Scrum

Introduction & Overview

Agenda

- Software is hard!
- Process - things to fix
- (Agile Processes)
- Scrum
- Conclusion

Software is hard!

- Wise words
- Software complexity
- Estimating
- Shipping / Deadlines

Software is hard: Wise words

- “Software Engineering” is a statement of hope, not fact.
- Rosenberg’s Law:
  “Software is easy to make, except when you want it to do something new. The only software worth making is software that does something new.”

Software is hard: Complexity

- Requirements
- Technology
- People

Complexity: Requirements

- Usually many stake-holders
- ...with different needs
- ...that change frequently
- ...and are difficult to articulate

Complexity: Requirements

- If you're doing top-down design, you produce a specification that stops at some level of granularity.
- The finer the granularity becomes,the more like code your specification becomes.
- The complexity of software is an essential property, not accidental.
  Hence, descriptions of software that abstract away complexity often abstract away its essence.

Complexity: Technology

- Rarely simple
- Often unreliable
- More than one piece at once

Complexity: People

- Skills
- Intelligence
- Viewpoints
- Attitudes
- Prejudices
- Daily mood

Estimating: Wise words

- It is impossible, by examining a piece of completed code, to determine, within a factor of 2, how many man-hours it took to write.
- If you can't tell how long finished software took to write, what's the chance of estimating it before writing the first line?

Shipping / Deadlines

- Constraints are key to building a great product - they're what makes creativity happen.
- If we had all the time and money in the world to build whatever we want, it would probably never be released.
- Deadlines are not the problem – what you do to meet them often is.

Yet another process?!?

- What's wrong with current processes?
- Defined vs. Empirical process control
- Agile processes
- Scrum

Process Problems?

- Project bidding is the source of the waterfall.
- Nebulous requirements and guesswork estimates form basis of contract negotiations.
- *Product owner* used to no further involvement afterwards.

Process Problems?

- Waterfall specs: “Tell us everything now, or else...”
  - Can lead to a lot of unused functionality.
  - Poor or missing communication during project.
  - Often gives rise to blame-shifting.

Process Problems?

- Project management believes it works to tell developers to work harder
  - Developers sustain the belief by cutting corners (typically quality).
  - Results in “unmaintainable” code after 3-5 years.

Process theory: Defined processes

- Well-defined
- Repeatable
- Predictable outcome

Process theory: Empirical processes

- VISIBILITY
  - everything relevant
  - truthfully
- INSPECTION
  - regularly
- ADAPTATION
  - to ensure goal is reached

Agile Processes

(what Agilists do)

- Talk more, write less
  (but write if you have to)
- Show software to users
- Acknowledge that requirements emerge
  (and all that this implies!)
- Progressive refinement of understanding
- Work on what gives highest ROI

Agile Manifesto

Individuals and Interactions	Processes and Tools
Usable Software	Comprehensive Documentation
Customer Collaboration	Contract Negotiation
Responding to change	Following a plan

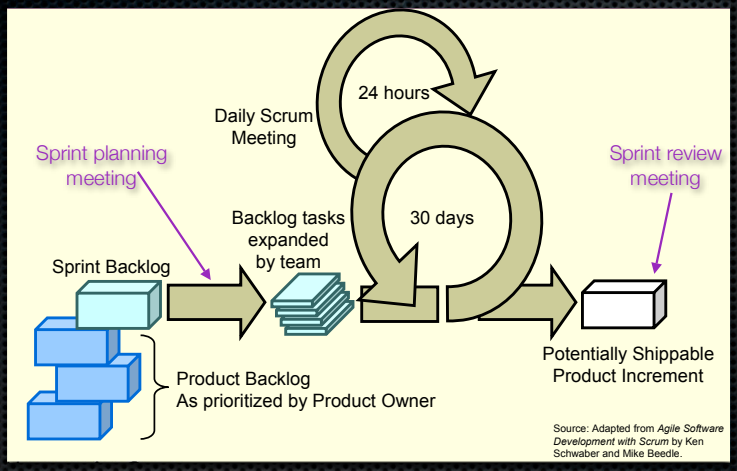
While there is value in the items on the right,
we value the items on the left more.

Scrum Process

- Support the way software development is *actually* done.
- Practices may look simple and unsophisticated, but...
- ...the dynamics they control and create have profound implications.

Scrum Process

- Principles
- Artifacts
- Roles
- Practice



Scrum Overview

Scrum Principles

- **Common Sense** (dealing with reality, not abstractions of it)
- The Art of the Possible (not the repeatable)
- Pigs & Chickens
- Self-organization and self-management
- Emergence (of requirements, technology, team capability)
- Time-boxing and visibility
- Inspection / Adaptation Cycle

Scrum Roles

- Product Owner
- Scrum Master
- Team

Role: Product Owner

- One person, empowered to make decisions for all customers and users.
- Develops and maintains Product Backlog
  - Prioritize backlog entries
  - Presents and Explains it to team
- Main responsibility is knowing what to build and in what sequence - manage the vision, ROI, and releases.
  - Cost, Date, Quality, and Functionality

Role: Scrum Master

- Represents Management to the Team.
- Typically filled by a Project Manager or a Team Leader.
- Responsible for:
  - Removing impediments to the team's progress.
  - Enforcing Scrum values and practices.

Role: Team

- Typically 7 ± 2 people.
- Cross-functional
- Members should be full-time
- Self-organizing
- Membership can change, but only between Sprints

Scrum Artifacts: Product Backlog *(what)*

- All functional & non-functional requirements.
- Managed *only* by Product Owner.
- Sorted in order of priority by Product Owner.
- Effort estimation (days)
- Originates from many sources.
- Visible to everyone.

Scrum Artifacts: Sprint Backlog *(how)*

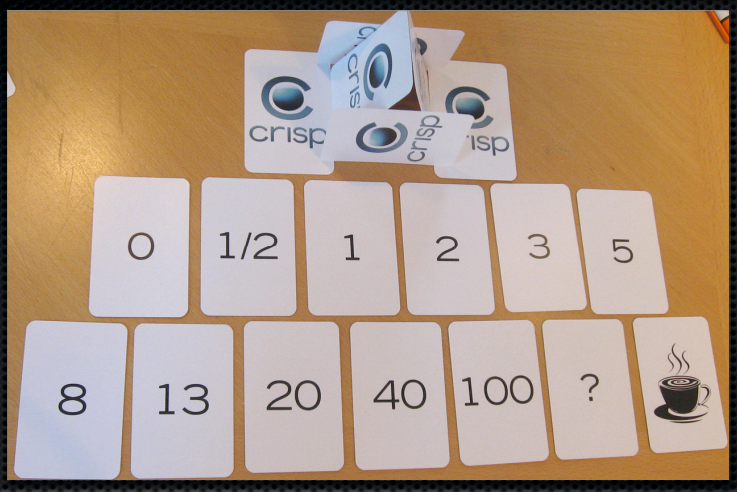
- Created together with Product Owner
- Based on highest priority Product Backlog items
- Each item broken down to list of tasks to perform
- Task effort estimation (4-16 hours)
- Valid only for *one* sprint.

Sprint

- Scrum projects make progress in a series of "sprints".
- Regular duration (typically 30 days).
- Design, coding + review, QA/testing, integration, documentation, ... until DONE
- Potentially releasable product after each sprint.
- New business functionality mandatory.

Sprint planning meeting

- Define a top priority goal / theme for the sprint.
- Achievable subset of product backlog becomes sprint backlog.
- Team estimates how long it takes to implement top priority backlog items
- Team decides which tasks are necessary to achieve the goal.



Estimation: Poker planning

Sprint: Commitment!

- No changes from Product Owner allowed during the sprint.
- Plan sprint durations around how long you can commit to keeping change out of the sprint.

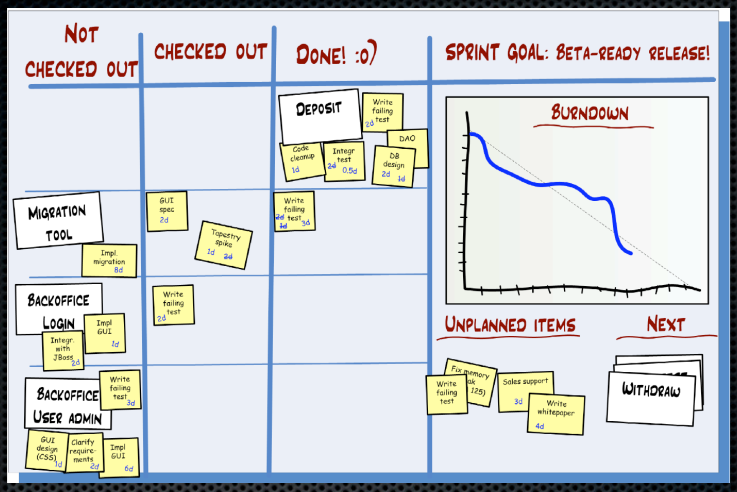
During the Sprint

- Daily scrums
- Maintain sprint backlog
- Make progress (or lack of it) visible, e.g. using a Burndown chart.

Daily scrum

- Daily meeting of 15 minutes.
- Three questions for each team member:
  1. What did you do yesterday?
  2. What will you do today?
  3. What obstacles are in your way.
- Chickens and pigs are invited, but only pigs can talk.
- Same place, same time every day.
- Not for problem solving.

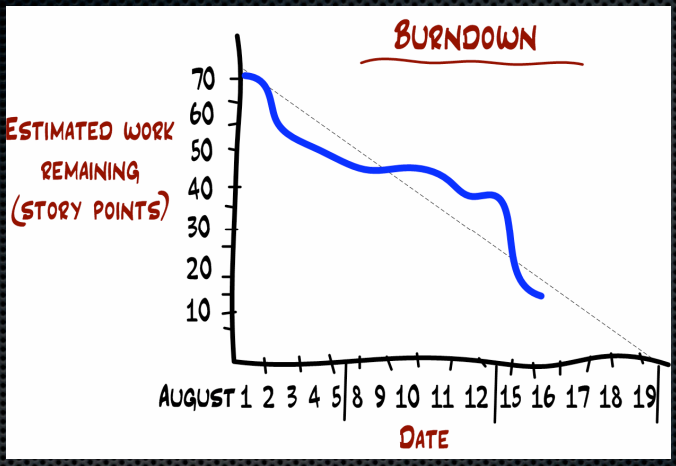
Daily scrum



Visibility: Taskboard

Maintain sprint backlog

- Team members sign up for and perform tasks (they are *not* assigned).
- ONLY the Team can:
  - Add, remove, and change tasks as needed to reach the sprint goal.
  - Adjust estimates of work remaining for tasks.
- Purpose is to make progress visible.



Burn-down chart

Sprint review meeting

- Team presents what it has accomplished.
- What went well, what can be improved (retrospective)
- Participants:
  - Product Owner, other stakeholders
  - Team and Scrum Master
  - Others that have interest...
- DEMO of new features.

Scrum Roles - Summary

Role	Team	Scrum Master	Product Owner
Responsi-bility	Self-organization, Self-management	Project success, Maintain quality	Project vision, Return of Investment
Authority	What tasks to do, How to do it	Ensure that Scrum rules are followed	Product Backlog items and prioritization
Job	Create releasable code	Facilitate work, Coaching, Remove hurdles	Funding, Release plan

Conclusion

- Support the way software development is *actually* done.
- Simple controls, powerful dynamics.
- Facilitate acceptance by providing a clear and simple interface to the rest of the organization.
- Improve creativity, empowerment, productivity, and the lives in general of the development team

Conclusion

- Project bidding should incorporate and acknowledge the use of Scrum up front.
- Scrum trades one traditional, long-term, high-risk project with many short low-risk sprints.
- Manageable risk and complexity.
- Meet deadlines by continuous adjustments to and re-prioritization of requirements.
- Improved long-term release planning (team velocity)

Other benefits

- Wraps existing engineering practices (XP, etc.)
- CMM Level 3 and ISO 9001 compliant
- It scales to multiple teams
- Integrate new employees

Scrum - the real thing