Transforming Software Development through Web 2.0 Collaboration

Brian Bryson, Solution Manager, Software Quality Management
Matt Holitza, Solution Manager, Software Change Management
Patterns of successful business transformation

*Enabled by the billion user Internet*

- **Narrow market focus** → **Expanded market focus**
  - Only 10% of the LEGO product line is sold in retail outlets -- niche products now account for nearly 15% of total ($1.1B) sales.

- **Top-down production** → **Peer production**
  - At P&G, R&D productivity has increased 60%--since 35% of new products have elements that originate *outside* of P&G.

- **Rigors of the ‘cathedral’** → **Openness of the ‘bazaar’**
  - Over half of surveyed developers use open source modules in their applications.*

The Wisdom of Crowds

"Under the right circumstances, groups are remarkably intelligent, and are often smarter than the smartest people in them."

James Surowiecki
Design principals of the collaboration economy

- Act globally
- Be open and transparent
- Let the process evolve
- Create low barriers to participation
- Promote peering and sharing
- Get the governance structures right
- Supply an infrastructure for collaboration
- Collaboration starts internally
Implications for the software delivery domain

Questions we asked...

If software teams adopted the principals of the collaboration economy...

1. How would software delivery teams and processes change?
2. How would we as a vendor change?
3. How would application lifecycle management change?
Team first principles

- Tools know the team
- Each team is different
- Teams are self-organizing
How would software delivery teams and processes change?

- Function first → People first
- Manual tasks → Automate what makes sense…
- Pre-determined tool function → Dynamically extensible
- Proprietary infrastructure → Internet standards
- Lengthy team and project onboarding → Day one productivity
- Rigid process → Creative collaboration
- Project post-mortems → Continuous steering
- Best guesstimates → Collective bottom-up estimates
- Ad hoc shared artifacts → Managed reuse
- Local standards → Industry standards
Recreating the virtual water cooler

- Instant messaging
- Virtual meeting room
- Application sharing
- Centralized info mgt; searching and indexing
- Configuration control of shared artifacts
- Co-browsing
- Electronic document routing and workflow
- Calendaring and scheduling
- Online event notification
- Project resource profiling
- White boards
- Online voting and polling
- Team member presence
- Tools for connected/disconnected use
- Threaded discussions
- Access to personal and project blogs
- Project dashboards and metrics
- Self-publication of content
- Self-administration of projects
- Lightweight peer-to-peer conferencing
- Lightweight group conferencing
- Non-intrusive auditing of changes
- Multiple levels of information visibility
- Personalization of content

Know what is going on, without having to ask.
Implications for the software delivery domain

Questions we asked...

If software teams adopted the principals of the collaboration economy...

1. How would software delivery teams and processes change?
2. How would we as a vendor change?
3. How would application lifecycle management change?
Designed to integrate both partner and competitor resources

Simplifying collaboration across the software delivery lifecycle

An open invitation to collaborate on a common integration architecture for software delivery

Open Services for Lifecycle Collaboration

A proposed architecture for sharing lifecycle resources

- Described on Jazz.net through illustrative resource definitions, service implementations, and use cases
- Inspired by Internet architecture
  - Loosely coupled integration with “just enough” standardization
  - Common resource formats and services
- Innovation from the Jazz technology platform

Barriers to sharing resources and assets across the software lifecycle

- Multiple vendors, open source projects, and in-house tools
- Private vocabularies, formats and stores
Inspired by Internet architecture

- Data specified independently of tools
- All data are resources with URLs
- Tools (multiple) access data through HTTP/APP
- References are embedded URLs
- Resources have representations
- Unprecedented extensibility
- Search and query through “structured indexes”, independent
The Jazz Project: Transforming software delivery

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

- A major investment by IBM to create a scalable, extensible, and collaborative application lifecycle management (C/ALM) platform.
- IBM’s vision of the future of software delivery—globally distributed, fluid and dynamic.
- An evolution of the Rational portfolio, which will evolve to support Jazz technology over time.
- A community at Jazz.net—where you can see Jazz-based products being built.
Jazz.net community site: Transparent development

Suppose we did our development out on the Internet?

- An transparent development laboratory where you can...
  - Communicate with the development team
  - Track the progress of builds and milestones
  - Get the latest product trials and betas
  - Join developers and product managers in discussion groups
  - Submit defect and enhancement requests

Suppose we let you try it for a team of 10 for FREE?

FREE!

DOWNLOAD Express-C

Free for 10 users! Zip file download with Client for Eclipse IDE and server. Client for Microsoft Visual Studio IDE downloaded separately:
Other Download Options...
Implications for the software delivery domain

Questions we asked...

If software teams adopted the principals of the collaboration economy...

1. How would software delivery teams and processes change?

2. How would we as a vendor change?

3. How would application lifecycle management change?
Hasn't this been done before?

- Many previous attempts at more integrated team software engineering environments
  - Assume integration around a database/repository
  - Design a data model for software engineering for the repository
  - Provide some sort of framework for tools to integrate around the repository

A services oriented and platform based approach...
The Jazz core offerings

- **Team Concert**
  - *Innovation Through Collaboration*
  - Unify by “thinking & working” in unison with real-time project health

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

- **Quality Manager**
  - *Collaborative Business-driven Quality*
  - Coordinate quality assurance plans, processes and resources

**Open Lifecycle Service Integrations**

**Rational Team Concert**
**Rational Requirements Composer**
**Rational Quality Manager**
**Business Partner Jazz Offerings**

**Rational**: ClearQuest
**Rational**: ClearCase
**Rational**: Build Forge
**Rational**: Requisite Pro
**Rational**: Asset Manager

**JAZZ TEAM SERVER**
Essential attributes of the Jazz Foundation

**Collaborate**
Deliver transparency of teams and projects for continuous, context-sensitive collaboration.

**Automate**
Automate non-creative tasks with automated processes and workflows.

**Report**
Deliver real-time insight into programs, projects and resource utilization.

Improve knowledge and practice maturity with an environment that develops individual and team talent.
Why collaborate?

- Accelerate project and team onboarding
- Build team cohesion
- Leverage talent across and beyond the enterprise
- Enable flexible resourcing

Deliver transparency of teams and projects for continuous, context-sensitive collaboration

Reduce costs
Why automate?

- Automate processes, workflows and tasks to improve individual and team agility
- Automate best practices that improve team agility
- Reduce tedious and time-consuming manual tasks
- Minimize non-productive wait times
- Minimize administrative busywork

Speed time-to-market
Why report?

Deliver real-time insight into programs, projects and resource utilization.

- Identify and resolve problems earlier in the software lifecycle
- Get fact-based metrics -- not estimates -- to improve decision making
- Leverage metrics for continuous individual and team capability improvement

Source: GBS Industry standard study
Defect cost derived in assuming it takes 8 hrs to find, fix and repair a defect when found in code and unit test.

$80/defect
$240/defect
$960/defect
$7,600/defect
Example: A team dashboard
Rational Requirements Composer
Harnessing the wisdom of the business and IT

Breaking down silos to get requirements right the first time
Rational Team Concert
Harnessing the wisdom of developers

Plan: Interactive real-time development planning around the world
Rational Team Concert
Harnessing the wisdom of developers

Estimate: Bottoms up estimates + Monte Carlo Analysis
Rational Team Concert
Harnessing the wisdom of developers

Develop: Collaboration in-context (Web 2.0 meets software change management)
Rational Team Concert
Harnessing the wisdom of developers

Manage: Aggregate relevant data for the team, the project, the individual
Rational Quality Manager: Improve operational efficiency

**What is needed:** Know what to expect from others and what others expect from you

- Stay on schedule and under budget.
- Dynamic Test Plan is always current
- Team and individual task workflow
- Enforceable process discipline
- Test Lab Management
- Requirements Through to Defects

Team event log

Individual Task List

Stay on schedule and under budget.
What analysts are saying...

- **Liz Barnett, Ez-Insight:** "With the Jazz project, Rational has developed **breakthrough technology** and is **poised to set the standard for collaborative ALM.**"

- **Paul Herzlich, Ovum:** "IBM has taken the opportunity to exploit the Jazz platform’s power inventively. Its ‘living’ test plan is a **masterpiece of applying new technology to a familiar problem**… What is being delivered will demonstrate convincingly that IBM is raising the standard for a test management product."

What customers are saying...

"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
Additional resources

- Download Rational Team Concert Express-C (10 free developers)
  http://jazz.net
- Find out more about Jazz foundation
  http://ibm.com/rational/jazz
- Explore Jazz product tutorials, demos and other developer learning resources
  http://ibm.com/developerworks/spaces/jazz
- Learn more about the Jazz technology and the future IBM Rational product roadmap
  http://ibm.com/rational/jazz/roadmap
Come see us in the Expo Hall!
Transforming Software Development through Web 2.0 Collaboration

Brian Bryson, Solution Manager, Software Quality Management
Matt Holitza, Solution Manager, Software Change Management