Innovate2011
The Rational Software Conference
11th and 12th of October
Let's build a smarter planet.

CICS: From waterfall to agile using RTC*

Nigel Hopper
CICS SM and API Team Leader

(*Or "Teaching the elephant to dance!")

30 billion transactions/day, >$300B/week

40+

~~35~~ years invested in applications

16,000 customers worldwide

CICS

30 million users

950,000 programmers earn their living from CICS

First GA'd when:
- *Nixon* was president
- Man *landed* on the moon

Over *900,000* concurrent users/system

One of the top 35 technologies that shaped the industry*

Used by 490 + of IBM"s top 500 customers

5000 packages from 2000 ISVs

http://www.ibm.com/ibm100/us/en/icons/cics/

50,000 CICS licenses

*According to Computerworld magazine

CICS TS code *is* complex!

- Started out over 40 years ago as a loose collection of programs
- Primarily written in assembler and PL/X
  - Now has Java, XML, Cobol and more!
- Eventually converted to domains
  - Currently in the region of 70 domains. Grouped in areas such as Application Services, Business Logic Applications, Base Runtime, CPSM
  - 3 APIs
  - Multiple tools such as CICS Explorer, CICS Deployment Assistant

Since 2007 we've worked on...

- Adopting Agile practices
- Adopting RTC for project planning and control
- Rewriting our build into Antz
- Migrating our source code to RTC
- Moving from green screen development to RDz
- Looking to use common tooling for development and service
- And there is still some way to go!

The CICS process until 2007

- Waterfall oriented to a cycle of:
  - Development
  - Functional Test
  - System Test
  - Quality
  - Translation/Packaging/Deliver
- 2 year release cycle
- Upfront commitment for the release

Pre-2007 process issues

- Up front commitment can be very inflexible
  - Change can require (5?) layers of management approval
- Work sized at – 'what will fit'
- Large overhead in
  - Planning
  - Tracking
  - Status recording
  - Managing change
- Coordination and scheduling of cross team work
  - Different teams, different priorities

Pre-2007 process issues (cont.)

- Late in release system test and beta issues
  - Integration problems
    - Difficult to identify
    - Lengthy to fix due to complexity
  - Beta issues
    - Raised late
    - Lack of time to make changes at this late stage
- Defect backlog
  - Release would often peak at 600+ defects
  - Main focus towards the end of release
- Often a great deal of post-release tidying up

Why Agile?

- IBM were sufficiently confident of it to make it the corporate direction for developing software
- “If CICS can do it, anyone can” message
- But how?
  - Little skills
  - Little knowledge
  - Still a 2 year release cycle
  - Who's benefit is this for?

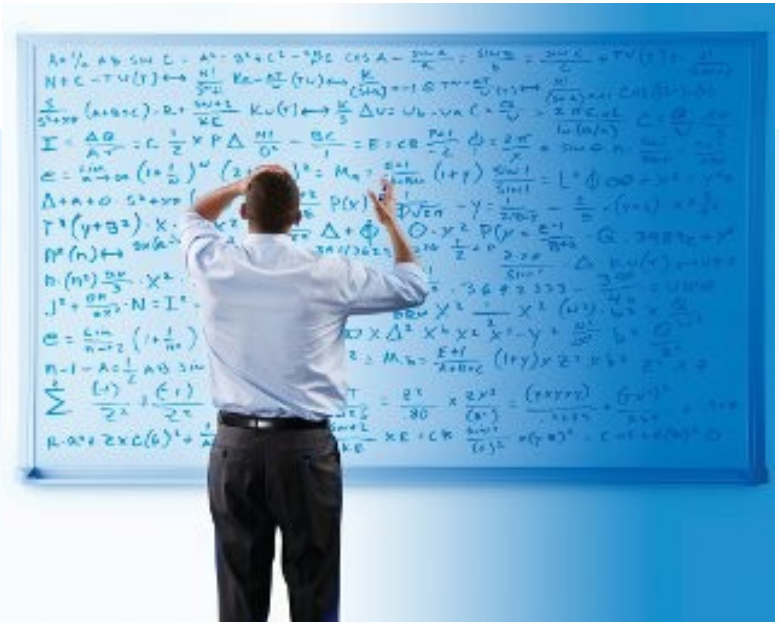
Initial Agile adoption

- In 2007 our CICS release became iterative
  - Still a 2 year release cycle
  - With 4 month iterations

- The intention was:
  - Work to be broken down into smaller 'chunks'
  - All development, functional testing and defects done in iteration
  - System/Integration testing done in following iteration
  - Beta every 4 months

Initial Agile adoption - Reality

- Still 2 year upfront commitment
  - Not seeing as much flexibility due to this
- Difficult cultural change
- Difficult to contain work to 4 months
  - Code not conducive to this approach
    - 6 million lines of code
    - Nearly 40 years of waterfall development
  - Requirements were too large
  - Agile was new to us



Initial Agile adoption – Reality (cont.)

- Tooling fragmented and not Agile 'friendly'
  - 10+ day code 'freeze' prior to beta shipping!
  - No way of integrating and prioritizing. For example:
    - Defects in one tool
    - Project tracking several others

- While the Agile term was used, reality was mini-waterfalls

Initial Agile adoption – benefits

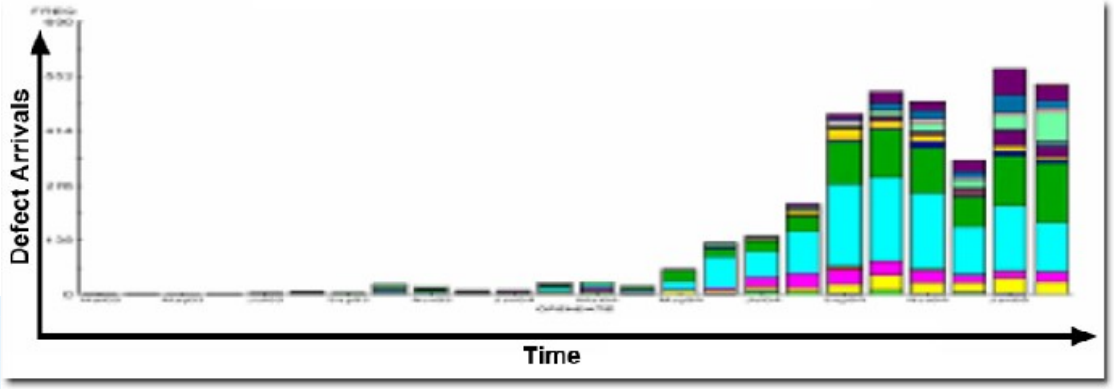
- Beta shipped every 4 months
  - Much earlier feedback
- System testing much earlier
- Defects found and handled much earlier
  - Peaked at 450
- Changed people's perception of what was possible
  - Proved work could be 'chunked'
  - Still a great deal of scepticism from certain 'mature' personnel!
- Feedback on quality in the release has been high

Agile: Deliver sooner, fix earlier

Waterfall Profile

Defects found later when they are more expensive to fix

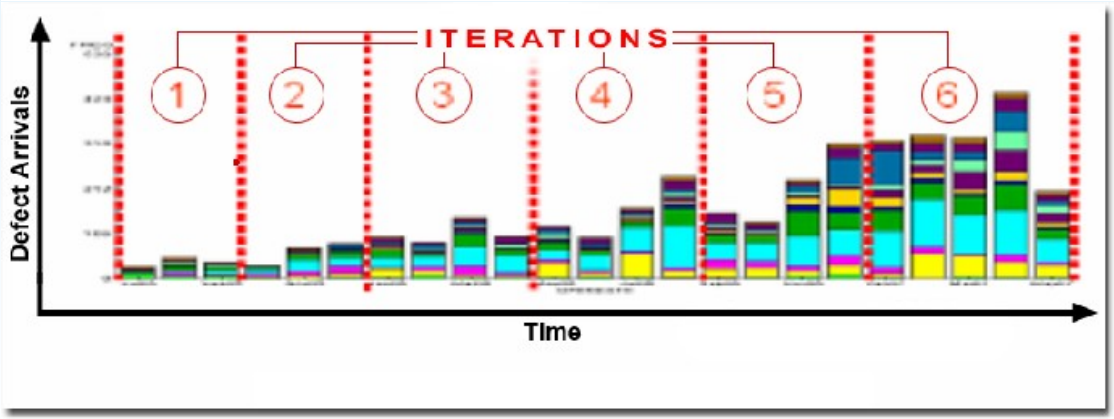
CICS TS for zOS
Last waterfall delivery



Agile Profile

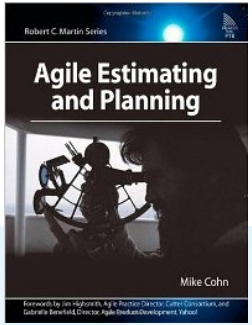
defects found early when they are cheaper to fix

CICS TS for zOS
First 'Agile' delivery



Timing is everything...

- Senior developers / testers / management had Agile education
  - More of an overview
  - Approximately 15 of us did a book review
    - Primarily team leaders and lead developers
    - Mike Cohn – Agile Estimating and Planning
- 2008 Rational Team Concert V1 beta shipped
  - Adopted by a couple of system testers for their infrastructure
  - Demonstrated it to the CICS team
  - Trialled by several CICS team members
- Hursley started centrally hosting Jazz servers

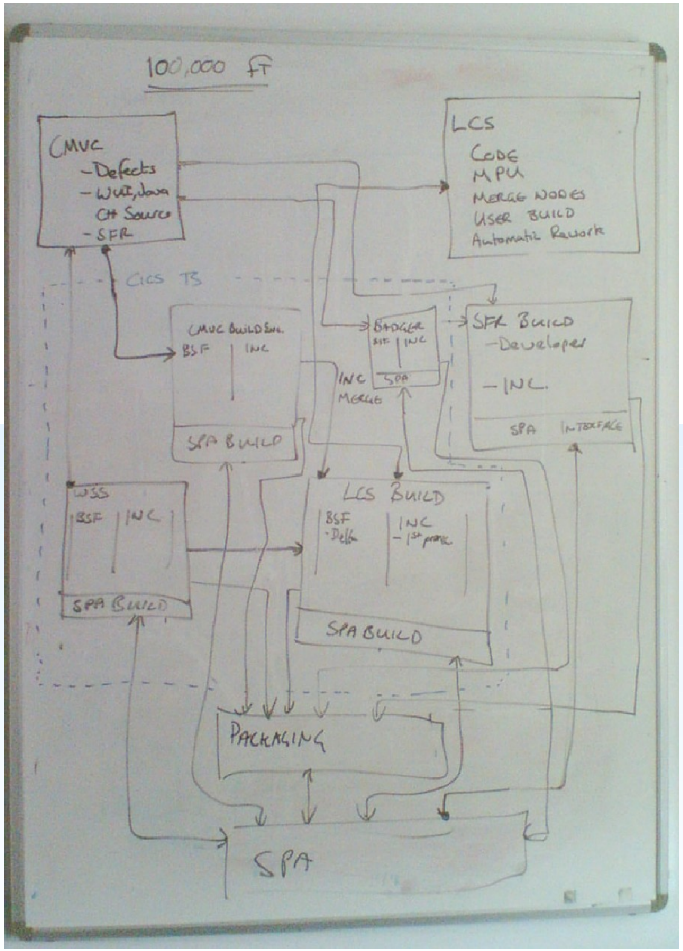


Timing is everything... (cont.)

- Timing was good - Major release of CICS shipped in 2009
- A significant turn around of staff meant a loss of important skills
- A review of tools and processes was undertaken

The way we were - Reality

- Using a wide variety of tools to manage project, for example:
  - 3 source code management systems
    - Completely alien to each other
    - Separate builds for two of the SCMs
    - Highly integrated build into main SCM
  - External database for customer requirements
  - Lotus Notes databases/documents for:
    - Internal requirements
    - Actions
    - Risks
- High learning curve for new starters



The Vision

Single non-proprietary environment for the delivery and service of future releases of CICS Transaction Server for zOS

- Where everyone (business, marketing, development, test, service, build, etc) can focus and collaborate via one single tool
- If we are really to adopt Agile principles, the tooling had to improve

Why RTC?

- Early trials showed great promise
- Hursley just started centrally hosting Jazz servers
- Highly configurable
  - We were in control!
- Provided great audit tracking
- Allowed for an end to end integrated development environment
  - Requirements, Approvals, Defects, Code, Tracking, Build and so on!
- Clearly a tool designed for Agile development

My RTC 'key' goals

- Key goal – use RTC 'out of the box'
  - Configuration – OK
  - Customization – Not OK
- Key goal – use industry standard Agile methods/terms
- Key goal – use a minimum number of roles – trust people
- Key goal – use RTC to implement RTC
- Key goal – big bang approach would not work

RTC Adoption: Making it happen

- First thing – 2 month iterations, 4 month betas
- Initial focus on work items, project planning and tracking
  - Epics, Stories. Tasks, Defects, Risks, Actions, etc.
- Long term focus on migrating source code and service delivery
  - Rewrite the build
  - Migrate LCS code
  - Service product through RTC
- Staged implementation
  - If you wait for it to be perfect, you will never start!
- Education – Combination workshops, mentoring, wiki help

RTC Adoption: Planning and Tracking

- Time line
  - July 2009 - Created RTC server
    - Infrastructure project, plus Main and Sandpit projects
  - August – October 2009 – Reviewed processes and configured server – 80-90% right
  - Mid-2009 – migration of Epics (requirements) to RTC
  - October 2009 – migration of Defects from CMVC to RTC
  - November 2009 – migration of CMVC source code to RTC
  - End of 2009 using RTC for all work outside of LCS/build
- All requirements, defects, plans, teams, etc., everything needed to work a project was coordinated through RTC
  - Including Epics, Stories, Defects, Tasks, Risks, Actions
  - PLUS editing of approximately 30% of the code

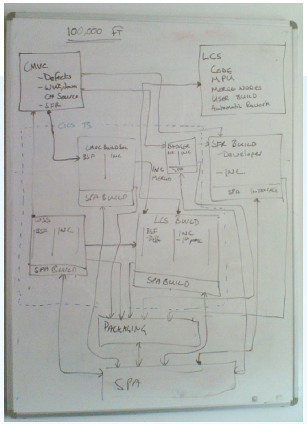
RTC Adoption: Planning and Tracking

- Many 'tweaks' since then
  - Use of dashboards for status meetings
  - New work items
    - Risks
    - Actions
  - Red/Amber/Green fields for stories
  - Minor changes to various:
    - Work flows
    - Enumerations
  - Swapped use of priority and severity in defects – My bad!
  - Templates for work item 'groups'

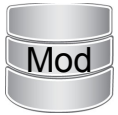
RTC Adoption: Development Environment

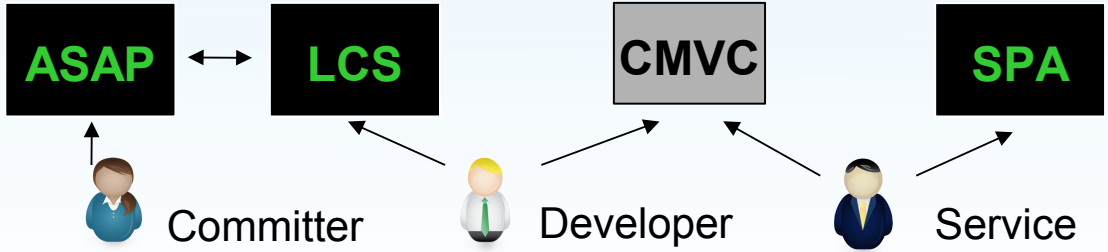
- The build rewrite
  - Start of build rewrite – January 2010
  - First integration build – September 2010
  - First beta shipped from Antz build – February 2011
  - RTC delivered CICS TS for zOS V4.2 - June 24, 2011
- The tooling
  - DTS started – June 2010
  - DTS ready – March 2011
  - Developer environment ready – April 2011
  - LCS Source code migrated to RTC – June 2011
  - RTC/RDz development environment rolled out – June/July 2011

Advancing the development environment



Ye olde environment

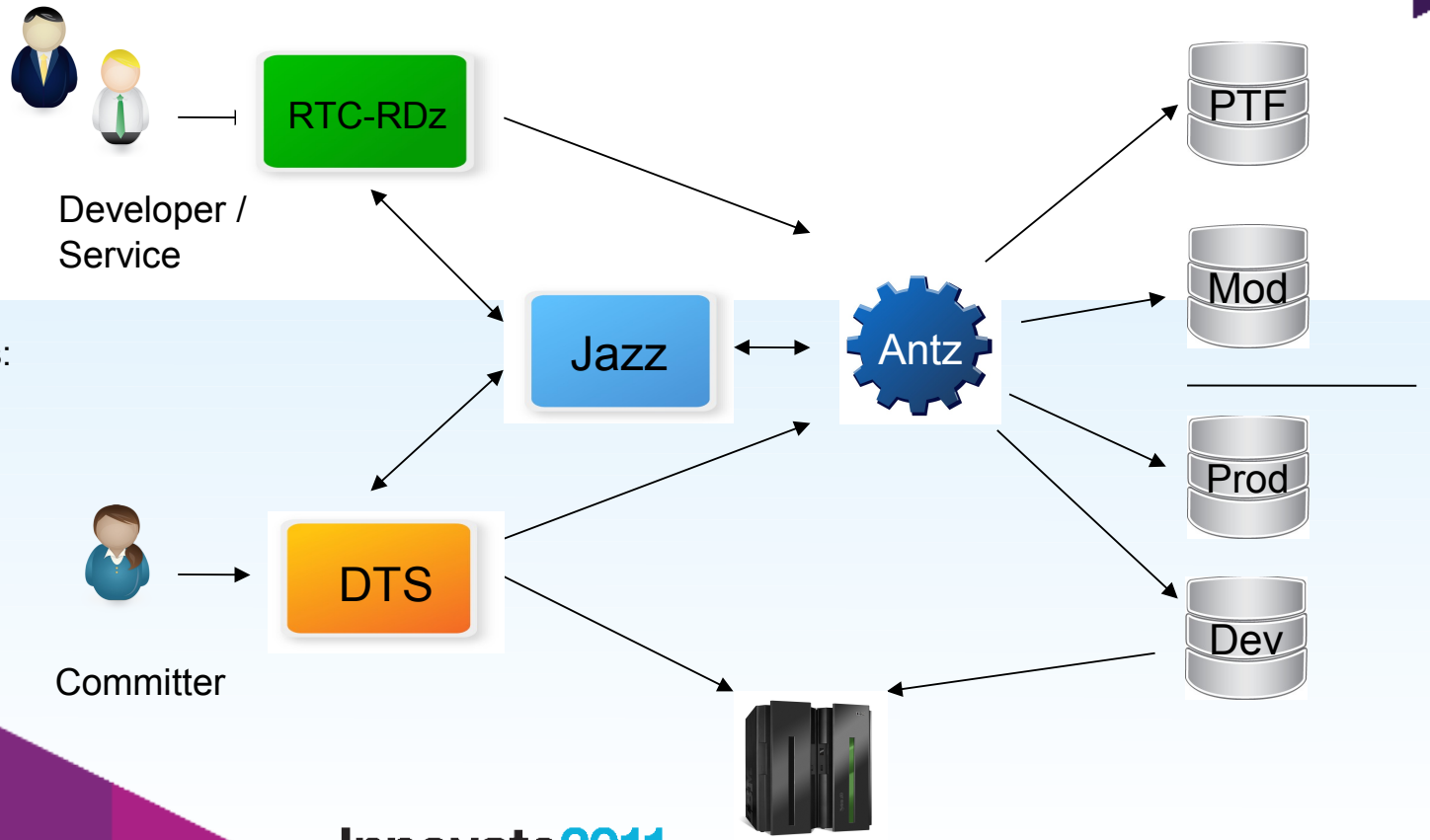




The new developer environment

- Rational Team Concert v3
- Rational Developer for System Z v8
- IBM Internal language editor plug-in
  - Better context aware editing
- Antz Eclipse plug-in for build control
  - Hursley constructed build tool
    - Ant is an open source build tool
    - We added extensions to support zOS
  - Antz now ships with RTC

Rational based development



RTC/Jazz manages:
- Release/Iterations
- Work items
- Plans
- Source code
- Streams
- Change sets
- Lots more!

Using the new tooling

- Cruise Eclipse plug-in
- Changing and unit testing the code (RTC/RDz)
  - Developers load a workspace with a project (CICS Domain)
  - Make changes using language aware editors
  - Check-in changes to RTC
  - Use RTC build definitions to request a Antz developer build
  - Build automatically delivers changes to developers load libraries
- Remote System Explorer within RDz
  - Starts CICS and allows job output to be examined
- RDz Debug perspective to debug CICS system code
  - Soooo much more screen real-estate
- Code reviewed through RTC approvals
- Developer delivers the code to the Delivery stream

Delivery and Test Service (DTS) Tool

- Requirement to control the promotion of code through:
  - Developer
  - Integration
  - Best so far (BSF)
  - Production (Increment)
  - Each level runs automated regression testing of increasing complexity
    - Integration to BSF to Production
- Previously two nodes with this approach (ASAP tool)
  - CICS
  - System management
  - Possible for one node to break the other
  - Stops complete teams or everyone from testing

The issues...

- RTC was not the problem, does what it says on the box
  - RTC is just a tool, it will do what you ask it to
  - Minor issues only
  - Changes are not set in stone
- People issues
  - They do not want to change - Why change if it currently works?
    - Not something new again?
    - How long is this one going to last?
    - Don't want to see the benefits
    - Don't want to try something new
  - Developers have used the same tool set for 20+ years
    - Single biggest change to development practices in 3 decades

The issues... (cont.)

- Process issues
  - We had some serious holes in our process – still very waterfall
  - Need agreement of many senior people
    - An Agile process that is clearly defined – somewhat of a contradiction
    - Still learning the Agile way
  - Some serious bun fights
    - Can often go around in circles
  - Sometimes you just need to make a decision

The issues... (cont.)

- There was a huge amount of inconsistency:
  - Between teams
  - Between tracking of work
  - In approving work
  - In artefacts created for delivery
    - content, designs, where stored,etc.
- Understanding the existing build
  - 20+ years of integrating a build into a library system!
  - Exactly what is GXP?
  - Any experience now retired!

The issues... (cont.)

- RTC/RDz environment: Many teething problems – to be expected
  - Build
    - Fine for production / integration
    - Many tweaks required for developer and service builds
  - Cruise plug-in missing some components
  - Service process
    - Technically flawed
    - Over engineered
    - Being revisited
  - New development process
    - Steep learning curve for some

The benefits...

- No longer a 2 year upfront commitment
- Integrated work item and reporting tools
  - Transparency of work items, dependencies and status
  - Ability to ignore what is green and focus on the issues
  - Risks/Actions integrated
- Responsiveness to business needs – aiming for greater value, earlier
  - Beta deliveries much earlier
  - Ability to adapt to change much easier
  - Flexible resource pool
  - Current release – much more Agile in terms of requirements and prioritisation

The benefits... (cont.)

- Dashboards
  - Instant status reporting – Everyone has a much clearer view of project
  - Weekly status (Scrum of Scrums)
    - The preparation for that meeting from 15-30mins to <5mins
    - Approximately 15 teams
    - Reduced the weekly status meeting from 90mins to 60mins
    - The meeting is now dealing with issues
  - Can have dashboards show information across multiple projects
  - Project, team and individual dashboards – all configurable
- Iteration quality checks
  - End of iteration criteria preparation from ~ 2-4 hours per team <15mins
  - End of iteration meeting from 2 hours to <1 hour

The benefits... (cont.)

- Mitigate against future development risks
  - Loss of specialized skills (particularly LCS)
- Common tool set and skills
  - Flexible Workforce
  - Far greater coordination of work
  - Improving our speed of delivery
  - Common tool set across
    - The department, teams, development and service
    - Hursley
    - IBM
- After 2 months of RTC/RDz – 90% working as well or better than VM

The benefits... (cont.)

- Now we have moved to a single stream set
  - Developer, Public, Best so far (BSF), Production (Increment)
  - No longer CICS and SM with the above
- Ability to cut new streams
  - Risky development
  - When beta required
    - Cut a new Production (INC) stream for beta
    - Development / Testing continues on other streams
    - Packaging and Regression testing on beta stream
    - Defects found in beta stream promoted to Production
    - Only those required for beta merged from Production to beta
    - No more code freezes!

The benefits... (cont.)

- Quality has improved
  - Defects in RTC – defects part of the backlog
  - Pre-2007 defects peaked at 600+ - Waterfall
  - 2007-2009 defects focused on much earlier (450 peak) – Agile/Iterative
  - 2009-2011 again lower levels of defects (350 peak) – Agile/Iterative+RTC
- Far greater ability to use Agile practices
  - Most teams working in 1 month iterations. Could not have been imagined 2 years ago
- Far greater collaboration across teams
- Far greater notification when things change

The benefits... (cont.)

- Far less post-release 'churn' – seems more in control
- Value for the engineer
  - Skills in a 'industry leading' environment
  - Much reduced learning curve
    - Eclipse or Web interface
  - Easier to move roles within CICS or Hursley
- You can trust people
- The more you use RTC, the more ways you find of using it
  - For example, community projects

Where are we now?

- Further adoption of Agile practices
  - Moved to 1 month iterations for most teams – Still 4 month betas
  - Story point sizing
  - User story definitions (Actor, goal, value)
  - Iteration planning, priority based, team driven commitment
  - Backlog monitoring
- All planning, development and service being done through RTC
- New customer requirements solution – linking into RTC (RRC?)
- Investigating RRC
- Further deployment of RQM
- Using CLM projects to link RRC/RTC/RQM for complete integration

Summary

- Moving from Waterfall to Agile development is possible
  - It doesn't have to be just 'new projects'
  - It can have a staged approach
  - Does not have to be for all aspects e.g. SCM
- Adopting RTC will provide a catalyst for change
  - Evangelists and the support of senior management are key
  - Re-examines processes in a new way
    - Helps to identified holes and errors in team processes

Summary (cont.)

- RTC / RDz does provide an end-to-end development environment
  - Flexible, auditable, functional, easy to pick up the basics
  - 90% of adoption driven by the teams, not management
  - Extensible through the Jazz Team Server and RRC and RQM
  - More you use it the more ways you find to use it
- Expect teething problems – be Agile in fixing them
- If you need multiple projects
  - Think ahead to the architectural design and configuration
  - Who needs access to what without restricting their work
  - Consider provider / consumer projects

www.ibm/software/rational

Backup Slides

www.ibm.com/uk/hursley/

Where is Hursley with Jazz today?

- Hursley – IBM's largest software lab in Europe
- Diverse heritage
  - Spitfire designed here
  - CICS TS for zOS
  - CICS TG
  - CICS Tools
  - WebSphere Application Server
  - WebSphere MQ
  - WebSphere MQ Broker
  - Java
  - Tivoli Netcool
  - Cast Iron
  - Many others

Jazz Adoption

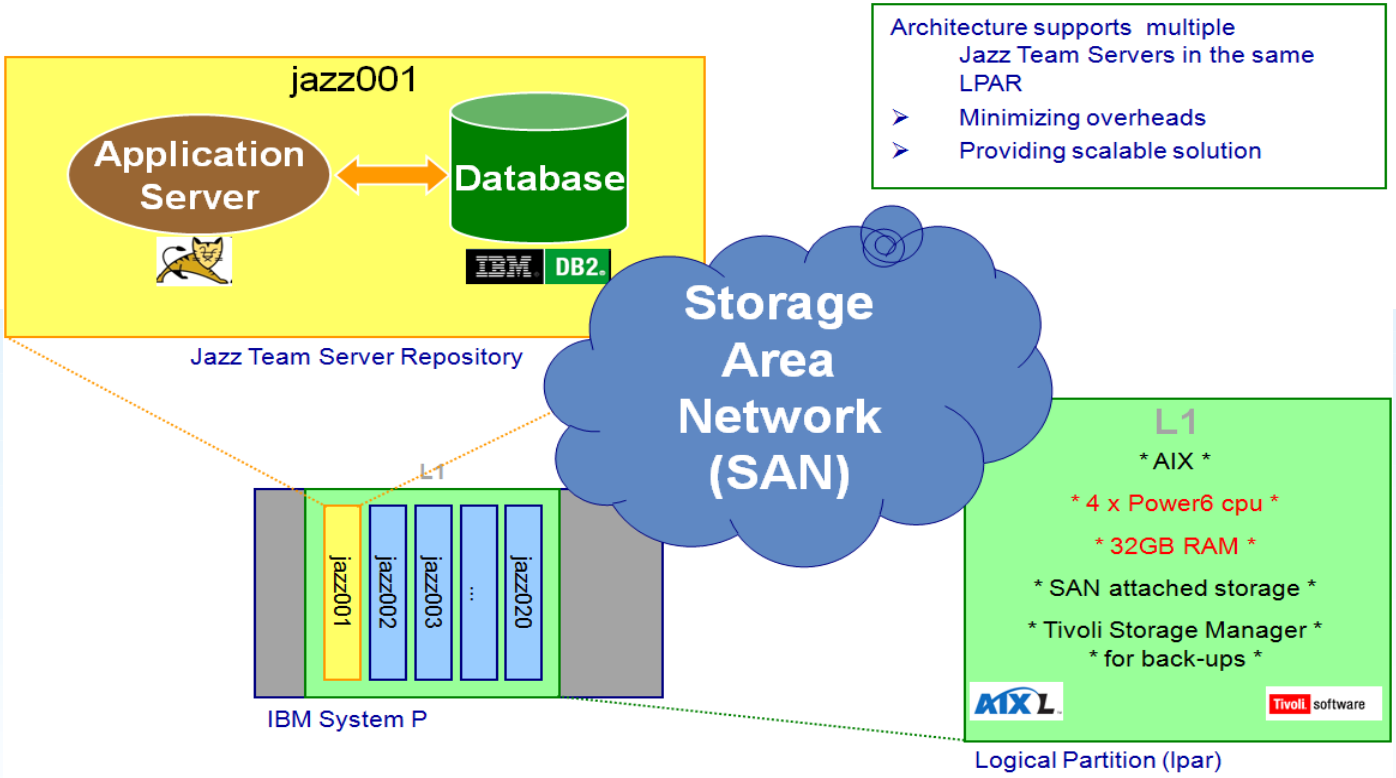
- Started with the provision of a central Jazz Service in June 2008
  - Hosting Jazz repositories, their back-up, infrastructure and maintenance
  - Agile approach to Jazz Service delivery
    - Customer rather than process focused
- Early in 2009 the Hursley Jazz Community was established
  - Monthly meetings to collaborate and share best practice
  - Provide help to users requesting it
  - Pull ideas and best practice from the Jazz Community
  - Give solutions based upon these ideas back out to our customers
- Then the Jazz @ Hursley website was set up to:
  - Provide access to background information
  - Provide access to the Jazz clients supported
  - Compliment, not replace, the official jazz.net site

Team Adoption – Hursley's portfolio

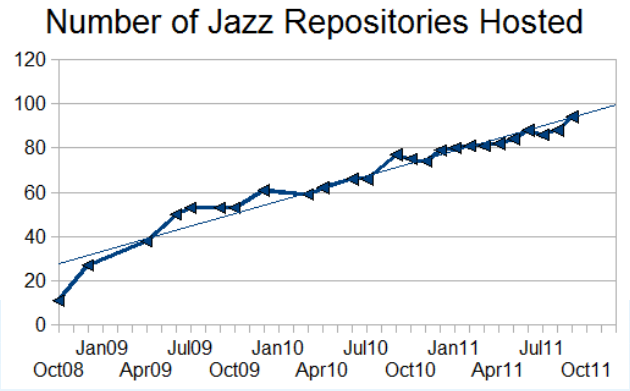
Product	RTC Work items	RTC SCM	RQM Test Mgmt
C1	Y	Plan	Y
C2	Y	Plan	N
M1	Y	Plan	Y
M2	Y	Plan*	Plan
W1	Y	Y	P4P
WB2	Y	Y	Y
W2	Y	Y	P4P
WB3	P4P	P4P	P4P
RR1	Plan	Plan	Plan
RR2	Y	Y	N
E1	Plan*	Plan*	Plan*
WB1	Y	Y	Plan*
IM1	Y	Plan	Y
Co1	Y	P4P	N
J1	Y	Y	N
J2	Y*	Y*	Y*
Ra1	Y	Synergy	Y
T1	ClearQuest P4P	ClearCase P4P	Y
ST1	Y		
H	Y	Y	Y

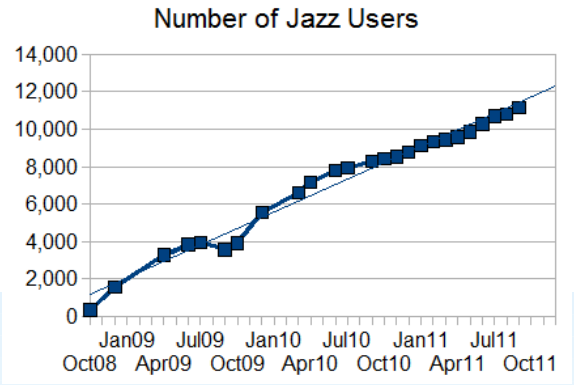
Circa 4Q2010

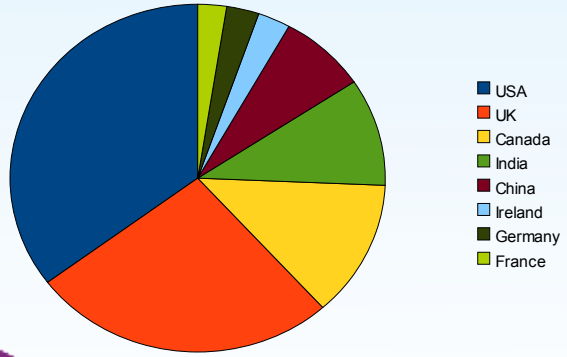
Hursley's centrally hosted approach



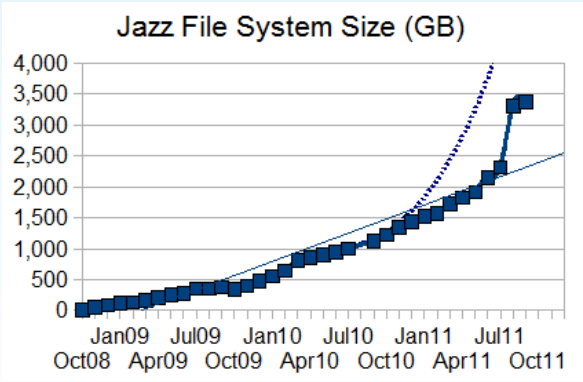
Hursley Statistics







Demographics

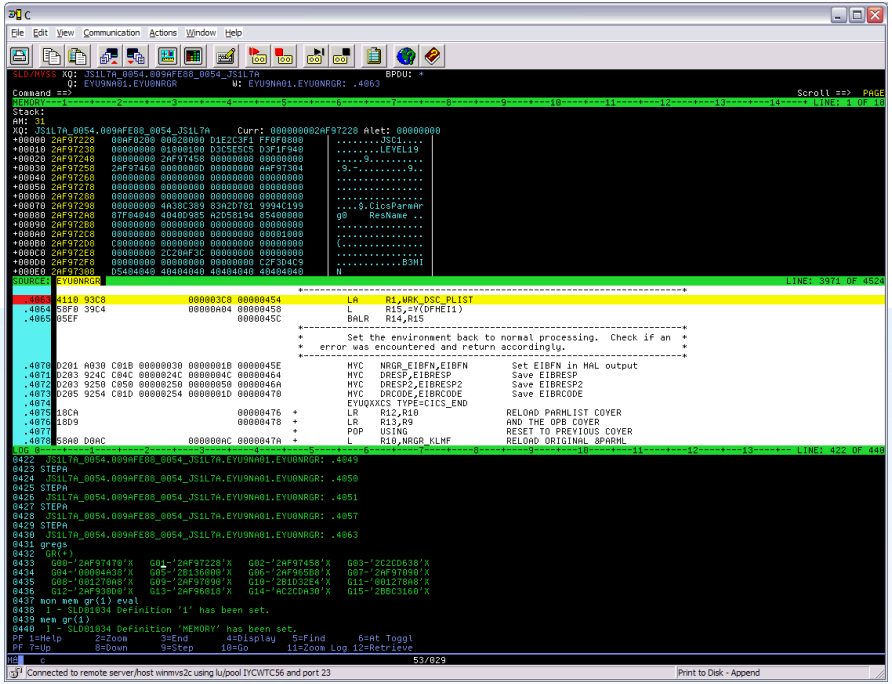


Further Backup Slides

Ye olde development way (Not that it's bad!)

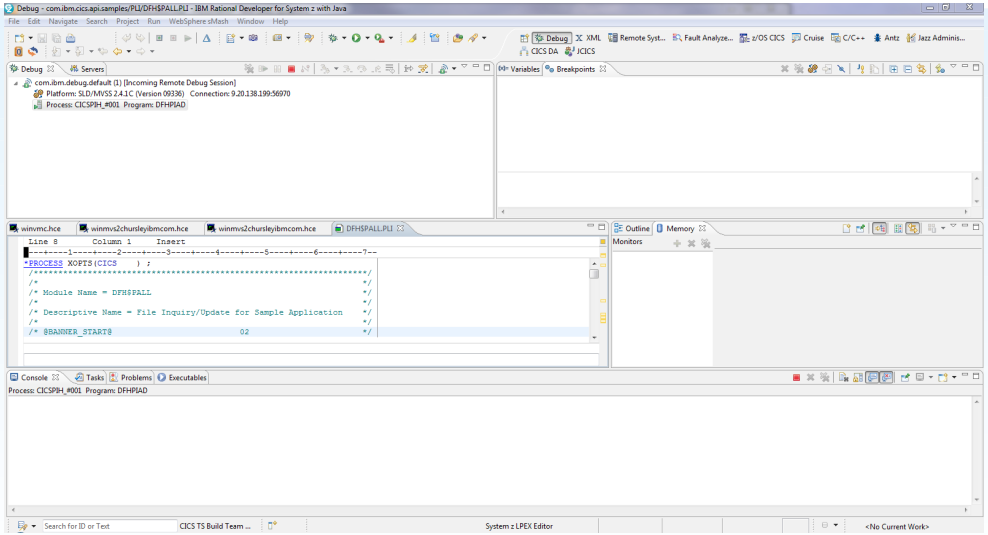
- VM environment
  - Cruise tool – Sophisticated library search and cross reference
  - MPU tool - Create an MPU
  - Check out using Alter associate to MPU
  - Edit using xedit in VM
  - Check in using catlg
  - Use MPU tool to correctly associate composites
  - MPUBATCH tool to build the code
  - Repeat until clean build
  - Use MVS2CALL to transfer to MVS
  - Repeat until unit tested
  - Use MPUREV to get code reviewed and approved
  - Use MPU to promote the code
- Pretty much all green screen

Debugging environment



SLD*
← Ye Olde Way

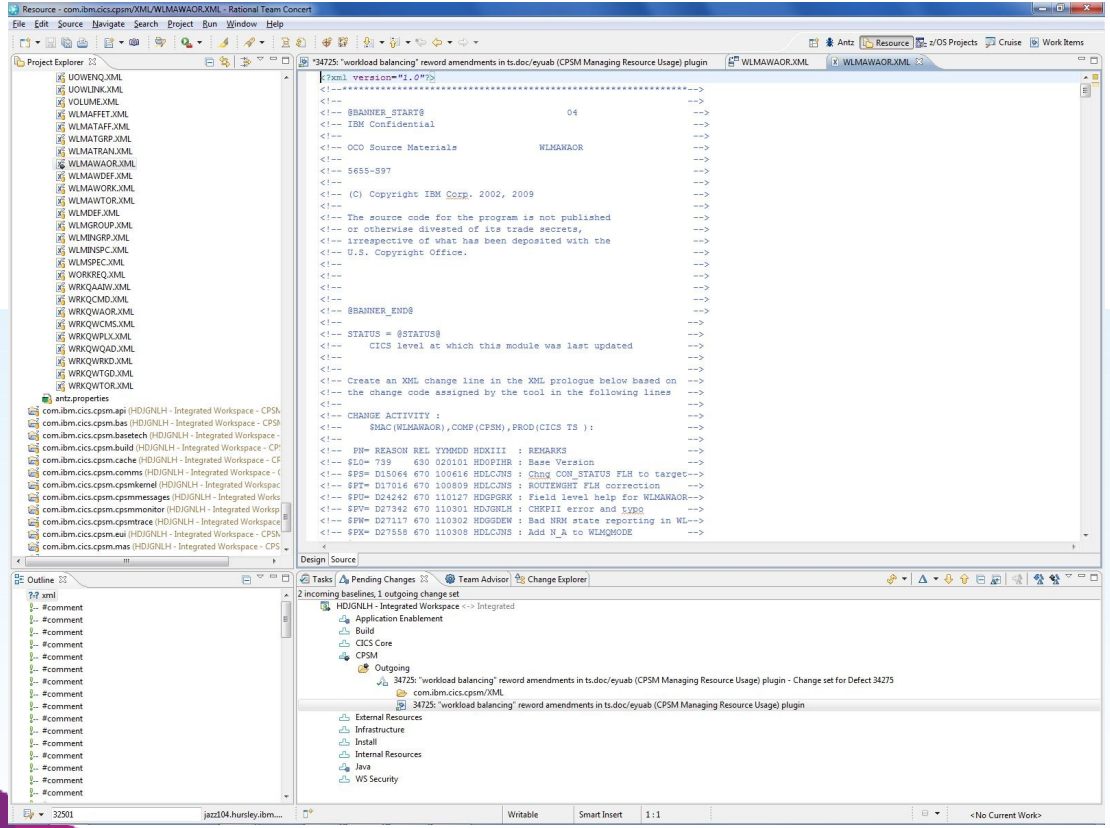
*SLD (Source Level Debugger) – Very powerful zOS debugging tool



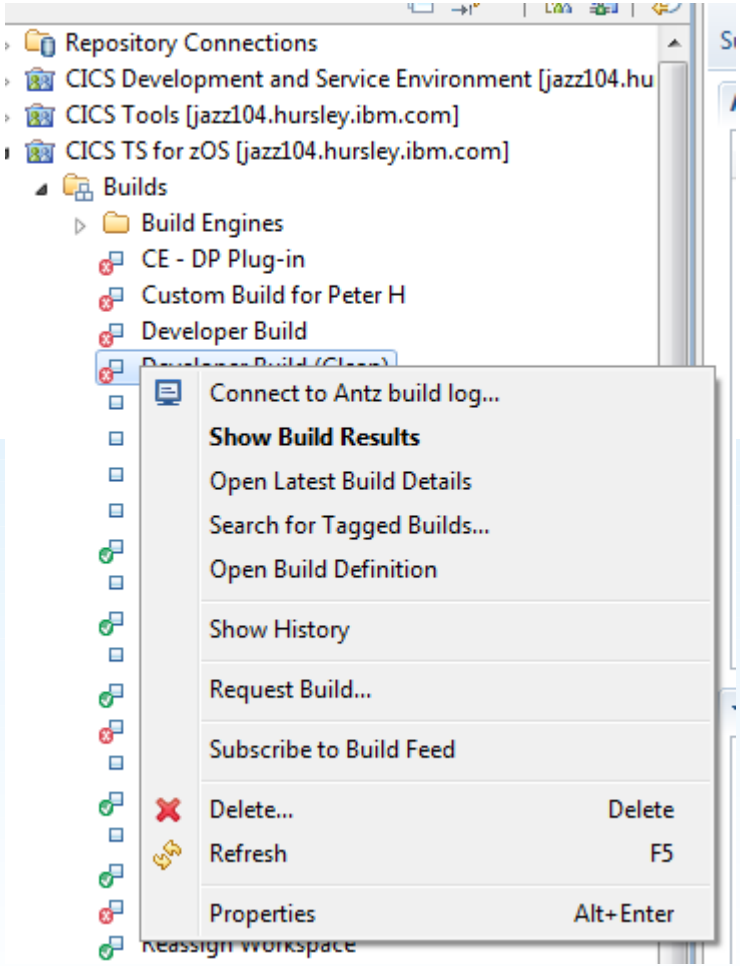
RDz Debug Perspective
The New Way →

Innovate2011

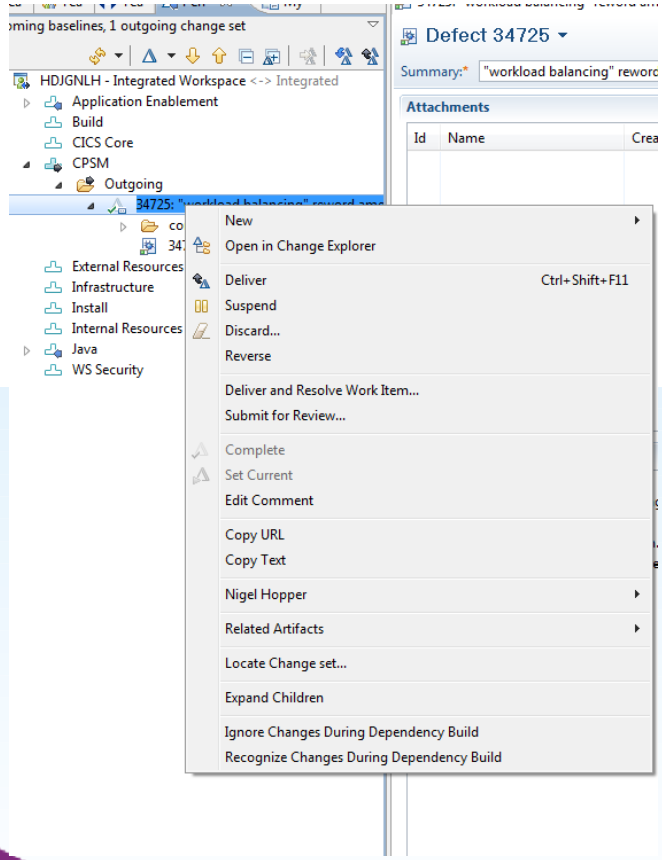
Code change

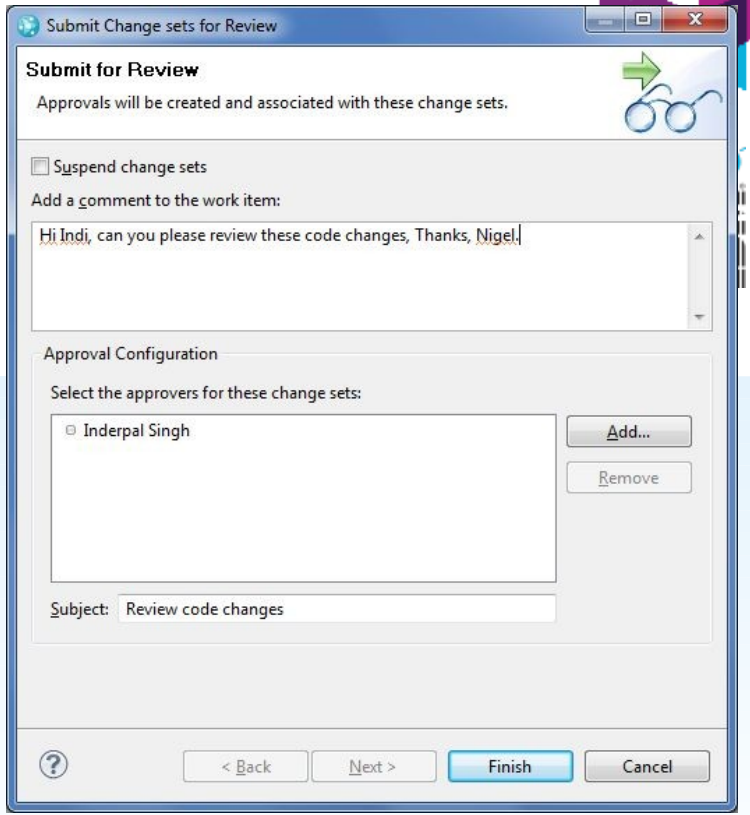

Resource - com.ibm.cics.cpsm/XML/WLMAWAOR.XML - Rational Team Concert
File Edit Source Navigate Search Project Run Window Help
Project Explorer
UOWENQ.XML
UOWLINK.XML
VOLUME.XML
WLMAFFET.XML
WLMATAFF.XML
WLMATGRP.XML
WLMATRAN.XML
WLMAWAOR.XML
WLMAWDEF.XML
WLMAWORK.XML
WLMAWTOR.XML
WLMDEF.XML
WLMGROUP.XML
WLMINGRP.XML
WLMINSPC.XML
WLMSPEC.XML
WORKREQ.XML
WRKQAAIW.XML
WRKQCMD.XML
WRKQWAOR.XML
WRKQWCMS.XML
WRKQWPLX.XML
WRKQWQAD.XML
WRKQWRKD.XML
WRKQWTGD.XML
WRKQWTOR.XML
antz.properties
com.ibm.cics.cpsm.api (HDJGNLH - Integrated Workspace - CPSM
com.ibm.cics.cpsm.bas (HDJGNLH - Integrated Workspace - CPSM
com.ibm.cics.cpsm.basetech (HDJGNLH - Integrated Workspace -
com.ibm.cics.cpsm.build (HDJGNLH - Integrated Workspace - CPS
com.ibm.cics.cpsm.cache (HDJGNLH - Integrated Workspace - CP
com.ibm.cics.cpsm.comms (HDJGNLH - Integrated Workspace - (
com.ibm.cics.cpsm.cpsmkernel (HDJGNLH - Integrated Workspac
com.ibm.cics.cpsm.cpsmmessages (HDJGNLH - Integrated Works
com.ibm.cics.cpsm.cpsmmonitor (HDJGNLH - Integrated Worksp
com.ibm.cics.cpsm.cpsmtrace (HDJGNLH - Integrated Workspace
com.ibm.cics.cpsm.eui (HDJGNLH - Integrated Workspace - CPSM
com.ibm.cics.cpsm.mas (HDJGNLH - Integrated Workspace - CPS
*34725: "workload balancing" reword amendments in ts.doc/eyuab (CPSM Managing Resource Usage) plugin
WLMAWAOR.XML
WLMAWAOR.XML
<?xml version="1.0"?>
<!-- @BANNER_START@ 04 -->
<!-- IBM Confidential -->
<!-- OCO Source Materials WLMAWAOR -->
<!-- 5655-S97 -->
<!-- (C) Copyright IBM Corp. 2002, 2009 -->
<!-- The source code for the program is not published -->
<!-- or otherwise divested of its trade secrets, -->
<!-- irrespective of what has been deposited with the -->
<!-- U.S. Copyright Office. -->
<!-- @BANNER_END@ -->
<!-- STATUS = @STATUS@ -->
<!-- CICS level at which this module was last updated -->
<!-- Create an XML change line in the XML prologue below based on -->
<!-- the change code assigned by the tool in the following lines -->
<!-- CHANGE ACTIVITY : -->
<!-- $MAC(WLMAWAOR),COMP(CPSM),PROD(CICS TS): -->
<!-- PN= REASON REL YYMMDD HDXIII : REMARKS -->
<!-- $L0= 739 630 020101 HD0PIHR : Base Version -->
<!-- $PS= D15064 670 100616 HDLCJNS : Chng CON_STATUS FLH to target-->
<!-- $PT= D17016 670 100809 HDLCJNS : ROUTEWGHT FLH correction -->
<!-- $PU= D24242 670 110127 HDGPGRK : Field level help for WLMAWAOR-->
<!-- $PV= D27342 670 110301 HDJGNLH : CHKPII error and typo -->
<!-- $PW= D27117 670 110302 HDGGDEN : Bad NRM state reporting in WL-->
<!-- $PX= D27558 670 110308 HDLCJNS : Add N_A to WLMQMODE -->
Design Source
Outline
Tasks Pending Changes Team Advisor Change Explorer
2 incoming baselines, 1 outgoing change set
HDJGNLH - Integrated Workspace <-> Integrated
Application Enablement
Build
CICS Core
CPSM
Outgoing
34725: "workload balancing" reword amendments in ts.doc/eyuab (CPSM Managing Resource Usage) plugin - Change set for Defect 34275
com.ibm.cics.cpsm/XML
34725: "workload balancing" reword amendments in ts.doc/eyuab (CPSM Managing Resource Usage) plugin
External Resources
Infrastructure
Install
Internal Resources
Java
WS Security
Writable
Smart Insert
1 : 1
<No Current Work>

Build


Repository Connections
CICS Development and Service Environment [jazz104.hu
CICS Tools [jazz104.hursley.ibm.com]
CICS TS for zOS [jazz104.hursley.ibm.com]
Builds
Build Engines
CE - DP Plug-in
Custom Build for Peter H
Developer Build
Connect to Antz build log...
Show Build Results
Open Latest Build Details
Search for Tagged Builds...
Open Build Definition
Show History
Request Build...
Subscribe to Build Feed
Delete... Delete
Refresh F5
Properties Alt+Enter

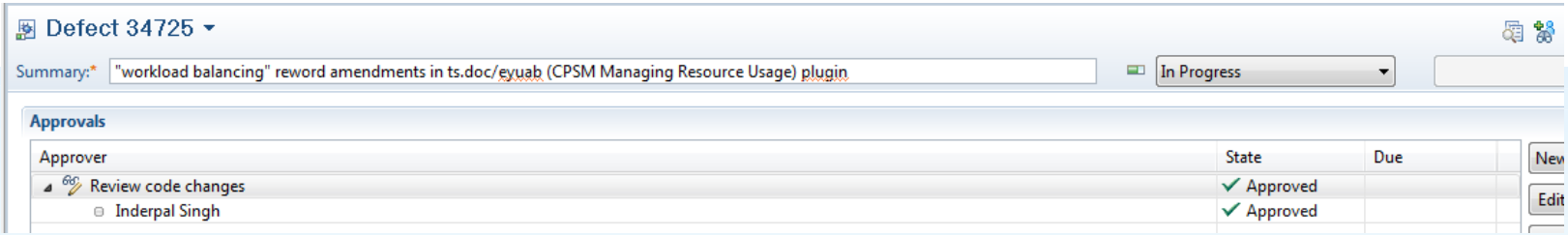
Review 1


ming baselines, 1 outgoing change set
HDJGNLH - Integrated Workspace <-> Integrated
Application Enablement
Build
CICS Core
CPSM
Outgoing
External Resources
Infrastructure
Install
Internal Resources
Java
WS Security
Defect 34725
Summary:*
"workload balancing" reworc
Attachments
Id
Name
Crea
New
Open in Change Explorer
Deliver
Ctrl+Shift+F11
Suspend
Discard...
Reverse
Deliver and Resolve Work Item...
Submit for Review...
Complete
Set Current
Edit Comment
Copy URL
Copy Text
Nigel Hopper
Related Artifacts
Locate Change set...
Expand Children
Ignore Changes During Dependency Build
Recognize Changes During Dependency Build


Submit Change sets for Review
Submit for Review
Approvals will be created and associated with these change sets.
Suspend change sets
Add a comment to the work item:
Hi Indi, can you please review these code changes, Thanks, Nigel.
Approval Configuration
Select the approvers for these change sets:
Inderpal Singh
Add...
Remove
Subject:
Review code changes
< Back
Next >
Finish
Cancel

Review 2

3. Nigel Hopper, 19 Aug 2011, 14:54
Hi Indi, Please could you check this, Thanks, Nigel.



Review 3a

Change Sets

CPSM - Nigel Hopper - Change set for Defect 34275 19-Aug-2011 14:54

WLMAWAOR.XML (after) (read-only)	WLMAWAOR.XML (before) (read-only)
These refresh requests will be	These refresh requests will be
issued by a routing region that is evaluating this region	issued by a routing region that is evaluating this region
as a possible target for a dynamic routing request.	as a possible target for a dynamic routing request.
<p>	<p>
The value range is from 0 to 2000, and represents	The value range is from 0 to 2000, and represents
units of milliseconds:	units of milliseconds:
</p>	</p>
When the Optimization status is not set to Active, then	When the Optimization status is not set to Active, then
this value will be set to 0, and WLM will ignore it.	this value will be set to 0, and WLM will ignore it.
When the Optimization status is set to Active, then	When the Optimization status is set to Active, then
a value of 0 means that a routing region will request a	a value of 0 means that a routing region will request a
status update of a target region on every occasion that	status update of a target region on every occasion that
it examines this region's status.	it examines this region's status.
Values between 1 and 2000 specify the minimum	Values between 1 and 2000 specify the minimum
time interval that must expire before the status of	time interval that must expire before the status of
this region can be refreshed.	this region can be refreshed.
<p>	<p>
A low interval value means that the CFDT Server will be	A low interval value means that the CFDT Server will be
polled more often for a status update, than for a higher	polled more often for a status update, than for a higher
value. For workloads in QUEUE mode, this	value. For workloads in QUEUE mode, this
will result in a task load more evenly distributed across	will result in a task load more evenly balanced across
the CICS regions in the workload target scope	the CICS regions in the workload target scope

Review 3b

Change Sets

Change set for Defect 34275

Modified on: 19 Aug 2011 14:54 (Last Week)
Author: Nigel Hopper
Comment: Change set for Defect 34275
Reasons: 34725: "workload balancing" reword amendments in ts.doc/eyuab (CPSM Managing Resource Usage) plugin
Changes: com.ibm.cics.cpsm/XML/WLMAWAOR.XML (Modified)

diff: com.ibm.cics.cpsm/XML/WLMAWAOR.XML - Before | After

```
@@ -40,6 +40,7 @@
 <!-- $PX= D27558 670 110308 HDLCJNS : Add N_A to WLMQMODE          -->
 <!-- $PY= D27737 670 110309 HDLCJNS : Add note to WLMTHRSH FLH      -->
 <!-- $PZ= D28323 670 110322 HDJGNLH : Dates and FLH Fixes           -->
+<!-- $D0= D34735 680 110819 HDJGNLH : FLH Fixes                     -->
 <!-- $L2= 868    650 061024 HDJLDD  : FLH WLM Views                 -->
 <!-- $L3= 868    650 061206 HDJLDD  : FLH Phrase Alterations        -->
 <!-- $L4= 868    650 070306 HDJLDD  : Remove invalid html tags      -->
@@ -457,7 +458,7 @@
            A low interval value means that the CFDT Server will be
            polled more often for a status update, than for a higher
            value.  For workloads in QUEUE mode, this
-           will result in a task load more evenly balanced across
+           will result in a task load more evenly distributed across
            the CICS regions in the workload target scope
            &lt;b&gt;
            (assuming all other health and link factors are equal)
@@ -478,7 +479,7 @@
            region using the MASs known to CICSplex views,
            or the CICS system definition views.
            &lt;/p&gt;
-         </eg:help>                                    <!--@PKC-->
+         </eg:help>                                    <!--@D0C-->
          <datatype>
            <basetype name='numeric'/>
            <eg:default>200</eg:default>
@@ -554,7 +555,7 @@
            the 1-25 scale, then that will cause an increase in
            the frequency of updates to the RS Server across the
            task load range. For workloads in QUEUE mode, this
-           will result in a task load more evenly balanced across
+           will result in a task load more evenly distributed across
            the CICS regions in the workload target scope
            &lt;b&gt;
            (assuming all other health and link factors are equal)
@@ -569,7 +570,7 @@
            region using the MASs known to CICSplex views,
            or the CICS system definition views.
            &lt;/p&gt;
-         </eg:help>                                    <!--@PKC-->
+         </eg:help>                                    <!--@D0C-->
          <datatype>
            <basetype name='numeric'/>
            <eg:default>15</eg:default>
```