

Agile Project Management



What Is Agile?

- Agile is a group of software development methodologies
 - Scrum
 - Extreme Programming (XP)
 - Lean
 - Etc.
- Key Characteristics:
 - Small increments
 - Adaptive to change
 - Collaborative



Defining Agility

- Individuals and interactions over processes and tools
 - Encourage engagement between functional areas
 - Avoid using documents to hand off information
- Working software over comprehensive documentation
 - Focus on incrementally attacking the problem
 - Stay releasable

Defining Agility

- Customer collaboration over contract negotiation
 - Prioritize based on business value
 - Work together to ensure that value is maximized
- Responding to change over following a plan
 - Plan just enough (no more than necessary)
 - Defer to the last responsible moment
 - Stay flexible and leverage what you've learned

Why Do It?

- It results in better software
 - Higher productivity (you get what you need quicker)
 - Higher quality
 - More customer satisfaction
 - More visibility
 - Better morale



Roles

- Product Owner
- Scrum Master
- Team Member



Product Owner

- Prioritizes the backlog
- Communicates what is important ... and what is not
- Is a proxy for the customer



Scrum Master

- Responsible for the process
- Facilitates agile meetings
- Helps to remove road blocks



Team Member

- Signs up for work
- Asks questions
- Collaborates with others
- Communicates progress / blocking issues
- Makes it happen



What Does It Look Like?

- Backlog
- Release
 - Release Planning
 - Iterations (1-4 weeks long)
 - Iteration Planning
 - Daily standup
 - Demo
 - Iteration Retrospective
 - Release Retrospective



The Backlog

- A ranked list of stories
- What is a story?
 - A scenario that we must do work to implement which results in business value
 - Typically in the form of: “As a <type of user>, I want <feature> so that <business value>”
 - Good stories meet the INVEST criteria



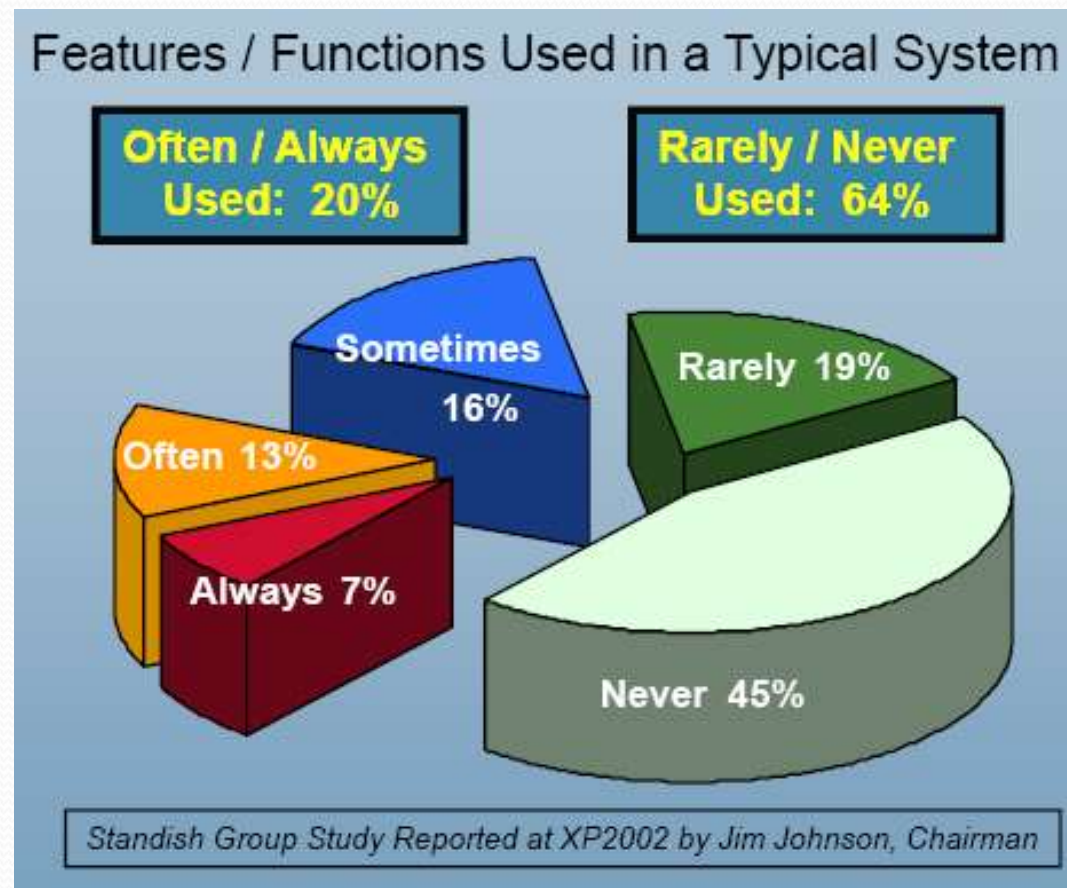
Example

Post a Job

- As a recruiter I want to be able to post a job to the web site so that I can generate interest in the position.

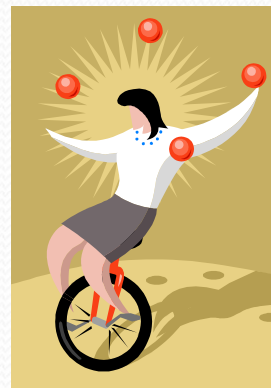


Why Prioritize?

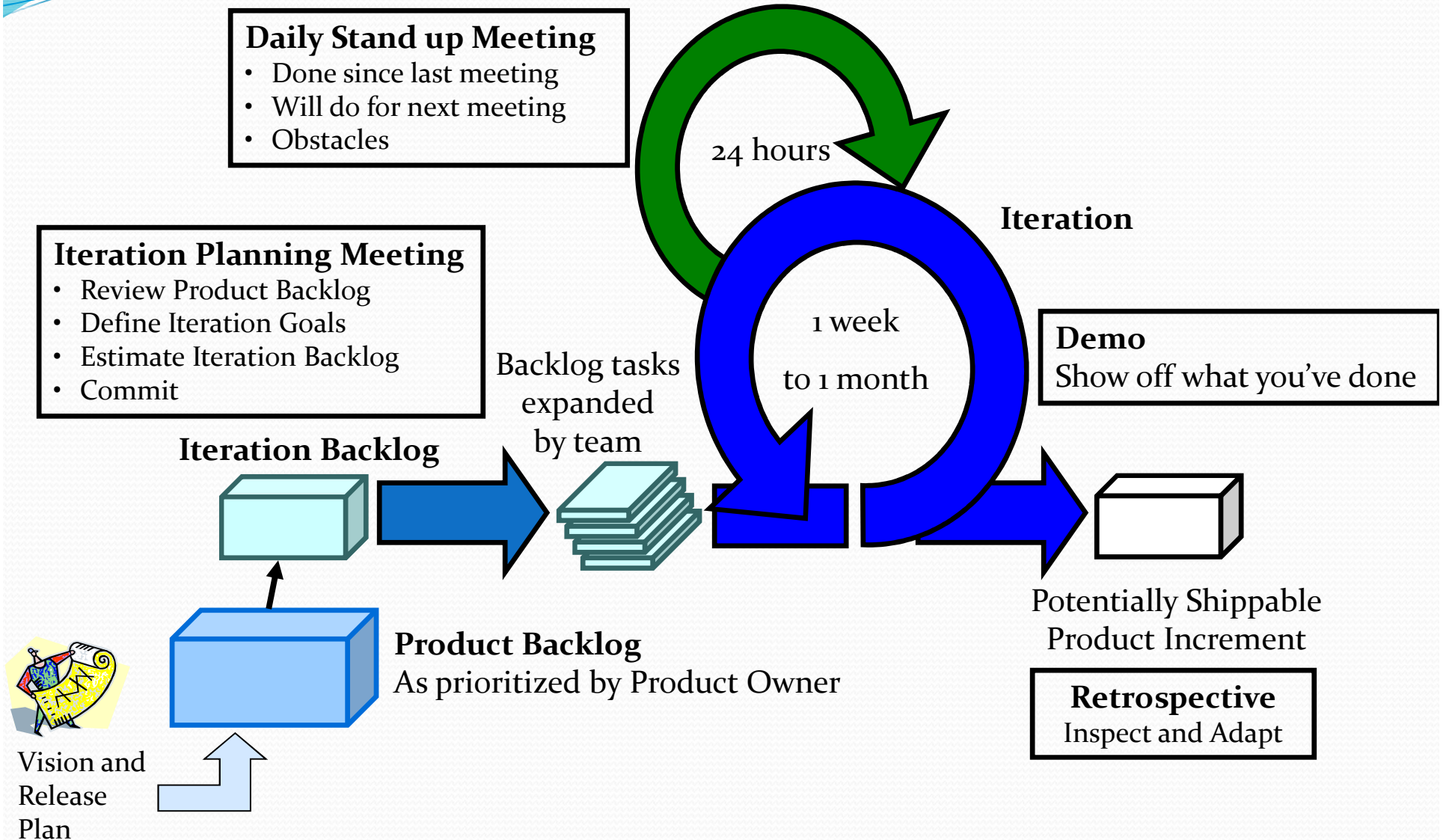


Prioritization Doesn't Stop

- The product owner re-prioritizes after each iteration
 - We've learned more about the business
 - Let's take advantage of that
- The further down the list something is, the less defined it will be and the less important it is to prioritize precisely



What Does an Iteration Look Like?



Iteration Planning

- Define scope as a team
- Define a clear understanding of “done”
- Plan just enough that you can commit



Before you Start



- Well Groomed Product Backlog
 - Prioritized
 - Estimated
- Iteration Theme/Goal

Name	Owner	Effort	Status	Public	Priority	User Priority	
User accepts license agreement	Walter Bodwell	1.5	In Progress	true	1	4	
User maps iterations to release	Walter Bodwell	2	Created	true	2	6,333	

Estimated

Prioritized

A Typical Iteration Planning Session

- Discuss Logistics
- Review iteration goals
- For each story (in Priority Order):
 - Understand it
 - Task it out
- Stop when “full” and commit

Typical Duration: 1-4 hours

Attendees:

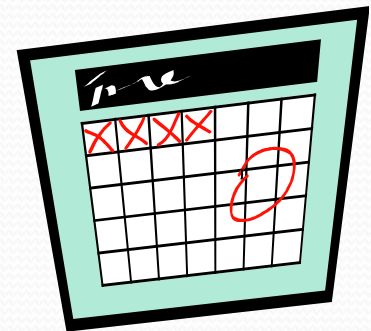
- Product owner
- Scrum master
- Delivery team

Materials:

- Stories (cards or online)
- Task planning material (cards, whiteboard, online)
- Planning/estimation materials (e.g. planning poker cards)

Discuss Logistics

- Review Historical Velocity
- Review Team Availability
 - Holidays / Vacations
 - Meetings
 - L3 Support, outside commitment, etc
- Review the Definition of Done



Review Iteration Goal(s)

- At a high level, what are we trying to accomplish this iteration
- Examples:
 - Improve reporting
 - Improve performance
 - Get ready for beta



Understand the Story

- Discuss the story
- Discuss why it is important
- Elaborate on acceptance criteria/tests
- Make priority adjustments
- Break down as needed



Task out the Story

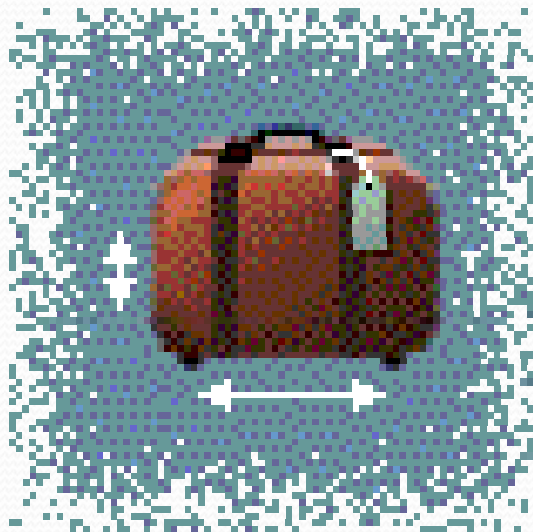
- Define tasks
- Estimate the work involved
- Double check ability to commit



The Product Owner can help
in avoiding less valuable work






Repeat

- Until the team cannot take on more
- Split stories as necessary



Commit



- Everyone agrees the iteration is doable
- Use disagreement and uneasiness in team members to drive out hidden risks, tasks, and issues
- Drive agreement with a fist of five
 -  • Absolutely, no question
 -  • I think this is good and will make it happen
 -  • I can support this
 -  • I'm uneasy about this and think we need to talk about it more
 -  • Let's continue discussing this idea in the parking lot

Managing your Tasks

Stories									
1.1	Iteration 4	Team A	All Owners	All Statuses	Refresh Log out Planigle				
Number of Stories: 5		Velocity Allocation: 24 of 19.33 (124%) - Team A		Utilization: 23 of 19 (121%) - Team A					
Name	Owner	Size	Time	Status	Public	Rank	User Rank		
- User searches for books by author, title or ISBN number	Sue Tester	8	7	In Progress	true	1	1		
Search by title showing just titles	Bob Developer	2		In Progress					
Add more details to results	Bob Developer	2		Not Started					
Add search by author or ISBN	Bob Developer	1		Not Started					
Test search	Sue Tester	2		Not Started					
+ User views detailed information on a book	Sue Tester	5	5	In Progress	true	2	4		
+ Administrator adds new books to site	Sue Tester	5	5	Not Started	true	6	5		
+ Administrator deletes book	Sue Tester	3	3	Not Started	true	7	6		
+ Administrator edits existing book info	Sue Tester	3	3	Not Started	true	8	7		

Tasks	Mon	Tues	Wed	Thurs	Fri
Code the user interface	8	4	8		
Code the middle tier	16	12	10	4	
Test the middle tier	8	16	16	11	8
Write online help	12				
Write the foo class	8	8	8	8	8
Add error logging			8	4	



Sprint 3 for MultiVue Install		IT Team Tasks		days remaining					
Task description	Commit	Status	256	32	0	0	0	0	0
Requirements Component									
Project Requirements Gathering	Campbell	Not started	16						
Formal Requirements Documentation	Campbell	Not started	8						
MultiVue Configuration Component									
Append Additional Demographics	Campbell	Not started	16						
SAP database Component									
Design SAP Database	Campbell	Not started	16						
Creation of the SAP Database	Campbell	Not started	4						
Create stored procedures on SAP database	Campbell	Not started	12						
SAP Code Component									
Creation of SAP .NET Component	Jan	Not started	16						
Creation of SAP Web Application	Jan	Not started	16						
SAP Security									
Creation of Security Administration Site	Campbell	Not started	24						
Secure Messaging	Campbell	Not started	12						
Security Integration	Jan	Not started	12						
SAP system testing	Campbell	Not started	8						
SAP System Verification	Campbell	Not started	8						
SAP Hardening									
Bug Fixing/ Cosmetic Changes	Mark	Not started	16						
Install in Live Environment	Campbell	Not started	16						
Biz Talk 2004 Component									
Extend ePEX-3 Adaptor	Campbell	Not started	4						
Extend Sw ift Adaptor	Mark	Not started	8						
Extend Upstream Schemas	Mark	Not started	8						
Create AIC schema	Campbell	Not started	4						
Create Mappings	Campbell	Not started	16						
Create SAP AIC	Mark	Not started	16						

Daily Standup

- What did I do yesterday?
- What will I do today?
- What's blocking me?

High Value

For the team

Quick

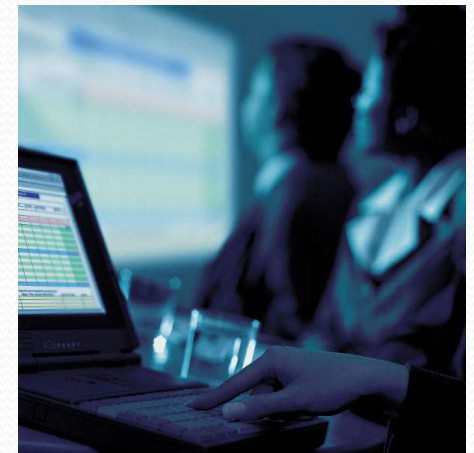
Parking Lot



Demo

- Show off what you got “done” in the iteration
- Should be from the user’s perspective
- No slides
- No code
- Just working software

If a customer could attend your demo,
you’re doing it right



Retrospective

- Review the process over the last iteration
- What went well?
- What went poorly?
- How can we do things better?
- Take the top 1-3 items and make sure you make progress on them in the next iteration



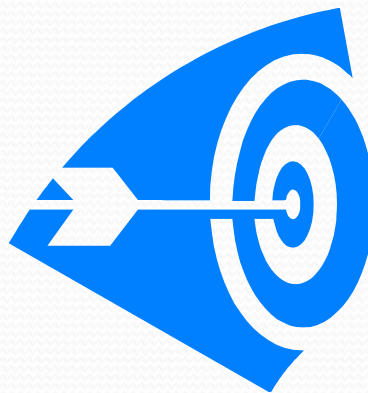
Improve

Estimating

- Identify a medium sized story that is well understood; call it a 5
- Now estimate other stories relative to that
- Is it about the same, $\frac{1}{2}$ as difficult, twice as difficult?
- Use Fibonacci numbers: 1, 2, 3, 5, 8, 13, 21
- If bigger than that or if too hard to estimate, split the story
- Tackle as a team; Planning poker can help (www.planningpoker.com)

Velocity

- Now that stories have sizes, you can track how many points you typically get done in an iteration
- Only count points for stories that get accepted in the iteration
- You can now use this to predict future completion rates



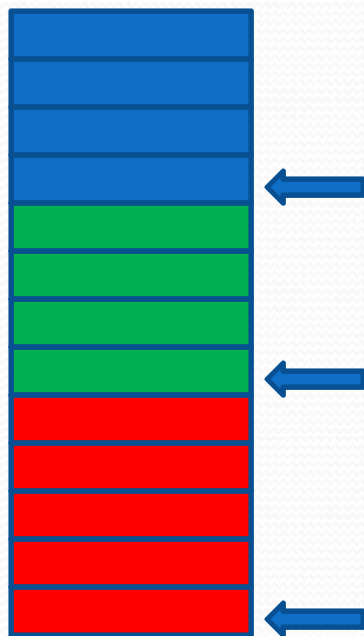
Structuring Teams

- It is preferable to have each team have the ability to complete its work by itself
- In other words, instead of a team per component, have teams with members who have knowledge of each component that will need to change to deliver something



Divvying Things Up

- Your goal is to divvy things up so that teams are working on items of around the same priority



Bad

If each team able to get 3 blocks in the release, the highlighted stories won't make it



Good

By better distributing stories amongst the teams, look which stories won't make it

Dependencies

- Approaches (go with the first one you can):
 - Structure the teams so that a single team can solve the problem end to end
 - Do the work within the same iteration
 - Implement the service and then use it
 - Stub out the service and implement it later
- Make sure the dependent teams (including external ones) are represented at release planning



Release Planning

- Kick off / Overview
- Break Out Sessions
- Review Results



Release Planning Deliverables

- Plan for each Iteration
- Assumptions
- Dependencies
- Risks



Release Planning Wrap Up

- Go through each iteration for each team
- Are things synched up across teams?
- Are you attacking the most important stories?
- Does the team believe in the results?



After The Meeting

- Capture the results in your tool of choice
- Update after each iteration



Anti-Goals of Release Planning

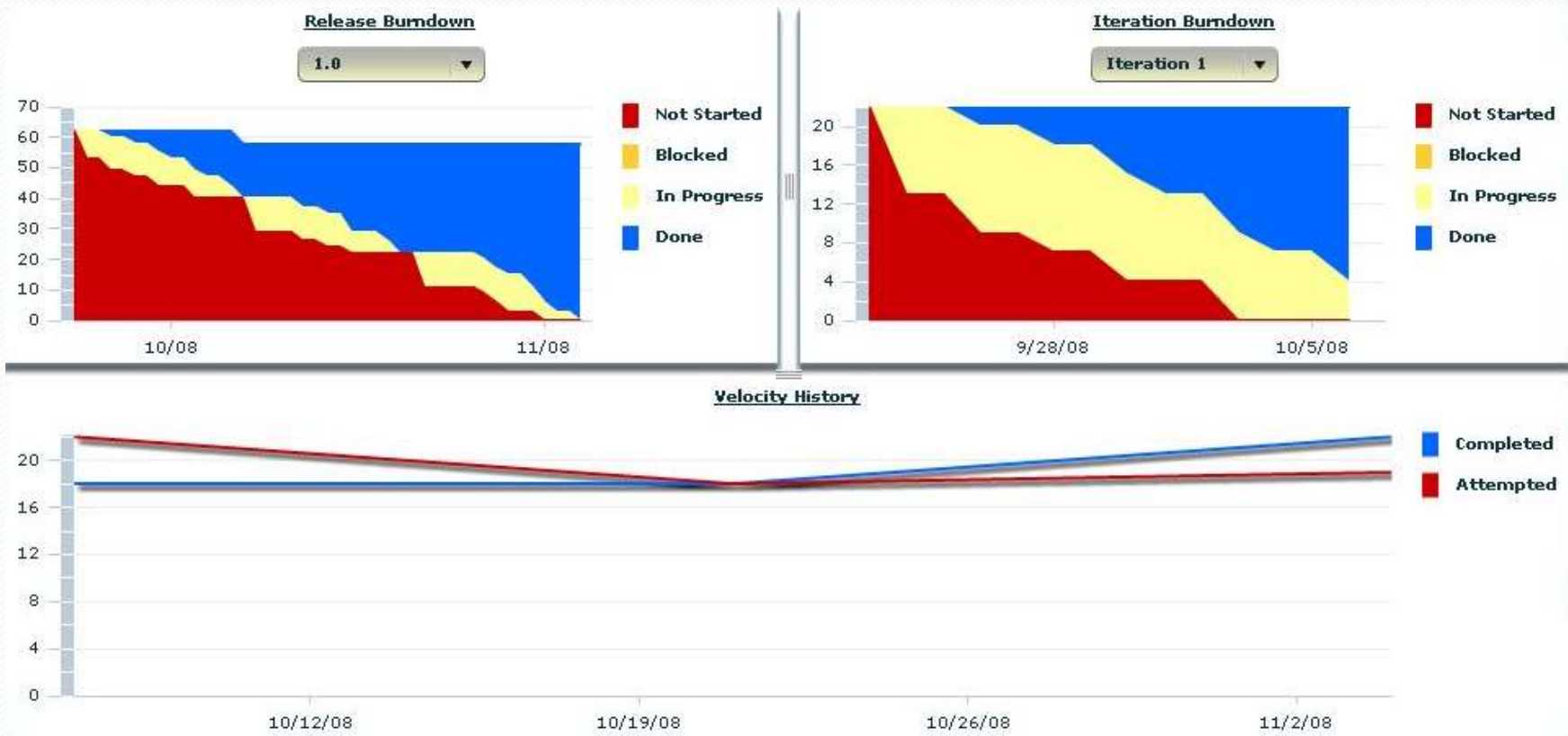


Release Planning is not a commitment!

Communicating the Future

- Themes give you room to be flexible
 - We know we're going to do something in this area
 - We'll decide as we go how much
- If a customer is asking about a particular feature, you can get into a discussion of priorities
 - Well, that's important, but we think this and this are more important, what do you think?
- Demos are a potential opportunity to get a customer involved
- Smaller, incremental releases generate feedback on what to dig into in more detail

Tracking the Release



Managing Risk

Waterfall

- Time, scope and resources “fixed”
- Changing one affects the others as well as quality
- Manage the plan
- Try to minimize change



Agile

- Time, resources and quality fixed
- Changing time or resources affects scope
- Manage the priorities
- Change as you learn more

Life in an Iteration

- Once in an iteration, scope is fixed
- Do the work in small increments
- Work closely with others
- It isn't done until it is really done
- If it doesn't add value, don't do it (or minimize)
- Leave decisions to the last responsible moment



It is a team effort

Self Organizing Teams

- The team members know how they can best contribute
- They figure out how to divvy the work up / attack the problem
- The scrum master facilitates and is part of the team

Feedback is key

- Do a little
- Get feedback
- Respond to feedback by doing a little more
- Automation helps decrease time to get feedback
 - Nightly/continuous build
 - Unit tests
 - Acceptance tests



Agile Documentation

- Keep to the minimal responsible amount of doc
- No more than you need at any point in time
 - Just enough to understand dependencies and mitigate risks
- Do you need this now?
 - If not, try to reduce or eliminate it
- Wiki's work well for collaborative design

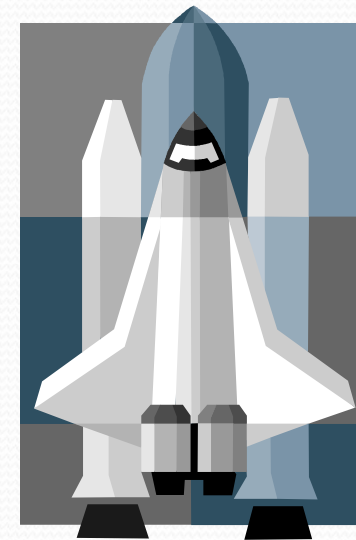


Management Is Not Enough!

- Engineering practices must change
 - Avoid specialization
 - Keep design simple and refactor as needed (YAGNI)
 - Create good automated regression tests
 - Integrate frequently
 - Peer review
- Consider
 - Test Driven Development (or Behavior Driven Development)
 - Pair Programming
 - Co-location
 - Dedicated team members

Staying Releasable

- Goal: Could release after any iteration
- Reality: Ability to do this will evolve over time
- Staying releasable gives you the ability to more easily change direction / take on new things
- It also tends to improve quality
- And predictability



Definition of Done

- You need to define for your environment
- Definition will evolve over time
- Example:
 - Unit tests written and passed
 - Acceptance tests automated and passed
 - User facing documentation written
 - Checked in to the build



Questions?

Walter Bodwell
Planigle

wbodwell@planigle.com

Twitter: @wbodwell

www.planigle.com

www.walterbodwell.com

