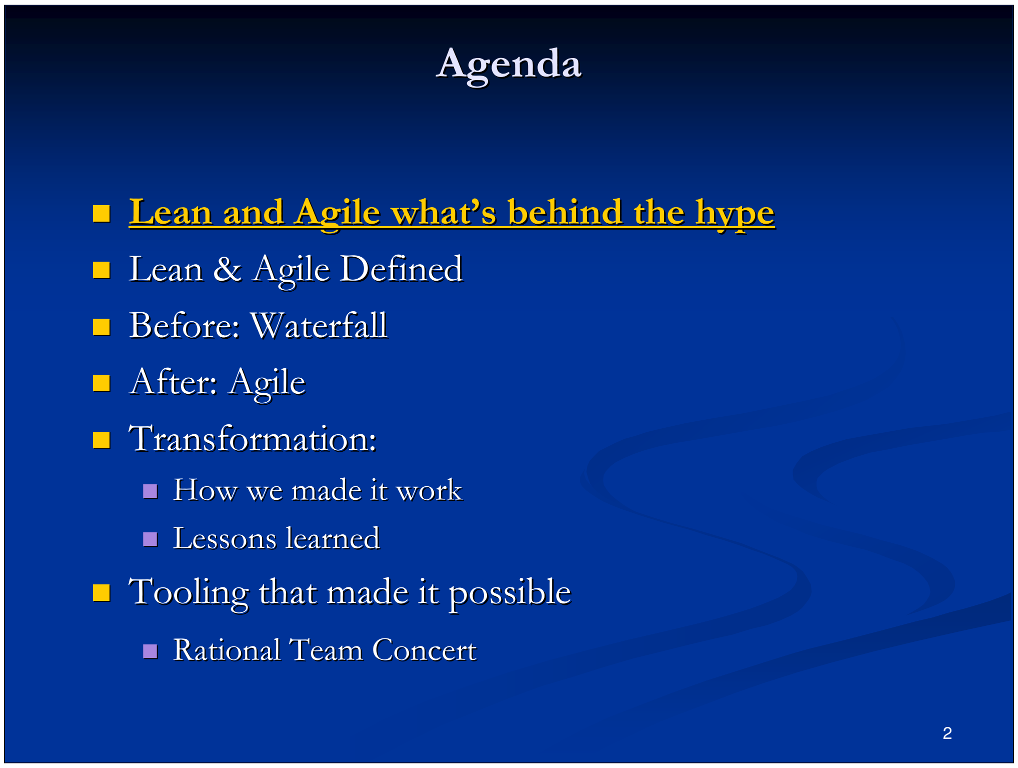
Lean / Agile Programming in a Mainframe World

by: Zamir Gonzalez

z Tools and Transformation Team



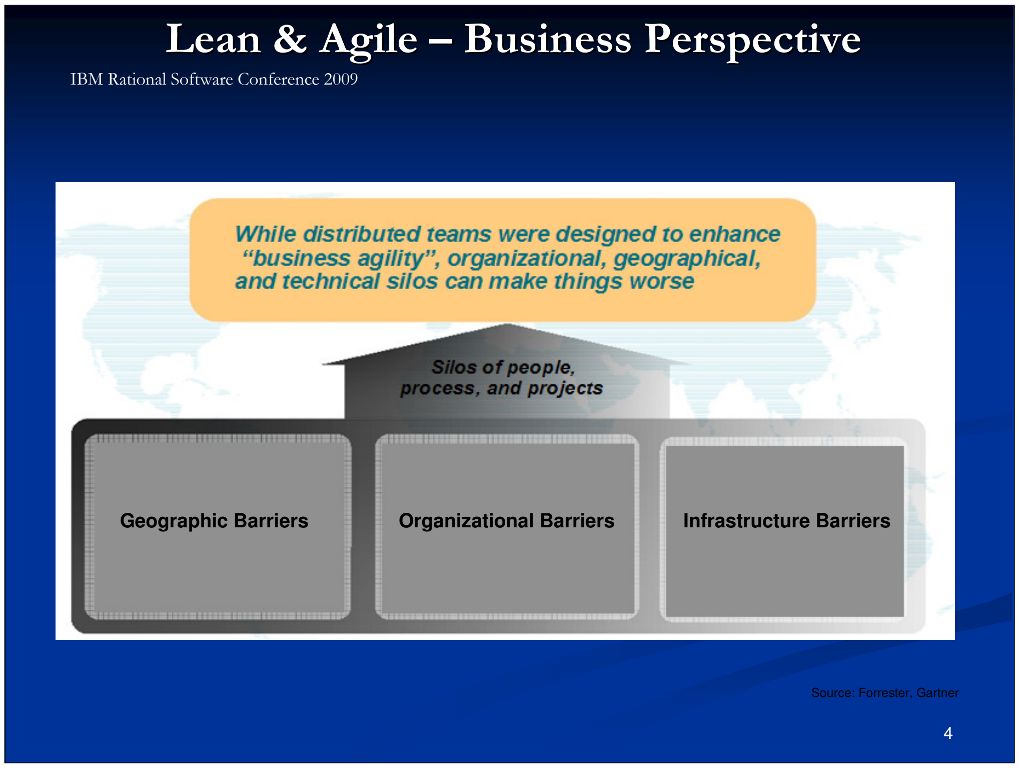
Agenda

- **Lean and Agile what's behind the hype**
- Lean & Agile Defined
- Before: Waterfall
- After: Agile
- Transformation:
  - How we made it work
  - Lessons learned
- Tooling that made it possible
  - Rational Team Concert

2

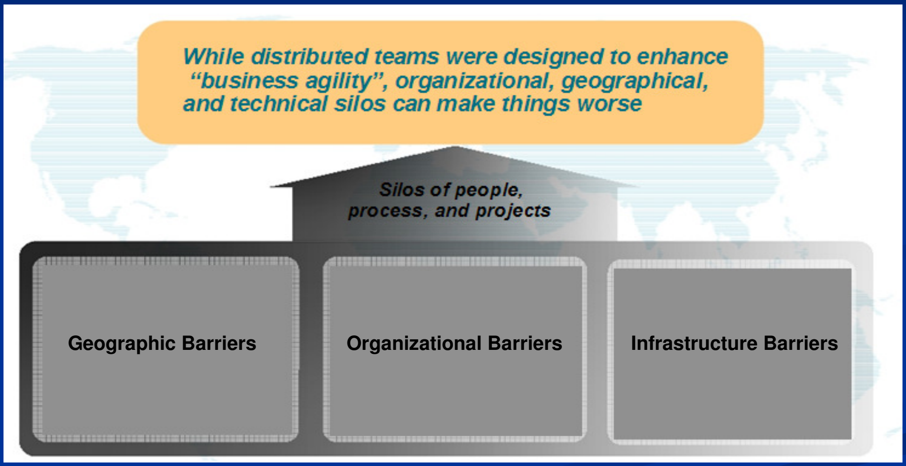
If it ain’t broke, don’t fix it

3



Lean & Agile – Business Perspective

IBM Rational Software Conference 2009


While distributed teams were designed to enhance "business agility", organizational, geographical, and technical silos can make things worse
Silos of people, process, and projects
Geographic Barriers
Organizational Barriers
Infrastructure Barriers

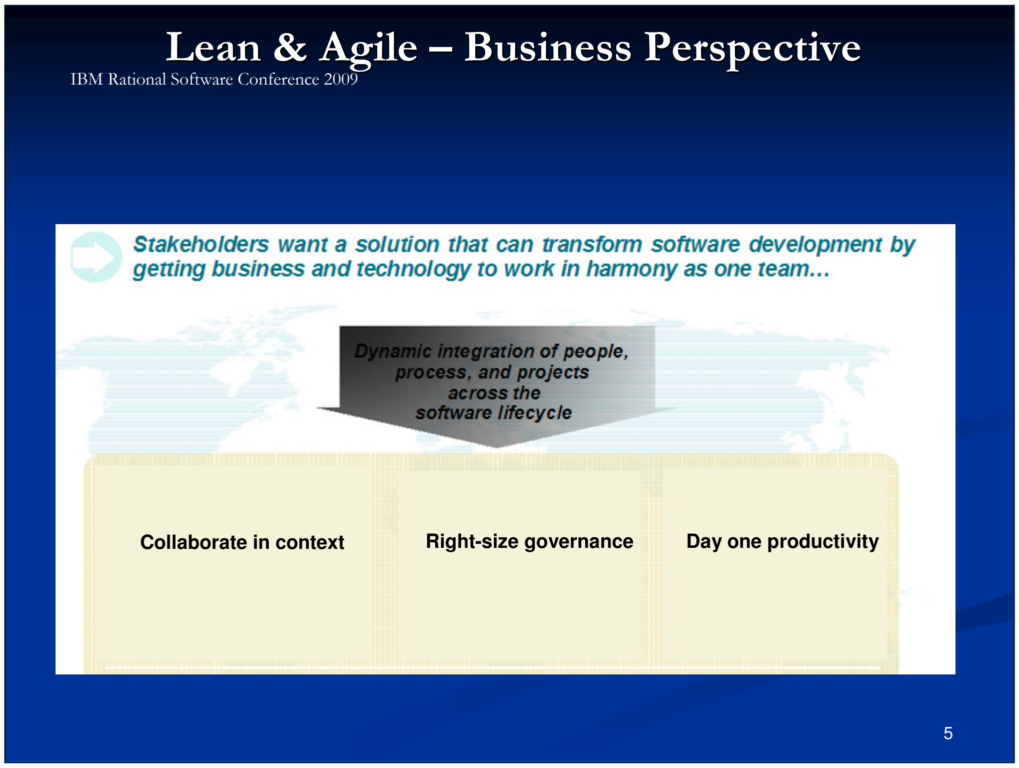
Source: Forrester, Gartner

4

Geographic Barriers: poor communication, language, culture and time differences, process gaps resulting in errors and rework, high degree of friction across teams

Organizational Barriers: lack of meaningful stakeholder input, poor line of business oversight, weak project governance, missed opportunities to leverage domain expertise

Infrastructure Barriers: incompatible tools and repositories, unreliable access to common artifacts, lengthy project and team on-boarding, brittle and inflexible tooling integrations



Lean & Agile – Business Perspective

IBM Rational Software Conference 2009

Stakeholders want a solution that can transform software development by getting business and technology to work in harmony as one team…

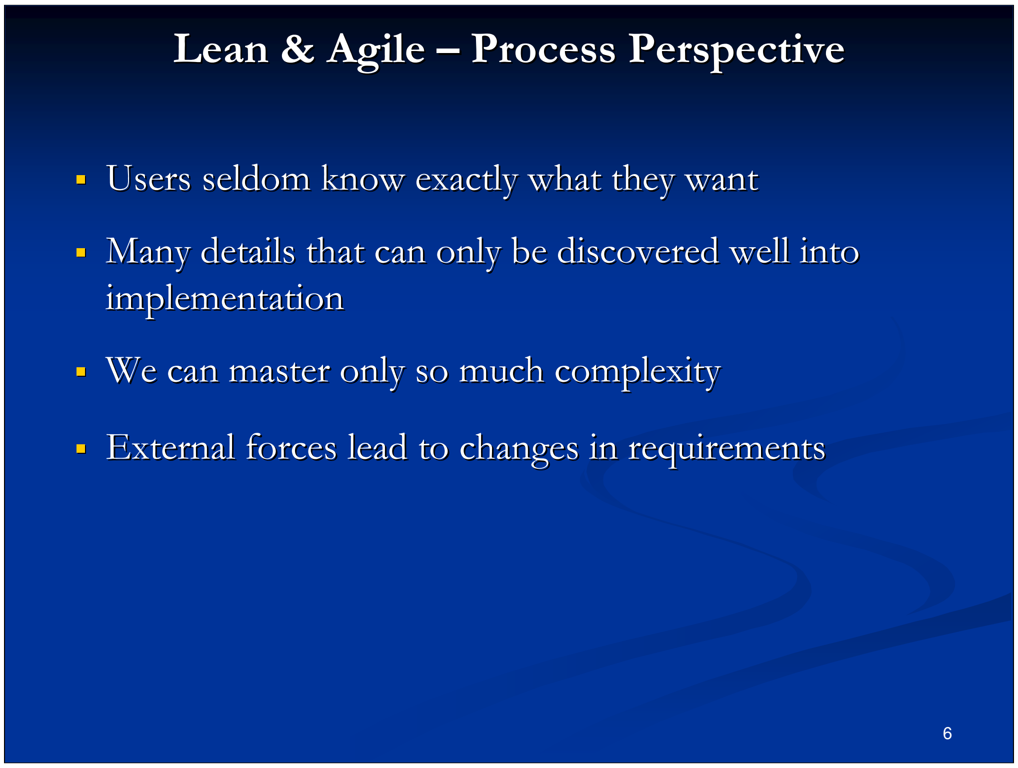
Dynamic integration of people, process, and projects across the software lifecycle
Collaborate in context
Right-size governance
Day one productivity

5

Collaborate in Context: enable team transparency, build team cohesion, automate hand-offs

Right-size governance: automate workflows through dynamic processes, automate data collection, real time reporting and alerts

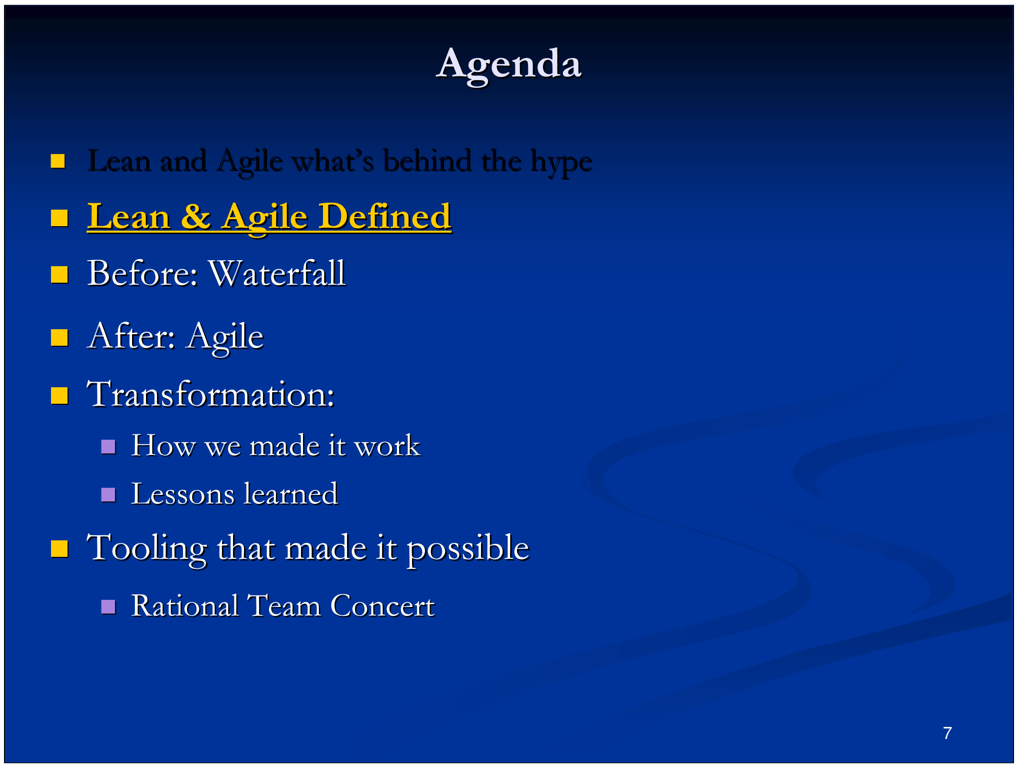
Day one productivity: dynamic provisioning of projects and teams, real-time release/iteration planning and workload balancing, unify teams



Lean & Agile – Process Perspective

- Users seldom know exactly what they want
- Many details that can only be discovered well into implementation
- We can master only so much complexity
- External forces lead to changes in requirements

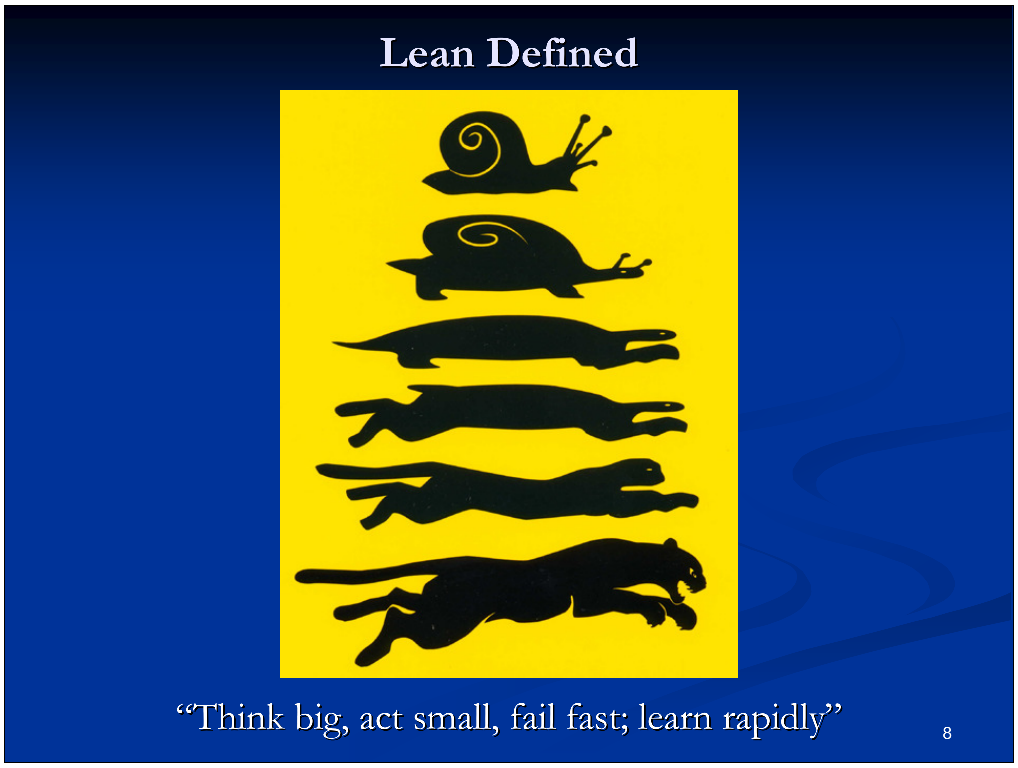
6



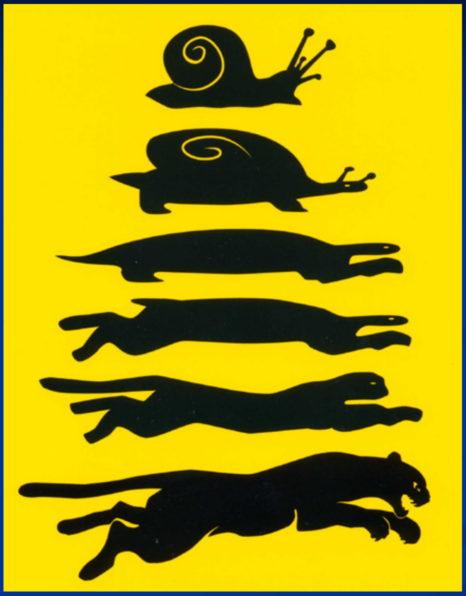
Agenda

- Lean and Agile what's behind the hype
- **Lean & Agile Defined**
- Before: Waterfall
- After: Agile
- Transformation:
  - How we made it work
  - Lessons learned
- Tooling that made it possible
  - Rational Team Concert

7



Lean Defined



"Think big, act small, fail fast; learn rapidly"

8

Lean Principles:

Eliminate waste: Everything not adding value to the customer is considered to be waste. This includes: unnecessary code and functionality, delay in the software development process, unclear requirements, bureaucracy, slow internal communication.

Amplify learning: scrums, short sprints allowing testing and feedback to come quickly, reflections session at end of sprint for group lessons learned

Decide as late as possible: use options-based approach for delaying decisions as much as possible until they can be made based on facts and not on uncertain assumptions and predictions. The more complex a system is, the more capacity for change should be built into it, thus enabling the delay of important and crucial commitments. The iterative approach promotes this principle – the ability to adapt to changes and correct mistakes, which might be very costly if discovered after the release of the system.

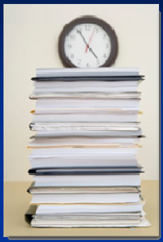
Deliver as fast as possible: -- just in time production ideology utilizing scrums and sprints

Empower the team -- developers should be given access to customer; the team leader should provide support and help in difficult situations, as well as make sure that skepticism does not ruin the team's spirit.

Build Integrity in -- The complete and automated building process should be accompanied by a complete and automated suite of developer and customer tests, having the same versioning, synchronization and semantics as the current state of the System. At the end the integrity should be verified with thorough testing, thus ensuring the System does what the customer expects it to.

See the whole

Lean: Eliminate waste


Internal Paperwork

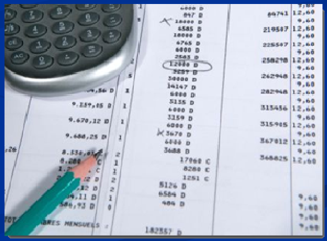
Backlog = Delivery Delay

Wait time = Lost $$$$

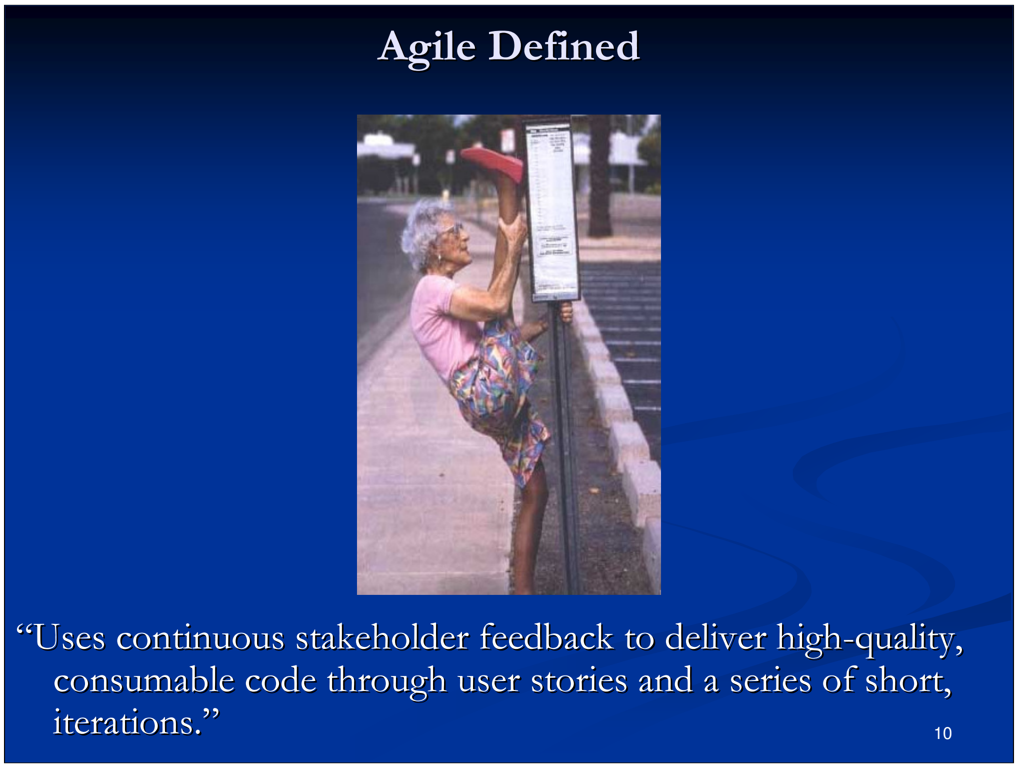
Red Tape & Change Control

Complexity

Defects


Extra Features Drive Cost Exponentially

9



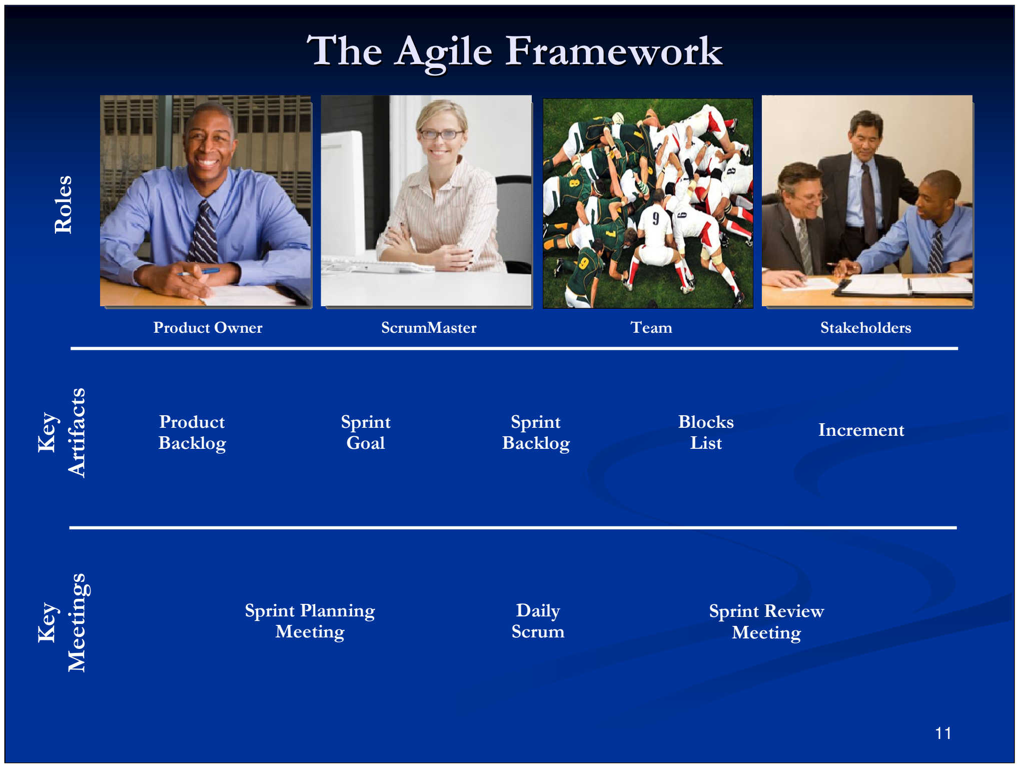
Agile Defined

"Uses continuous stakeholder feedback to deliver high-quality, consumable code through user stories and a series of short, iterations."

10

Core principles

- "fits just right" process
- continuous testing and validation
- consistent team collaboration
- rapid response to change
- ongoing customer involvement
- Frequent delivery of working software



The Agile Framework

Roles	Product Owner	ScrumMaster	Team	Stakeholders	
Key Artifacts	Product Backlog	Sprint Goal	Sprint Backlog	Blocks List	Increment
Key Meetings	Sprint Planning Meeting	Daily Scrum	Sprint Review Meeting		

11

At Start of Iteration

Development Process to be used (One Page)

Current Candidate List

At Start of Coding

List of Prioritized Selected Use Cases / Features to be delivered <u>this</u> iteration

Latest Architecture / Model Design Docs (not maintained : Frozen point in time. NOT auditable)

At End of Iteration

Demo

Delivered Code

Test Cases

Reflection / Status

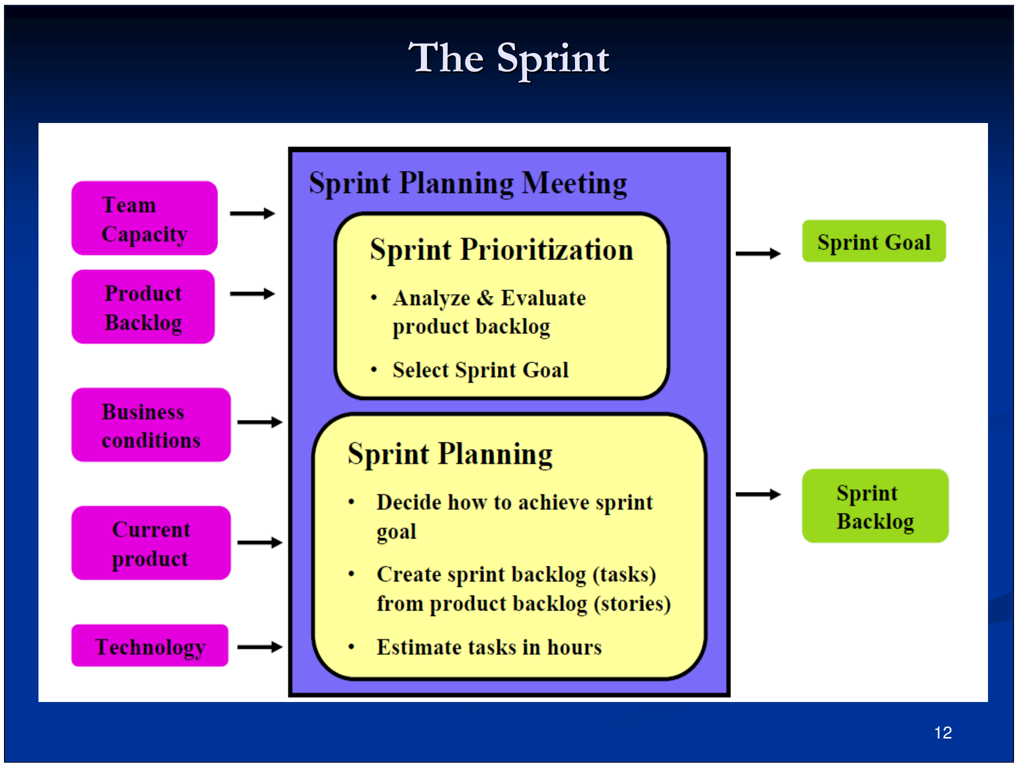
List of Use Cases / Features actually delivered (complete and tested)

Use Cases / Features not delivered (input to reflection)

Revised Development Process for next iteration

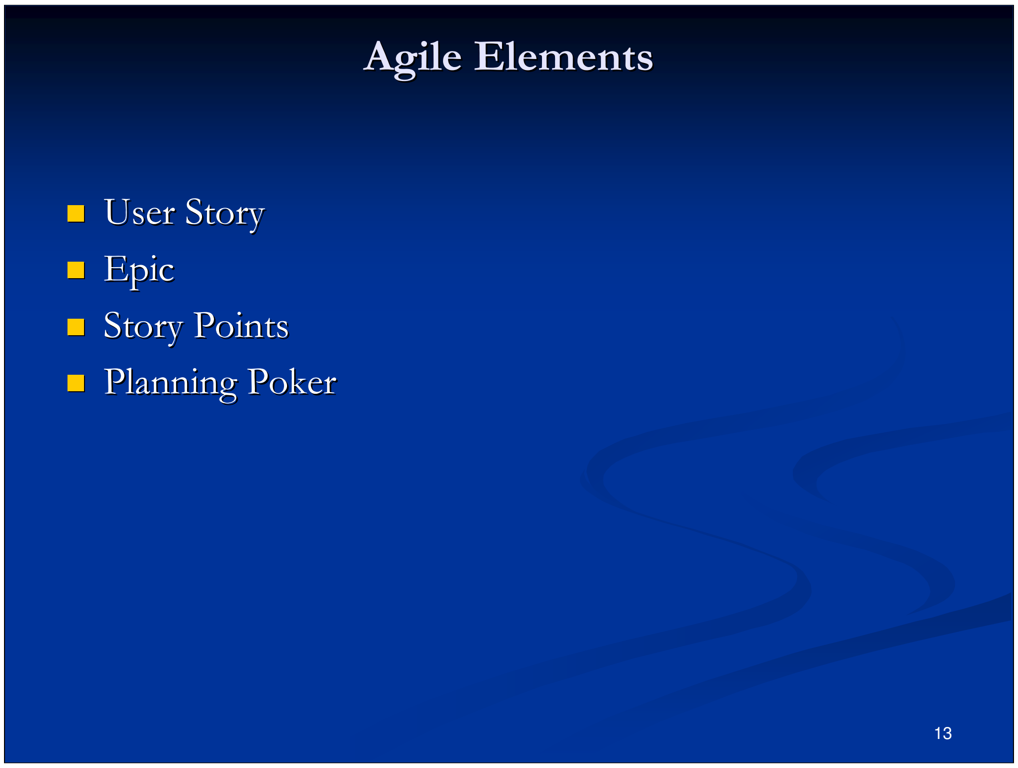
Product Backlog:

- A list of all desired work on the project
- Ideally expressed such that each item has value to the stakeholders
- Prioritized by the Product Owner
- Reprioritized at the start of each Sprint



The Sprint

12



Agile Elements

- User Story
- Epic
- Story Points
- Planning Poker

13

Roles:

Allows the team to think of the product in terms of solving needs of real people

Identifies a type of user engaged in reaching some goals w/ your product

Provides the team insight into what the person is engaged in doing – although not necess

Goals:

Needs to represent the user's goal

Should not be the technical solution

Should not be focused on advantages to the programmers developing the product

Business Value:

Allows the benefits to the customer to be apparent

Provides insight so stories can be intelligently prioritized

They are relative to the points of your other User Stories

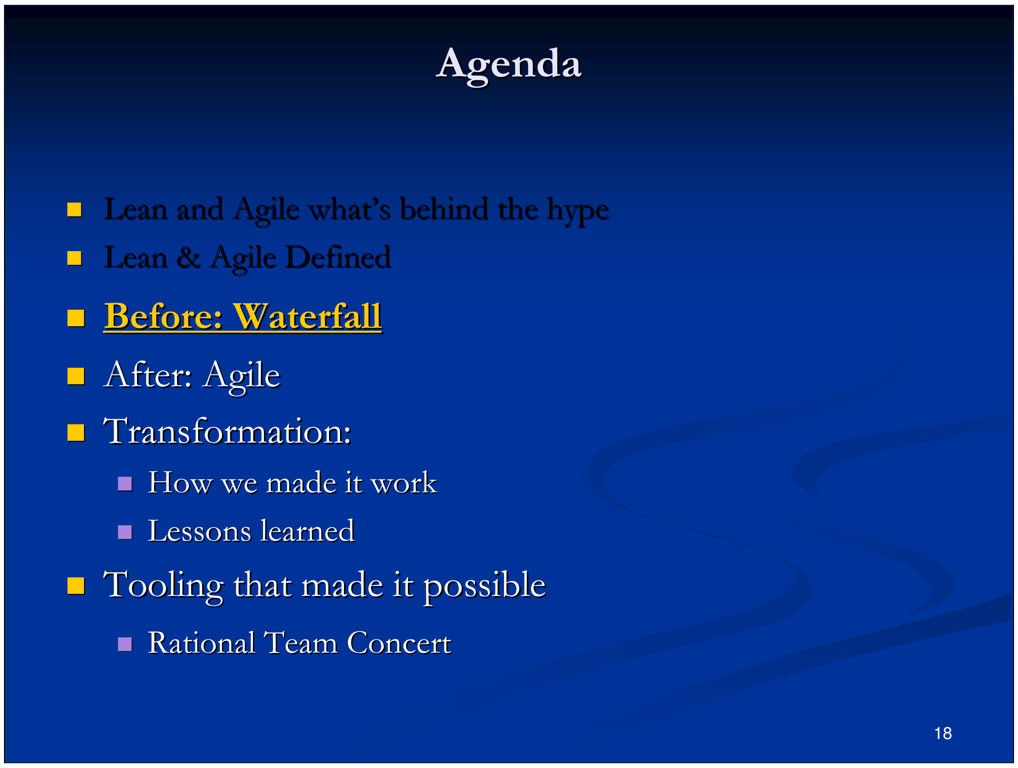
There is no sense of time in the measure

Establish 1 as a very simple effort, 5 as average, etc.

The team collaborates to size each user story

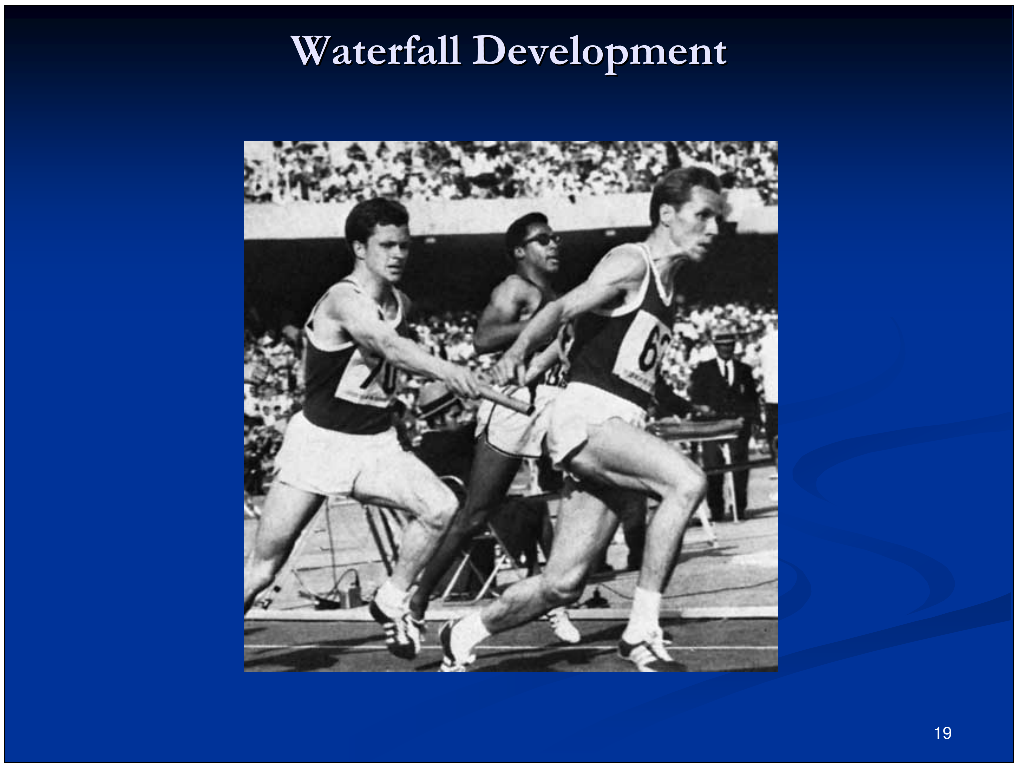
Accuracy is improved with history

- A way to help teams to estimate User Story Points
  - Each player has a Planning Poker deck
  - Each deck consists of 13 cards: ?, 0, ½, 1, 2, 3, 5, 8, 13, 20, 40, 100, and inf
- Every User Story is estimated
  - Points should include all the work to complete the story within the sprint
    - No matter what your role is, your card should take into account all required work
  - Players place their cards simultaneously
  - Players who play a higher or lower value explains why, and whole team play



Agenda

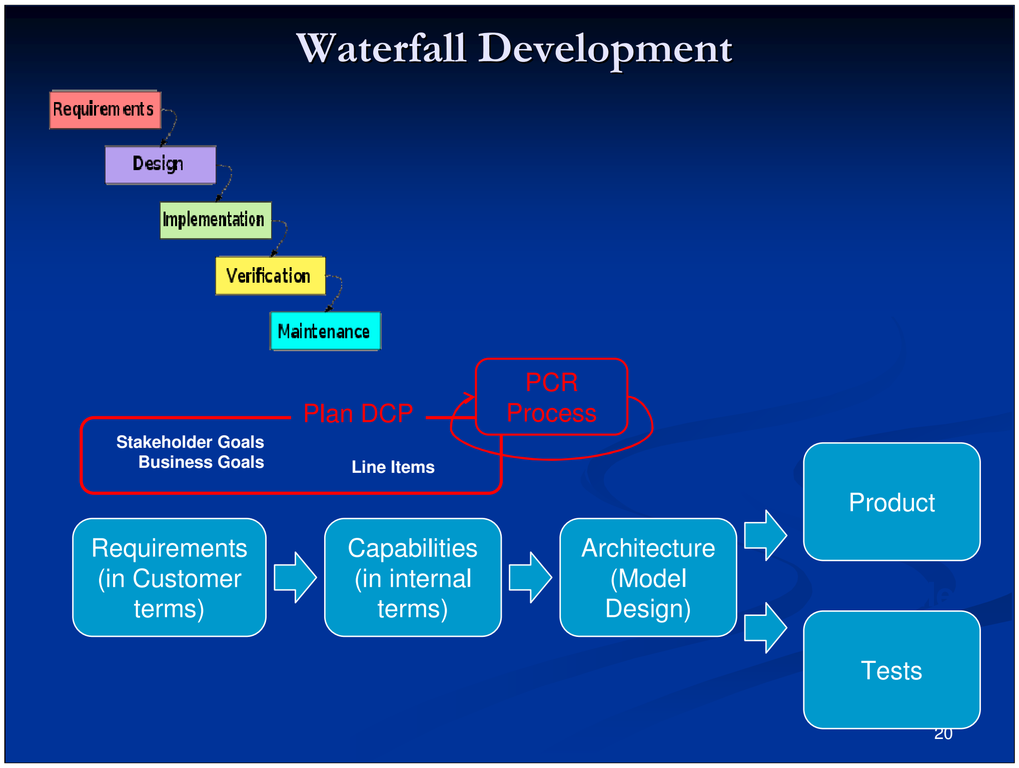
- Lean and Agile what's behind the hype
- Lean & Agile Defined
- **Before: Waterfall**
- After: Agile
- Transformation:
  - How we made it work
  - Lessons learned
- Tooling that made it possible
  - Rational Team Concert



Waterfall Development

19

Waterfall is very much like a relay race with one fixed phases and sub teams waiting for traditional handoffs before they can begin.



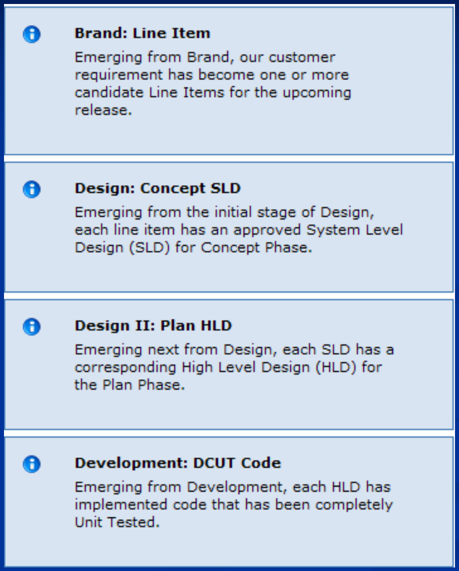
Waterfall Development

IPD: Integrated Process Development

PCR: Process Change Request

DCP: Decision Checkpoint

zOS Process: Development



Brand: Line Item

Emerging from Brand, our customer requirement has become one or more candidate Line Items for the upcoming release.

Design: Concept SLD

Emerging from the initial stage of Design, each line item has an approved System Level Design (SLD) for Concept Phase.

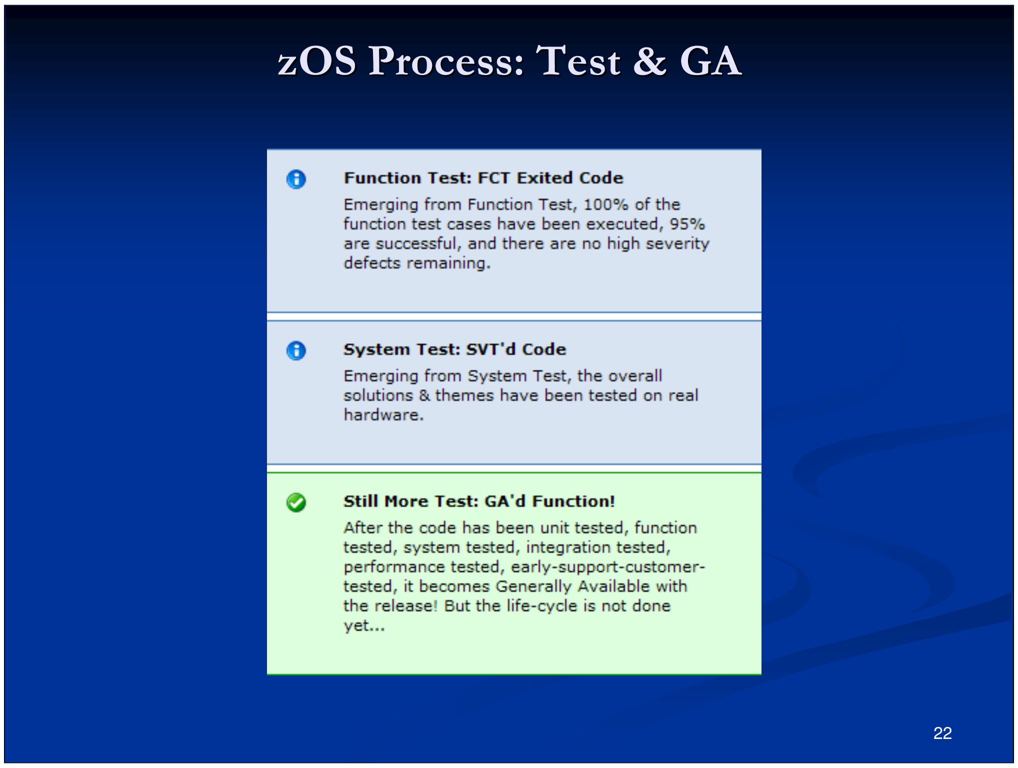
Design II: Plan HLD

Emerging next from Design, each SLD has a corresponding High Level Design (HLD) for the Plan Phase.

Development: DCUT Code

Emerging from Development, each HLD has implemented code that has been completely Unit Tested.

21



zOS Process: Test & GA

Function Test: FCT Exited Code

Emerging from Function Test, 100% of the function test cases have been executed, 95% are successful, and there are no high severity defects remaining.

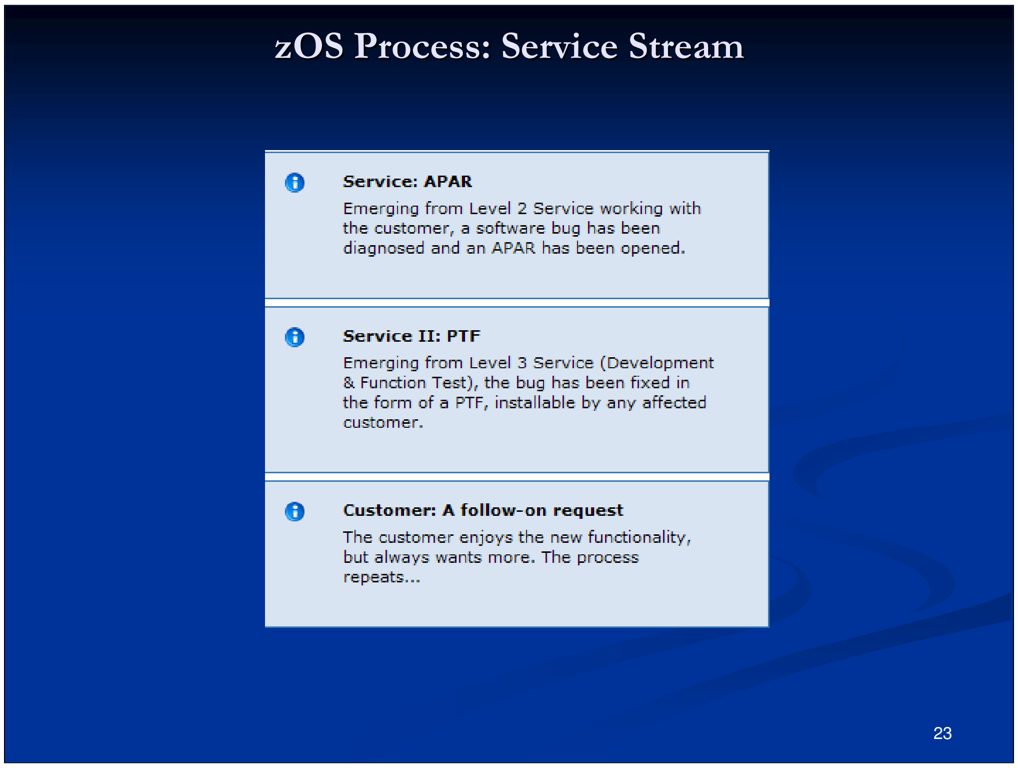
System Test: SVT'd Code

Emerging from System Test, the overall solutions & themes have been tested on real hardware.

Still More Test: GA'd Function!

After the code has been unit tested, function tested, system tested, integration tested, performance tested, early-support-customer-tested, it becomes Generally Available with the release! But the life-cycle is not done yet...

22



zOS Process: Service Stream

Service: APAR

Emerging from Level 2 Service working with the customer, a software bug has been diagnosed and an APAR has been opened.

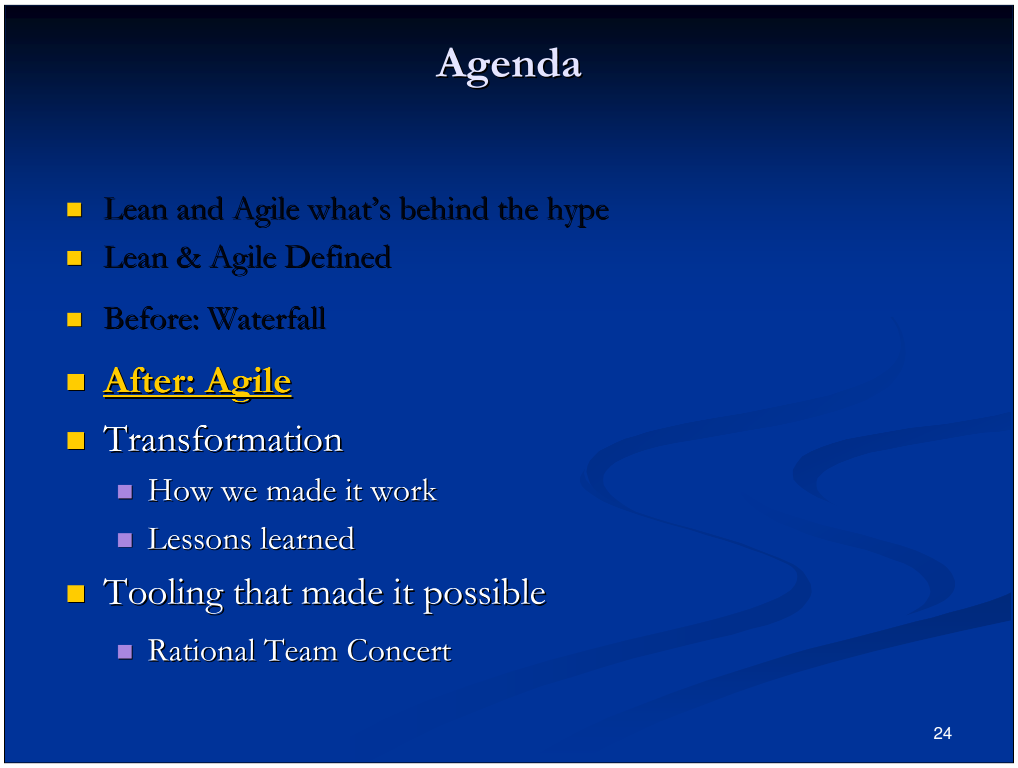
Service II: PTF

Emerging from Level 3 Service (Development & Function Test), the bug has been fixed in the form of a PTF, installable by any affected customer.

Customer: A follow-on request

The customer enjoys the new functionality, but always wants more. The process repeats...

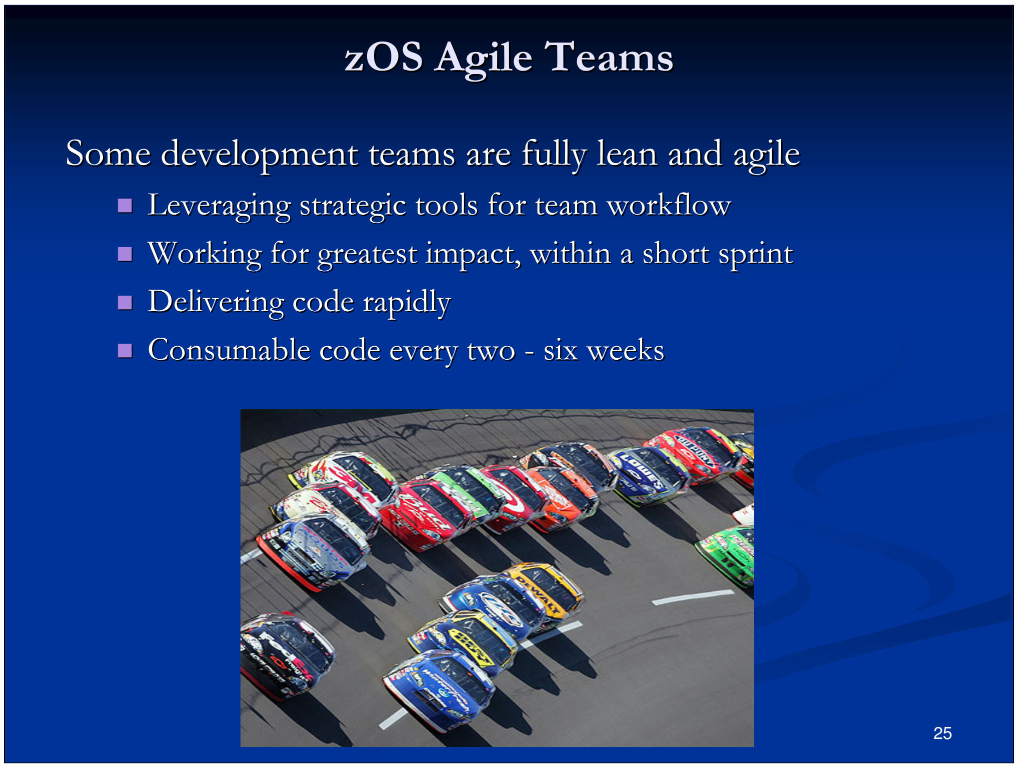
23



Agenda

- Lean and Agile what's behind the hype
- Lean & Agile Defined
- Before: Waterfall
- **After: Agile**
- Transformation
  - How we made it work
  - Lessons learned
- Tooling that made it possible
  - Rational Team Concert

24

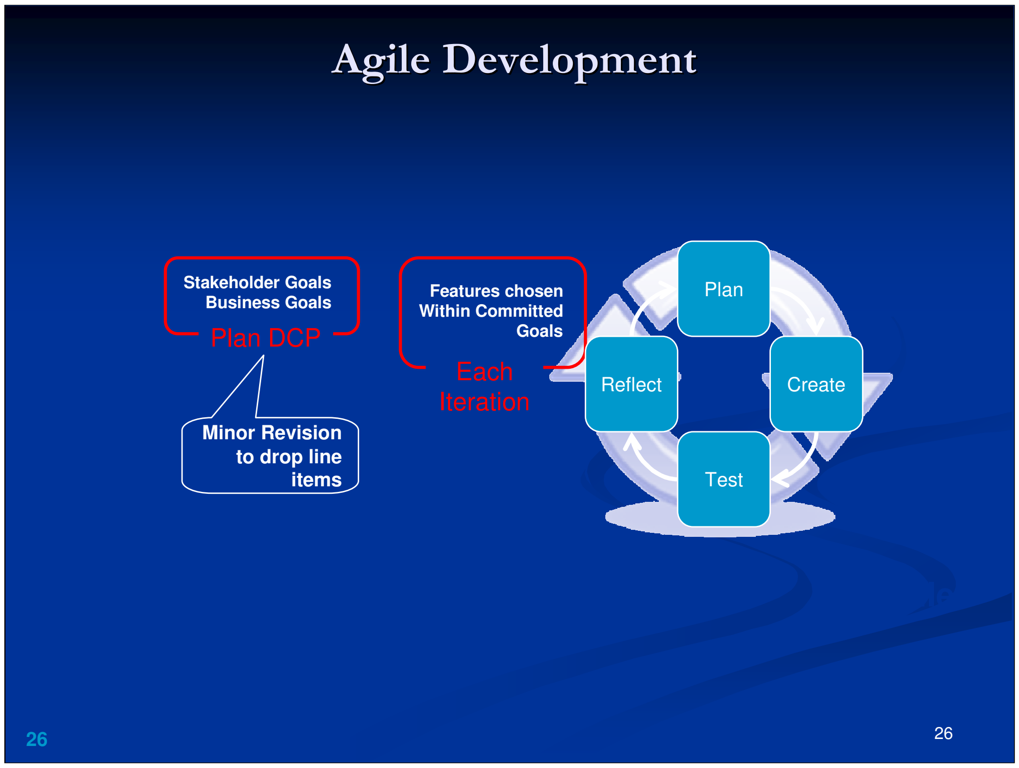


zOS Agile Teams

Some development teams are fully lean and agile

- Leveraging strategic tools for team workflow
- Working for greatest impact, within a short sprint
- Delivering code rapidly
- Consumable code every two - six weeks

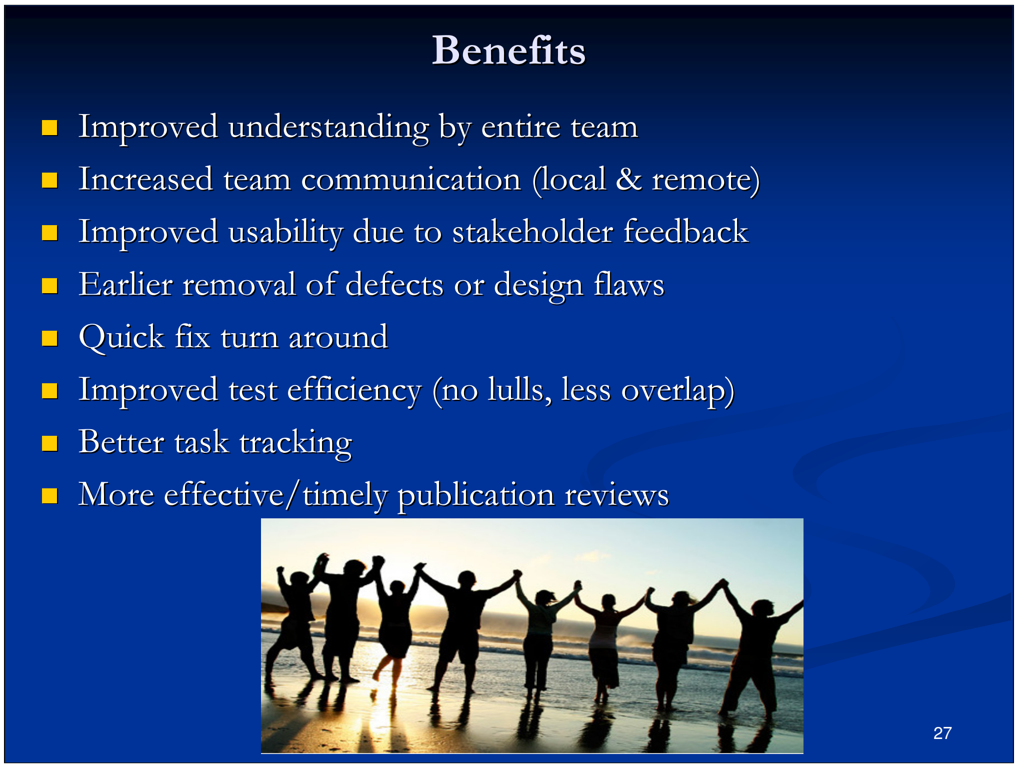
25



Agile Development

IPD: Integrated Process Development

PCR: Process Change Request



Benefits

- Improved understanding by entire team
- Increased team communication (local & remote)
- Improved usability due to stakeholder feedback
- Earlier removal of defects or design flaws
- Quick fix turn around
- Improved test efficiency (no lulls, less overlap)
- Better task tracking
- More effective/timely publication reviews

27

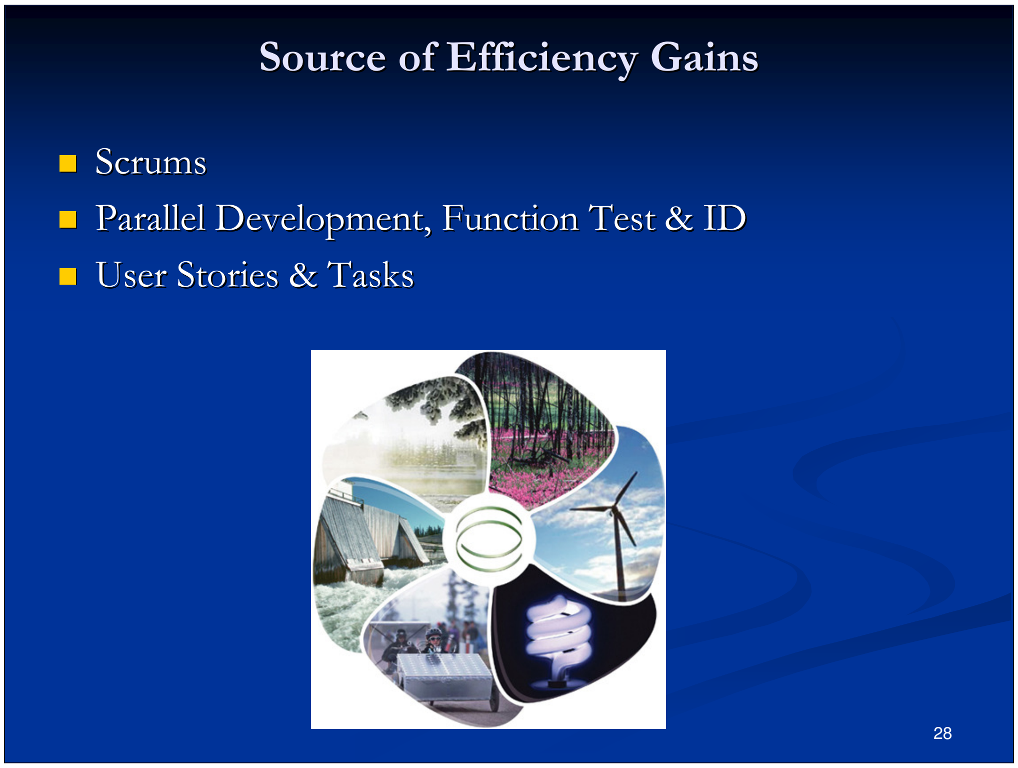
Earlier removal of defects as a result of parallel efforts:

- solution provided to testers within days

-in waterfall, would have been found months later, with formal defect process taking weeks

Increased team communication as a result of planning poker and smaller user stories

- improved understanding by entire team
- allows for better planning
- increases morale



Source of Efficiency Gains

- Scrums
- Parallel Development, Function Test & ID
- User Stories & Tasks

28

Scrums

increased and improved communications

continually re-focuses work activities and priorities

Parallel Development and Function Test

FVT test cases available and executed during Unit Test

Shift-left of discovery of errors, closer to when code was developed

not having to write up all defects

Function testers influencing code development

actively participating in design and code inspections

pointing out defects before code is written

User Stories & Tasks

rebalancing tasks among team enhanced

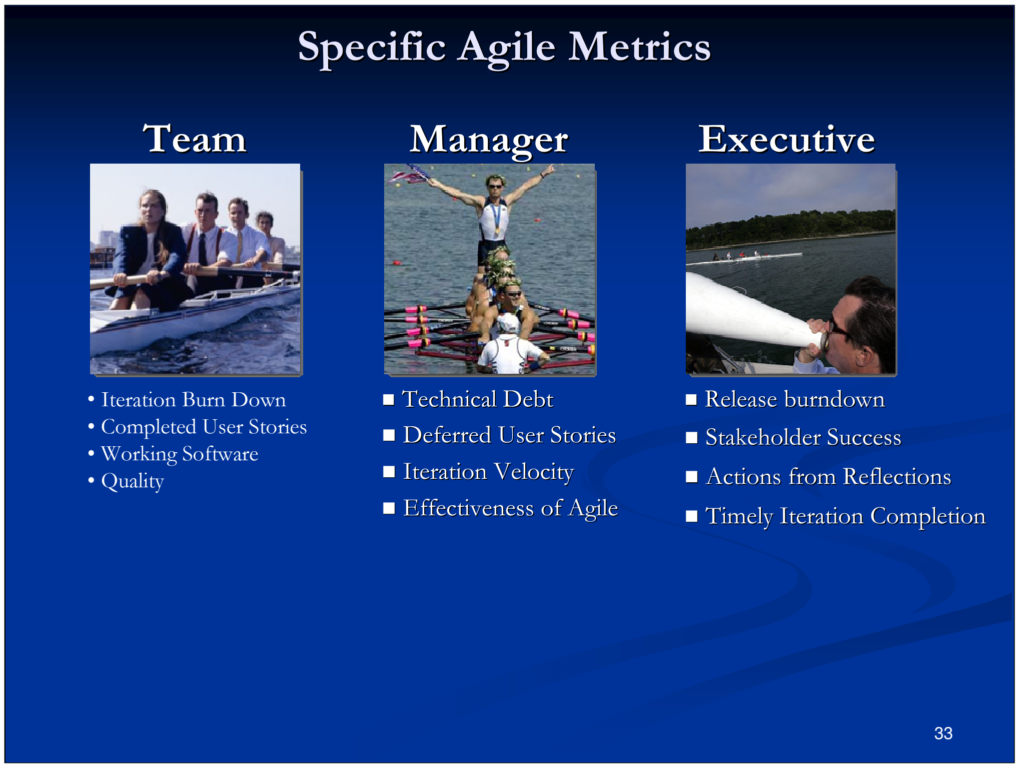
apportioning time over the effort -- spreading out the overtime

pieces of a line item can be deferred

- Predictable pace
    - Overtime peaks are distributed more evenly
    - Far easier to plan/estimate smaller bits of work
- Design churn minimized
- Real “consumables” available earlier
    - Earlier feedback from System Test, Level 2, exploiters, etc.

Voorstellen:
-wie ben je
-Wat doe je
-Waar zit je
-Wat hoop je / verwacht je

Huisregels:
-Interactie!
-Open minded
-Pauze om 19.30u
-Toetsing o.b.v. attitude en inzicht



Specific Agile Metrics

Team	Manager	Executive
• Iteration Burn Down • Completed User Stories • Working Software • Quality	■ Technical Debt ■ Deferred User Stories ■ Iteration Velocity ■ Effectiveness of Agile	■ Release burndown ■ Stakeholder Success ■ Actions from Reflections ■ Timely Iteration Completion

33

Burn Down charts give visibility into a project's progress. They show the progress made against predictions, and open the door to discussions about how best to proceed, including the difficult discussions about whether to cut scope or extend the schedule.

Velocity: How much work you did in your previous iteration. It's usually measured in Story Points.

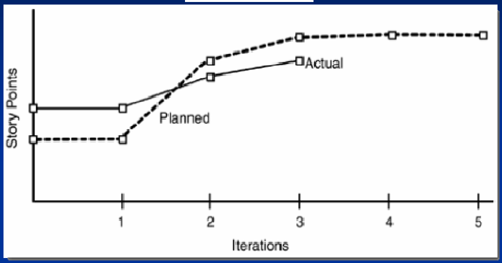
Tech Debt: The underlying cause of the inability to develop new features due to a defect burden

Working Software Remember one of the major guiding principles of Agile is: Working software Over comprehensive documentation. Attributes of working software include: (but not limited to)
Tested, stable, concrete (no sev1 or sev2), demoable to customers

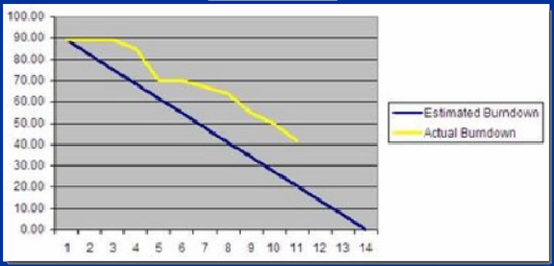
Taken together, these iteration metrics and their trend over time provide an ongoing indicator of the team's real progress.

Visualizations

Velocity



Burndown



Executive Dashboard

Project name	Target eGA date	Status
Rational Team Concert 1.0	06/30/08	🟢
CC/CQ 7.1	06/26/08	🟡
Build Forge 7.Next	10/15/08	🟡
Software Analyzer 7.0	04/29/08	🟢

34

Here are some of the more common Agile metrics

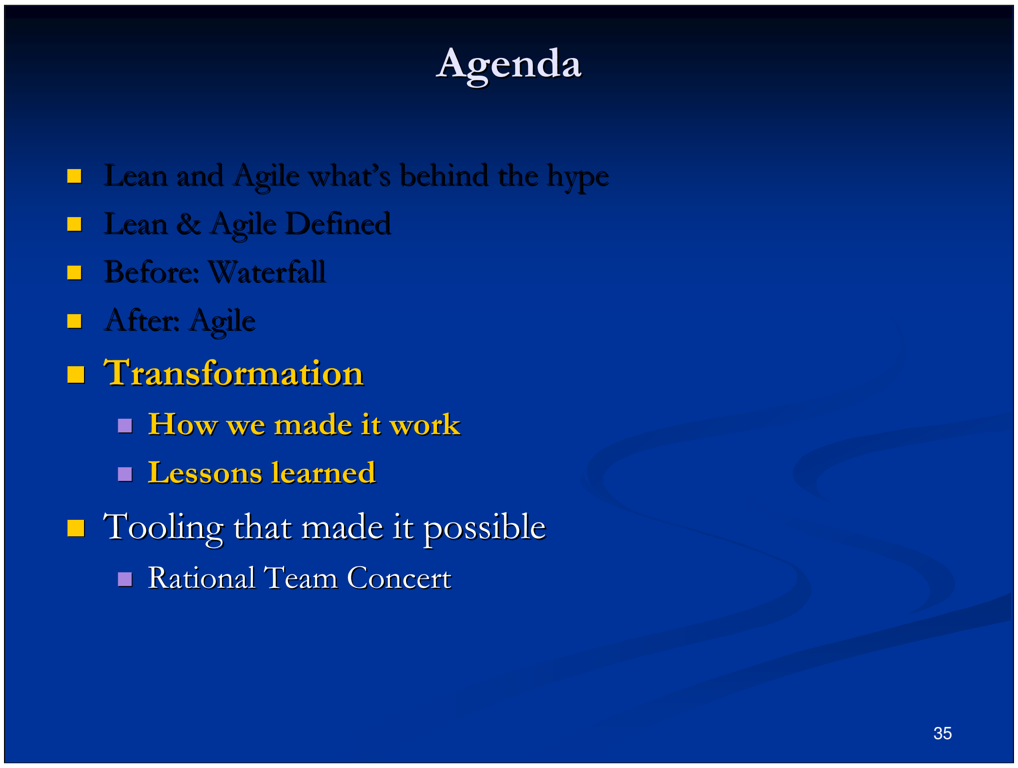
Velocity Chart: A velocity chart shows the sum of estimates of the work delivered across all iterations. Typically velocity will stabilize through the life of a project unless the project team make-up varies widely or the length of the iteration changes. As such, velocity can be used for future planning purposes.

Iteration Burndown: Task progress provides a very telling measure of overall iteration progress and has the potential (though it often does not) to remain at a constant rate throughout an iteration. The Burndown Chart shows a trended view of task progress and is the most common measure of iteration progress.

Executive Dashboard: This sample shows a typical summary of project status for the Rational brand.

Shown is the CC/CQ/RTC/BF project schedules:

The idea here is the quick visual assessment. Ie. Red = Risk



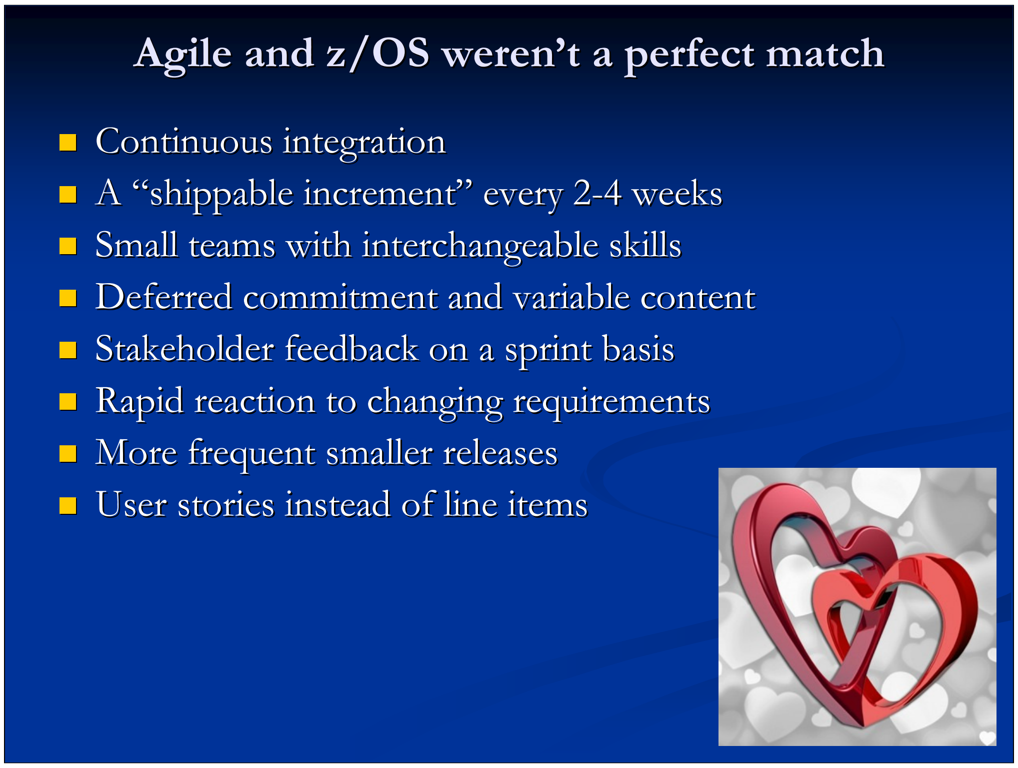
Agenda

- Lean and Agile what's behind the hype
- Lean & Agile Defined
- Before: Waterfall
- After: Agile
- Transformation
  - How we made it work
  - Lessons learned
- Tooling that made it possible
  - Rational Team Concert

35

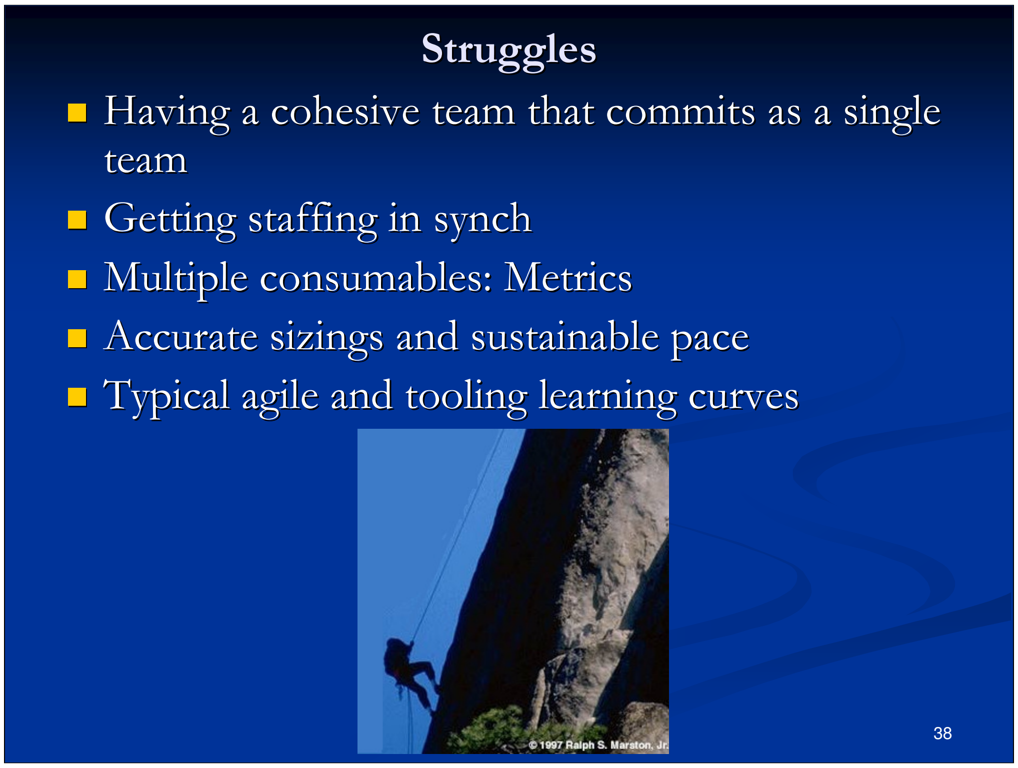
Transformation

36



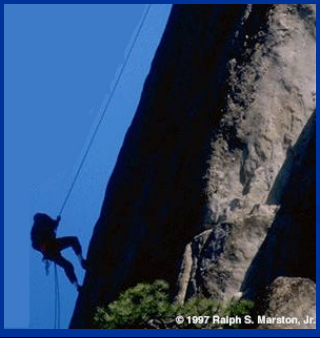
Agile and z/OS weren't a perfect match

- Continuous integration
- A "shippable increment" every 2-4 weeks
- Small teams with interchangeable skills
- Deferred commitment and variable content
- Stakeholder feedback on a sprint basis
- Rapid reaction to changing requirements
- More frequent smaller releases
- User stories instead of line items



Struggles

- Having a cohesive team that commits as a single team
- Getting staffing in synch
- Multiple consumables: Metrics
- Accurate sizings and sustainable pace
- Typical agile and tooling learning curves



38

Cohesive team:

-- breaking down silos

-- dealing with disparate tools and processes

-- dealing with politics of different management chains

Getting Staffing in synch

-- How many testers per developer? What is the right sprint load for “done done done”

Multiple consumables

-- some things being done agile, some waterfall,

-- metrics and reporting requirements disparate

Sizings and pace:

-- what is the real pace/team velocity a team can sustain taking into account, meetings, etc that take place in a day? 6hrs?

-- how to correctly play planning poker by all team members, translation of points to measures of time

Prudently Agile – did what made sense

- Establish team view of agile
- Find ways to eliminate waste
- Identify different types of stakeholders
- Formalize the concept of "stretch" functions
- Improve estimation and planning
- Foster a "whole team" approach
- Continuously refine processes

- A lean approach to unit test
  - A single set of testcases assists with two test tasks (FVT & UT)
  - Defect recording not required for UT bugs
  - Either Testers write all testcases, dev executes during UT or dev can help create testcases
- FVT (Function Test) works in parallel with design and development
  - Variations/Test Definition included with design materials
  - Testcases developed in synch with code and executed as part of UT

Sprints are still defined at a team level usually revolving around a single development team and

Design: as per Lean/Agile tenants try to delay decision making: degree of design details and ove

Whole Team: current definition of “whole team” function test and development and/or informatio

Definition of done will vary but must be defined by the whole team at Sprint planning meeting to e

Discipline representatives: Development, Function Test, System Test, Level 2, & ID represented

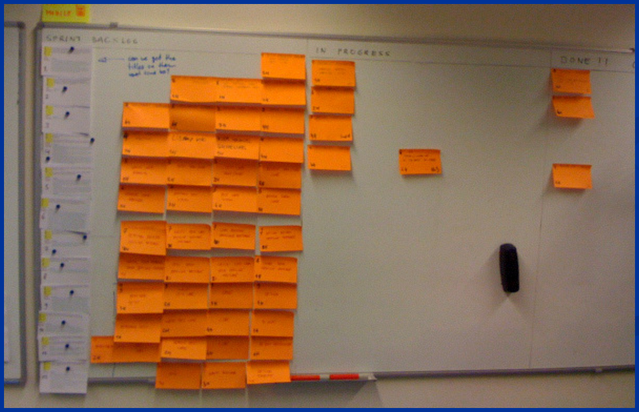
Scrum team

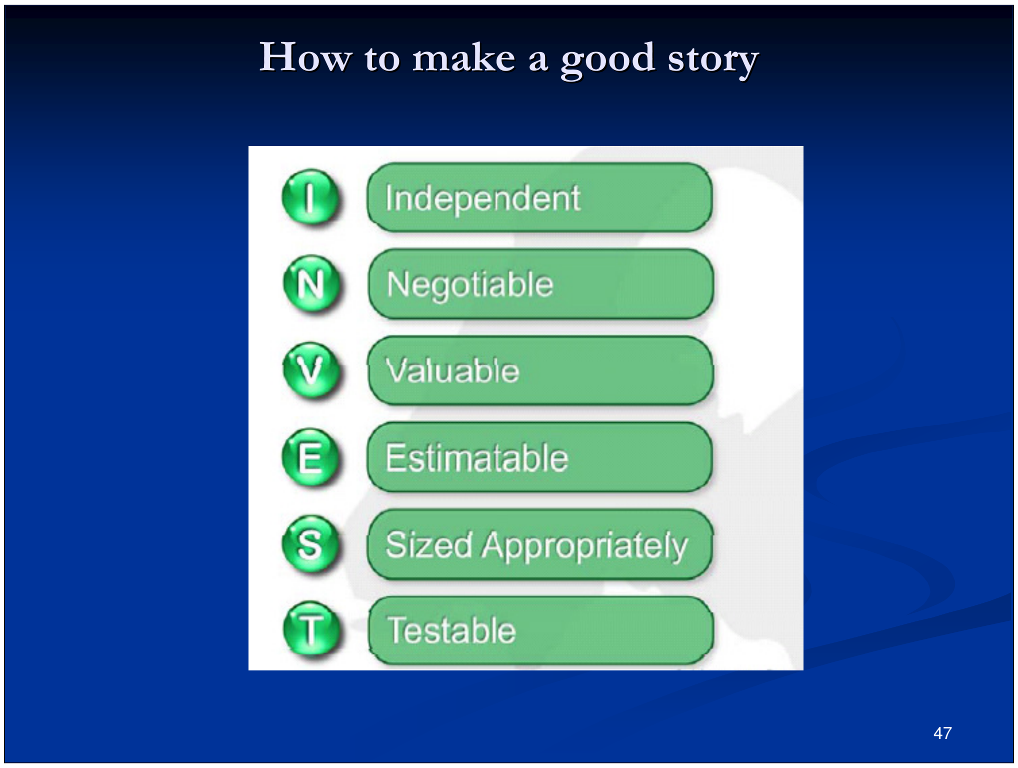
- Scrum team members have tasks to do within the sprint
- Development & Function Test at a minimum

How to Create Product backlogs

Write product backlog items with different levels of detail:

- Fine grained for items about to be worked on
- Coarse grained for items further in the future





How to make a good story

I Independent
N Negotiable
V Valuable
E Estimatable
S Sized Appropriately
T Testable
47

Independent: dependencies lead to problems estimating and prioritizing.

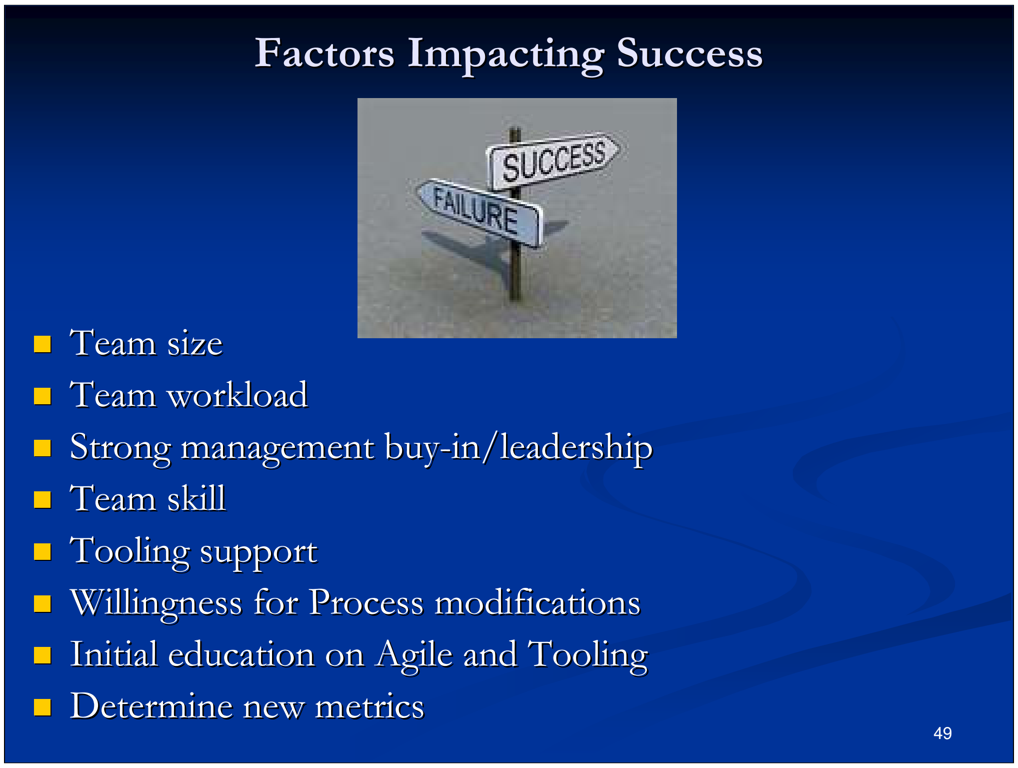
Negotiable: stories are not contracts, leave or imply some flexibility

Valuable: to users or customers, not developers. Rewrite developer stories to reflect value to users or customers

Estimatable: because plans are based on user stories, need to be able to estimate them

Sized Appropriately: small enough to complete in one sprint

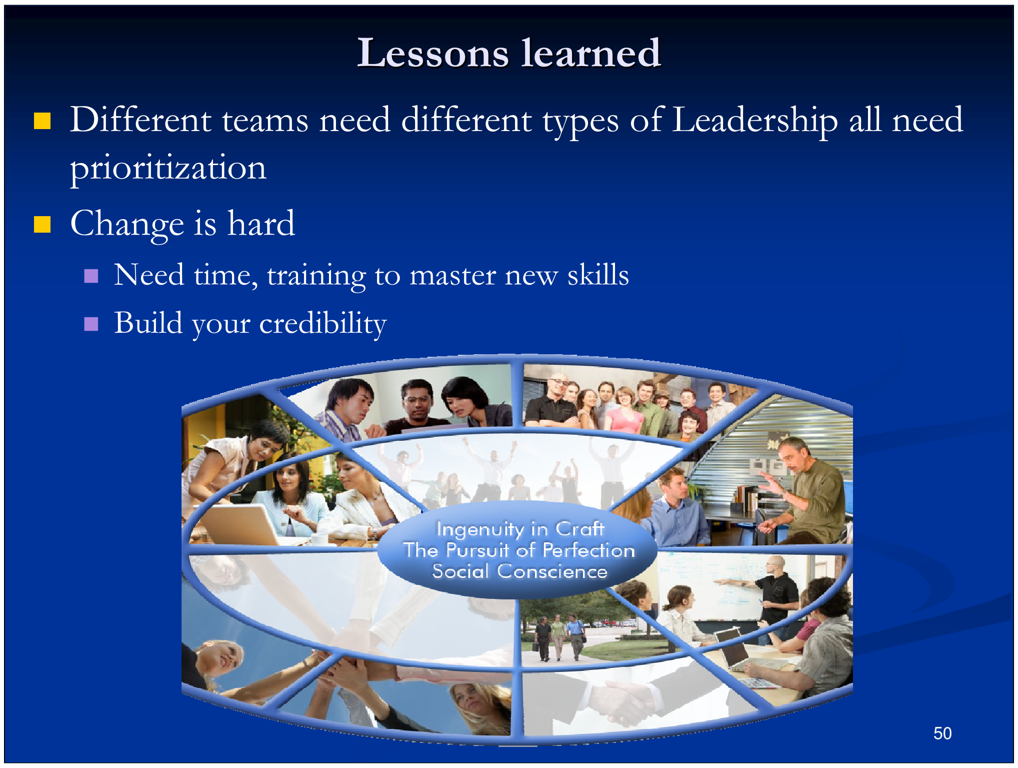
Testable: testable so that you have a easy what of knowing when finished. Done or not done



Factors Impacting Success

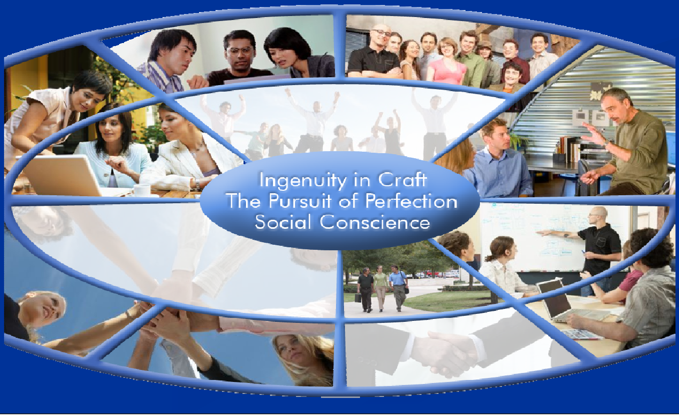
- Team size
- Team workload
- Strong management buy-in/leadership
- Team skill
- Tooling support
- Willingness for Process modifications
- Initial education on Agile and Tooling
- Determine new metrics

49

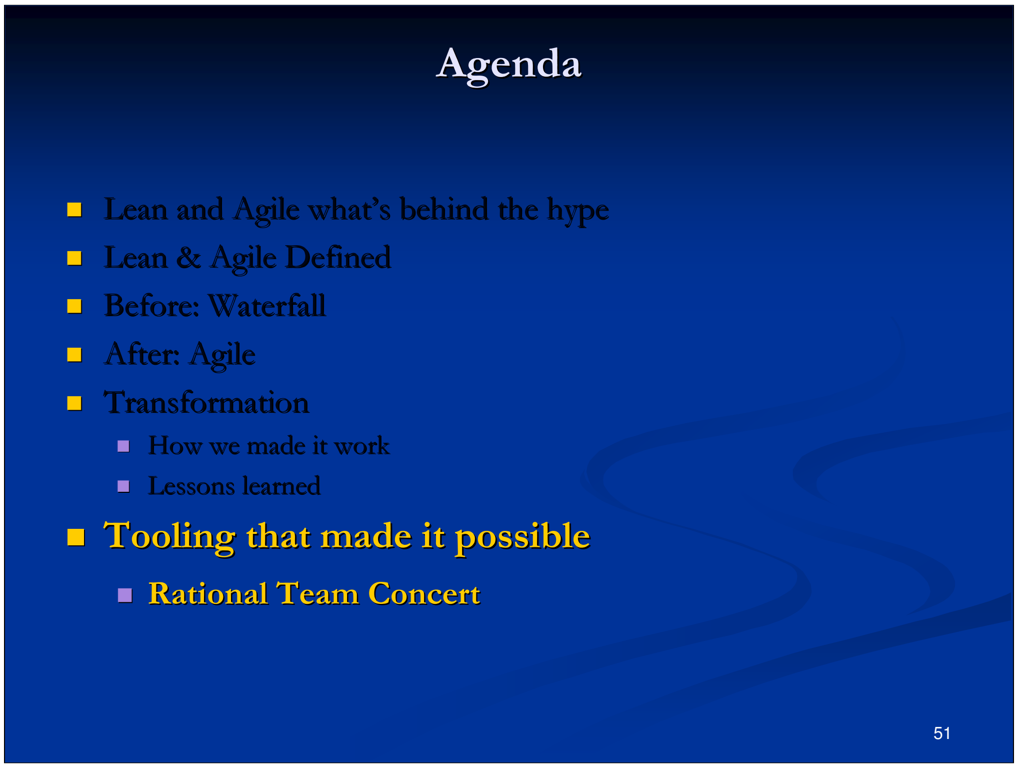


Lessons learned

- Different teams need different types of Leadership all need prioritization
- Change is hard
  - Need time, training to master new skills
  - Build your credibility



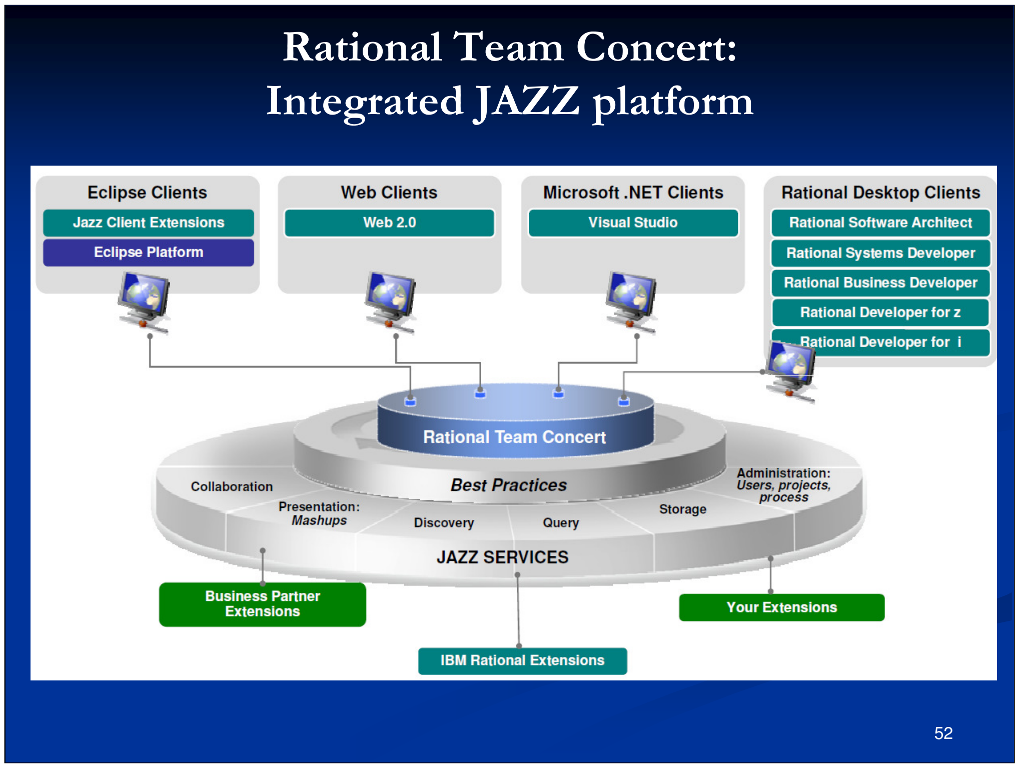
50



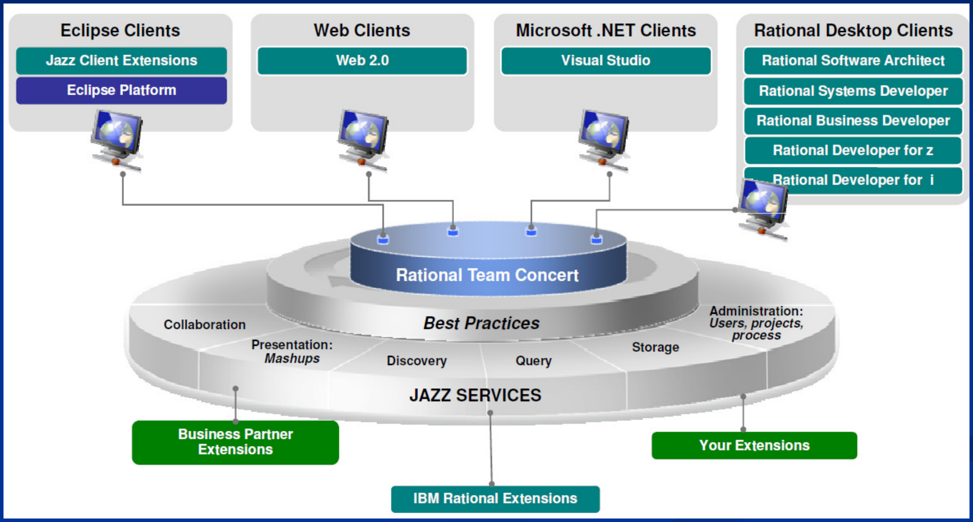
Agenda

- Lean and Agile what's behind the hype
- Lean & Agile Defined
- Before: Waterfall
- After: Agile
- Transformation
  - How we made it work
  - Lessons learned
- **Tooling that made it possible**
  - **Rational Team Concert**

51

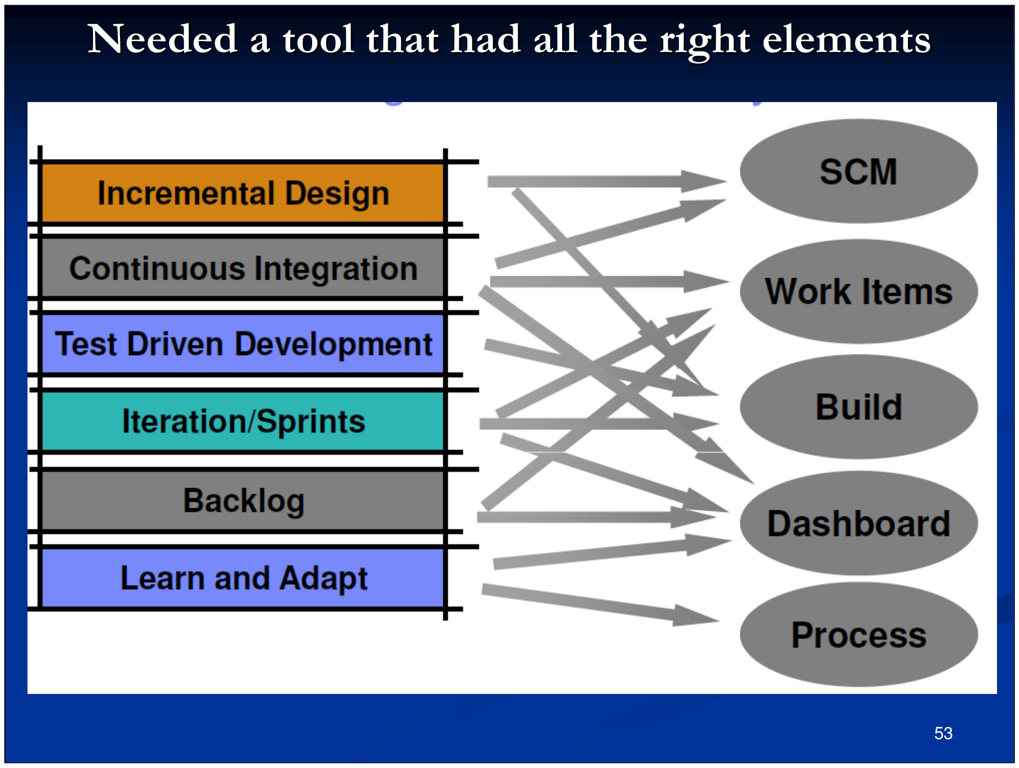


Rational Team Concert: Integrated JAZZ platform

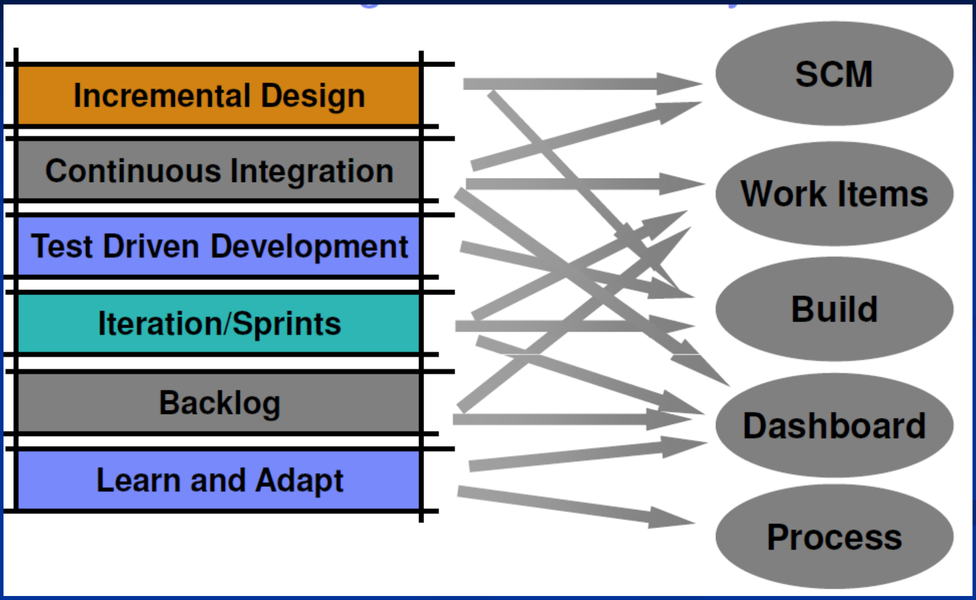


52

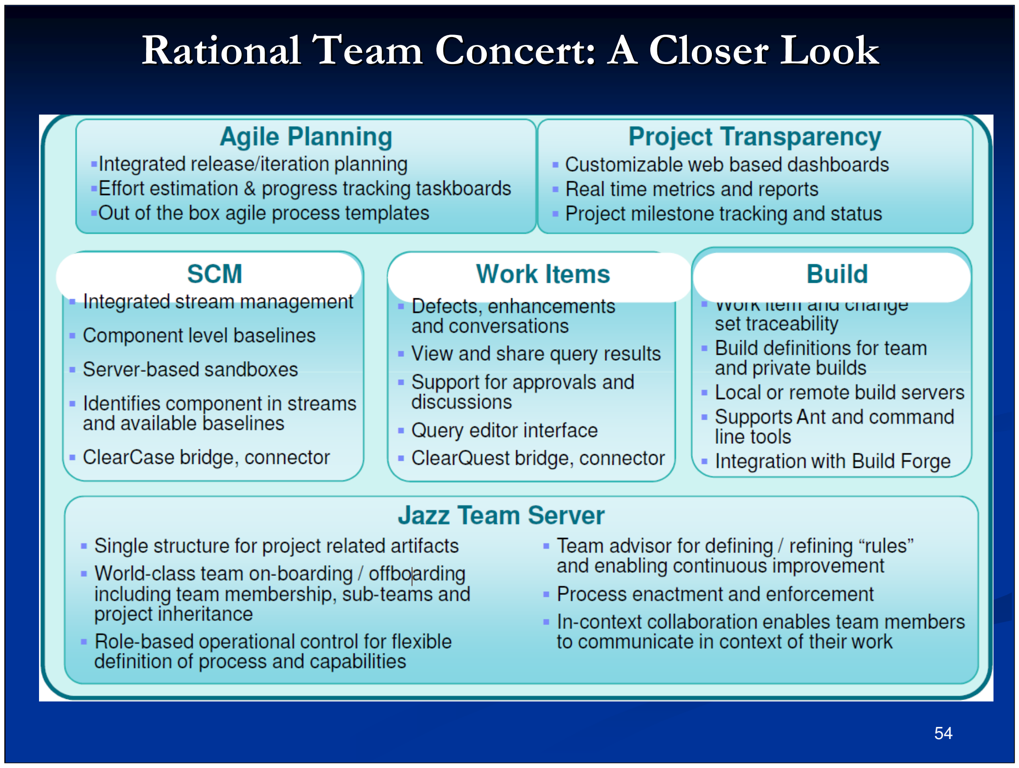
RTC built on the Jazz Platform is open and extensible
RTC provides integrated end to end support of any development process
RTC provides both planning and automated status to keep teams on track
RTC provides unique, in context, collaboration among software developers
You can adopt RTC in an incremental way using your existing artifacts



Needed a tool that had all the right elements



53



Rational Team Concert: A Closer Look

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

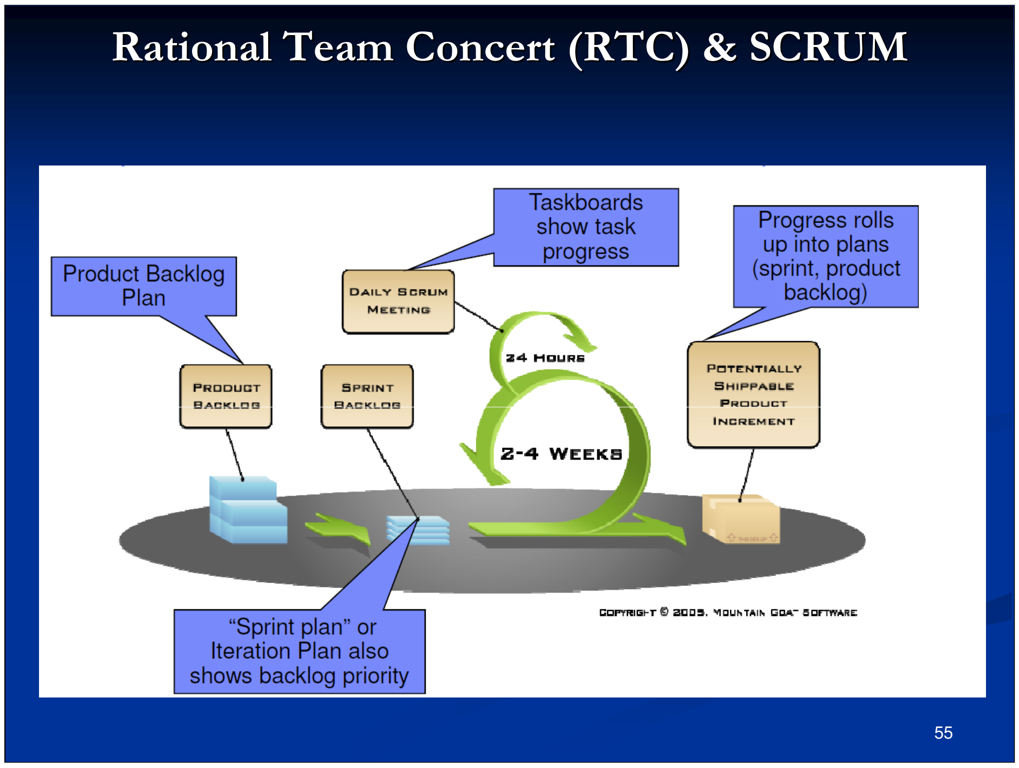
SCM
- Integrated stream management
- Component level baselines
- Server-based sandboxes
- Identifies component in streams and available baselines
- ClearCase bridge, connector

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

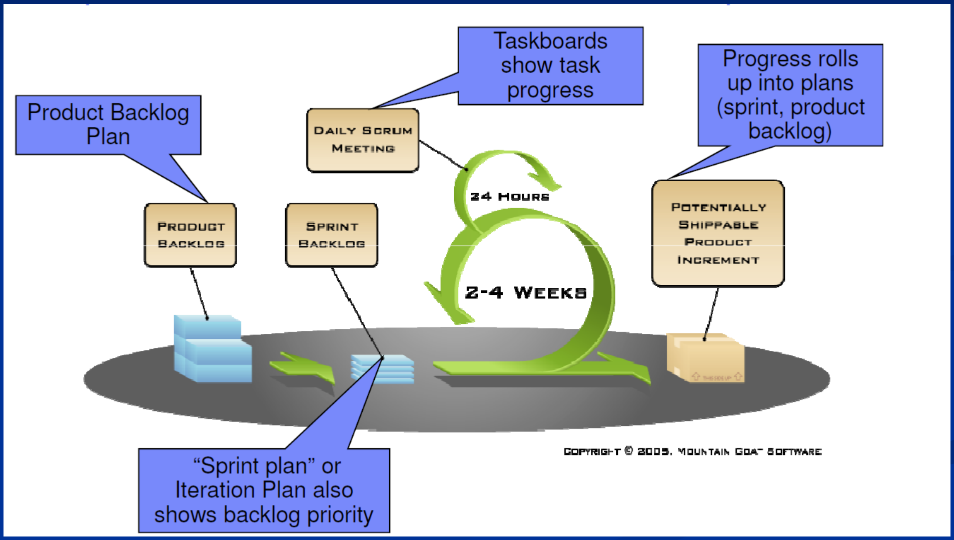
Build
- Work item and change set traceability
- Build definitions for team and private builds
- Local or remote build servers
- Supports Ant and command line tools
- Integration with Build Forge

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

54



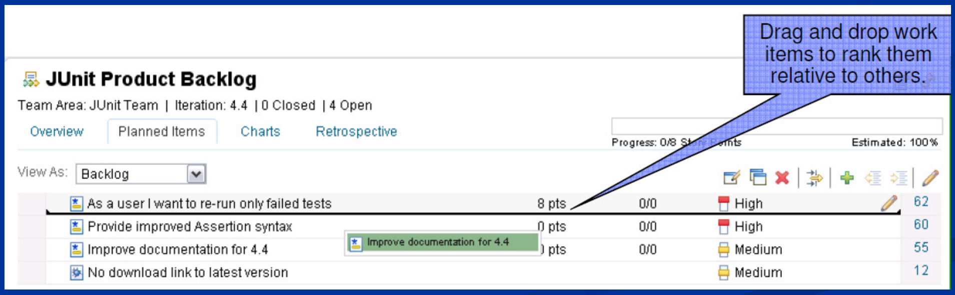
Rational Team Concert (RTC) & SCRUM


Product Backlog Plan
Taskboards show task progress
Progress rolls up into plans (sprint, product backlog)
Daily Scrum Meeting
24 Hours
Product Backlog
Sprint Backlog
2-4 Weeks
Potentially Shippable Product Increment
"Sprint plan" or Iteration Plan also shows backlog priority
Copyright © 2005, Mountain Goat Software

55

Backlog Plan Mode

- Good for managing SCRUM backlog
- Support coarse & fine grained prioritization
- Ranking is reflected in all planning views



JUnit Product Backlog

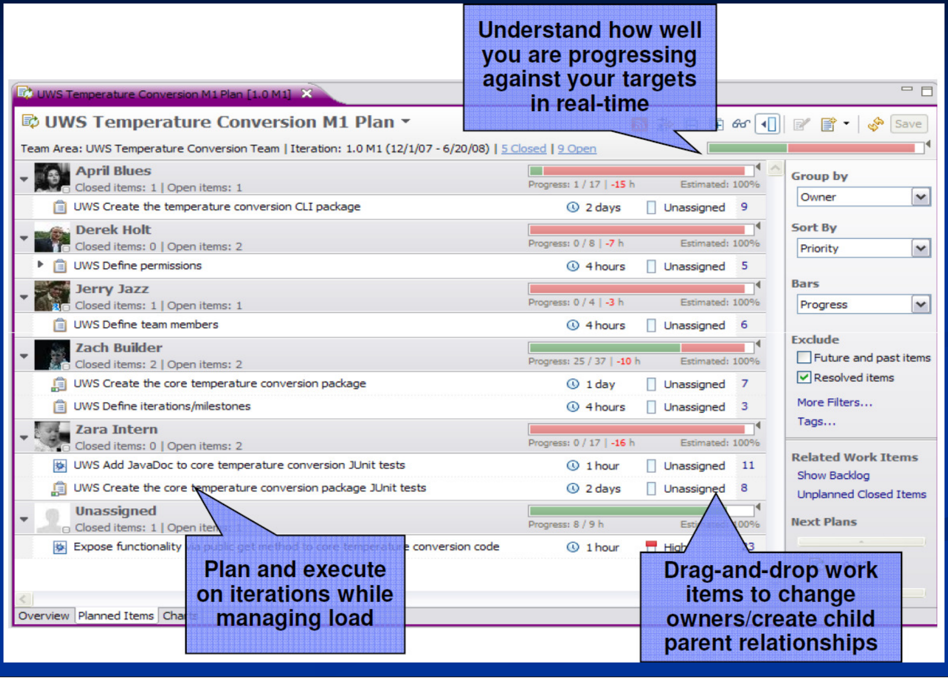
Team Area: JUnit Team | Iteration: 4.4 | 0 Closed | 4 Open

Overview
Planned Items
Charts
Retrospective
Progress: 0/8 Story Points
Estimated: 100%
View As: Backlog
Improve documentation for 4.4
Drag and drop work items to rank them relative to others.

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

56

Iteration “Sprint” Planning



Understand how well you are progressing against your targets in real-time

UWS Temperature Conversion M1 Plan [1.0 M1]

UWS Temperature Conversion M1 Plan

Team Area: UWS Temperature Conversion Team | Iteration: 1.0 M1 (12/1/07 - 6/20/08) | 5 Closed | 9 Open

April Blues Closed items: 1 | Open items: 1	Progress: 1 / 17 | -15 h Estimated: 100%		
UWS Create the temperature conversion CLI package	2 days	Unassigned	9
Derek Holt Closed items: 0 | Open items: 2	Progress: 0 / 8 | -7 h Estimated: 100%		
UWS Define permissions	4 hours	Unassigned	5
Jerry Jazz Closed items: 1 | Open items: 1	Progress: 0 / 4 | -3 h Estimated: 100%		
UWS Define team members	4 hours	Unassigned	6
Zach Builder Closed items: 2 | Open items: 2	Progress: 25 / 37 | -10 h Estimated: 100%		
UWS Create the core temperature conversion package	1 day	Unassigned	7
UWS Define iterations/milestones	4 hours	Unassigned	3
Zara Intern Closed items: 0 | Open items: 2	Progress: 0 / 17 | -16 h Estimated: 100%		
UWS Add JavaDoc to core temperature conversion JUnit tests	1 hour	Unassigned	11
UWS Create the core temperature conversion package JUnit tests	2 days	Unassigned	8
Unassigned Closed items: 1 | Open ite	Progress: 8 / 9 h Esti 100%		
Expose functionality via public get method to core temperature conversion code	1 hour	High	3

Group by: Owner

Sort By: Priority

Bars: Progress

Exclude
- ☐ Future and past items
- ☑ Resolved items

More Filters…

Tags…

Related Work Items

Show Backlog

Unplanned Closed Items

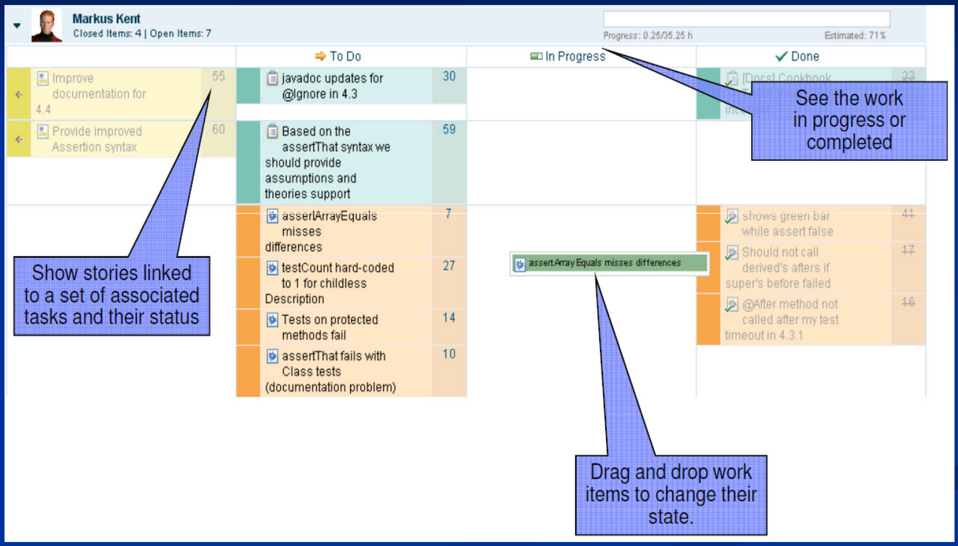
Next Plans

Overview | Planned Items | Char

Plan and execute on iterations while managing load

Drag-and-drop work items to change owners/create child parent relationships

Taskboards



Markus Kent
Closed Items: 4 | Open Items: 7

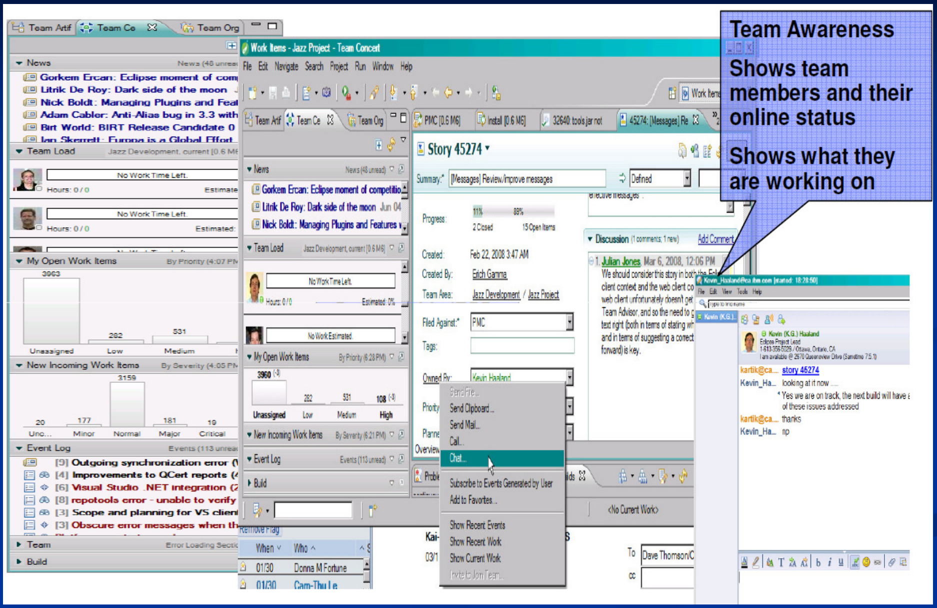
Progress: 0.25/35.25 h   Estimated: 71%

	To Do		In Progress	Done	
Improve documentation for 4.4 (55)	javadoc updates for @Ignore in 4.3	30		[Docs] Cookbook … inc…	~~23~~
Provide improved Assertion syntax (60)	Based on the assertThat syntax we should provide assumptions and theories support	59			
	assertArrayEquals misses differences	7	assertArrayEquals misses differences	shows green bar while assert false	~~41~~
	testCount hard-coded to 1 for childless Description	27		Should not call derived's afters if super's before failed	~~17~~
	Tests on protected methods fail	14		@After method not called after my test timeout in 4.3.1	~~16~~
	assertThat fails with Class tests (documentation problem)	10			

See the work in progress or completed
Show stories linked to a set of associated tasks and their status
Drag and drop work items to change their state.

58

In context collaboration



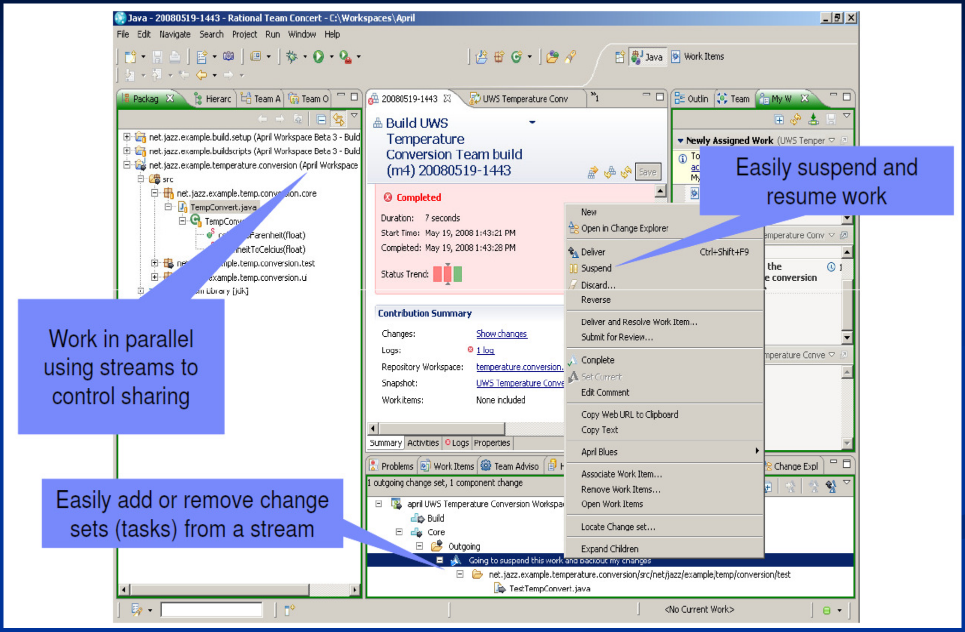
Team Awareness

Shows team members and their online status

Shows what they are working on

59

SCM is stream and component based


Java - 20080519-1443 - Rational Team Concert - C:\Workspaces\April
File Edit Navigate Search Project Run Window Help
Build UWS Temperature Conversion Team build (m4) 20080519-1443
Completed
Duration: 7 seconds
Start Time: May 19, 2008 1:43:21 PM
Completed: May 19, 2008 1:43:28 PM
Status Trend:
Contribution Summary
Changes: Show changes
Logs: 1 log
Repository Workspace: temperature.conversion
Snapshot: UWS Temperature Conve
Work items: None included
Summary Activities Logs Properties
New
Open in Change Explorer
Deliver Ctrl+Shift+F9
Suspend
Discard...
Reverse
Deliver and Resolve Work Item...
Submit for Review...
Complete
Set Current
Edit Comment
Copy Web URL to Clipboard
Copy Text
April Blues
Associate Work Item...
Remove Work Items...
Open Work Items
Locate Change set...
Expand Children
Newly Assigned Work
1 outgoing change set, 1 component change
april UWS Temperature Conversion Workspace
Build
Core
Outgoing
Going to suspend this work and backout my changes
net.jazz.example.temperature.conversion/src/net/jazz/example/temp/conversion/test
TestTempConvert.java
<No Current Work>

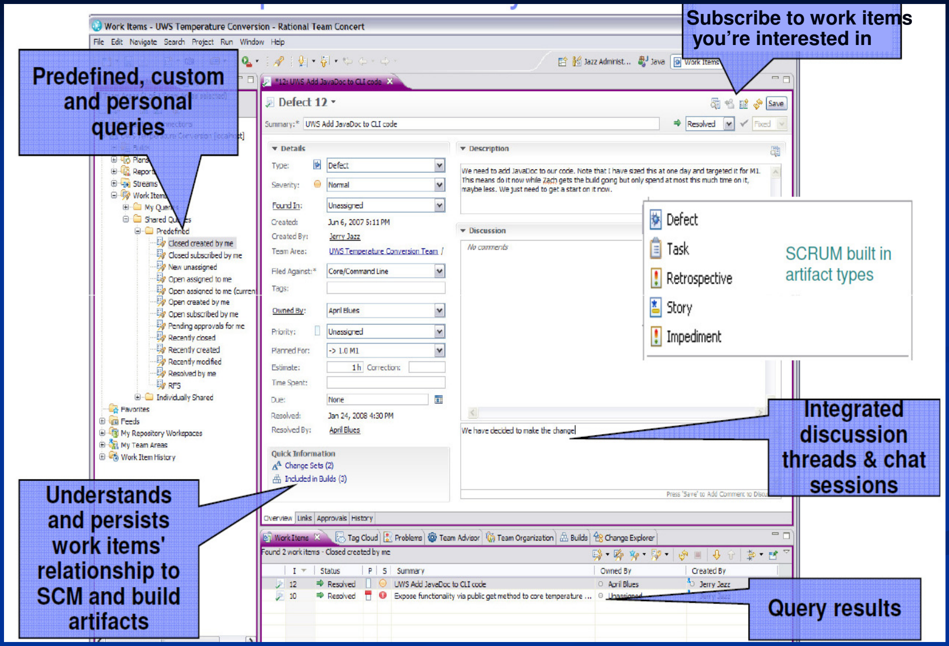
Easily suspend and resume work

Work in parallel using streams to control sharing

Easily add or remove change sets (tasks) from a stream

60

Work items capture traceability & effort


Work Items - UWS Temperature Conversion - Rational Team Concert
File Edit Navigate Search Project Run Window Help
Jazz Administ... Java Work Items
Closed created by me
Closed subscribed by me
New unassigned
Open assigned to me
Open assigned to me (current)
Open created by me
Open subscribed by me
Pending approvals for me
Recently closed
Recently created
Recently modified
Resolved by me
RFS
Individually Shared
Favorites
Feeds
My Repository Workspaces
My Team Areas
Work Item History
*12: UWS Add JavaDoc to CLI code
Save
Overview Links Approvals History
Work Items Tag Cloud Problems Team Advisor Team Organization Builds Change Explorer
Defect
Task
Retrospective
Story
Impediment
Press 'Save' to Add Comment to Discu

Predefined, custom and personal queries

Subscribe to work items you're interested in

SCRUM built in artifact types

Integrated discussion threads & chat sessions

Understands and persists work items' relationship to SCM and build artifacts

Query results

Defect 12

Summary:* UWS Add JavaDoc to CLI code — Resolved — Fixed

Details

Type:	Defect
Severity:	Normal
Found In:	Unassigned
Created:	Jun 6, 2007 5:11 PM
Created By:	Jerry Jazz
Team Area:	UWS Temperature Conversion Team /
Filed Against:*	Core/Command Line
Tags:	
Owned By:	April Blues
Priority:	Unassigned
Planned For:	-> 1.0 M1
Estimate:	1 h Correction:
Time Spent:	
Due:	None
Resolved:	Jan 24, 2008 4:30 PM
Resolved By:	April Blues

Quick Information

Change Sets (2)
Included in Builds (3)

Description

We need to add JavaDoc to our code. Note that I have sized this at one day and targeted it for M1. This means do it now while Zach gets the build going but only spend at most this much time on it, maybe less. We just need to get a start on it now.

Discussion

No comments

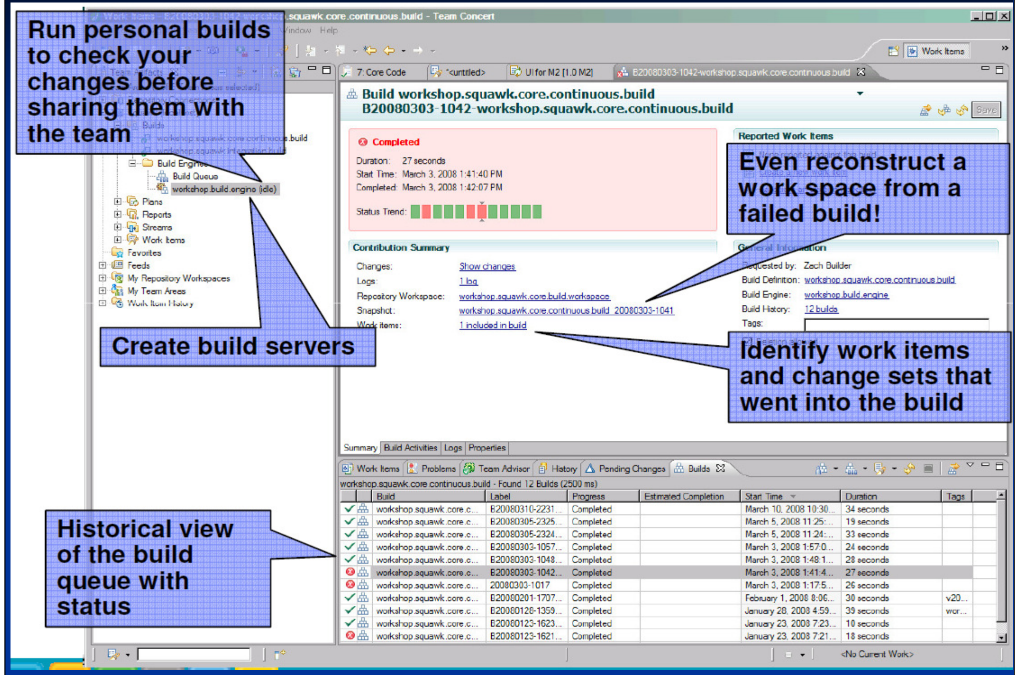
We have decided to make the change

Found 2 work items - Closed created by me

I	Status	P	S	Summary	Owned By	Created By
12	Resolved			UWS Add JaveDoc to CLI code	April Blues	Jerry Jazz
10	Resolved			Expose functionality via public get method to core temperature ...	Unassigned	Jerry Jazz

61

Builds - Extensible Continuous Integration



Build workshop.squawk.core.continuous.build
B20080303-1042-workshop.squawk.core.continuous.build

Completed

Duration: 27 seconds
Start Time: March 3, 2008 1:41:40 PM
Completed: March 3, 2008 1:42:07 PM

Status Trend: 🟩🟥🟩🟩🟩🟥🟥🟩🟩🟩🟩🟩

Contribution Summary

Changes:	Show changes
Logs:	1 log
Repository Workspace:	workshop.squawk.core.build.workspace
Snapshot:	workshop.squawk.core.continuous.build_20080303-1041
Work items:	1 included in build

General Information

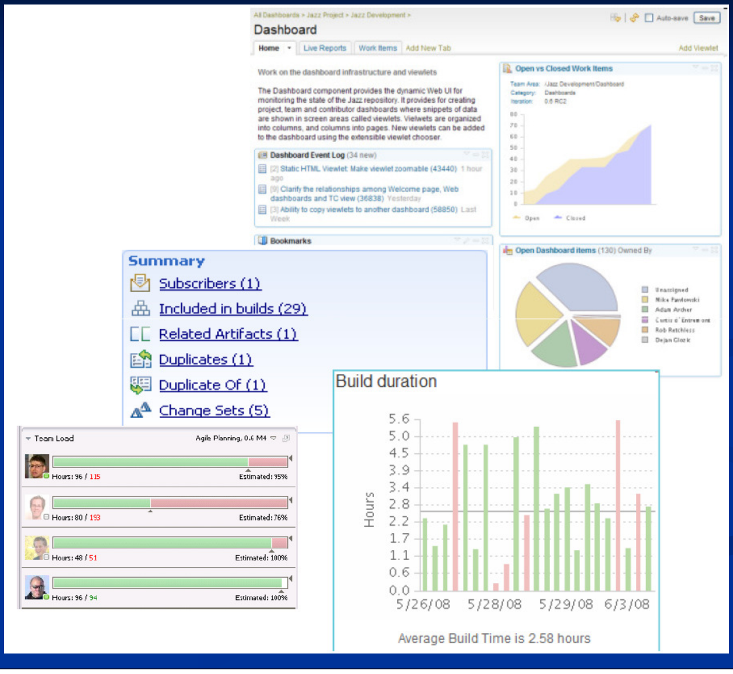
Requested by:	Zach Builder
Build Definition:	workshop.squawk.core.continuous.build
Build Engine:	workshop.build.engine
Build History:	12 builds
Tags:	

workshop.squawk.core.continuous.build - Found 12 Builds (2500 ms)

	Build	Label	Progress	Estimated Completion	Start Time	Duration	Tags
✓	workshop.squawk.core.c...	B20080310-2231...	Completed		March 10, 2008 10:30...	34 seconds	
✓	workshop.squawk.core.c...	B20080305-2325...	Completed		March 5, 2008 11:25:...	19 seconds	
✓	workshop.squawk.core.c...	B20080305-2324...	Completed		March 5, 2008 11:24:...	33 seconds	
✓	workshop.squawk.core.c...	B20080303-1057...	Completed		March 3, 2008 1:57:0...	24 seconds	
✓	workshop.squawk.core.c...	B20080303-1048...	Completed		March 3, 2008 1:48:1...	28 seconds	
⊗	workshop.squawk.core.c...	B20080303-1042...	Completed		March 3, 2008 1:41:4...	27 seconds	
⊗	workshop.squawk.core.c...	20080303-1017	Completed		March 3, 2008 1:17:5...	26 seconds	
✓	workshop.squawk.core.c...	B20080201-1707...	Completed		February 1, 2008 8:06...	30 seconds	v20...
✓	workshop.squawk.core.c...	B20080128-1359...	Completed		January 28, 2008 4:59...	39 seconds	wor...
✓	workshop.squawk.core.c...	B20080123-1623...	Completed		January 23, 2008 7:23...	10 seconds	
⊗	workshop.squawk.core.c...	B20080123-1621...	Completed		January 23, 2008 7:21...	18 seconds	

Transparency across disciplines and process

IBM Rational Software Conference 2009



All Dashboards > Jazz Project > Jazz Development >

Dashboard

Home | Live Reports | Work Items | Add New Tab | Add Viewlet

Auto-save | Save

Work on the dashboard infrastructure and viewlets

The Dashboard component provides the dynamic Web UI for monitoring the state of the Jazz repository. It provides for creating project, team and contributor dashboards where snippets of data are shown in screen areas called viewlets. Vielwets are organized into columns, and columns into pages. New viewlets can be added to the dashboard using the extensible viewlet chooser.

Dashboard Event Log (34 new)

- [2] Static HTML Viewlet: Make viewlet zoomable (43440) 1 hour ago
- [9] Clarify the relationships among Welcome page, Web dashboards and TC view (36838) Yesterday
- [3] Ability to copy viewlets to another dashboard (58850) Last Week

Bookmarks

Open vs Closed Work Items

Team Area: /Jazz Development/Dashboard
Category: Dashboards
Iteration: 0.6 RC2

Open Dashboard items (130) Owned By

Summary

- Subscribers (1)
- Included in builds (29)
- Related Artifacts (1)
- Duplicates (1)
- Duplicate Of (1)
- Change Sets (5)

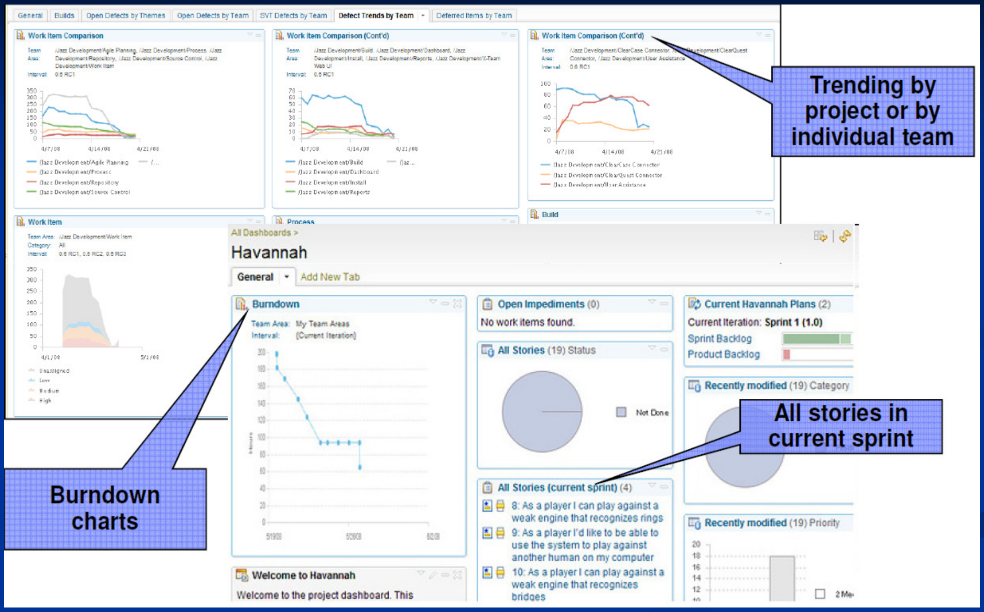
Team Load — Agile Planning, 0.6 M4

Hours: 96 / 115	Estimated: 95%
Hours: 80 / 193	Estimated: 76%
Hours: 48 / 51	Estimated: 100%
Hours: 96 / 94	Estimated: 100%

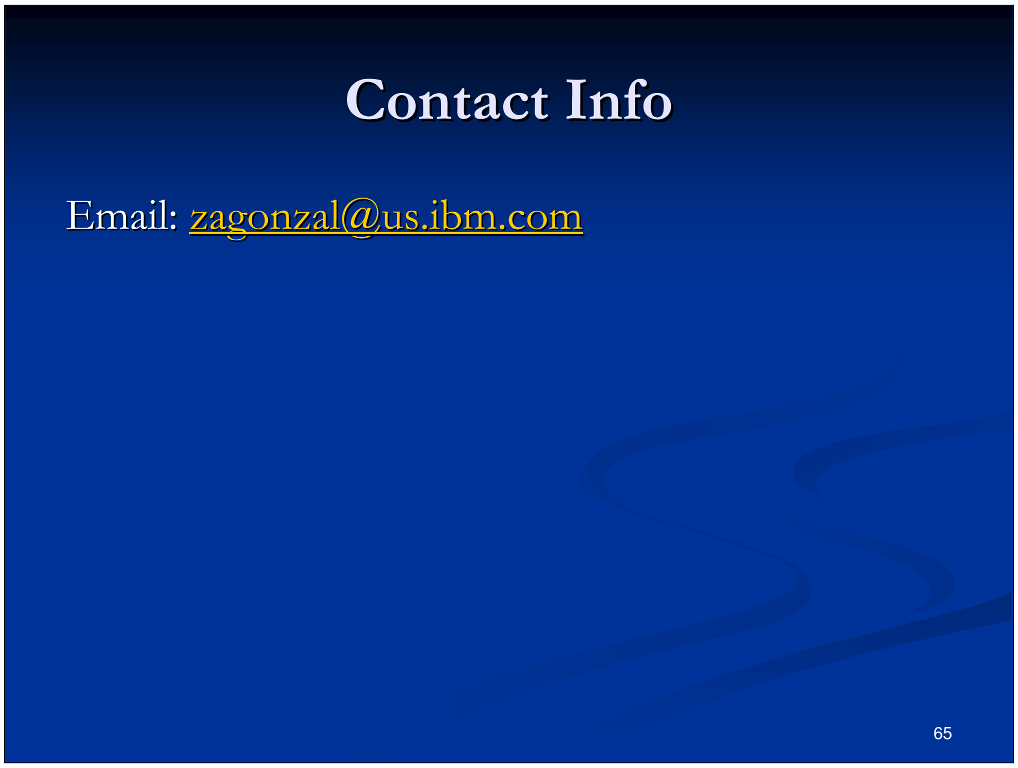
Build duration

Average Build Time is 2.58 hours

Customized Dashboards


Trending by project or by individual team
Burndown charts
All stories in current sprint
Work Item Comparison
Havannah
Burndown
Open Impediments (0)
No work items found.
Current Havannah Plans (2)
All Stories (19) Status
Recently modified (19) Category
All Stories (current sprint) (4)
Recently modified (19) Priority
Welcome to Havannah

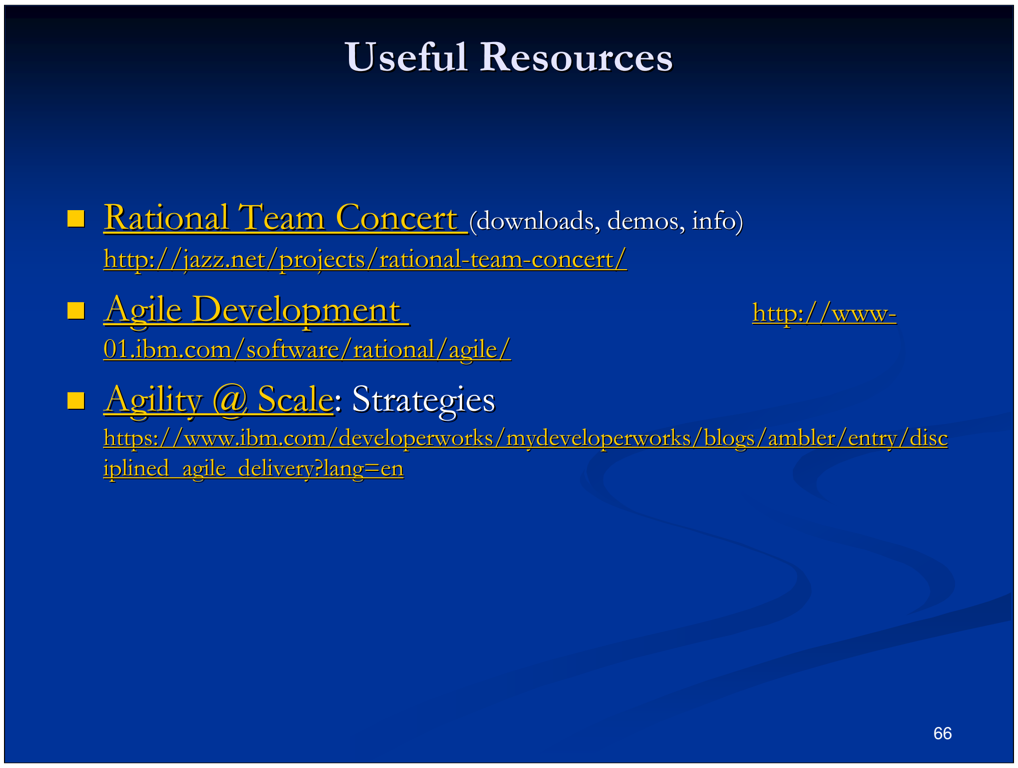
64



Contact Info

Email: zagonzal@us.ibm.com

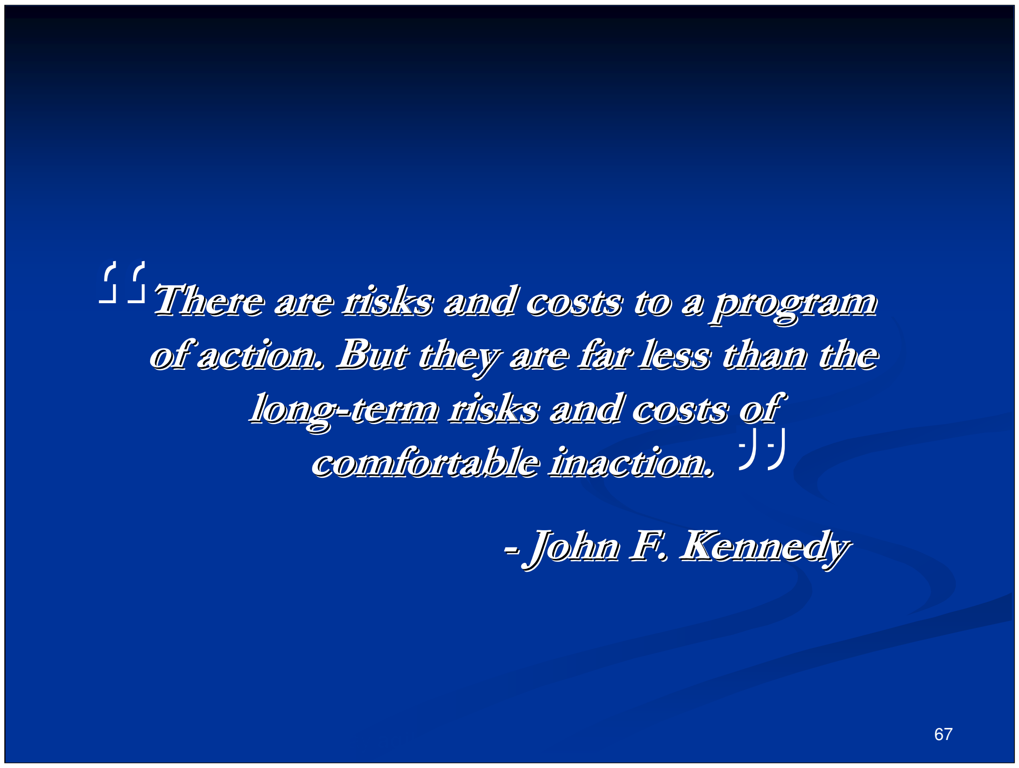
65



Useful Resources

- Rational Team Concert (downloads, demos, info) http://jazz.net/projects/rational-team-concert/
- Agile Development http://www-01.ibm.com/software/rational/agile/
- Agility @ Scale: Strategies https://www.ibm.com/developerworks/mydeveloperworks/blogs/ambler/entry/disciplined_agile_delivery?lang=en

66



"There are risks and costs to a program of action. But they are far less than the long-term risks and costs of comfortable inaction."

- John F. Kennedy

67

As quoted by May, 2007.

Backup

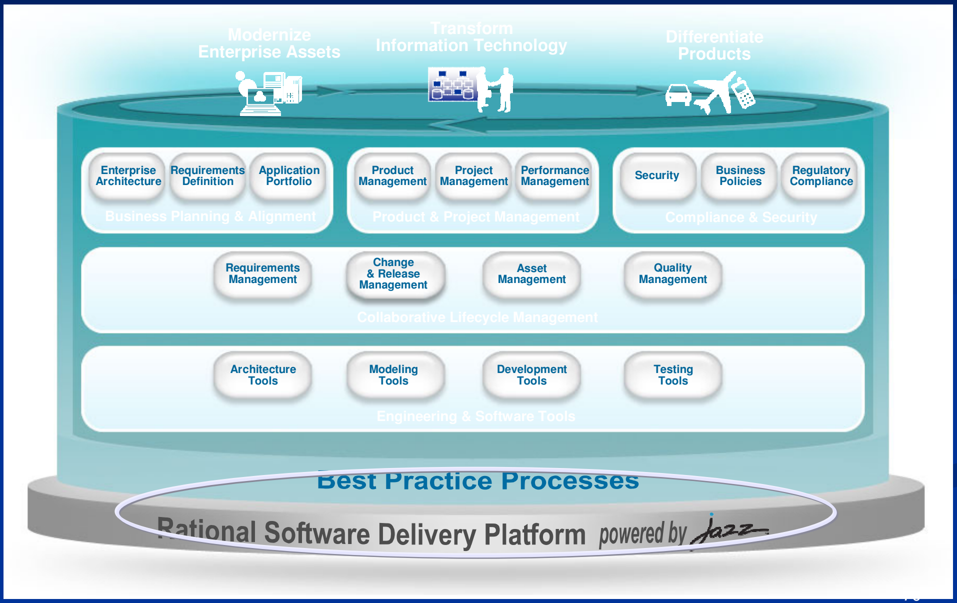
MY COMPUTER DOESN'T WORK! THE HARD DRIVE CRASHED! WHAT DO I DO?!
DID YOU BACK UP?
WHY? IS IT GONNA BLOW?!
STEINFELD ©96

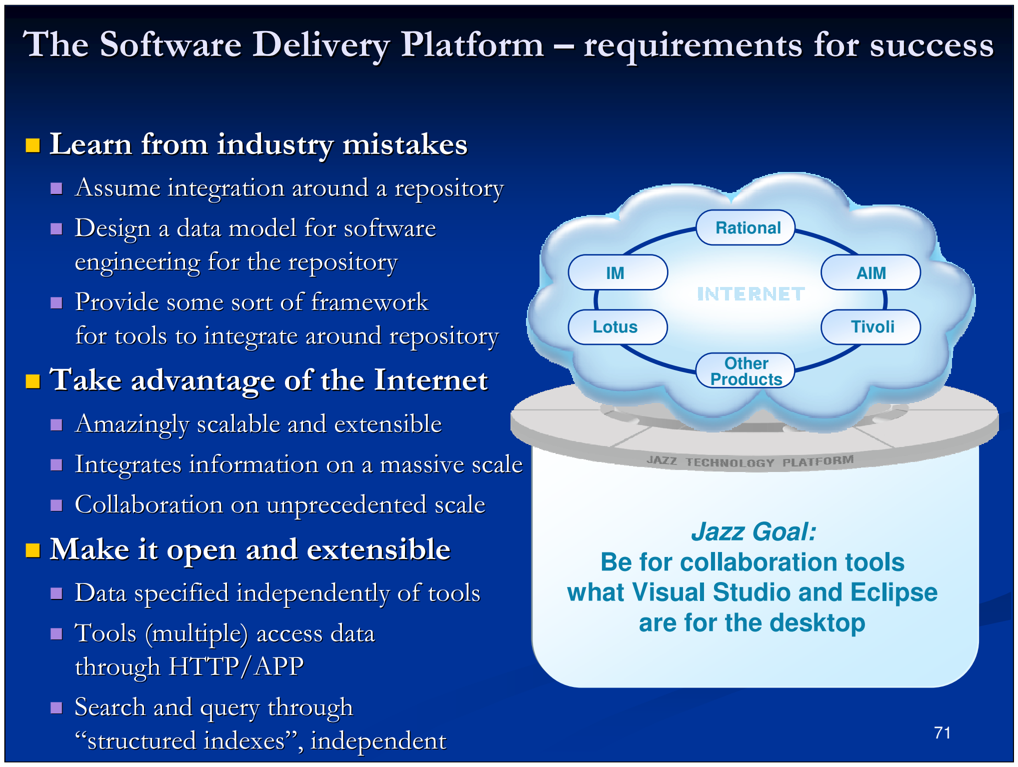
68

Jazz Platform Overview

69

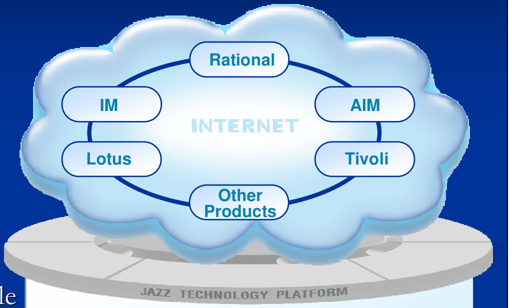
Strategy: Software Delivery Platform powered by Jazz





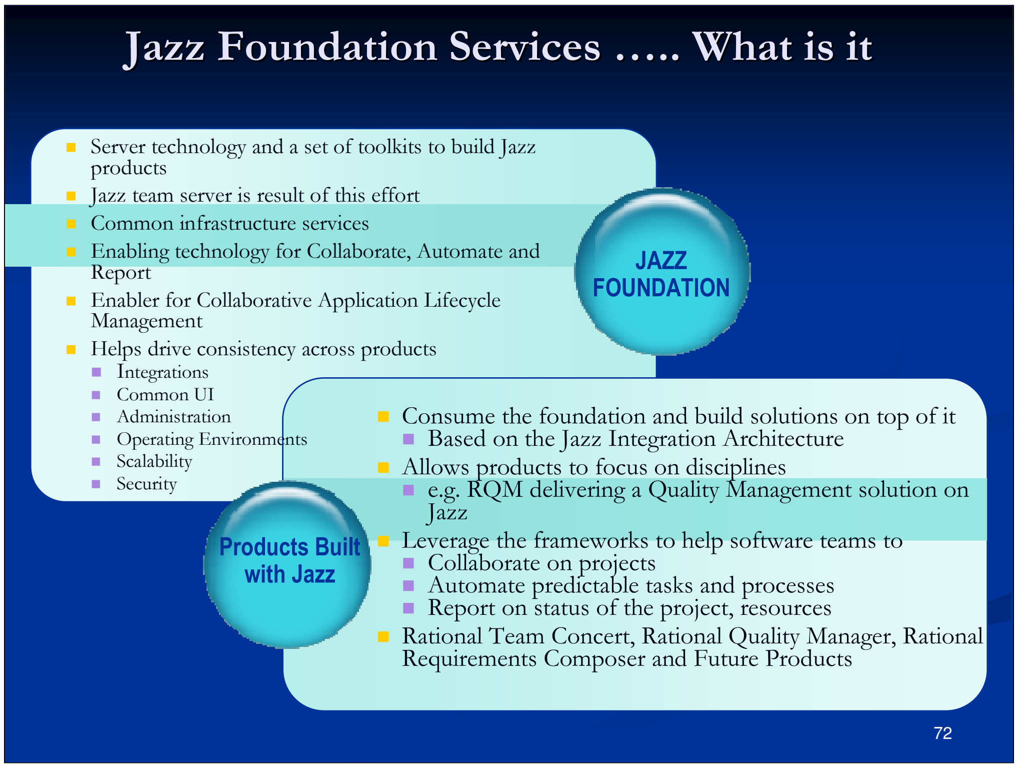
The Software Delivery Platform – requirements for success

- **Learn from industry mistakes**
  - Assume integration around a repository
  - Design a data model for software engineering for the repository
  - Provide some sort of framework for tools to integrate around repository
- **Take advantage of the Internet**
  - Amazingly scalable and extensible
  - Integrates information on a massive scale
  - Collaboration on unprecedented scale
- **Make it open and extensible**
  - Data specified independently of tools
  - Tools (multiple) access data through HTTP/APP
  - Search and query through "structured indexes", independent



Jazz Goal:
Be for collaboration tools what Visual Studio and Eclipse are for the desktop

71



Jazz Foundation Services ….. What is it

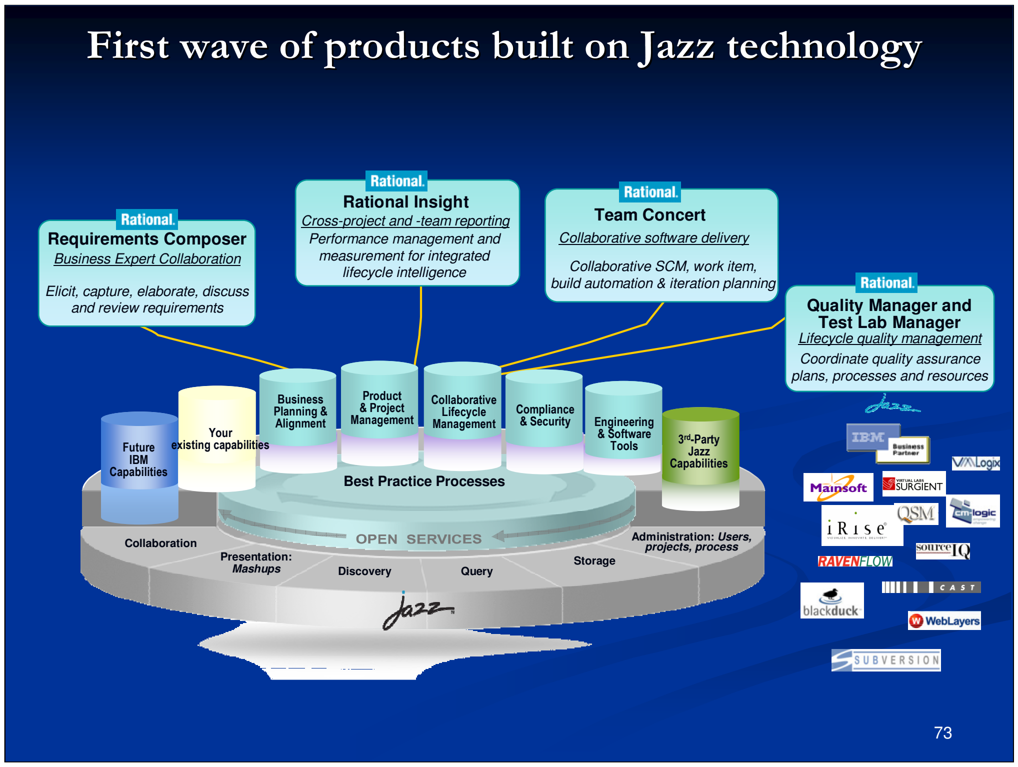
JAZZ FOUNDATION

- Server technology and a set of toolkits to build Jazz products
- Jazz team server is result of this effort
- Common infrastructure services
- Enabling technology for Collaborate, Automate and Report
- Enabler for Collaborative Application Lifecycle Management
- Helps drive consistency across products
  - Integrations
  - Common UI
  - Administration
  - Operating Environments
  - Scalability
  - Security

Products Built with Jazz

- Consume the foundation and build solutions on top of it
  - Based on the Jazz Integration Architecture
- Allows products to focus on disciplines
  - e.g. RQM delivering a Quality Management solution on Jazz
- Leverage the frameworks to help software teams to
  - Collaborate on projects
  - Automate predictable tasks and processes
  - Report on status of the project, resources
- Rational Team Concert, Rational Quality Manager, Rational Requirements Composer and Future Products

72



First wave of products built on Jazz technology

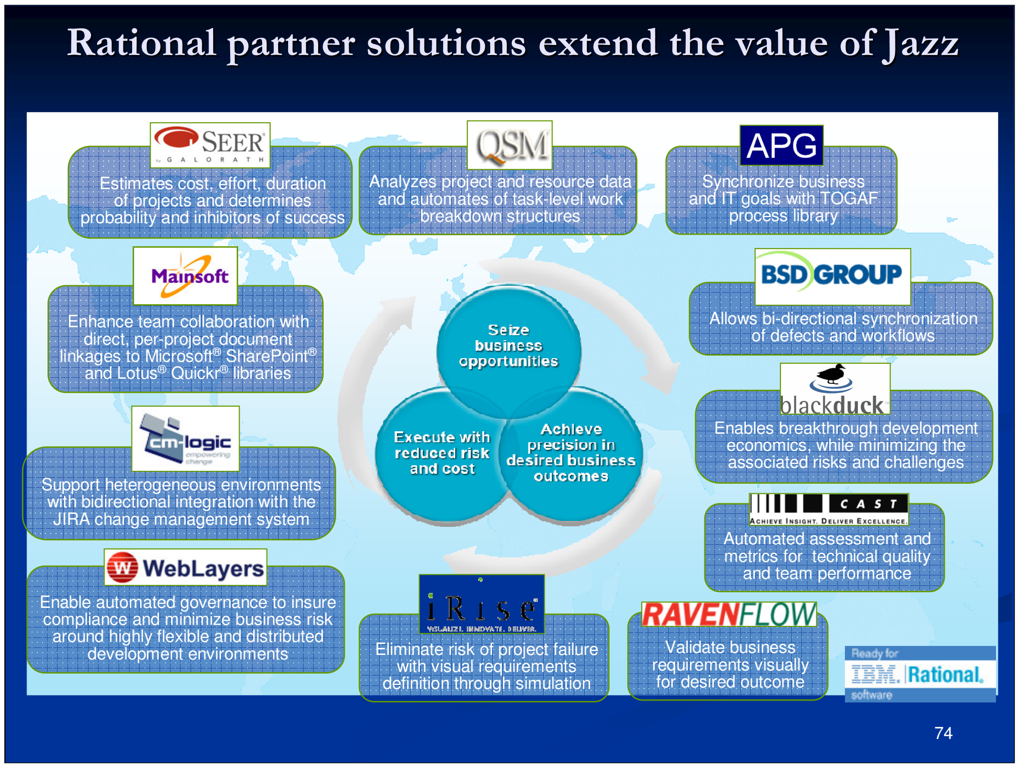
Rational. **Requirements Composer**
Business Expert Collaboration
Elicit, capture, elaborate, discuss and review requirements

Rational. **Rational Insight**
Cross-project and -team reporting
Performance management and measurement for integrated lifecycle intelligence

Rational. **Team Concert**
Collaborative software delivery
Collaborative SCM, work item, build automation & iteration planning

Rational. **Quality Manager and Test Lab Manager**
Lifecycle quality management
Coordinate quality assurance plans, processes and resources

73



Rational partner solutions extend the value of Jazz

SEER by GALORATH: Estimates cost, effort, duration of projects and determines probability and inhibitors of success

QSM: Analyzes project and resource data and automates of task-level work breakdown structures

APG: Synchronize business and IT goals with TOGAF process library

Mainsoft: Enhance team collaboration with direct, per-project document linkages to Microsoft® SharePoint® and Lotus® Quickr® libraries

BSD GROUP: Allows bi-directional synchronization of defects and workflows

cm-logic: Support heterogeneous environments with bidirectional integration with the JIRA change management system

blackduck: Enables breakthrough development economics, while minimizing the associated risks and challenges

CAST: Automated assessment and metrics for technical quality and team performance

WebLayers: Enable automated governance to insure compliance and minimize business risk around highly flexible and distributed development environments

iRise: Eliminate risk of project failure with visual requirements definition through simulation

RAVENFLOW: Validate business requirements visually for desired outcome

Seize business opportunities
Execute with reduced risk and cost
Achieve precision in desired business outcomes
Ready for IBM Rational software

74

Jazz Based Product Suite

Rational Team Concert

IBM Rational Team Concert is a team-aware software development platform that integrates work item tracking, builds, source control, and agile planning. Rational Team Concert interoperates with other products by providing Visual Studio integration and connectors for ClearCase and ClearQuest.

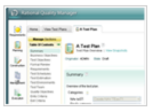
Rational Team Concert for System z

Rational Team Concert for System z provides distributed users with all of the capabilities of Rational Team Concert hosted on the robust System z platform.

Rational Change Management Express

IBM Rational Change Management Express exposes a subset of IBM Rational Team Concert to provide a robust collection of change management features, including work item tracking, process awareness and customization, team awareness, and project health viewing through team reports and Web dashboards.

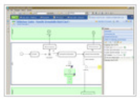
Rational Quality Manager and Rational Test Lab Manager



Rational Quality Manager is a centralized test management environment that helps increase the efficiency and quality of software delivery through test planning, workflow control, tracking and traceability, and metrics reporting. Rational Test Lab Manager, an extended component of Rational Quality Manager, helps to improve the efficiency of the test lab environment and optimize its utilization, cutting workload and saving on test infrastructure.

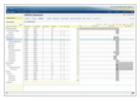
Jazz Based Product Suite

Rational Requirements Composer



IBM Rational Requirements Composer provides a platform for collaborative requirements definition that enables business analysts, client stakeholders and software development teams to elicit, capture, elaborate, discuss, review, and validate requirements using a variety of requirements definition techniques and collaboration capabilities.

Rational Project Conductor



IBM Rational Project Conductor is a project and resource management system optimized for software and systems delivery. It enables project and program managers to plan, schedule, and staff projects, with the right resources working on the right tasks. It provides management with control and visibility over project status and progress, and serves as the central repository for project and program data.

Rational Build Forge

IBM Rational Build Forge is a process execution framework that automates, orchestrates, manages, and tracks all the processes through each handoff within the software development lifecycle to create an automated software factory. Rational Build Forge integrates into your current environment and supports major development languages, scripts, tools, and platforms, allowing you to preserve your existing investments while adding valuable automation, acceleration, notification, and scheduling capabilities.

76