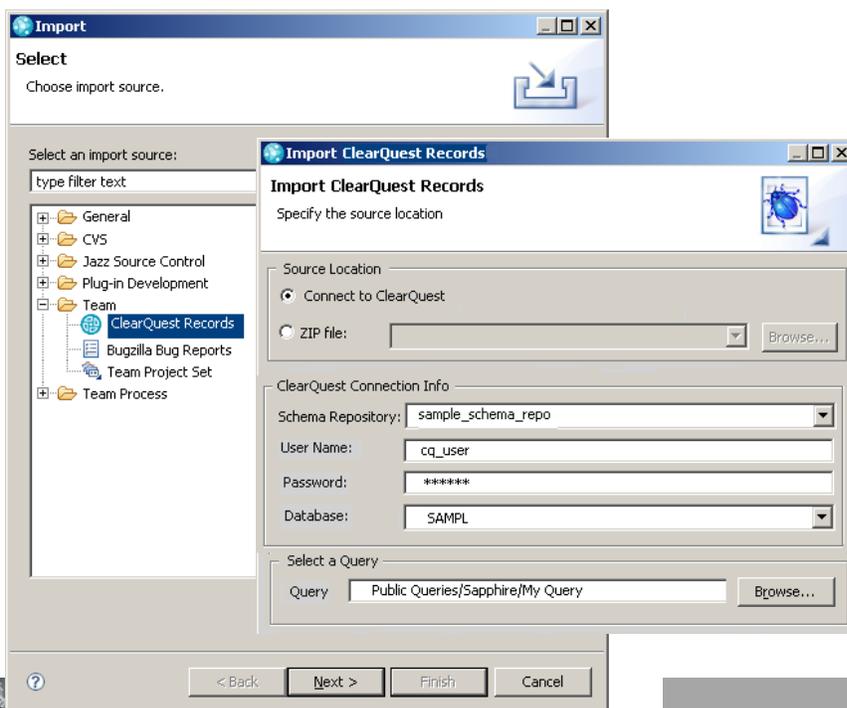
- Reduces Time/Network Traffic
 - ▶ No Waiting for Data Synchronization so Key Use Cases Will Be Faster
 - ▶ Quicker Access to Record Information
- Lower Administration
 - ▶ alternate mechanism for linking ClearQuest with Rational Team Concert; create & maintain sync rules not required
- More Manageable Repositories
 - ▶ No Data Replication - Helps Minimize Database Growth
- Integrated User Experience Reduces Training Costs
 - ▶ ClearQuest Bridge UI Provides Integration On The Glass Between the Rational Team Concert and ClearQuest
 - ▶ Appears As If You Are Working In a Jazz-based Environment



ClearQuest Importer Wizard

Provide ability to move select ClearQuest records to a Rational Team Concert project

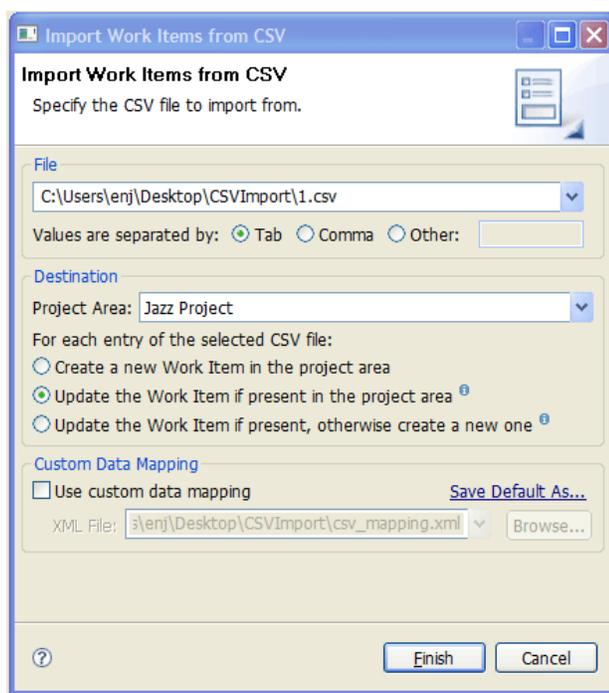
- Import ClearQuest records to an intermediate XML format
- Select scope of import using ClearQuest query
- Create a mapping file to map ClearQuest Record types to work item attributes
 - ▶ Sample provided
- Import into an Rational Team Concert project once mapping has been validated.



CSV Importer

Streamlines migrations

- Create new work items or update existing work items
- First row or custom XML mappings
- Allows for work items to be imported from other systems



Lab #10 Overview

- In this lab the instructor will demonstrate how to synchronize code changes between Rational Team Concert and Rational ClearCase



IBM
Software
Group

Project Growth and Multi-stream Development

An IBM Proof of Technology



© 2009 IBM Corporation

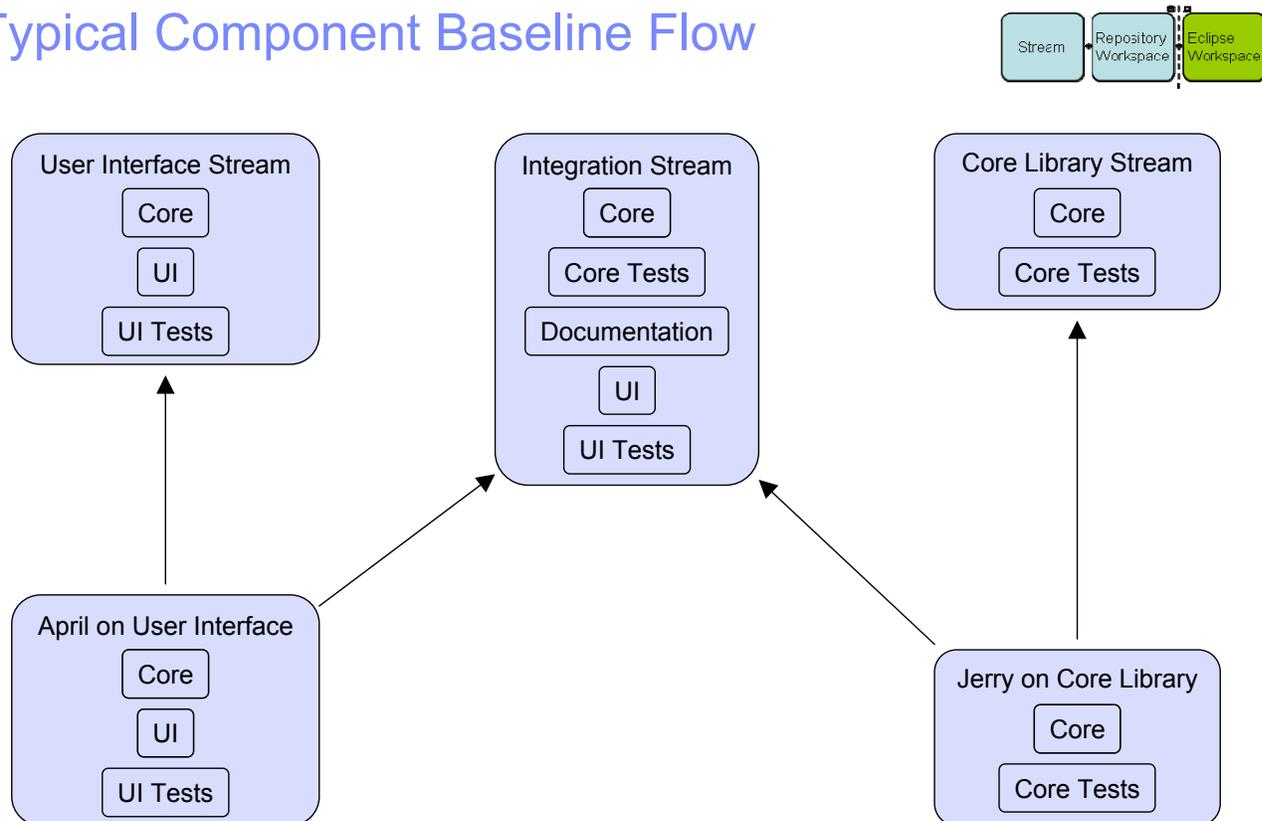
Objectives

- Understand the support for parallel development in Rational Team Concert

Growth and Adding Teams

- Project growth leads to multiple inter-dependent teams
- Each team needs its own stream
 - Quickly collaborate and share changes with each other
 - Run builds on a scheduled basis, as well as ad hoc
- Enhance ability for cross team collaboration and communication
 - Manage cross team dependencies
 - Project build stability and transparency
- Need a stream for cross team sharing and project builds
 - Well known change adoption schedule and process
 - Consistent and repeatable successful full project builds

Typical Component Baseline Flow



Other Scenarios

- Maintenance
 - ▶ New stream for maintenance
 - ▶ Created from final release snapshot
 - ▶ Isolated from daily development
 - ▶ Easy to move changes between streams
- Community exploration
 - ▶ Single person exploration can use a repository workspace
 - ▶ Community exploration will require sharing and collaboration
 - ▶ New stream created from a development stream snapshot

Concepts Learned

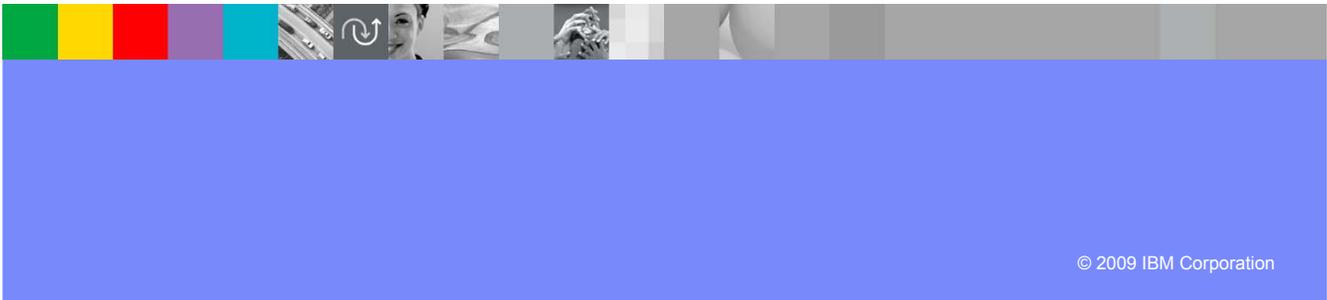
- In this module you explored the support for parallel development offered by Rational Team Concert.
- You have explored handling multiple streams and the sharing of component dependencies between them.
- Rational Team Concert enables
 - ▶ Increased team productivity by allowing parallel development
 - ▶ Enhances the delivery policies and process while improving baseline consistency
 - ▶ Supports seamless interaction for globally distributed teams
 - ▶ Establishes traceability and transparency among project artifacts



IBM
Software
Group

Session summary

An IBM Proof of Technology



© 2009 IBM Corporation

Session summary

- We have described current collaboration challenges with distributed teams
- We have explored how 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

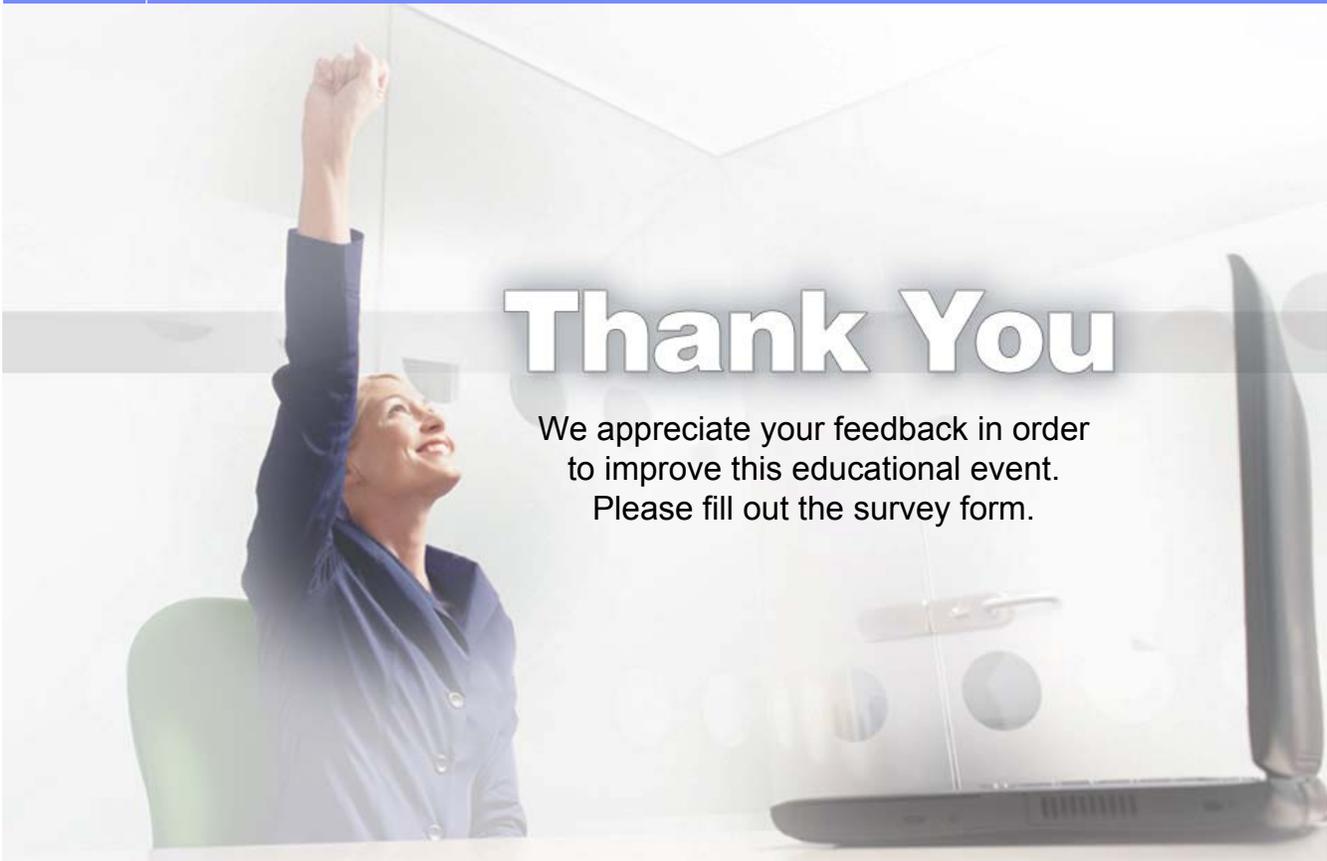
Next steps

- Engage your local Rational 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://www.ibm.com/developerworks/spaces/jazz>

Additional resources

- Learn more about and download free trials of Rational Team Concert at <http://ibm.com/rational/rtc>
- Explore Rational Team Concert tutorials, demos and other developer learning resources <http://ibm.com/developerworks/spaces/jazz>
- 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 2009. All rights reserved.

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 other IBM products and services are trademarks of the International Business Machines Corporation, in the United States, other countries or both. Other company, product, or service names may be trademarks or service marks of others.

