

Software Development With Kanban

Radford University, September 4th 2013

About Me

Racker for 4 years



2009-2011: Mail Backend

2011-present: Cloud Control Panel

2012-present (1 year): Team lead

Before That:




Our Product

CLOUD SERVER

radford is great

⚙ Actions ▾

Server Details

Server Status	Active
ID	b89a136a-5f1a-428d-b328-ae26a4b71263
System Image	 Arch 2013.6 · Rebuild...
Size	512 MB RAM, 20 GB Disk · Resize...
Disk Partition	Automatic ?
Monitoring Agent	Not installed, host checks unavailable · Install Agent...
Region	Dallas (DFW)
Server Type	Next Generation Server
Reverse DNS	0 Records · Add Record...
Created Date	Aug 1, 2013 at 5:55 PM
Last Updated	0 minutes ago

Managing Your Server

LOG INTO YOUR SERVER NOW

For Linux, use the command below to log in via SSH. Open a terminal application and run:

```
ssh root@198.61.203.202
```

[More Remote Login Commands »](#)

HELP ME WITH...

- [Configuring Basic Security](#)
- [Rebuilding My Server](#)
- [Creating a Monitoring Check](#)

[Learn More about Cloud Servers »](#)

WHAT'S NEXT?

- [Configuring and Using DNS](#)
- [Load Balancing My Servers](#)
- [Configuring Outbound Email](#)

[Visit Our Knowledge Center »](#)

Our Team

20 developers in two locations

- Blacksburg, VA
- Porte Allegre, Brazil

4 quality engineers

2 product team members

1 project manager

Agile Development Methodologies

Agile is a bucket term for software development practices that emphasize iterative change.

Less Agile	More Agile
quarterly releases	weekly releases
code review per iteration	code review per story
strictly followed requirements documents	changing requirements when it makes sense for business
QE team must certify releases	everyone is responsible for release quality

The Agile Grocery Store

“Agile” is not one formal set of practices

Agile practices and values solve problems around communication and delivery

Select values and practices that address your team's problems



Common Agile Themes

Deliver value in small increments: *stories*

Encourage *communication* between developers and business members

Release and test frequently to reduce risky integrations

An Example Agile System

Scrum:

- *sprints*: iterations with a set length of time
- stories prioritized at the start of a sprint
- at the start of a sprint, development team commits to a certain amount of work
- demo and release at the end of a sprint

Personal Comment

I have three years of experience with scrum

I've done two week, three week, and one month sprints

Scrum is great

General Comments on Scrum

Scrum addresses a certain set of problems

- 6 month releases
- Constantly shifting priorities
- Developers working on things that don't have business value

Iterations are arbitrary lengths of time

- (Shorter iterations are generally better)

Problems Our Team Faced

Large team

- multiple streams at work at once
- lots of overhead putting together iterations

Releasing not a big deal

- 5-6 deploys per day since project start

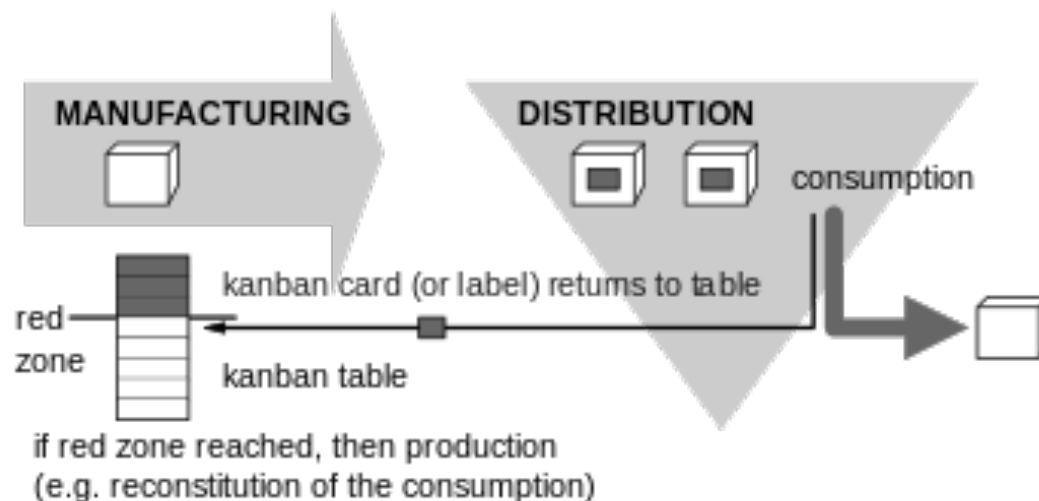
Greenfield development

- some work more experimental in nature

Kanban in Manufacturing

Disclaimer: I am not a manufacturer :)

- When a card is removed from table, order replacements
- Defective products do not move forward
- *Pull system*: ask for work when idle



Kanban Ideas for Software Dev

Limit work in progress

Focus on quality

Deliver often

How do our team processes enforce these practices?

Limit Work in Progress

Make “working on something” a choice

Focus only on the things that are *valuable*

“Valuable” is a *team* decision, some examples:

- delivers business value to customers
- enables faster delivery of future features
- enables better understanding of customer application use, to inform future work

Limit Work in Progress

Okay, you have 4 developers on a team.

What should your “in development” (max # of stories happening at once) limit be?

Limit Work in Progress

4 developers working on 4 things at once may encourage the wrong things:

- Encourages story “tunnel vision”
- May lead to specialized knowledge
- Story “ownership” can mean design conversations don’t happen

Limit Work in Progress

Some candidate 'in dev' limits:

- 3 developers - 2 stories
- 4/5 developers - 3 stories
- 6 developers - 4 stories

This relies on “stories” being similarly sized units of work!

Focus on Quality

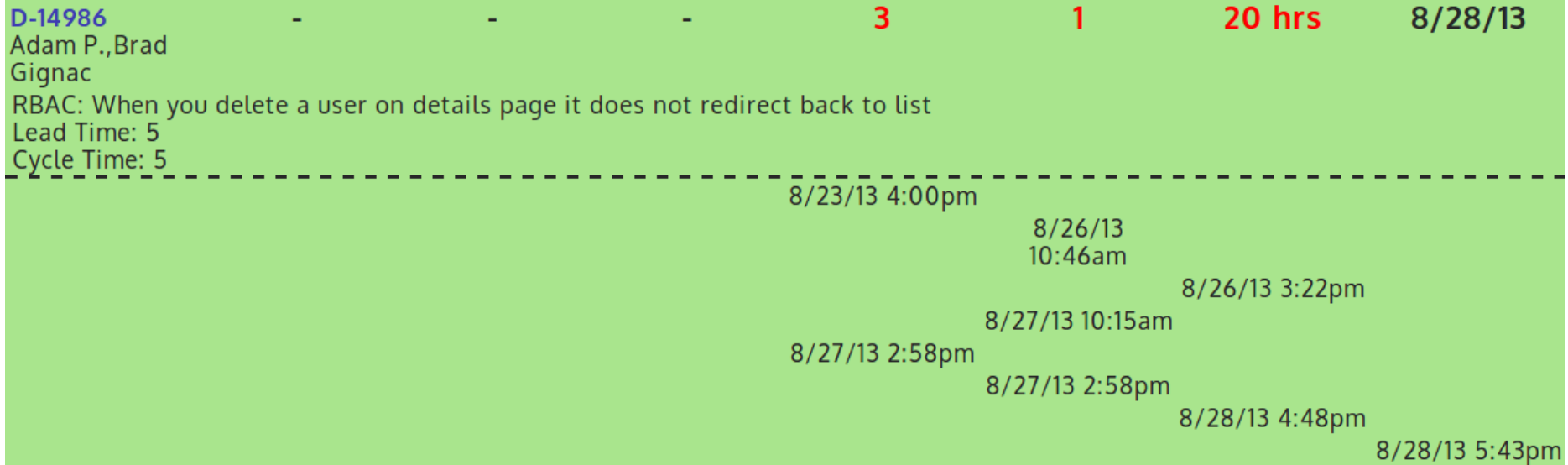
Software Defects are rework

Track defect rate and aim for 0%

Focus on Quality

Track stories that “move back” between phases

This story moved from “dev done” to “in dev”



Focus on Quality

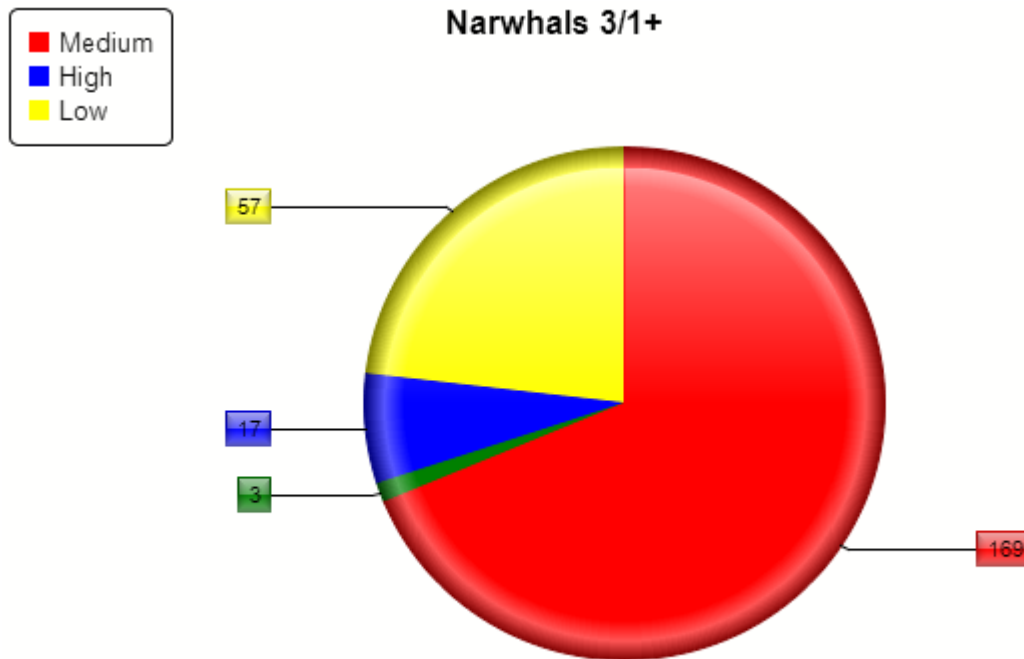
We have an infinite amount of work :)

By working on something you are implicitly choosing **not** to work on other things

Track story priority:

- High: “critical fix”
- Medium: “on roadmap”
- Low: “not on roadmap, doesn’t matter if it gets fixed”

An Example Priority Graph



Focus on Quality

Phases of a story card

- “Ready for Development”
 - work is waiting to be pulled in
- “In development”
 - work is waiting to be finished
- “Dev done”
 - work is waiting to be certified by a product manager

Testing is “baked in” with development work

Work Visualization With Storyboard

Ready for Dev 5	In Dev 2	Dev Done 5	Accepted
<div><div><div><div><div></div><div>B-54244</div><div></div></div><div>Additional Flavors: Remember value selected per flavor class when switching back and forth</div><div>M</div></div></div></div>	<div><div><div><div><div></div><div>B-54243</div><div></div></div><div>Additional Flavors: Show more information on Server Details (requires underline tooltip)</div><div>Bill W.</div><div>M</div></div></div></div>	<div><div><div><div><div></div><div>D-14643</div><div></div></div><div>Cloud files last modified timestamp shows 12am instead of 12pm</div><div>Hartsock</div><div>M</div></div></div></div>	
<div><div><div><div><div></div><div>B-55499</div><div></div></div><div>Additional Flavors: Show pricing</div><div></div></div></div></div>	<div><div><div><div><div></div><div>B-55733</div><div></div></div><div>Additional Flavors: Use old flavor selection for first gen servers</div><div>Jlgar</div><div></div></div></div></div>		
<div><div><div><div><div></div><div>B-55694</div><div></div></div><div>Additional Flavors: Sort RAM slider in ascending order</div><div></div></div></div></div>			

Deliver Often

Multiple Deploys Per Day

- Tests pass
- Code review
- Merge
- Deploy to preprod (like production)
- Deploy to production
- Process takes 1-2 hours

Deliver Often

 [Back to Dashboard](#)

 [Status](#)

 [Changes](#)

 [Workspace](#)

 [Build with Parameters](#)

 [Delete Project](#)

 [Configure](#)

 [Dependency Graph](#)

Build History [\(trend\)](#)

	#179	Sep 3, 2013 4:57:25 PM
	#178	Sep 3, 2013 3:41:22 PM
	#177	Sep 3, 2013 1:54:18 PM
	#176	Aug 30, 2013 7:39:02 PM
	#175	Aug 30, 2013 5:58:02 PM
	#174	Aug 30, 2013 5:52:06 PM
	#173	Aug 29, 2013 8:44:51 PM

Project Reach_Production




[Workspace](#)



[Recent Changes](#)

Downstream Projects

-  [Reach_Production_DFW](#)
-  [Reach_Production_Sydney](#)

Permalinks

- [Last build \(#179\), 20 hr ago](#)
- [Last stable build \(#179\), 20 hr ago](#)
- [Last successful build \(#179\), 20 hr ago](#)
- [Last failed build \(#174\), 4 days 19 hr ago](#)
- [Last unsuccessful build \(#174\), 4 days 19 hr ago](#)

Understanding Your Problems

Effectiveness relies on accurately understanding your problems

You need to constantly evaluate your team's effectiveness

Top performance is hard and if you don't think you are having problems you need to look further :)

Understanding Your Problems

Retrospectives

- team meetings where we assess how things are going
- important that everyone talks

Understanding Your Problems

Root Cause Analysis

- when there is a problem, ask “why”
- to that answer, ask “why”
- repeat (usually 5 times)
- agile methodologies stress “why”s around testing and communication

Our Advantages

Developers and leaders all have multiple years of working with agile software development

Most code is new (almost all is < 2 years of age)

Continuous Delivery since project start

Integrated Testing since project start

Our Current Challenges

Product development across other teams is still iteration-based

- We are a UI development team
- We have limited ability to impact concerns around server provisioning, etc
- How can we get more iterative feedback to inform project planning?

Making estimation uniform across multiple streams of work

To Recap

Agile methods stress iteration

Select methodologies that address your problems

References

1. Kanban: Successful Evolutionary Change for Your Technology Business -- David J. Anderson)
2. Agile Manifesto -- <http://agilemanifesto.org>
3. Wikipedia: Kanban -- <http://en.wikipedia.org/wiki/Kanban>