Scrum

The Art of Doing Twice the Work in Half the Time

479,107 (7%) open Scrum jobs in the United States - 3,715 in Denmark

Who We Are

Scrum Inc. is the Agile leadership company of Dr. Jeff Sutherland, co-creator of Scrum. We are based at the MIT Cambridge Innovation Center, MA.

CEO Jeff Sutherland helps companies achieve the full benefits of Scrum leading our comprehensive suite of support services and leadership training:
• Scaling the methodology to an ever-expanding set of industries, processes and business challenges Training (Scrum Master, Product Owner, Agile Leadership, online courses, etc.)
• Consulting (linking Scrum and business strategy, customizing Scrum)
• Coaching (hands-on support to Scrum teams)

Chie f Content Officer JJ Sutherland maintains the Scrum framework by:
• Capturing and codifying evolving best practices (Scrum Guide)
• Conducting original research on organizational behavior
• Publishing (3 books) and productizing ScrumLab

Principal Hardware Engineer Joe Justice leads our hardware consulting practice:
• Worldwide consulting at leading hardware companies
• 700-800% performance improvement in hardware development
• Builds 100 mpg cars in his garage with help from 500 people in 32 countries

We run our company using Scrum as the primary management framework, making us a living laboratory on the cutting edge of “Enterprise Scrum”

Find out more at www.scruminc.com.
As a group we need **Introductions** in order to work together effectively
Group Introductions

- Who’s in your group?
- Pair introductions
- Talk to each other and line up across the room by level of Scrum experience
- Line up in a second dimension by job function
- What companies, industries, non-software application are represented?
Self-Organize Teams

- Based on line exercise, divide up into cross-functional teams.

- Then:
  - Select a team name
  - Select a Vision Holder
  - Select a Servant Leader
  - Create a learning backlog – what do you hope to get out of the class individually and as a team
Team Name - Learning Backlog

Do

Doing

Done
“Release Plan” for Our Two Days

Day 1
- Introduction & Teams
- Scrum Origins
- Airplane Exercise
- Agile Manifesto
- The Scrum Framework

Day 2
- Project Leader
- User Stories
- Create Stories
- Failed Scrum I
- Estimate Stories
- Failed Scrum II
- Release Planning

Day 2
- Portfolio Planning
- Sprint Planning
- Daily Scrum
- Scrum Board
- Patterns
- Swarming
- Interrupts
- Clean Code

Day 4
- Done
- Sprint Review
- Scrumming the Scrum
- Scaling
- Team Backlogs
- XP Game
- Course Wrap-up & Retro
As a Scrum Master I need to understand the Why of Scrum in order to get the benefits of Scrum
Disruptive Leadership with Scrum

Leadership
Lean
Cross Functional Teams
Continuous Improvement
Agility
Delighting the Customer
Innovation
Scaling
Leadership

EMC
Ericsson
Workday
Cisco
Salesforce
Walmart
Visa
Macy’s
Stubhub
Symantec
Adobe
Intuit
Twitter
Paypal
Version One
Lean IT
BBC
Deloitte
Google
UK Cabinet
Sky News
GOTO
## Why You Hate Work

By TONY SCHWARTZ and CHRISTINE PORATH MAY 30, 2014

### White-Collar Salt Mine

A 2013 survey of 12,115 workers worldwide found that many lacked a fulfilling workplace.

<table>
<thead>
<tr>
<th>DO NOT HAVE THIS AT WORK</th>
<th>70%</th>
<th>DO HAVE THIS</th>
</tr>
</thead>
<tbody>
<tr>
<td>66</td>
<td>Regular time for creative or strategic thinking</td>
<td>21</td>
</tr>
<tr>
<td>60</td>
<td>Ability to focus on one thing at a time</td>
<td>31</td>
</tr>
<tr>
<td>50</td>
<td>Opportunities to do what is most enjoyed</td>
<td>33</td>
</tr>
<tr>
<td>50</td>
<td>Level of meaning and significance</td>
<td>36</td>
</tr>
<tr>
<td>49</td>
<td>Connection to your company's mission</td>
<td>25</td>
</tr>
<tr>
<td>48</td>
<td>A sense of community</td>
<td>35</td>
</tr>
<tr>
<td>47</td>
<td>Opportunities for learning and growth</td>
<td>38</td>
</tr>
<tr>
<td>46</td>
<td>Opportunities to do what you do best</td>
<td>36</td>
</tr>
<tr>
<td>45</td>
<td>Ability to prioritize your tasks</td>
<td>36</td>
</tr>
<tr>
<td>43</td>
<td>Overall positive energy</td>
<td>36</td>
</tr>
<tr>
<td>40</td>
<td>Understanding of how to be successful</td>
<td>40</td>
</tr>
<tr>
<td>40</td>
<td>Ability to balance work and home life</td>
<td>37</td>
</tr>
<tr>
<td>40</td>
<td>Ability to disengage from work</td>
<td>42</td>
</tr>
<tr>
<td>40</td>
<td>Comfort in truly being yourself</td>
<td>45</td>
</tr>
</tbody>
</table>

Source: The Energy Project

94 percent of those surveyed were in white-collar jobs; 6 percent were in blue-collar jobs.
People Can’t Get Things Done!

The best-laid schemes o' mice an' men
Gang aft agley,
An' lea'e us nought but grief an' pain,
For promis'd joy!

Robert Burns
Twice as much sweat in half the time
Make Work Visible
The shadows are lengthening for me. The twilight is here... But I want you to know that when I cross the river, my last conscious thoughts will be of the Corps, and the Corps, and the Corps.
Plans are Worthless, Planning is Everything!
Plans are Worthless, Planning is Everything!

U.S. Air Force Academy
Cellular Architecture: Programming the Human Biocomputer
Project Architecture: 84% Failure Rate

Death March at the Bank
Landing the Project

YouTube: RF-4E of the Hellenic Air Force
Make Work Visible!
Bootstrapping

ACCION International

Nonprofit organization

Accion International is a global nonprofit organization that supports microfinance institutions in their work to provide financial services to low-income clients. [Wikipedia]

Founder: Joseph Blatchford

People also search for

- Women's World Banking
- FINCA International
- Grameen Foundation
- Freedom from Hunger

Grameen Bank

Microfinance company

The Grameen Bank is a Nobel Peace Prize-winning microfinance organization and community development bank founded in Bangladesh. It makes small loans to the impoverished without requiring collateral. [Wikipedia]

Founder: Muhammad Yunus
Founded: 1983

People also search for

- Grameen Foundation
- Grameen America
- BRAC
Organizational Architecture: 3 Constraints

- Conway's law

  - organizations which design systems ... are constrained to produce designs which are copies of the communication structures of these organizations

- Brook’s Law

  - adding people to a late project makes it later

- Drucker’s “Cuckoo Effect”

  - any innovation in a corporation will stimulate the corporate immune system to create antibodies that destroy it
If there was a Nobel Prize for management, and if there was any justice in the world, I believe that the prize would be awarded, among others, to Jeff Sutherland, Ken Schwaber and Mike Cohn for their contributions to the invention of Scrum.

Steve Denning, Forbes 29 Apr 2011
Takeuchi and Nonaka

The New New Product Development Game

Harvard Business Review, Jan 1986

Type A – Isolated cycles of work

NASA

Type B – Overlapping work

Fuji Xerox

Scrum Project Management

Type C – All at once

Honda, Canon, 3M, …
Characteristics of Great Teams

• Transcendent Goals
• Autonomy
• Cross-Fertilization
Agile Leadership

- Sales, Marketing, Finance
- Manufacturing
- Technology, Software
- Families and weddings
- Education
- Agriculture
- Government
- Space
Changing the World

Higher grades
Faster learning
More fun
Leadership Development
Involves handicapped
Motivated students
Self-discipline

eduscrum.com
Scrum is a Productivity Superweapon - It is Shockingly Efficient

Rick Horgan, Sr. Editor, Crown Business
How to Play the Game

- **Goal:** See how good your team can get at making many airplanes
  - Each airplane must be made from $\frac{1}{4}$ of a sheet of Letter/A4-sized paper
  - Each team member may only do 1 “fold” of the paper at a time. You must then pass the airplane to another team member to do the next fold.
  - Planes must have a blunt tip (so no injury if hit in the eye)
- **Each airplane must tested and shown to fly 3 meters in the testing area.**
  - Planes may only be tested once; if it fails, it must be discarded.
  - Only successfully tested planes count towards your goal.
  - Work in progress (partially folded airplanes) must be discarded at the end of each Sprint.
- Teams are responsible for self-organizing, and deciding among themselves how to manage the work, assign roles, etc.
  - Teams are not in competition with each other – only with themselves.
One Person, One Fold
Product Owner Tests
World Record = 35

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>10 November 2019</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1/6</td>
<td>9/10</td>
<td>2/3</td>
<td></td>
</tr>
<tr>
<td>New Zealand</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Awesome</td>
<td>3/2</td>
<td>12/14</td>
<td>0/1</td>
<td></td>
</tr>
<tr>
<td>Sparkle</td>
<td>6/2</td>
<td>8/1</td>
<td>15/10</td>
<td></td>
</tr>
<tr>
<td>Loch Ness</td>
<td>1/2</td>
<td>1/2</td>
<td>12/3</td>
<td></td>
</tr>
<tr>
<td>Transformers</td>
<td>0/3</td>
<td>12/15</td>
<td>25/4</td>
<td></td>
</tr>
<tr>
<td>Sixlions</td>
<td>0/10</td>
<td>20/1</td>
<td>36/1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>15/5</td>
<td>62/1</td>
<td>125/1</td>
<td></td>
</tr>
</tbody>
</table>

Note: Task Allocation chart

High WIP speed
As a Scrum Master understanding the Agile Manifesto will help me implement good Scrum
Agile competition goes beyond lean manufacturing by permitting the customer, jointly with the vendor or provider, to determine what the product will be.

For agile competitors, the ability to individualize products comes at little or no increase in manufacturing cost. It does, however, exact a cost: It requires major changes in organization, management philosophy, and operations.

Managers need training to lead Agile teams.
Agile is the Future

Agile Manifesto

- Respond to Change
- Working Software
- Delight the Customer
- Great Teams
The Four Horseman of the Apocalypse

- Old Way of Thinking
- Organizational Debt
- Technical Debt
- Not Ready
  - Not Done

Impediments
Nokia’s mobile brand is officially dead. After Microsoft bought the company, it took only months for it to decide that using the Nokia brand meant little and it could walk away from it without offending too many customers or worrying about losing market share. It was an historic moment, but it was a necessary one in Microsoft’s mind, and it was perhaps an end of era in the industry.
Management becomes Leadership

- Strategic Vision
- Customer First
- Support the Teams
- Shipping Product

Leadership
Customers Often Don’t Know What They Want Until They See It! *Humphrey’s Law*

Customers loved this...

...Until they tried this...
Agile Values Scale

Scrum Values

Focus  Respect  Courage  Commitment
Scrum Values

• Focus
  “When you are not practicing, remember, someone somewhere is practicing and when you meet him he will win.” Ed Macauley

• Commitment
  Kung Fu means “hard training married man” who's commitment is for a lifetime

• Courage
  “Creativity takes courage.” Henri Matisse

• Openness
  “When a good idea comes, part of my job is to move it around, just see what different people think, get people talking about it ..” Steve Jobs

• Respect
  “Steve Job’s reputation for abrasiveness tarnishes his legacy... and surely this had a derogatory effect on the morale and even the performance of Apple.” Moses Ma
Market Capitalization

- Apple – $660 billion
- Google – $362 billion
- Facebook – $222 billion
- IBM – $155 billion
- GM – $54 billion

Whereas firms with a vertical mindset like IBM, are struggling with declining revenues and bloody cost-cutting reorganizations, firms in the horizontal world of Agile, like Apple and Google, are busy growing and inventing the future.

*Stephen Denning 26 Jan 2015*
Applicability of Scrum
Ogunnaike and Ray’s Process Control Requirements

Adapted from Snowden’s Cynefin Framework - see Wikipedia
Defined Process

- Traditional waterfall development is a “defined process.” A plan is defined at the beginning and precisely followed to the end.
- This assembly line approach requires minimizing deviations from plan to be successful.
- On average 65% of requirements change during software development causing waterfall projects to have an 14% worldwide success rate during the past decade. (Jim Johnson, Standish Group, 2011)
Empirical Process

- Controlling a process that has many unexpected changes requires introducing a feedback loop in order to inspect and adapt.
- Product is built iteratively and incrementally where each set of features is fully operational after a short cycle. Results are inspected and changes are made in repeated cycles as work progresses.
- Inspecting and adapting require full transparency of the work process to be successful.
- During the past decade, the worldwide success rate of software projects developed with empirical processes has been triple the success rate of defined projects.

Empirical plan with a new input after each cycle
As a Servant Leader I need to understand the Scrum Framework in order to implement Scrum
Scrum has Three Roles

- **Vision Holder**
  - Define and prioritize the features of the Backlog
  - Decide on release date and content
  - Responsible for the profitability of the product (ROI)

- **Servant Leader**
  - Facilitates the Scrum process and Team self-organization
  - Removes obstacles and shields the team from interference
  - Responsible for improving performance of the team

- **Team**
  - cross-functional (incl. testing)
  - self-organizing/-managing group of individuals, has autonomy regarding how to achieve its commitments
  - typically 3-9 people
Scrum has Four Meetings

• **Sprint Planning**
  • Vision Holder presents READY backlog to Servant Leader and Team
  • Deliverable is Sprint Backlog

• **Daily Scrum**
  • Team self-organizes to improve performance
  • Deliverable is new daily plan for implementation and impediment removal

• **Sprint Review**
  • Team presents backlog that is DONE to Vision Holder and Stakeholders
  • Deliverable is velocity (what Vision Holder confirms is DONE), feedback (used to update Backlog), and potentially shippable Increment of Work

• **Retrospective**
  • Servant Leader and team identify the top process improvement
  • Deliverable is the KAIZEN to put at top of Sprint Backlog for next Sprint.
Scrum Makes Work Visible

- Scrum Artifacts
  - Product Backlog
  - Sprint Backlog
  - Amount of work remaining in Sprint

- Useful tools
  - Scrum Board
  - Burndown Chart
    - Show work remaining
    - Helps calibrate velocity
Sprint - Iterative and Incremental
The Scrum Framework

Welcome to ScrumLab Open

How Scrum Works

- **Product Owner**
- **SM - ScrumMaster**
- **Team**
- **Customer**

**Input from End-Users, Customers, Team and Other Stakeholders**

**Sprint Review**

**Sprint Retrospective**

**Daily Standup**

24 hrs

**24 hrs**

**Sprint 1-4 Weeks**

**Back to ScrumLab**
As a Scrum Master I need to understand Roles & Responsibilities in order to implement Scrum
Scrum - a Self-Organizing Team

3 Social Objects
- Scrum Board
- Points
- Velocity
- Burndown Chart

5 Ceremonies
- Sprint Planning
- Daily Scrum
- Backlog Refinement
- Sprint Review
- Sprint Retrospective

3 Roles
- Product Owner
- Scrum Master
- Team

Product Backlog
- Sprint Planning
- Sprint Backlog

Make Work Visible
- Daily Scrum
- Sprint Replan

Sprint Refinement
- Ready Backlog
- Estimates

Product Increment
- Done/Velocity/Feedback
- Kaizen
- Acceptance Tests

Delivers Revenue
Delivers Velocity and Happiness
Delivers Product with Quality

Delivers Revenue
Delivers Velocity
Delivers Product with Quality

Delivers Revenue
Delivers Velocity
Delivers Product with Quality
Exercise: Project Manager/Leader

- As a team, write down all responsibilities of a traditional project manager/leader
- Put one responsibility on each sticky note
- Time: 4 minutes
How a Traditional Project Manager Transforms to Scrum

Jeff Sutherland, Ph.D., Nafis Ahmad, PMP

Abstract—Transitioning from a traditional project manager to Scrum is challenging. The PMI Project Manager manages the project by ensuring that intermediate deliverables are delivered at various stages of the project. Agile development emphasizes the need for producing tangible results as soon as possible and as often as possible. The resulting role of an Agile project manager is fundamentally different from a PMI Project Manager. We have provided a mapping between the PMI responsibilities to Scrum and show a project manager how to more easily make the transition to an agile practice.

Keywords: PMI; PMBOK, Project Manager; Scrum; Agile

II. TRADITIONAL PROJECT MANAGEMENT (TPM)

A traditional project typically has the following characteristics:

- Phased: TPM is divided into distinct phases of homogeneous activities like requirements, design, implementation, verification and maintenance phases. A “technical transfer” or handoff, transitions the project from one phase to another.
- Sequential: The phases of the project are typically sequential where one phase starts when the previous phases are completed and ‘perfected’.
Exercise: Project Leader

- Sort Project Leader responsibilities on sticky notes into these five categories
  - Product Owner
  - Scrum Master
  - Team
  - Waste
  - Other

- Time: 5 minutes
Scrum Master Role & Responsibilities

- Facilitator
- Knowledgeable about the Scrum process
- Coach – Team and PO to enhance team performance
- Removes impediments
- Protects the team from interruptions
- Holds Scrum values and practices
The Scrum Master Owns The Process

- Scrum is a simple framework that requires consistent discipline

- Scrum Master owns the process
  - Facilitates Daily Stand-Up
  - Facilitates Sprint Planning
  - Facilitates Retrospective
  - Protects the team
  - Removes impediments
Product Owner Owns The **WHAT**

- Have a compelling product vision that is executable, generates lots of cash, and arouses passion in the team, company, and customers.

- Build a roadmap for rolling out the vision that everyone can see and sign up for.

- Build a Product Backlog of “enabling specifications” that are “just enough, and just in time.”

- Spend half the time with customers, sales, and marketing.

- Spend the other half working closely with the team clarifying specifications.
A Successful Product Owner...

• Delivers:
  • The right product set to excite customers
  • At the right time
  • In the order that maximizes business value

• Responds dynamically to change faster than competitors
• Clarifies customer need to development teams so that uncertainty is removed and developer velocity is maximized

• *The Product Owner is ultimately accountable for winning in the market!*
Scrum Master

- Works with the Product Owner to:
  - Find techniques for effective Product Backlog management;
  - Clearly communicate vision, goals, and Product Backlog items to the Team;
  - Teach the Scrum Team to create clear and concise Product Backlog items;
  - Understand long-term product planning in a Scrum environment;
  - Understand and implement the values of the Agile Manifesto; and,
  - Facilitate Scrum events such as release planning
Basic Truths about Teams

- **Team motivation**
  - People are most productive when they manage themselves.
  - People take their commitment more seriously than other people’s commitment for them.
  - People do the best they can.
  - Under pressure to “work harder,” developers automatically and increasingly reduce quality.

- **Team performance**
  - Teams and people do their best work when they aren’t interrupted.
  - Teams improve most when they solve their own problems.
  - Broad-band, face-to-face communications is the most productive way for teams to work together.

- **Team composition**
  - Teams are more productive than the same number of individuals
  - Maximum effective team size is around 7-8 people.
  - Products are more robust when a team has all of the cross-functional skills focused on the work
  - Changes in team composition reduce productivity.

Source: Ken Schwaber
Teams are:

• Cross-functional - most members can do more than one thing
• Self-organizing - they decide **how** they will work
• Self-managing - they decide **how much** work they can do in a Sprint
• Collaborative - they work together to achieve the Sprint goal
• No more than 3 - 9 people
Leadership Responsibilities

- Provide challenging goals for the teams
- Create a business plan/organization that works
  - Eliminate organizational debt
  - Provide all resources the teams need
- Identify and remove impediments for the teams
  - Know velocity of teams
  - Remove waste - eliminate technical debt
- Hold Product Owners accountable for value delivered per point
- Hold Scrum Masters accountable for process improvement and team happiness
As a Scrum Master I need to understand How to Get Started With Scrum in order to be effective in my job.
Scrum is simple but not easy

- A systematic approach to implementing Scrum will give you the most rapid benefit for the least effort.
- The Shu Ha Ri concept from the martial arts can help.
Shu State - beginners mind

- Scrum Master sets up process as in the Scrum Guide. See scrumguides.org.
- Team has a known velocity at a sustainable pace.
- Retrospective is used to enhance performance.
- Review Core Scrum at agileatlas.org for Scrum Alliance version of Scrum Guide.
Ha State - at least 200% acceleration

- Team has software done with all features tested and no bugs at the end of a sprint.
- Product owner has ready backlog at beginning of the sprint (good user stories)
- Team has data showing velocity and quality has at least doubled.
- Team is positioned to work towards hyperproductivity, the design goal of Scrum
Ri State - at least 400% acceleration

- There are three proven methods for consistently moving teams into the Ri state:
- Process Intensive: Systematic’s approach to get Ready Ready to be Done Done
- At CMMI Level 5 Systematic has every team follow the same rigorous process
- Coach Intensive: Shock Therapy
  - A strong coach trains the team in the martial art of Scrum
- Team Intensive: Teams that Finish Early Acclerate Faster
  - Use of a Pattern Language makes improvement Fast, Easy, and Fun
Team of One in Richmond State

• Team needs to be hyperproductive.
• But what does a great Scrum Master do?

http://www.youtube.com/watch?v=Hzgzm5m7oU
As a Scrum Master I need to know
How to Play the Game
to enable a high performing team
The agile process is the universal remedy for software development project failure. Software applications developed through the agile process have three times the success rate of traditional waterfall method and a much lower percentage of time and cost overruns. The secret is the trial and error and delivery of the iterative process.
It’s all about technique ...
Mark your calendars for 24 May — 29 May, 2015, when we’ll come together at Quinta da Pacheca near Porto, for the seventh ScrumPloP.

- The ScrumPloP Mission
- Call for Participation: What are patterns and pattern languages all about, and what does one do at a PLoP?
- Submission Guidelines: How do I write a pattern?
- Logistics, Cost, Registration, etc.

Program
- Attendee roster
- About PLoPs: The community where patterns grow
- The venue is Quinta da Pacheca near Porto — a beautiful place in the middle of the port wine region of Portugal. It is about 75 minutes by car (1 hour 50 minutes by train) from Porto. The price per attendee will be about €1000, which includes food and lodging. (Our menu and wine selection will affect price.)
- Calendar

TO THE PATTERNS ➤

- Works in progress (authors working on drafts go here)
  - ScrumPloP Product Backlog

Jeff Sutherland @ Scrum, Inc:
A Pattern Language for Hyperproductivity

Jeff Sutherland, the inventor of Scrum, is a charter member of the Scrum Patterns group. He is the author of most of these Scrum PLoP® patterns — patterns he teaches to get teams off to a good start, and to get great teams to a hyperproductive state:

- How do you get started? (Stable Teams)
- How do you successfully pull backlog items into a Sprint? (Yesterday’s Weather)
- How do you do stuff done? (Swarming: One-Piece Continuous Flow)
- How do you deal with interruptions during the Sprint? (Illegitimus non Interruptus)
- How do get defect free at the end of the Sprint? (Daily Clean Code)
- How do you deal with surprises? (Emergency Procedures)
- How do you ensure you continuously improve? (Scrumming the Scrum)
- How do you get teams to have fun? (Happiness Metric)
- How do you do hyperproductive? (Teams that finish early accelerate faster)

Jeff presented these at Agile 2013.
As a Scrum Master I need to dedicate up to 10% of my team’s time to helping the Product Owner with Product Backlog Refinement in order to double my teams performance.

See ScrumPlop.org: Definition of Ready, Release Plan
Definition of Ready - Documented Velocity Improvement

• Systematic CMMI 5 data consistently shows 200% increase in velocity with Ready checklist.

• American Healthways, a multi-billion dollar company implemented a Definition of Ready for their help desk in went from 1000 tickets a week to 2000 tickets a week in two one week sprints. (Private communication, Mike Dwyer)

• Scrum Inc focused on Definition of Ready for three one week sprints and increased velocity by 300%.
Only Allow Ready Backlog into Sprint Planning

- Maximum Sprint Planning time is 2 hours per week of sprint
- Optimal Sprint Planning time is less than one hour per week of sprint
- Only Ready backlog can optimize sprint planning
- The Product Owner cannot get backlog Ready without help from the team
  - Only the team can estimate
  - The team needs to help the Product Owner break down large stories into small stories
Product Backlog

- The Product Backlog consists of work to be done ordered by business value
- Anyone can put anything on the backlog
- Product Owner is the final authority on ordering the backlog.
- The backlog consists of Product Backlog Items (PBIs)
- The majority of Scrum teams use User Stories as PBIs.
User Story

- A UserStory is a story, told by the user, specifying how the system is supposed to work, written on a card, and of a complexity permitting estimation of how long it will take to implement.
- The UserStory promises as much subsequent conversation as necessary to fill in the details of what is wanted.
- The cards themselves are used as tokens in the planning process after assessment of business value and [possibly] risk.
- The customer prioritizes the stories and schedules them for implementation. -- RonJeffries
User Story Templates

- As a <role> I would like to be able to <action> to achieve <business value>

As who, I want what so that why.
What’s Wrong with This Story?

As a captain I'd like the log function to automatically use today's star date.
Break Epics into Stories

**As a frequent flyer I want to**
book flights customized to my preferences, so I save time

**As a frequent flyer I want to**
book a trip using miles so that I can save money

**As a frequent flyer I want to**
easily book a trip I take often **So that I can save time**

**As a premium frequent flyer I want to**
request an upgrade **So I can be more comfortable**
User Story - Guidelines

- Immediately actionable
- Negotiable
- Valuable
- Estimable
- Sized to fit
- Testable

Modified from Bill Wake – www.xp123.com

- User Stories slice through all layers
- Customer facing value delivered every sprint
- Slices accumulate in value

Stuff not needed

GUI
Client
Server
DB
schema
One Company’s Definition of Ready
User Story, Acceptance Tests, Examples, Wire frame, Estimates

As a system user I would like to be able to generate a report that shows clinician's past, present and future scheduled patients that I can filter by clinician and/or patient.

The exact role has not yet been identified.

1. Report Name
2. Optionally Select Service Line, but not required.
3. It is required to select how you want this report filtered by Clinician and/or Patient
4. Select the start date; click the calendar icon and the calendar date range picker pops up as a modal. Must select a date range; a from and to date.
5. When the <Ok> button is clicked the report will generate and transition to it.
6. When the <Cancel> button is clicked the report will be canceled and nothing else happens.
7. When the <X> button is clicked the report will be canceled and nothing else happens.

All past schedule patient and/or clinician names are printed in black and all future schedules printed in, "Blue".

Examples:
Past Scheduled: Future Scheduled:
Clinician 1  Clinician 1
Patient 1  Patient 1
Patient 2  Patient 2
Patient 3  Patient 3
User Stories Improve Communication Effectiveness

Communication effectiveness

- Effective
- Ineffective

Richness ("temperature") of communication channel

- Cold
- Hot

- Document
- Audio recording
- Video recording
- 2 people on phone
- 2 people at whiteboard
- Google Hangout

Source: research from McCarthy and Monk (1994)
Irrelevant Information

Spec 1

A
B
C

20 hrs

Same spec
+ irrelevant details

A
B
C

39 hrs

Source: How to avoid impact from irrelevant and misleading info on your cost estimates, Simula research labs estimation seminar, Oslo, Norway, 2006
Enabling Specification

- User stories notes may be enough for a web site
- For a complex system you need enabling specification
  - Short - 3-5 pages for a feature
  - Usually a lightweight use case
  - Product Owner documents critical information in collaboration with team
    - User experience (design)
    - Business logic
    - Data structures
- Stories are derived from the enabling specification
- The enabling specification is a living document
  - Updated over time
  - Global picture of the feature
  - Allows traceability of stories back to product design
Use the Definition of Ready to Improve Team Performance
Create User Stories Exercise
Exercise Background

- General situation
  - Today is Jan 1.
  - The government has announced Outlook will be outlawed from April 1 to save money!
- We are greatsoftware.com
  - Our goal:
    - Create “bookmeeting.com” an online resource booking service targeted primarily towards those that use Outlook for booking meeting rooms.
    - March 1: Pilot customer will start using it.
    - April 1: Commercial rollout.
- Our context:
  - We have built similar services before, so we have a functioning team, development environment, and operational environment.
  - This project is top priority, we have a 5-person Scrum team ready to start today.

Source: Henrik Kniberg
Bookmeeting.com

- **Product Vision**
  - Welcome to bookmeeting.com! It doesn’t get any simpler than this.
  - Getting started guide:
    - Create an account at bookmeeting.com
    - Set up rooms
    - Your company can now surf to bookmeeting.com and book meetings!
  - Payment
    - Pay per month (rate may vary depending on number of rooms)

- **Roles**
  - **Booker**
    - Creates and administrates the meeting booking, invites participants
  - **Participant**
    - Attends meetings. Receives invitations & updates.
  - **Admin**
    - Sets up rooms and users.
  - **Buyer**
    - Buys the service and pays for its use.
  - **Seller**
    - Hosts the service and charges for its use. Initially greatsoftware.com.

Source: Henrik Kniberg
Quick Example of Building an Initial Product Backlog

Story map

Lachlan Heasman 2011
Exercise

• Create product backlog
  • Goal: Pile of users stories
  • Acceptance criteria for one story

• Time: 10 minutes

Write on sticky notes.
Use a thick pen.
Use the story template.

As a X
I want Y
so that Z

As a X
I want Y
so that Z

As a X
I want Y
so that Z

As a X
I want Y
so that Z

As a X
I want Y
so that Z

As a X
I want Y
so that Z
As a Scrum Master I need to know how to go from Waterfall to Scrum in order to do my job.
Case Study: Death March

This is a real story with some details changed to protect the guilty!
It demonstrates:
1. How to fail with Scrum
2. How to rescue a failed Scrum
3. How to convert a waterfall team into a Scrum team
Symptom: Waterfall process (under Scrum banner)

- Requirements: Q1, Q2, Q3, Q4 (2013)
- Coding: Q1, Q2 (2014)
- Testing?: Q1, Q2 (2014)
- Release: Q2 (2014)

We are here
Symptom: Long, detailed requirements specifications
Symptom: Lack of trust & commitment
Strategy: Implement Scrum

- Create product backlog
- Estimate product backlog
- Execute 2 sprints, measure velocity

• Show us where we stand
• Help us move faster

2014

Jan  Feb

We are here
Step 1: Change Definition of Done

- Old definition of done:
  - Code checked in
- New definition of done:
  - Releasable
  - Tester added to team
Step 2: Create a product backlog

Features left to implement

Features implemented but not tested & integrated
Software Estimation Error

- Effect of uncertainties over time

Relative Size Range

- Operational Concept
- Life Cycle Objectives
- Life Cycle Architecture
- Initial Operating Capability

Phases and Milestones

Feasibility - Plans/Rqts. - Design - Develop and Test

© USC-CSE
Points vs. Hours

- Rand Corporation received a grant from U.S. DOD in the 1940’s to determine best way to estimate tough projects
  - Discovered estimation in hours has high error rate and wide variance
  - Found people could put things in relative size piles best
  - Experts need to estimate independently - avoid anchoring
- Fibonacci growth pattern easiest for humans
  - Seen everywhere in nature
  - RAND called it the Delphi technique
Why Points are Better Than Hours

Cone of Uncertainty

Estimation Error

Iteration

Gray Line - Hours
Red Line - Points

Agile Estimating Strategy

• Don’t estimate time
  • Estimate relative size of stories
  • Measure velocity per sprint

• Estimates are done by the people who are going to do the work
  • Not by the people who want the work done
  • Team allocate 10% of sprint time to Product Owner

• Estimate continuously during the project, not all up front

• Prefer verbal communication over detailed, written specifications
The Fibonacci Sequence

- Barry Boehme called it the Wideband Delphi Technique for software

http://www.youtube.com/watch?v=ahXIMUkSXX0
As a Scrum Master I need to know how to Estimate Stories in order to know velocity and pull the right amount of work into a sprint.
<table>
<thead>
<tr>
<th>Items</th>
<th>Pieces</th>
<th>Time (hrs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item 1</td>
<td>106 pcs</td>
<td>0.3 hrs</td>
</tr>
<tr>
<td>Item 2</td>
<td>132 pcs</td>
<td>0.5 hrs</td>
</tr>
<tr>
<td>Item 3</td>
<td>286 pcs</td>
<td>2 hrs</td>
</tr>
<tr>
<td>Item 4</td>
<td>706 pcs</td>
<td>2 hrs</td>
</tr>
<tr>
<td>Item 5</td>
<td>1636 pcs</td>
<td>8 hrs</td>
</tr>
</tbody>
</table>

(1-4) (1-3) (4-24)
Case Study Continued
Step 3: Estimate product backlog

Features left to implement

Total: 180 points

Features implemented but not tested & integrated

Total: 70 points
Step 4: Execute 2 sprints

<table>
<thead>
<tr>
<th>Estimated Velocity</th>
<th>Actual Velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sprint 1</td>
<td>30</td>
</tr>
<tr>
<td>Sprint 2</td>
<td>25</td>
</tr>
</tbody>
</table>
Step 5: Face reality & Revise the plan

Backlog = 250 points
Velocity = 10 points/sprint

25 sprints ➔ > 1 year until release!

2013
Q1 ➔ Q2 ➔ Q3 ➔ Q4
Promised release

2014
Q1 ➔ Q2
Earliest likely release

We are here

Q1 Q2 Q3 Q4

Promised release ➔ Earliest likely release

2013 ➔ 2014

© 2006-2015 Scrum Inc.
Step 6: Act

Backlog = 250 points

Velocity = 10 points/sprint

Reduce total scope
Incremental releases

Fix impediments
- Pressure on team
- Ineffective build & test environment
- Lack of teamwork, discipline & motivation
- Disruptions & context switching
- Unrealistic expectations
- ROOT CAUSE: Company not focused

Overall priorities
1. Operations
2. Project X
3. Anything else
**Result**

2014

- Q1
- Q2
- Q3
- Q4

- Originally promised release (big-bang)

2015

- Q1
- Q2

- Earliest likely release if process hadn’t changed (big-bang)

**Velocity**

- 2014
  - Q1: 9-10
  - Q2: estimated 25-30
  - Q3: actual 25-30

- 2015
  - Q1: 25-30
  - Q2: 25-30
Case 3: Take-away points

- Waterfall is still waterfall even if you call it Scrum
  - Know your tools, get training & coaching early.
- Don’t believe your plan
  - There is no “the plan must be right because we promised”.
  - Make sure you have reliable feedback loops & a changeable plan.
- There is no “too low velocity”
  - Just actual velocity, and a realistic or unrealistic plan.
- Build quality in
  - Don’t postpone test & integration, that gives a false velocity.
- Having good people isn’t enough
  - An inappropriate process can cause even a great team to fail.
- Dramatic improvements can be made quickly
  - With a strong management team that has access to empirical data and is willing to focus.
As a Scrum Master I Need to Help the Product Owner do **Release Planning** to Deliver Working Product to End Users
Scrum Scales “Fractally” Rather than “Hierarchically”
First Scaled Scrum

IDX Systems 1996-2000 (now GE Healthcare)

- Managers self-organized company into teams
- Managers became leaders
  - Directors ran Scrum of Scrums
  - VPs became leaders of sites with multiple Scrum of Scrums
- Grew to over 600 developers in eight business units

- All products on maximum 3 month release cycle
- Whole company on 6 month release train

Hyperproductive Strategy

Audited Results
Product Backlog

Administrate users

View Invoice in HTML, PDF, or Excel format

As a helpdesk operator I want to see who is logged in

Operations manual

100 simultaneous users

Register new user

Edit existing user

Find user

Delete user

View Invoice in HTML, PDF, or Excel format

As a helpdesk operator I want to see who is logged in

Find user

Operations manual

100 simultaneous users

Delete user

Source: Henrik Kniberg
Splitting Stories and Breaking Out Tasks

Split stories

- Administrate users
- User admin: Register new user (5)
- User admin: Find user (3)
- User admin: Edit existing user (2)
- User admin: Delete user (8)

Break into tasks (normally during sprint planning meeting)

- Write failing test
- Create DB schema
- Write form validation
- Do GUI design
- Write server-side logic
- Do integration test

Source: Henrik Kniberg
Product Owner Responsibility

- Get one sprint READY backlog
  - Team can get started
- Get two sprints READY backlog
  - Team can accelerate sprint to sprint
- Build out Release Plan
  - Company can predict revenue
- Build one year roadmap
  - Customers can be briefed on company strategy
Release Cycles

- Goal: every sprint results in potentially releasable product increment.
- Product owner decides when to release.
Backlog Maintenance

- April 2008
- May 2008
- June 2008
- Q3 2008
- Q4 2008
- 2009
- 2010
- 2011
- 2012
Release Planning Prerequisites

- Vision for release
- Product backlog prioritized and estimated
- Historical data
  - Velocity of teams
  - Emerging requirements
  - Undone work
  - Customer feedback that must be dealt with immediately after release
- If historical data is not available, velocity, emerging requirements, and undone work must be estimated after the first few sprints
Release Planning Meeting Participants

- Includes stakeholders, Product Owner team, and all parties that execute the release
- Scrum Masters and cross functional teams
- Third party team teams (waterfall teams or external vendors)
- Sales, marketing, and operations
- Could be half a day for small release with Ready backlog
- Could be multiple days for large release or backlog that needs refinement
Release Planning Agenda

- Background, business and competitive climate (PO)
  - Release goals (PO)
  - Current product and development state (PO)
- Product Backlog refinement for release (All)
- Capture and discussion of issues (All)
- Technical Issues (Dev teams)
  - Technology
  - Testing Challenges and Strategies
  - Dependencies
  - Engineering Standards and Practices
  - Hardening and Hackathon
  - Development regimen
  - Build
  - Test
  - Continuous integration
- Team interactions (Dev teams)
  - Scrum of Scrums
  - Special handling of multi-team Sprint Planning, Sprint Reviews
  - Multi-team retrospectives
- Tentative schedule (PO)
Measuring Velocity

Beginning of sprint

Sprint Backlog

Estimated velocity = 26

End of sprint

Sprint Backlog

Actual velocity = 18

Product Backlog

8
5
5
3
5
5
3
5
8
5
5
3
5
8
5
3
5
Release Burndown Chart Key to On Time Project Delivery

- Answers the key question “Will we be done on time?”
- Useful for “what if?” analysis and managing tradeoffs of Scope, Velocity and Time
- Vital for identifying and addressing unreasonable expectations

Source: Henrik Kniberg
Critical Estimates for Release Date

- Product Backlog Estimates (based on Definition of Done)
- Undone Work (anything beyond DoD needed to deploy)
- Emerging Requirements (historical data)
- Customer Issues post release (historical data)
- Example:
  - For a healthcare company in Houston, for every 100 points estimated there are 20 points of undone work (User Acceptance Testing) plus 40 points of emerging requirements plus 60 points of customer feedback when new features go live for the first time.
  - Plan must include $100 + 20 + 40 + 60 = 220$ points for every 100 points of initial estimate
- **Release plan must be updated based on real data after every sprint**
SCRUM: the Art of Doing Twice the Work in Half the Time

Sébastien Chabal
Modular Framework for Scaling Scrum

Product Ownership Cycle
- Executive Action Team
- Backlog Prioritization
- Backlog Decomposition & Refinement
- Release Planning

Product & Release Feedback
- Scrum Master Cycle
- Continuous Improvement & Impediment Removal
- Cross-Team Coordination
- Release Management

Team-Level Process
- Team-Level Process

Organization Level
- Enterprise
- Business Unit
- Team

Delighting the Customer
- Metrics & Transparency

Strategic Vision
- Executive Action Team
- Backlog Prioritization
- Backlog Decomposition & Refinement
- Release Planning

Team-Systemic Coordination
- Team-Level Process
- Product & Release Feedback
- Continuous Improvement & Impediment Removal
- Cross-Team Coordination
- Release Management

Metrics & Transparency
- Product & Release Feedback
- Continuous Improvement & Impediment Removal
- Cross-Team Coordination
- Release Management
As a Product Owner I Need to Know How to Do Portfolio Planning to Deliver Multiple Products to End Users
Calculating Net Present Value

\[ \sum_{t=0}^{N} \frac{C_t}{(1+r)^t} \]

*Where*
- \( C_t \) is the net cash flow in time period \( t \)
- \( r \) is the discount rate
- \( t \) is the time period
- \( N \) is the total number of time periods considered

**Illustrative Example:**
- \( C_0 = -$50; \ C_4 = -$30; \ C_6 = $45; \ C_{10} = $100 \)
- \( r \) is 5%

\[
\text{NPV} = \frac{100}{(1+.05)^{10}} = $61.40
\]

Cash Flow ($)

<table>
<thead>
<tr>
<th>Time (t)</th>
<th>-50</th>
<th>-30</th>
<th>45</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>t0</td>
<td>Today</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

NPV = $20

\[ = \frac{100}{(1+.05)^{10}} = $61.40 \]
The Calculation Is Easily Automated

Download an NPV calculation Excel tool at www.scruminc.com/npvtemplate
NPV/Point Drives Better Decision Making

One metric to encapsulate return on investment
1. Calculate Epic NPV
2. Can also include “intangible” benefits
   • Use Planning Poker to estimate business value relative to reference activity with known cash flows
3. Estimate story points to complete Epic
4. Divide total NPV by estimate of points
   • Answers: How can we get most return with least effort?

Focuses team on optimizing returns
• Eliminates most internal power politics
• Encourage teams to think in business case terms
• Highlights key assumptions and variables to confirm
• Supports after-action retrospective to improve accuracy
• Improves ability to forecast financial results
Prioritize Possible Epics by NPV/Point
Minimum Level Set by Current Rev/Point Run Rate

NPV/point

Available quarterly team capacity for Epics (based on yesterday’s weather)

NPV/Point floor (based on current rev/point run-rate)
Sprint Planning
Sprint Planning Meeting

**Jackass team, sprint 15**

**Sprint goal**
- Beta-ready release!

**Sprint backlog**
- Deposit (5)
- Migration tool (13)
- Backoffice login (3)
- Backoffice user admin (5)
  (Estimated velocity = 26)

**Schedule**
- Sprint period: 2006-11-06 to 2006-11-24
- Sprint demo: 2006-11-24, 13:00, in the cafeteria
- Daily scrum: 9:30 – 9:45, in conference room Jimbo

**Team**
- Jim
- Erica (scrum master)
- Tom (75%)
- Niklas
- Eva
- John
Sprint planning meeting - example

GOAL: Beta-ready release!

- Goal
- Present backlog
- Reprioritize, Re-estimate, split stories, combine stories
- Break out tasks
- Estimate velocity, draw the line

09/02/15

Henrik Kniberg
The Sprint Commitment

- Team’s commitment to the Product Owner:
  - “We promise that ...”
    - We believe we can reach the sprint goal.
    - We will do everything in our power to reach the goal and will inform you immediately if we have problems.
    - Code will be potentially shippable at the end of the sprint.
    - If we fall behind schedule we will negotiate with the Product Owner to decide what to do.
    - If we get ahead of schedule we will add stories from the product backlog in priority order.
    - We will display our progress and status on a daily basis.
    - Every story we do is complete.
- Caveat
  - Estimates are estimates. We will be early some times and late other times. We will document this normal variation with our sprint velocity.
As a Scrum Master I need to run a good daily meeting to help the team improve performance.
Purpose of Daily Scrum

• Build team focus
• All arms linked - intense collaboration
• Mental attitude - we will crush impediments
• Create team spirit
Daily Scrum Meeting

15 minutes

• What did I do yesterday that helped the Team meet the Sprint goal?
• What will I do today to help the Team meet the Sprint goal?
• Do I see any impediment that prevents me or the Team from meeting the Sprint goal?
Scrum of Scrums

- Scrum is an object-oriented organizational framework
- The organization will need to be refactored to maximize flow
- Small steps regularly
- Large changes periodically

Communication Paths
n(n-1)/2 per team
5(4)/2 = 10
24 teams(10) = 240 + a few cross team
80% less comm

Waterfall Comm Paths
n(n-1)/2 for 120 people
120(119)/2 = 7140
Scrum of Scrums as Release Team

Zero Defect Release

After failed software releases we adopted a program Scrum-Of-Scrums...

- Very uncomfortable for people in the beginning
- Huge impact on communications and problem resolution

“I was reluctant at first but the Daily Scrum of Scrums was the key reason this is the best launch in our history...”

Manufacturing Manager

Adapted from Slides By Chris Sullivan
As a Scrum Master I need to act on Scrum Board Warning Signs in order to improve team performance.
### Sprint: Day 0

<table>
<thead>
<tr>
<th>To Do:</th>
<th>Doing:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Kaizen</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Buffer 33</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Deposit</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Migration Tool</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Backend Login</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Backend User Admin</strong></td>
<td></td>
</tr>
</tbody>
</table>

#### Burndown

- **Sprint Goal:** Get a ready release!

#### Points vs. Days

<table>
<thead>
<tr>
<th>Points</th>
<th>Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>90</td>
<td>1</td>
</tr>
<tr>
<td>80</td>
<td>2</td>
</tr>
<tr>
<td>70</td>
<td>3</td>
</tr>
<tr>
<td>60</td>
<td>4</td>
</tr>
<tr>
<td>50</td>
<td>5</td>
</tr>
<tr>
<td>40</td>
<td>6</td>
</tr>
<tr>
<td>30</td>
<td>7</td>
</tr>
<tr>
<td>20</td>
<td>8</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
</tr>
</tbody>
</table>

#### Task Details

- **Migration**: 8 pts
- **Tapestry**: 8 pts
- **Integrate w/boss**: 8 pts
- **Implement GUI**: 5 pts
- **GUI Spec**: 5 pts
- **Implement GUI**: 5 pts
- **Clarify Req**: 3 pts

#### Daily Clean Code 3 pts

- **Completed Tasks**:
  - Write a failing test: 2 pts.
  - Integration test: 3 pts.
  - Code cleanup: 2 pts.
  - DAO: 3 pts.
  - Code cleanup: 2 pts.
  - Tapestry: 8 pts.

#### Remaining Tasks

- **Write a failing test**: 2 pts.
- **Integration test**: 2 pts.
- **Code cleanup**: 3 pts.
- **DAO**: 3 pts.
- **Tapestry**: 8 pts.
Sprint Backlog: After First Meeting

**Sprint Goal:**
Get a ready release!

**Burndown**

<table>
<thead>
<tr>
<th>Points</th>
<th>Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

**To Do:**
- Daily Check Codes
- Daily Clean Codes
- Test
- Code cleanup 2 pts
- Migration 8 pts
- GUI spec
- Migration 8 pts
- Add login 8 pts
- GUI design
- Implement GUI 5 pts
- Migration Tool 8 pts
- Admin
- Deposit
- Backend

**Doing:**
- Daily Check Codes
- Test
- Code cleanup 2 pts
- Migration 8 pts
- GUI spec
- Migration 8 pts
- Add login 8 pts
- GUI design
- Implement GUI 5 pts
- Migration Tool 8 pts
- Admin
- Deposit

**Done:**
# Sprint Backlog: Day X

## To Do:
- **Kaizen**
  - Daily Check Code (3pts)
- **Buffer 23**
  - Sales Support (3pts)
- **Migration Tool**
  - Code cleanup (3pts)
  - Migration (3pts)
  - GUI spec (3pts)
  - Migration Tool (3pts)
- **Backend Login**
  - Implement (2pts)
  - Implement (GUI) (2pts)
  - GUI spec (2pts)
  - Implement (GUI) (2pts)
- **Backend User Admin**
  - Implement (GUI) (2pts)
  - Clarify Req (2pts)
  - Implement (GUI) (2pts)

## Doing:
- **Daily Check Code** (3pts)
- **Sales Support** (3pts)

## Done!
- **Fix Bug** (2pts)
- **Write a failing test** (2pts)
- **White Paper** (3pts)
- **Code cleanup** (3pts)
- **GUI spec** (3pts)
- **Code cleanup** (3pts)
- **GUI spec** (3pts)

## Sprint Goal:
Get a ready release!

## Burndown
- Points
- Days
- Points
- Days
Warning Sign #1

To Do:  | Doing:  | Done!
---|---|---
Kaizen  | Daily Code  |  
Buffer 33  |  |  
Deposit  |  |  
Migration Tool  |  |  
Backend Login  |  |  

Sprint Goal:
Get a ready release!

Burndown

Points

Days

0 10 20 30 40 50 60 70 80 90 100

write a failing test.. 2 pts.
Integration test 2 pts.
Implementation GUI 5 pts.
Migration Tool 8 pts.
Tapestry spike 8 pts.
Integrate w/ boss 8 pts.
Implement GUI 5 pts.
Clarify Req 3 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

Migration

tapestry

daily code

DB Design

Integration test 2 pts.
Implementation GUI 5 pts.
Migration Tool 8 pts.
Tapestry spike 8 pts.
Integrate w/ boss 8 pts.
Implement GUI 5 pts.
Clarify Req 3 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

Backend User Admin

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.

write a failing test.. 2 pts.
write a failing test.. 2 pts.
write a failing test.. 2 pts.
**Warning Sign #2**

**Sprint Goal:**
Get a ready release!

### Burndown

<table>
<thead>
<tr>
<th>Points</th>
<th>Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

---

### To Do:

- **Kaizen**
  - Daily Clean Costs

- **Buffer 3**
  - Sales Support 3 pts
  - Migration Tool 3 pts

- **Deposit**
  - Integration tool 2 pts
  - DB Design 2 pts
  - Migration 8 pts

- **Migration Tool**
  - Tapestry ry 8 pts
  - Migration 8 pts

- **Backend Login**
  - Backend Security 8 pts
  - Implement GUI 5 pts

- **Backend User Admin**
  - Backend Login 3 pts
  - GUI Design 5 pts
  - Implement GUI 5 pts

---

### Doing:

- Fix Bug 2 pts
- White Paper 5 pts
- Customer Direct 13 pts

---

### Done!

- GUI Spec 2 pts
- Code cleanup 2 pts
- Migration 8 pts
- Tapestry ry spike 8 pts
- Integrate w/boss 8 pts
- Implement GUI 5 pts
- GUI design 5 pts
- Implement GUI 5 pts
- Clarify Req 3 pts
- Fix Bug 2 pts
- White Paper 5 pts
- Customer Direct 13 pts
- Marketing Demo 5 pts
- Sale Support 3 pts
- Fix Bug 2 pts
- Write a failing test.. 2 pts
- DAO 3 pts
### Warning Sign #3

#### To Do:
- Kaizen
- Buffer 12
- Deposit
- Migration Tool
- Backend Login
- Backend User Admin

#### Doing:
- Daily Clean Code
- Fix Bug: 2 pts
- White Paper: 5 pts
- Customer Cleanup: 13 pts
- DB Design: 4 pts
- DAO: 3 pts
- Migration: 8 pts
- GUI Design: 5 pts
- Implement GUI: 5 pts
- Write a failing test: 2 pts
- Integrate with boss: 8 pts
- Implement GUI: 5 pts
- Fix Bug: 2 pts
- White Paper: 5 pts

#### Done!
- Daily Clean Code
- Fix Bug: 2 pts
- White Paper: 5 pts
- Customer Cleanup: 13 pts
- DB Design: 4 pts
- DAO: 3 pts
- Migration: 8 pts
- GUI Design: 5 pts
- Implement GUI: 5 pts
- Write a failing test: 2 pts
- Integrate with boss: 8 pts

#### Sprint Goal:
Get a ready release!

#### Burndown

<table>
<thead>
<tr>
<th>Points</th>
<th>Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>90</td>
<td>10</td>
</tr>
<tr>
<td>80</td>
<td>20</td>
</tr>
<tr>
<td>70</td>
<td>30</td>
</tr>
<tr>
<td>60</td>
<td>40</td>
</tr>
<tr>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>40</td>
<td>60</td>
</tr>
<tr>
<td>30</td>
<td>70</td>
</tr>
<tr>
<td>20</td>
<td>80</td>
</tr>
<tr>
<td>10</td>
<td>90</td>
</tr>
<tr>
<td>0</td>
<td>100</td>
</tr>
</tbody>
</table>
Source: Jim Coplien
As a Scrum Master I need to understand Common Pitfalls and Solutions to enable a high performing team.
Scrum Team Hyperproductive Pattern Language

Teams that Finish Early Accelerate Faster

- **Stable Teams** - How you get started
- **Yesterday's Weather** - How you pull backlog into a sprint
- **Swarming** - How you get work done quickly
- **Interrupt Pattern** - How to deal with interruptions during the sprint
- **Daily Clean Code** - How to get defect-free product at sprint end
- **Scrum Emergency Procedure** - Stop the line
- **Scrumming the Scrum** - How to ensure you improve continuously
- **Happiness metric** - How to ensure teams aren’t overburdened

Teams That Finish Early Accelerate Faster: A Pattern Language for High Performing Scrum Teams
47th Hawaii International Conference on System Sciences (HICSS)
By Jeff Sutherland, Neil Harrison, Joel Riddle
January 2014
As a Servant Leader I need to encourage **Swarming**
to enable a high performing team
Exercise: Never Keep Customers Waiting

Never keep a customer waiting
Start early = Finish early
Round 2 – Swarming

Policy

Limit WIP (work in process)

Current limit = 1 customer at a time

DAVE

LISA

Dave
Lisa
Bob
Eric
Maria
Discussion – what was the difference? Why?
Weinberg Table of Project Switching Waste

Table 2-1: Waste Caused by Project Switching

<table>
<thead>
<tr>
<th>Number of Simultaneous Projects</th>
<th>Percent of Working Time Available per Project</th>
<th>Loss to Context Switching</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100%</td>
<td>0%</td>
</tr>
<tr>
<td>2</td>
<td>40%</td>
<td>20%</td>
</tr>
<tr>
<td>3</td>
<td>20%</td>
<td>40%</td>
</tr>
<tr>
<td>4</td>
<td>10%</td>
<td>60%</td>
</tr>
<tr>
<td>5</td>
<td>5%</td>
<td>75%</td>
</tr>
</tbody>
</table>

Prioritizing Between Projects

Adapted from Henrik Kniberg

Traditional strategy: “Everything is important! Do it all at once!”

Agile strategy: “Prioritize & focus!”

Adapted from Henrik Kniberg
Swarming
Care about the whole product

Not just your little task

I'm more efficient if I just do my tasks

Source: Revised after Henrik Kniberg

This product rocks!

Boy are we effective as a team!

Not just your little task

I'm more efficient if I just do my tasks

Source: Revised after Henrik Kniberg

© 2011-2014 Jeff Sutherland
Enterprise Level Swarming
As a Scrum Master I need to manage interruptions in order for the team to be successful.
Illigitimus Non Interruptus

On Buffer Overflow **ABORT**, Replan, Dates Slip
As a Scrum Master my Team needs to have Working Product to double production and quality.
Lean Results - Same with Scrum

Red River Army Depot
Humvee Repair Facility

<table>
<thead>
<tr>
<th>Productivity / Cost</th>
<th></th>
<th>Membership</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Improvement</td>
</tr>
<tr>
<td>HMMWV;</td>
<td>70%</td>
<td>400 to 120 man-hrs / vehicle</td>
</tr>
<tr>
<td>HEMTT;</td>
<td>41%</td>
<td>2003 to 1172 man-hrs / vehicle</td>
</tr>
<tr>
<td>SEE;</td>
<td>46%</td>
<td>1600 to 870 man-hrs / vehicle</td>
</tr>
<tr>
<td><strong>OVERALL:</strong></td>
<td>52%</td>
<td><strong>27% increase in personnel</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Delivery</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Improved output</td>
</tr>
<tr>
<td>HMMWV;</td>
<td>3900%</td>
<td>3 to 120 vehicles / week</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Flow time decreased by 75% from 40 to 10 days</td>
</tr>
<tr>
<td>HEMTT;</td>
<td>260%</td>
<td>5 to 18 vehicles / week</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Flow time decreased by 75% from 120 to 30 days</td>
</tr>
</tbody>
</table>

Workers rehabilitate vehicle frames on an assembly line at Red River. The depot looked to Toyota's "Lean manufacturing" process to streamline and expedite repairs.

National Public Radio
# In Process Inventory is the Biggest Waste

<table>
<thead>
<tr>
<th>Muda 無駄</th>
<th>In-Process Inventory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overproduction</td>
<td>Partially implemented user stories, bug debt and incomplete work carried forward. Requires multiple handling, creates overhead and stress.</td>
</tr>
<tr>
<td>Extra Processing</td>
<td>Teams create low-priority features and make them self-justifying. This work squeezes capacity from the high-priority work.</td>
</tr>
<tr>
<td>Transportation</td>
<td>Bug debt, reactivations, triage, redundant testing, relearning of others’ code, handling broken dependencies.</td>
</tr>
<tr>
<td>Motion</td>
<td>Handoffs across roles, teams, divisions, and so on.</td>
</tr>
<tr>
<td>Waiting</td>
<td>Managing tools, access rights, data transfer, lab setup, parallel release work.</td>
</tr>
<tr>
<td>Correction</td>
<td>Delays, blocking bugs, incomplete incoming components or dependencies.</td>
</tr>
<tr>
<td></td>
<td>Scrap and rework of code.</td>
</tr>
</tbody>
</table>

| Mura 斑    | Unevenness                  | Varying granularity of work, creating unpredictability in the flow. |
|           | Inconsistency               | Different definitions of done, process variations that make assessment of “potentially shippable” impossible |

| Muri 無理 | Absurdity | Stress due to excessive scope. |
|          | Unreasonableness | Expectations of heroic actions and commitments to perform heroic actions. |
|          | Overburden | Stress due to excessive overhead. |
The 7 Wastes of Software Development

- Partially done work
- Extra features
- Lost knowledge
- Handoffs
- Task switching
- Delays
- Defects

Features and functions used in a typical system:

- Always: 7%
- Often: 13%
- Sometimes: 16%
- Rarely: 19%
- Never: 45%

**2/3 of the stuff we build is rarely or never used!**

**Only 1/5 of the stuff we build is used often or always!**

Source: Standish Group Study Reported at XP2002 by Jim Johnson, Chairman

There is surely nothing quite so useless as doing with great efficiency what should not be done at all.

*Peter Drucker*
Systematic Strategy for Lean

Tools divided into three dimensions

Level\Dimension

Production

Management

People

Thinking tools are best transformed by people and projects
Impediments

Data driven removal of impediments using control charts from 11/2007

Examples on causes:

• Special competencies
• Disk full
• Setup misunderstood
• COTS failed

Root cause analysis of time to fix a bug automatically generates Scrum Master’s impediment list.
Daily Clean Code Pattern
scrumplop.org

... bugs turn into features at midnight ...
Here we discuss bugs that arise within the sprint. Preexisting bugs should be prioritized by the Product Owner and managed in the Product Backlog. Bugs appearing from outside the sprint such as customer emergencies should be handled by the Illigitimus non Interrupus pattern.

Velocity is limited because a team spends time dealing with too many bugs.
It is natural to want to keep a list of bugs. There are several forces that encourage this.

- One of the most compelling reasons to put bugs on a bug list is that developers are busy with other tasks, and shouldn’t be interrupted.
- Managers can see that adding new features increases revenue, but fixing bugs does not apparently increase revenue. If the team has a fuzzy Definition of Done, work might be considered Done.
- Bugs that arrive might be considered low priority, and it’s nice to have a place to put them.
- In short, deferring the fixing of bugs until later is borrowing against your future velocity.
Therefore: Fix all bugs in less than a day.
As a Scrum Master I need to optimize Flow in order for the team to be successful.
Theory of Constraints – Smooth Flow

1. Reduce intake
2. Fix
3. Increase intake
4. Fix next
Case Study:
Developing Products >5x Faster

A real-life example of applying Value Stream Mapping and Scrum to dramatically speed up product development.
Games out of date
⇒ Missed market windows
⇒ Demotivated teams
⇒ Overhead costs

3.5 m value added

Process cycle efficiency = 14%

Estimate

2 m cycle time = 12x faster

Preliminary result

3-4 m cycle time = 6-8x faster

Source: Henrik Kniberg
Case Study

Take-away Points

• Speeding up product development is often a matter of improving the process rather than adding people.
• Value stream mapping is a great tool for spotting bottlenecks.
• Scrum is a great tool for removing bottlenecks.
  • But beware the trap – suboptimization!
• The pictures make it look easy....
  • But executing the change is usually hard.

Source: Henrik Kniberg
As a Scrum Master I Need My Team to Have Working Software at the End of a Sprint to be Agile
Definition of Done

Default Definition of Done
• Releasable

Default Definition of Done
• Unit/Integration tested
• Ready for acceptance test
• Deployed on demo server

Default Definition of Done
• Acceptance tested
• Release notes written
• Releasable
• No increased technical debt

What’s else must be done before shipping the code?
- For example “customer acceptance test + user documentation”
Why not? Who does it? When? What happens if a problem turns up?
Burn up this work in release burndown!

Source: Henrik Kniberg
Exercise: Why Are Teams Not Done?

- Write down on stickie notes reasons for not done.
- One per note. As many notes as you can create in 4 minutes.
- Bring them to the front of the room.
Why Is It So Important to Have Working Product?

Teams That Finish Early Accelerate Faster!
Teams that Finish Early Accelerate Faster

- **Stable Teams** - How you get started
- **Yesterday’s Weather** - How you pull backlog into a sprint
- **Daily Clean Code** - How to get defect-free product at sprint end
- **Swarming** - How you get work done quickly
- **Interrupt Pattern** - How to deal with interruptions during the sprint
- **Stop the Line** - How to deal with emergencies
- **Scrumming the Scrum** - How to ensure you improve continuously
- **Happiness metric** - How to ensure teams aren’t overburdened

Teams That Finish Early Accelerate Faster: A Pattern Language for High Performing Scrum Teams
47th Hawaii International Conference on System Sciences (HICSS)
By Jeff Sutherland, Neil Harrison, Joel Riddle
January 2014
ScrumLab Vision Statement - scruminc.com

- **FOR** experienced Scrum practitioners (Jill) who are “in the trenches”
- **WHO need** clear and actionable information to answer specific Scrum questions whenever they arise
- **ScrumLab** is the authoritative, curated on-demand source for Scrum Inc.’s leading thinking
- **That:**
  - Clearly explains Scrum and its underlying principles (i.e. why it works)
  - Shares successful advanced practices for different contexts
  - Provides actionable solutions to implement successfully
  - Is available whenever you need it
- **Unlike** other online Scrum resources
- **Our product** captures decades of successful experience and wisdom from the co-creator of Scrum and his hand-picked team of thought leaders
Why Don’t Teams Have Working Software

- Poor definition of DONE
- Stories not READY
- Dysfunctional leadership
- Technical debt
- Organizational Debt
- Ineffective coaching

Source: ScrumInc/VersionOne Workshop 14 Oct 2014
Poor Definition of DONE

• **Definition of DONE unclear**
  • It is impossible to be DONE if you don’t know what DONE is.

• **Lack of consistent quality standards**
  • Definition of DONE does not include “working software”.
  • Dysfunctional Product Owner accepts stories that are not done.
Stories Not Ready

• **Sizing stories**
  • Bad estimates - inconsistent use of story points
  • Taking stories to big into sprint
  • Taking to many stories into sprint

• **Poorly written stories**
  • Stories not clear, particularly acceptance criteria
  • Unidentified dependencies
Dysfunctional Leadership

• Too many projects in pipeline (Context Switching)
• Everything is top priority
• Pressure to get things done delays projects and reduces quality
• Lack of understanding of Scrum
• Fear of exposure or change in responsibilities
• No continuous integration and/or no testing at all (Obamacare)
Technical Debt

- Not finishing sprints creates bad code - 24x delay
- Legacy code is often accumulation of mountain of technical debt which reduces velocity
  - Severely aggravated by not using current technology for continuous integration and automated testing
  - Technical debt is incurred by running development too close to maximum which generates short cuts, lack of refactoring, loss of creativity, demotivation, and sloppy craftsmanship

Microsoft TFS Mountain of Technical Debt - Scrum reduced bugs from 30000 to 2000 - Agile Software Development with Vision Studio, 2011
Poor Coaching

- **Silo’s and specialization** cripple velocity
  - specialized test teams are the worst example
- **Developers not functioning as a team**
  - minimal collaboration, no swarming
- **No continuous improvement** flatlines velocity
  - no happiness
  - no interrupt pattern
  - no scrumming the scrum
- “**Pretend Agile**” - no teamwork, no working software, no customer collaboration, and no effective response to change
Batch Size / Iteration Length

Process / External Factors

Work Item Cycle Time or Lead Time

Scale = 5
< 1 week

Scale = 15
~ 2 week sprint

Scale = 30
~ 1 month
Systematic Approach to Getting To Done

• Implementing the Definition of Done
• Ensuring that backlog is Ready
• Training management
• Technical debt remediation plan
• Refactoring the organization
• Upgrading coaching and Scrum Master positions
Systematic Scrum Model

Disciplines

Clarify features

Automated test
Continuous Integration

Verify sprint delivery

Valu

READY

SPRINT

DONE

Velocit

Feature

Story

Scrum and CMMI - Going from Good to Great: Are You Ready Ready to Be Done Done
Implementing Done

- **Definition of Done** must include integration testing and test capacity must exceed coding capacity
- **Testers must be on the Scrum team** - no separate test teams
- **Do not take too much into sprint.** Use Patterns.
  - Use “Yesterday’s Weather” pattern
  - “Illigitimus Non Interruptus”, and
  - “Scrum Emergency Procedure”
- **Use automated build system** combining new and old code (continuous integration)
- Systematically build **automated acceptance tests** which prevent top priority problems first
- **Bugs fixed in less than a day**
  - “Daily Clean Code”
  - 70% of defects are integration defects. Testing them later will take up to 24 times more testers!
Implementing Ready

- Scrum Guide updated to include concept of Ready
- Team agrees on common **Definition of Ready**
- Only Ready Stories into Sprint Backlog
- **Backlog Refinement assures Ready** state.
- A good Ready state can **triple velocity**. Spend the time needed to get the backlog Ready.
<table>
<thead>
<tr>
<th>Stage</th>
<th>Description</th>
</tr>
</thead>
</table>
| New Card Nursery | - All inputs accepted  
- **Promotion:** Product Owner determines this story matches product goals |
| Elementary School | - Analysts decompose  
- User experience experts research context  
- Business alignment needs identified  
- **Promotion:** Matches release goals |
| Junior High | - Card details, acceptance criteria, UI pre-work (wireframes, visual and content prototypes)  
- Legal & compliance issues reviewed  
- **Promotion:** Alignment with key stakeholders on features, functions, and visuals |
| High School | - Ready for sprint  
- Candidates for Release Planning/Sprint Planning  
- Minimal refinement expected on core User Experience |
Functional Leadership

- Agile competition goes beyond lean manufacturing by permitting the customer, jointly with the vendor or provider, to determine what the product will be.
- For agile competitors, the ability to individualize products comes at little or no increase in manufacturing cost. It does, however, exact a cost: **It requires major changes in organization, management philosophy, and operations.**
- Managers need to be trained in how to lead Agile teams.
Leadership Responsibilities

- Provide challenging goals for the teams
- Create a business plan and operation that works
  - Set up the teams (in collaboration with teams)
  - Provide all resources the teams need
- Identify and remove impediments for the teams
  - Know velocity of teams
  - Understand what slows teams down (impediment list)
  - Remove waste (first-things-first)
- Hold P.O. accountable for value delivered per point
- Hold S.M. accountable for process improvement and team happiness
Fix Technical Debt

- **Remediate**
  - 80% of bugs come from 20% of code (or less)

- **Stop the Pain**
  - Systematically build acceptance tests into the build - highest priority first

- **Reduce the Debt**
  - Team build business case for Product Owner -
  - How many points for Tech Debt could go to value creation? (How long will it take to remove debt?)

- **Management commits to systematic improvement of product**
  - Reduce operational costs
  - Increase sales
Eliminate Organizational Debt

- Identify blocking teams and fix them
- Move towards cross-functional feature teams
- Train and hire T-shaped people
- Build out an object-oriented architecture for the product
- Use value stream mapping to identify queues, wait states, and other waste across the organization
Spotify Succeeds with Excellent Coaching

• Hires great workers
• Every team has a coach
  • Coaches are responsible for 1 process improvement every Sprint
  • Improvement backlog and they measure improvement continuously
  • Coaching has radically improved output of high performance teams.
• In the last year, 33% of all Spotify Teams have moved to continuous deployment multiple times per sprint.
### Table 2 – Example cash flow improvement

<table>
<thead>
<tr>
<th>Cycle Time</th>
<th>Forecast Date</th>
<th>Forecast Cost</th>
<th>Cash flow Benefit (cost savings + FY revenue)</th>
<th>Benefit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current</td>
<td>15-Jul</td>
<td>1000K</td>
<td>$0 + $60K = $60K</td>
<td>0%</td>
</tr>
<tr>
<td>10% Decrease</td>
<td>27-May</td>
<td>912K</td>
<td>$87K + $90K = $177K</td>
<td>296% Better</td>
</tr>
<tr>
<td>20% Decrease</td>
<td>4-Apr</td>
<td>820K</td>
<td>$120K + $145K = $265K</td>
<td>442% Better</td>
</tr>
</tbody>
</table>

Best Metrics for Coaches

- **Time to fix a defect.** If this averages less than 24 hours the team’s velocity will double.
- **Measure of swarming.** How well do individuals and interactions generate performance.
- Measure flow = actual work to do a story/calendar time to done
- If this is over 50% team velocity will double again
Conclusions

- Bad Agile is caused by six primary factors
  - Poor definition of DONE
  - Stories not READY
  - Dysfunctional leadership
  - Technical
  - Organizational debt
  - Ineffective coaching
- Systematically focusing on remediating these issues will consistently produce high performing teams with 200-400% improvement in production.
- Failure to focus on them will add yet another team to the 58% of teams that are “Bad Agile” leading to unhappy customers, lost revenue, and lower stock prices.
As a Scrum Master I need my Team to show Working Software at the Sprint Review to get feedback from Stakeholders
Sprint Review

What have we accomplished?

- Team demonstrates working code to stakeholders
- Only 100% completed stories are demonstrated
  - Partially complete stories are ignored
  - Product Owner confirms what is Done
- Direct feedback from stakeholders
- Feedback incorporated into the product backlog
Deliverables from the Sprint Review

- Potentially shippable increment of product
- Velocity (what is Done)
- Feedback (update Product Backlog)
As a Scrum Master I need to Scrum the Scrum to get an effective Retrospective
Sprint Retrospective

What happened?

- **Week 1**
  - New desks installed
  - First story ready for test
  - Sam sick

- **Week 2**
  - Server crashed
  - Story #25 removed from sprint
  - Big argument

- **Week 3**
  - Half-day conference
  - New build server
  - LAN shootout
  - Team flow
  - Sprint demo

Source: Henrik Kniberg
Sprint Retrospective

Source: Henrik Kniberg
Powering Up the Retrospective

- Identify the top process improvement
- Create acceptance tests
- Put it in the sprint backlog as top priority

- This is a pattern called Scrumming the Scrum
Sprint Retrospective

Long term effect

Velocity

Sprint

Effective velocity over time (with retrospectives)

Effective velocity over time (without retrospectives)

Source: Henrik Kniberg
Starting Up a New Scrum Team

- Scrum, Inc. started up a new team in December 2010. Half of the team knew nothing about Scrum.
  - One week sprints
  - Scrumming the Scrum
  - Happiness Metric
Happiness Metric Moves Team Members Towards the Upper Right Quadrant

Happier People Function Better

- Doctors in a positive mode show three times the intelligence and creativity and diagnose 19% faster.
- Optimistic salespeople outsell pessimistic counterparts by 56%.
- Happier CEOs are 15% more productive.
- Happier managers improve customer satisfaction by 42%.
- Research shows that happiness causes better performance.
Happiness Metric

- On a scale of 1 - 5 we rate
- How do you feel about your role?
- How do you feel about the company?
- What would make you feel better?
- With data from Happiness Metric, order individual and company improvements by value - then Scrum the Scrum

Vote for highest value

Estimate value
“Rules”

- Only address one impediment
- Put the kaizen in the backlog for the sprint - Scrum the Scrum
- Action should usually yield results quickly
- Communicate actions (and success or not) back to the Team
- And the hardest rule: **Use common sense.**
Results: Scrumming the Scrum Using Happiness Metric

Raw Scrum Inc. Velocity History
(not adjusted for fluctuation in team capacity by sprint)

16x output with
4x FTEs = 4x
Acceptance Tests
Modern Agile Acceptance Model

- The move toward Acceptance Test Driven Development requires a more complete model of progressing requirements acceptance
- Example model (sharper definition of done)
  - Conditions of Satisfaction - broad terms
    - Acceptance Criteria - further refined
      - Examples - actual scenarios or data
      - Executable Examples - Executable Spec
Simple Scenarios

- Suppose we are creating a registration function
- It consists of specifying email (as User ID) and Password

Enter your Email and Password to Register

<table>
<thead>
<tr>
<th>Email Address:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Password:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Register</th>
</tr>
</thead>
</table>
Acceptance

Condition of Satisfaction

• Conditions of Satisfaction are high level criteria established with an initial Epic/Feature/User Story
  • In our previous example, the conditions of acceptance for the password would be:
    • Ensure the password is not easy to crack.

• The PO would define the conditions of acceptance in concert with business stakeholders
Acceptance Examples Make It Real

• Actual examples are the ultimate way to communicate requirements:

<table>
<thead>
<tr>
<th>Password</th>
<th>Expected</th>
<th>Expected Message</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>abc123</td>
<td>Invalid</td>
<td>Your password must be at least 8 characters and no more than 12 characters long</td>
<td></td>
</tr>
<tr>
<td>abcdefghi</td>
<td>Invalid</td>
<td>Your password must contain at least one character and one number</td>
<td></td>
</tr>
<tr>
<td>1aaaaaaaaa</td>
<td>Valid</td>
<td></td>
<td>Why valid?</td>
</tr>
<tr>
<td>ajx972dab</td>
<td>Valid</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

• The PO works closely with a tester, developer, and business stakeholder to define these criteria
Acceptance

Acceptance Criteria

• Acceptance Criteria specifies the details that the team can then build against:
  • Following the example for Password:
    • Must be at least 8 characters and no more than 12
    • Must contain only alpha-numberics and the period
    • Must contain at least one digit
    • Must contain at least one character
    • Etc. (there may be more criteria)

• The PO works closely with a tester, developer, and business stakeholder to define these criteria
Acceptance Test Driven Development (ATDD) Tools: Fit and Cucumber

FIT (Framework for Integrated Test) and Fitnesse (Wiki front end)
- Test specified in table format
- Developers generates classes (“fixtures”) to hook into application
- Users/testers use Wiki or Excel to specify inputs and outputs

Cucumber
- New tool for natural language scenarios

In order to ensure my account is correct
As a Registered User
I want to check my recent activity

Scenario: Recent Account Activity
Given I am a registered user “Jsmith”
And I am logged in with password “xyx123”
And I have had account activity in the last 45 days
And I am on the home page
When I click the “Recent Activity” button
Then I should see the “Account Activity” Page
And I should see a list of my activity over the last 45 days

Scenario: No Recent Account Activity
Given I am a registered user “Jsmith”
And I am logged in with password “xyx123”
And I have had no account activity in the last 45 days
And I am on the home page
When I click the “Recent Activity” button
Then I should see the “Select Account Activity Period” Page
And I should get a message: “You had no activity in the last 45 days, please select a time frame to review”

<table>
<thead>
<tr>
<th>numerator</th>
<th>denominator</th>
<th>quotient</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>12.6</td>
<td>3</td>
<td>4.2</td>
</tr>
<tr>
<td>100</td>
<td>4</td>
<td>25 expected</td>
</tr>
</tbody>
</table>
Engineering Practices and Scrum
Scrum & XP

Team

Scrum

Daily Scrum

Whole team

XP

Coding standard

Sprint planning meeting

Refactoring

Planning game

Burndown chart

Customer tests

Pair programming

TDD

Sustainable pace

Continuous integration

Simple design

Sprint review

Collective ownership

Metaphor

Small releases

Product backlog

Source: Henrik Kniberg
Feedback Cycles

- Sprint demo
- Daily Scrum
- Continuous integration
- Unit test
- Pair programming

Source: Henrik Kniberg
How Google Tests Software

- Google early adopter of agile methods and Scrum
- Google also uses agile testing at enterprise scale
- 15,000 developers run 75+ million tests per day

- 15,000+ developers in 40+ offices
- 4,000+ projects under active development
- 5,500+ submissions per day on average
- Single monolithic code tree with mixed language code
- Development on one branch - submissions at head
- All builds from source
- 20+ sustained code changes per minute with 60+ peaks
- 50% of code changes monthly
- 75+ million test cases run per day

As a class group we need a Course Review in order to retain knowledge.
Folding

Fold 5 boats using the prototype in the bag as a model

Business Value: 300
Estimated Complexity: 1 - 2 - 3 - 5 - 8 - 13 - 21
Story 26 - Iteration 2
XP Game - Backlog Refinement

- Elect a Product Owner (PO) and Scrum Master (SM) in each team
- PO fetch product backlog
- SM fetch paper, balloons, etc.
- For each story in backlog:
  - Team estimates relative complexity (story points)
  - PO prioritize stories
- Teams estimates how many cards can get done in three minute sprint
- Product Owner captures estimates and actuals
- Sum of business value of all cards / sum of story points for all cards
Having Fun Yet?
As a class group we need a Course Retrospective in order to wrap up effectively.
For those who want certification ...

- Read agileatlas.org on Core Scrum and the Scrum Guide at scrumguides.org before taking the test.
- You will receive email from the Scrum Alliance with login instructions to take the test.
- Students who pass the test will receive a list of missed questions with correct answers highlighted.
- Students who fail will receive a list of missed questions without answers.
- The test can be taken one additional time at no charge. After that a fee of $25 per additional attempt will be charged.
- A passing score is a least 24 out of 35 questions correct.
http://www.youtube.com/watch?v=u6XAPnuFjJc
Stay Connected

Scruminc.com
• For up coming events, new content releases, and more!

ScrumLab
• articles, online courses, tools, and papers on all things scrum
• www.scruminc.com/scrumlabs

Blog
• http://www.scruminc.com/category/blog/

Online Courses
• advance your scrum with our online courses. Visit the Scrumlab store to view upcoming topics.

Twitter, Facebook, and G+
• @ScrumInc, @jeffsutherland, scrum and scrum inc.
Advanced Topic: Distributed Scrum
• With distributed agile development it is possible to tap into new
global markets and make best use of globally available talent, while
potentially reducing costs.
• Teams at patterns & practices have been successfully using this
approach for a number of years but its success should not be taken
as a given.
• The decision to distribute your project should be a conscious one
and the decision maker(s) must understand that in doing so they:
  • reduce the project’s likelihood of success
  • increase the delivery time
  • reduce the team’s performance and increase its dysfunction
• The risk/reward tradeoff needs to be clearly understood before
deciding to distribute your team(s).
Current State of Distributed Scrum
Distributed/Outsourcing Styles

- Isolated Scrums
- Distributed Scrum of Scrums
- Distributed Daily Meeting
Outsourcing

- What happens if you outsource $2M of development?
  - Industry data show 20% cost savings on average

- Outsourcing from PatientKeeper to Indian waterfall team:
  - Two years of data showed breakeven point occurs when Indian developer costs 10% of American Scrum developer
  - Actual Indian cost is 30%

- $2M of Scrum development at my company costs $6M when outsourced to waterfall teams
Future (desired) State
Distributed Scrum: Agile Project Management with Outsourced Development Teams (PDF)

Big Island, Hawaii
January 03-January 06
ISBN: 0-7695-2755-8
Jeff Sutherland, Patientkeeper, USA
Anton Viktorov, StarSoft Dev. Labs, Russia
Jack Blount, SirsiDynix, USA
Nikolai Puntikov, StarSoft Dev. Labs, Russia

DOI Bookmark: http://doi.ieeecomputerociety.org/10.1109/HICSS.2007.180

ABSTRACT

Agile project management with Scrum derives from best business practices in companies like Fuji-Xerox, Honda, Canon, and Toyota. Toyota routinely achieves four times the productivity and 12 times the quality of competitors. Can Scrum do the same for globally distributed teams? Two Agile companies, SirsiDynix and StarSoft Development Laboratories achieved comparable performance developing a Java application with over 1,000,000 lines of code. During 2005, a distributed team of 56 Scrum developers working from Provo, Utah; Waterloo, Canada; and St. Petersburg, Russia, delivered 671,688 lines of production Java code. At 15.3 function points per developer/month, this is the most productive Java project ever documented. SirsiDynix best practices are similar to those observed on distributed Scrum teams at IDX Systems, radically different than those promoted by PMBOK, and counterintuitive to practices advocated by the Scrum Alliance. This paper analyzes and recommends best practices for globally distributed Agile teams.
SirsDynix - 12,500 library systems in 42 countries

- Over a million lines of Java code
SirsiDynix Distributed Scrum

- 56 developers distributed across sites

- SirsiDynix
  - Provo, Utah
  - Denver, CO
  - Waterloo, Canada

- Starsoft Development Labs
  - St. Petersburg, Russia
SirsDynix Distributed Scrum

Scrum daily meetings

- Local Team Meeting
- St. Petersburg, Russia 17:45pm
- 7:45am Provo, Utah
- Scrum Team Meeting
Sirsidynix Distributed Scrum
## Velocity in Function Points/Dev month

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Person Months</td>
<td>54</td>
<td>540</td>
<td>827</td>
</tr>
<tr>
<td>Lines of Java</td>
<td>51,000</td>
<td>58,000</td>
<td>671,688</td>
</tr>
<tr>
<td>Function Points</td>
<td>959</td>
<td>900</td>
<td>12673</td>
</tr>
<tr>
<td>Function Points per Dev/Month</td>
<td>17.8</td>
<td>2</td>
<td>15.3</td>
</tr>
</tbody>
</table>
SirsiDynix Challenges

- Builds were stable only at Sprint boundaries
- No XP in U.S, only in Russia, did not have equal talent across teams
- No face to face meetings
- Low test coverage
- Poor refactoring practice
- Company merger created competitive products
- Sirsi now owned Dynix and killed Dynix product
Research Issue

- SirsiDynix was a retrospective study of a single data point
- Even if quality was perfect, it does not prove anyone else can do it.
- Even worse, if you observe a finding after the fact, you cannot infer causality
- Is SirsiDynix a lucky accident? Or maybe an unlucky accident?
SirsiDynix Opportunity

- A prospective study should be done
  - full XP technical practices
  - multiple projects
  - meet or exceed SirsiDynix velocity
  - ensure quality levels in the top 1% of the software industry.
Setting up a prospective study

- Define the distributed team model before projects start
- Assure consistent talent, tools, process, and organization across geographies
- Establish high quality data gathering techniques on velocity, quality, cost and environmental factors.
- Run a consistent team model on a series of projects and look for comparable results
- Demonstrate that local velocity = distributed velocity
- Demonstrate that local quality = distributed quality
- Demonstrate linear scaling at constant velocity per developer
Since 2006, Xebia (Netherlands) started localized projects with half Dutch and half Indian team members. After establishing localized hyperproductivity, they move the Indian members of the team to India and show increasing velocity with fully distributed teams. After running XP engineering practices inside many distributed Scrum projects, Xebia has systematically productized a model very similar to the SirsiDynix model [1] for high performance, distributed, offshore teams with outstanding quality.
Xebia Case study: Building a new railway information system (ProRail)
Xebia/ProRail Example

- ProRail rescued a failed waterfall project to build a new scheduling system and automated railway station signs at all Netherlands railway stations.
- An 8 person half-Dutch and half-Indian Scrum team started the project and established local velocity.
- After establishing local velocity at 5 times other waterfall vendors on the project, the Indian half of the team went back to India.
Xebia/ProRail Definition of Done

- Scrum teams run all XP practices inside the Scrum including intensive pair programming.
- The customer completes acceptance testing on all features during each Sprint.
- Done at the end of the Sprint means customer has accepted the code as ready for production.
- Defect rates are less than 1 per 1000 lines of code and steadily getting lower.
Xebia/ProRail Defect Tracking

- Defect rate gets lower and lower as code base increases in size.
- 95% of defects found inside iteration are eliminated before the end of the iteration.
Xebia Team Characteristics

• TDD, pair programming, continuous integration. Same tools and techniques onshore and offshore.
• Daily Skype video Scrum meeting of team across geographies.
• SmartBoards, wikis, and other tools used to enhance communication.
• Indians say it feels exactly the same in India as it does in Amsterdam. They do the same thing in the same way.
Integrating analysis, testing and implementation

- Analysts, testers and developers define together how a user story will be tested using FitNesse
- Testers write functional tests while the user story is being implemented
- Developers write ‘fixtures’, which link tests with the production code
- Regression tests are run continuously
- Bugs are found and fixed within the Sprint
# ATDD with FitNesse

### 1. Process a single standard DVS message for a CTA.

**DVS**
- departure: AMF, track 1, fase a time 10:10 to UT
- abbreviated route stations: HVS, WP, BRN
- actual stop stations: HVS, WP, BRN, UT
- trainType: ICE International
- carrier: syntus
- process

<table>
<thead>
<tr>
<th>check publication on display</th>
<th>AMF-TB7-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>check total trains</td>
<td>1</td>
</tr>
<tr>
<td>train</td>
<td>1</td>
</tr>
<tr>
<td>departure time</td>
<td>10:10</td>
</tr>
<tr>
<td>destination</td>
<td>Utrecht Centraal</td>
</tr>
<tr>
<td>track phase</td>
<td>1a</td>
</tr>
<tr>
<td>via</td>
<td>via Hilversum, Weesp, Baarn</td>
</tr>
</tbody>
</table>

**Check CTA display**

<table>
<thead>
<tr>
<th>check publication on display</th>
<th>AMF-ICTA-1a</th>
</tr>
</thead>
<tbody>
<tr>
<td>check departure time</td>
<td>10:10</td>
</tr>
<tr>
<td>check train type</td>
<td>ICE International</td>
</tr>
<tr>
<td>check carrier</td>
<td>syntus</td>
</tr>
<tr>
<td>check destination</td>
<td>Utrecht Centraal</td>
</tr>
<tr>
<td>check via</td>
<td>via Hilversum en Baarn <strong>expected</strong></td>
</tr>
<tr>
<td>check via</td>
<td>via Hilversum, Weesp en Baarn <strong>actual</strong></td>
</tr>
</tbody>
</table>
Resolving ‘us’ against ‘them’

- Several team members in India had earlier experience with offshore development
  - Most had experienced a lot of distrust and misunderstanding
- Being in one team with onshore colleagues made a huge difference
  - Working jointly to solve a problem is a great way of resolving conflict
- Robbers Cave Experiment (1954):
  - 2 separate groups of boy scouts - strong ‘us’ against ‘them’ feeling
  - Only resolved when faced with a shared problem (water supply was cut off)
Resolving Cultural Differences

- One of the teams had local velocity decrease after distributing the team.
- Root cause analysis indicated the Indians were waiting for the senior Indian developer to tell them what to do.
- The same day this was determined, the Dutch ScrumMaster became a team member and the lead Indian developer became the ScrumMaster with the goal of eliminating the impediment.
- Distributed velocity immediately went up to previously established local velocity.
## Dutch Velocity vs. Russian Velocity

1. M. Cohn, User Stories Applied for Agile Development. Addison-Wesley, 2004

<table>
<thead>
<tr>
<th></th>
<th>Dutch Velocity</th>
<th>Russian Velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person Months</td>
<td>827</td>
<td>125</td>
</tr>
<tr>
<td>Lines of Java</td>
<td>671,688</td>
<td>100,000</td>
</tr>
<tr>
<td>Function Points</td>
<td>12673</td>
<td>1887</td>
</tr>
<tr>
<td>Function Points per Dev/Month</td>
<td>15.3</td>
<td>15.1</td>
</tr>
</tbody>
</table>
Linear Scalability of Large Scrum Projects

Xebia Fully Distributed Scrum
# Large Scale Implementation of Fully Distributed Scrum Model

<table>
<thead>
<tr>
<th>Date</th>
<th>Event Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>09/2006</td>
<td>First presentations about agile software development</td>
</tr>
<tr>
<td>1.2.2007</td>
<td>Start of the first sprint (pilot: Intranet Germany)</td>
</tr>
<tr>
<td>&gt;60</td>
<td>Agile Teams</td>
</tr>
<tr>
<td>&gt;420</td>
<td>Team Members</td>
</tr>
<tr>
<td>&gt;140</td>
<td>Certified ScrumMasters</td>
</tr>
<tr>
<td>&gt;40</td>
<td>Certified Scrum Product Owner</td>
</tr>
<tr>
<td>3</td>
<td>Scrum Coaches</td>
</tr>
<tr>
<td>high</td>
<td>Support and Commitment from CIO</td>
</tr>
<tr>
<td>ETT</td>
<td>Enterprise Transition Team is established (currently sprint 15)</td>
</tr>
</tbody>
</table>
Root Case Analysis and Interventions
Examples

- Management and Product Owners onshore, all development offshore
- Teams scattered around the world
• Investment issues to resolve with distributed teams
  • How will you retain intellectual property onshore?
  • How will an onshore product owner have local access to technical expertise?
  • How will you measure offshore performance and move back onshore if you are losing money?
“It always seems impossible until it’s done.”
_Nelson Mandela_