Agile Software Development -Experiences from the Trenches

Johannes Rieken

IBM Rational Zurich Research Lab



Manifesto for Agile Software Development

Individuals and interactions over processes and tools Working software over comprehensive documentation Customer collaboration over contract negotiation Responding to change over following a plan

(see http://agilemanifesto.org/)



What is Agile

An iterative and incremental (evolutionary) approach performed in a highly collaborative manner with just the right amount of ceremony to produce high quality software in a cost effective and timely manner which meets the changing needs of its stakeholders.

So Agile is not a method. Its more like a conceptual framework (abstract class) with concrete implementations like, Eclipse Way, OpenUp, Scrum, ... or the My Company Way ©.





Agile is Not

Low ceremony

- It is (very) formal and has (very) specific practices

Do what you want (cowboy coding)

- Requires lots of discipline

Easy to do

- See above 🙂
- Needs cultural changes
- A silver bullet





Core Principles

- Short development iterations (3 6 weeks)
- Ongoing customer / stakeholder involvement
- Ongoing investment in code quality (refactoring)
- Retrospectives (improve the process)
- Self organizing teams



Short iterations – the main driver

- Allows feedback / quality checks any n weeks
- Each milestone is a miniature development cycle
 - plan, execute, test, ship, retrospective
- The iteration outcome (library, product, ...) must be shippable / consumed
 - Other teams, betas, demos, ...





The Eclipse Way Practices





7

What is behind the Eclipse Way

Practices underpinned with values

- ship quality on time
- Used, developed and improved over time

Practices are from all kinds of sources

- XP, Scrum, Crystal Clear, RUP, ...
- Patterns Organizational Patterns of Agile Software Development Coplien
- It is not low ceremony
 - Approvals, verifications, reviews

It is agile: incremental, iterative, collaborative, transparent, customizable

> Many effective teams work like this



- 4 week iterations ⇒ end with an end of iteration demo
- 8 week milestones ⇒ announced with New & Noteworthy ⇒ retrospective at the end

Our Roles

- Project management committee (PMC)
 - -Accountable for release plan
 - -Themes to work on
 - -Facilitator, coordinator
 - encourages participatory decisions
 - e.g. top 5 architectural issues
- Component lead / Team
 - Accountable for iteration plan, test plans
- Contributor
 - Accountable for estimates, code, tests, design
 - Plays many roles
 - Developer, Tester, Architect
 - Customer support, Release Engineering







Iteration Plan Input





Iteration Plan Input - continued

PMC

- Requirements (release plan): traceability, ...
- Themes: improve performance, ...
- Concrete tasks: externalize strings
- Community / Stakeholders
 - Requirements
 - Enhancements
 - Defect reports
- Team
 - Architectural issues
 - Redesigns
 - Product ideas
- Defect Backlog
 - Previous test passes
 - Self hosting





Iteration Plan

- Goal: an estimated 4 weeks plan approved by the team, PMC, and stakeholders
- In the RTC project the steps to get there differ from team to team
 - Team lead creates initial plan based on input from PMC, stakeholders
 - Team decides on defect backlog, architectural issues, ...
 - Team members estimate their work
 - PMC buddy approves the plan
- In Scrum the sprint (iteration) plan is defined in a sprint meeting:
 - Stakeholders (product owner) are part of that meeting. They manage a prioritized list of product requirements, enhancements
 - Team manages defect backlog, architectural issues, ...
 - Team estimates

Iteration Checkpoints

- Daily (stand-up meeting, team lead contributor discussion, ...)
 - What have you done yesterday
 - What are you going to do today
 - Any road blocks
 - How many work is remaining
- 2 times a week (Planning call)
 - Inter team issues
 - Progress on PMC and Community work
- Every two weeks
 - Iteration Plan walkthrough with PMC buddy
- End of iteration
 - New & Noteworthy / Milestone
 - Demo for PMC, stakeholders and other teams



Tracking Progress

To hit the end target of an iteration it is essential for the team to track its progress. Teams in RTC do so using Agile Planning tools and Dashboards.

Init Plan * Team Areas: Xinit Team [Streadous 4.4 m2 (\$/26/08 - 6/25/08)] Science [127.0mm] Provents: Xinit Team [Streadous 4.4 m2 (\$/26/08 - 6/25/08)] Science [127.0mm] Provents: Xinit Team [Streadous 4.4 m2 (\$/26/08 - 6/25/08)] Science [127.0mm] Provents: Xinit Team [Streadous 4.4 m2 (\$/26/08 - 6/25/08)] Science [127.0mm] Provents: Xinit Team [Streadous 4.4 m2 (\$/26/08 - 6/25/08)] Science [127.0mm] Provents: Xinit Team [Streadous 4.4 m2 (\$/26/08 - 6/25/08)] Science [127.0mm] Provents: Xinit Team [Streadous 4.4 m2 (\$/26/08 - 6/25/08)] Science [127.0mm] Provents: Xinit Team [Streadous 4.4 m2 (\$/26/08 - 6/25/08)] Science [127.0mm] Provents: Xinit Team [Streadous 4.4 m2 (\$/26/08 - 6/25/08)] Science [127.0mm] Provents: Xinit Team [Streadous 4.4 m2 (\$/26/08 - 20.0mm] Provents: Xinit Team [Streadous 4.4 m2 (\$/26/08 - 20.0mm] Provents: Xinit Team [Streadous 4.4 m2 (\$/26.0mm] Provents: Xinit Team [Streadous 4.4 m2 m2 m2	Home Current Milestone Trends + Add New Tab	
Team Areas: Jahr Team Ileration: 4.4 m2 (5/16/08 - 6/15/08) <u>5.Concol</u> <u>17.Concol</u> Progress: 44 / 208 -38 h Image: 4 h <		
Bill Cassavelii Popues: # / 12 (+7). Binmadd 100% Past (1 doed) Popues: # / 12 (+7). Binmadd 100% Past (1 doed) Sort By Sort By Conference in Europe Out of office from May 19, 2008 to Jun 2, 2008 Sort By Dut of office from May 19, 2008 to Jun 2, 2008 Sort By Sort By Public (2 open) @ 2 hours Unassigned 40 Past (2 doed) @ 30 mins Unassigned 40 Past (2 doed) @ 2 hours Unassigned 40 Past (2 doed) @ 2 hours Popues 20 / 56 = -0 > Past (2 doed) @ 2 hours Popues 20 / 56 = -0 > Past (2 doed) @ 2 hours Popues 20 / 56 = -0 > Past (2 doed) @ 2 hours Popues 20 / 56 = -0 > Past (2 doed) @ 2 hours Popues 20 / 56 = -0 > Past (2 doed) @ 2 hours Popues 20 / 56 = -0 > Past (2 doed) @ 2 hours Popues 20 / 56 = -0 > Past (2 doed) @ 2 hours Popues 20 / 56 = -0 > Past (2 doed) @ 2 hours Popue 2 / 50 / 60 / 70 / 70 / 70 / 70 / 70 / 70 / 7	Conce Work items X = X R Barradown	
We After method not called after my test timeout in 4.3.1 ③ I hour Unassigned 16 Future Expressive/Planned Work (0 open) Proceeding Rended Work (0 open) Programed Work (0 open) Programed Work (0 open) Programed Violation Programe 0 (0 open items: 1)	The function of the function o	Ex Flanning
verview Planned Items Charts		

Single Iteration



Jasa.

😼 | 🔶 🗋 Auto-save 🛛 Save.

New Work Items by Severity Team Aver User Development Soils Planning Category: All Interval: 0.6 RC1, 0.6 RC2, 0.6 RC5, 0.6 RC3, 0.6 RC4

Unclassi Hinor Nom al Hajor Critical Biochar

Development - Isolate Work not people

- Repository workspaces Provides individual isolation.
 You don't have to make your changes visible to the team just to backup or use the repository features.
- Streams Provides team / project isolation.
- Suspend and Resume Provides task level isolation for personal work.
- Team areas Provides process isolation.



Stream setup for RTC Development



17

Build Process

To be able to ship a product every n weeks building it must be absolutely painless and fully automated. And the build quality must be verified using automatic unit tests.

A 120080618-1642 🕅				
🏝 Build weekly.integration.jazz I20080618-1642 🔻	ar 🖑 🆑 Save		Command Prompt	
✓ Completed Durator: 3 hours, 7 minutes, 10 seconds Start Time: June 18, 2008 10:42: 17 PM Completed: June 19, 2008 10:42: 17 PM Completed: June 19, 2008 11:49:27 AM Status Trend: ✓ Request Build Request Build Request Build Request Build be executed on the first avail ✓ Request Build Request Build be executed on the first avail ✓ Request Build Request Build Status ✓ Request Build Reques	Reported Work Items None reported against the build Create a new work Item Image: State and executing work Item	versus	$\begin{array}{c} 08.7.4.2.2068 & 12.22 & PH & (D1R) \\ 12.4.8.2067 & 01.4.4 & PH & (D1R) \\ 0.7.4.2.2067 & 01.2.6 & PH & (D1R) \\ 12.4.8.2067 & 01.2.6 & PH & (D1R) \\ 12.4.8.2067 & 01.2.6 & PH & (D1R) \\ 12.4.8.2067 & 01.42 & PH & (D1R) \\ 0.7.8.2067 & 01.42 & PH & (D1R) \\ 0.7.8.2067 & 01.42 & PH & (D1R) \\ 0.7.8.2067 & 01.42 & PH & (D1R) \\ 12.4.8.2067 & 01.49 & PH & (D1R) \\ 12.4.8.2067 & 01.27 & PH & (D1R) \\ 12.4.8.2067 & 01.27 & PH & (D1R) \\ 0.7.8.2068 & 01.49 & PH & (D1R) \\ 0.7.8.2068 & 01.419 & PH & (D1R) \\ 0.7.8.2069 & 01.48 & PH & (D1R) \\ 0.7.8.2069 & $	<pre>com.ibm.team.uvorkitem.ueb com.ibm.team.uvorkitem.ueb com.ibm.team.zuvich.featur.tests.tools com.ibm.team.zuvich.tests.tools com.ibm.team.zuvich.tests.tools com.ibm.sprocess.com.ibm.team.server.orgi.feature continuous.uvorkitem.com.ibm.team.server.orgi.feature continuous.uvorkitem.com.ibm.team.uvorkitem.server.featu com.ibm.team.zuvich.tests.com.ibm.team.uvorkitem.server.featu com.ibm.suvorkitem.com.ibm.team.uvorkitem.server.featu com.ibm.suvorkitem.com.ibm.team.uvorkitem.server.featu com.ibm.suvorkitem.com.ibm.team.uvorkitem.server.featu com.ibm.suvorkitem.com.ibm.team.uvorkitem.server.featu com.ibm.suvorkitem.com.ibm.team.uvorkitem.server.featu com.ibm.suvorkitem.com.ibm.team.uvorkitem.server.featu com.suvorkitem.com.ibm.team.server.featu com.suvorkitem.server.feature com.suvorkitem.server.server.feature com.suvorkitem.server.server.feature com.suvorkitem.server.sever.sever.feature com.server.server.seve</pre>
Summary Acondes Compliander Some Logs Downloads External Links Properties				



Staged Builds

- Team builds team's integration stream / n times a day
 - discover component problems
- Nightly builds project continuous integration stream
 - discover integration problems between components



Coverage

Build Tracking

Equally important is to track whether a build has compile errors and/or failing test case. The majority of the builds should be green.



Agile @ Scale: Component Based Development

Component based

- A team is responsible for one or more component at one site co-location
 - "architecture follows organization"
- Components are distributed across sites

API first

- An API is a commitment
- Producer / consumer relationships

Team Concert supports component based development

- Team owns a stream
- Team shares changes in a stream
- Team owns a component
- Stream references components



Eclipse Components



Stabilization

Each iteration ends with a stabilization phase. In the RTC project this is the last week of a 4 weeks iteration

First two days is testing

- Teams create test plan
- Teams do actual testing
- Next two days is fixing
 - Only critical defects are fixed
 - Every fix requires a team lead approval & a code review
- Last day is sanity check day
 - Rebuild only for stop ships
 - All teams sign off on Milestone build







Process Awareness

"While many aspects of process might be automatable, we found that productive processes emphasize the creative value added by the people in the process."

> Organizational Patterns of Agile Software Development – Coplien

- Support many different practices and processes
- Rational Team Concert is Process neutral
- Reactive, not controlling
- Specific to the team, development phase
- Reduce team member mistakes
- Free the team members minds



Retrospective

After each iteration teams reflect on what worked well and what didn't.

- Retrospectives are captured in a special work item
- Define actions how to tune the process to get more effective
- PMC does a retrospective as well

🛿 💭 45392: MS Retrospective - Jazz Project 🎇 15							
🖟 Retrospective 45392 🝷							
Summary: M5 R	etrospective						
▼ Details		▼ Description					
Created:	Feb 22, 2008 5:58 PM						
Created By:	Erich Gamma	Highlights of Jazz M5					
Team Area:	Work Item / Jazz Project	- editable Wiki now available on Linux (and Mac OS X 'Leopard') since XULRunner is gone					
Filed Against:	Work Items	- Team Central UI overhaul - Reusability of Work Them entition components					
Owned By:	André Weinand	- User specific mail configuration - Structured process spec editors (instead XML editing) - HTML based Work Item channe formatting					
Planned For:	0.6 M5	- Web UI: more in sync with Eclipse UI					
Resolved:	Feb 28, 2008 12:19 AM	Repo:					
Resolved By:	André Weinand	- Virtual REST service: foundation for Work Item export service					
		What worked well Overall					
Quick Information		- build times down to 2 hours					
Subscribers	(2): DB, EG	- push of products (aka deliverables) to our server					
🕖 Attachment	s (1): 10615	- deltas are great					
Copies (1):	50595	- short twin-iterations are good for getting cross team dependencies fixed early (in M5D1 instead of M5)					
E Copied Fron	n: 37453	What worked well in our teams					
C Mentions (2)	- Numbers: resolved fixed APT(313), Work Items & Foundation UI (483)					
Mentioned By (1)		- iteration planning worked very well					
		 - most of the reactives from the original Jazz to breadse plant are finished - endgame week: build input for "big features" on Monday, "small fixes" on Tuesday -> testable build available early 					
		What didn't work well Overall - private builds sometimes run too long - we should not have to debug build problems (e.g. disabled ijt) - process spec pluggable editors are late - continuous stream is often in undefined state it would be cool of a green continuous build could automatically push a baseline to continuous stream - provisioning war					



Endgame

Convergence process applied before release

- Sequence of test-fix passes
 - community event

With each pass the costs for fixing are increased

- higher burden to work on fix for a problem
- higher burden to release a fix for a problem
- focus on higher priority problems



What Else is Important

Transparency

- Who is doing what
- How good is the progress

Traceability

. . .

- Which change sets are in a build
- Who fixed the bug
- ...

Team First

- Joining a team
- Team controls the way it works





Recommendations

- Make sure that you can build the product by pressing a button
 - Run integration build if necessary
- Ensure quality with automated unit tests
- Being iterative is the key. It will induce almost all the rest.
 - Start with 4 6 weeks iteration and stick to the rhythm. Adjust (after a while) if necessary
 - Ensure that the iteration produces a deliverable that is consumed
 - Plan, execute, test
 - Review what you have done and improve (the process)