PRACTICAL ROADMAP TO GREAT SCRUM

SYSTEMATICALLY ACHIEVING HYPERPRODUCTIVITY


Monday, August 24, 2009
Jeff Sutherland, Ph.D.

- Chairman, Scrum Training Institute
- CEO Scrum, Inc. and Senior Advisor, OpenView Venture Partners
- Agile coach for OpenView Venture Partners portfolio companies
- CTO/VP Engineering for 9 software companies
- Created first Scrum at Easel Corp. in 1993. Rolled out Scrum in next 5 companies
- Achieved hyperproductive state in all companies. Signatory of Agile Manifesto and founder of Agile Alliance

- [http://jeffsutherland.com/scrum](http://jeffsutherland.com/scrum)
- [jeff@scruminc.com](mailto:jeff@scruminc.com)
Techniques or Methodologies Used

Source: Forrester Research December 2008
Global Agile Company Online Survey

Base: 241 technology industry professionals in a variety of roles, including but not limited to development (numbers have been rounded)
Scrum is a Simple Framework

- **Roles**
  - Product Owner
  - ScrumMaster
  - Team

- **Meetings**
  - Daily Meeting
  - Sprint Review
  - Sprint Planning

- **Artifacts**
  - Product Backlog
  - Sprint Backlog
  - Burndown Charts

- **Scrum**
Keys to high performance Scrum ...

Daily Meeting

Sprint

READY

IMPEDEMENTS

DONE

Value

Velocity

© Jeff Sutherland 1993-2009
DONE - the key to doubling performance

The best data in the world on doubling performance by focusing on DONE at the end of a Sprint comes from a CMMI 5 company.

Hundreds of teams run the same process and they all double productivity and cut defects by 40%.

All Scrum teams can do this easily (if they remove impediments)
READY - the key to the second doubling of performance

The Product Owner can easily double the velocity of a Scrum team by getting Product Backlog to a high READY state.

READY state can be measured by the process efficiency of story execution.

When you double process efficiency you will be running at four times waterfall performance.
SELF-ORGANIZATION - the third doubling

- Individual self-organizes work
- Team self-organizes around goals
- Architecture self-organizes around working code
- Product emerges through iterative adaptation
- Collaborative approach as opposed to authoritative approach
- Flat organizational structure
# Russian vs. Dutch Velocity

## Distributed/outsourced teams

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></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/ Mon</td>
<td>15.3</td>
<td>15.1</td>
</tr>
</tbody>
</table>

1. M. Cohn, User Stories Applied for Agile Development. Addison-Wesley, 2004
Benchmarked Out of the Box

Scrum looked at projects off the chart


Scrum: A Pattern Language for Hyperproductive Software Development


Every team can achieve hyperproductivity


C. Jakobsen and J. Sutherland, "Scrum and CMMI – Going from Good to Great: are you ready-ready to be done-done?," in Agile 2009, Chicago, 2009.
"FÅ GJORT DOBBELT SÅ MYE TIL HALVE PRISEN!"
Going from Good to Great with Scrum
Are you READY to be DONE?

Carsten Ruseng Jakobsen and Jeff Sutherland

Carsten.Ruseng.Jakobsen@systematic.com, jeff@scruminc.com

Monday, August 24, 2009
Systematic Experience Reports

C. Jakobsen and J. Sutherland, "Scrum and CMMI — Going from Good to Great: are you ready-ready to be done-done?," in Agile 2009, Chicago, 2009.


How can we systematically go hyperproductive?

Systematic Software Engineering A/S

- Established in 1985 and now Denmark’s largest privately-owned software and systems company
- **500+ employees:** 71% hold a MSc or PhD in software engineering
- High employee satisfaction – attractive workplace for ambitious software engineers
- Dun & Bradstreet credit rating: **AAA**

- **CMMI Level 5 and ISO 9001:2000 and AQAP 2110 + 150**
- **Supplier of products and projects to more than 27 countries, export share is 60%**
Directive from Strategic Planning Session in summer 2005:
Future Improvements should be primarily based on Lean
Projects combining agile methods with CMMI are more successful in producing higher quality software that more effectively meets customer needs at a faster pace.

- Systematic Software Engineering works at CMMI level 5 and uses Lean product development as a driver for optimizing software processes. Valuable experience has been gained by combining Agile practices from Scrum with CMMI.

Early pilot projects at Systematic showed productivity on Scrum teams almost twice that of traditional teams

- Other projects that demonstrated a story based test driven approach to software development reduced defects found during final test by 40%.

We assert that Scrum and CMMI together bring a more powerful combination of adaptability and predictability to the marketplace than either one alone and suggest how other companies can combine them.
Customers demand more complexity and more speed

- Management of complexity requires process discipline, and management of increased speed of change requires adaptability.
- CMMI primarily provides process discipline and Scrum enhances adaptability.
- Is it possible to integrate CMMI and agile practices like Scrum to achieve the benefits from both – or even more?
Scrum implements Lean

**P1** Eliminate waste
- Tool 1: Eliminate Waste
- Tool 2: Value Stream Mapping
- Tool 3: Feedback
- Tool 4: Iterations
- Tool 5: Synchronization
- Tool 6: Set-based development

**P2** Amplify Learning
- Tool 7: Options Thinking
- Tool 8: Latest Responsible Moment
- Tool 9: Decision Making

**P3** Responsible decisions
- Tool 10: Pull
- Tool 11: Queue Theory

**P4** Fast Delivery
- Tool 12: Cost of DealY

**P5** Empower team
- Tool 13: Self-determination
- Tool 14: Motivation
- Tool 15: Leadership
- Tool 16: Expertise

**P6** Build integrity in
- Tool 17: Perceived Integritet
- Tool 18: Conceptual Integritet
- Tool 19: Refactoring
- Tool 20: Test

**P7** See the Whole
- Tool 21: Measures
- Tool 22: Contracts

© Jeff Sutherland 1993-2009
Systematic’s new model for Lean SW development

Tools can be divided into three dimensions

<table>
<thead>
<tr>
<th>Value</th>
<th>Flow</th>
<th>Pull</th>
<th>Perfection</th>
</tr>
</thead>
<tbody>
<tr>
<td>P6 Build Integrity In T19 Refactoring T20 Test</td>
<td>P2 Amplify learning T5 Synchronization T4 Iterations</td>
<td>P2 Amplify Learning T3 Feedback T6 Setbased development</td>
<td>P6 Build Integrity In T18 Conceptual integrity T17 Perceived integrity</td>
</tr>
<tr>
<td>P1 Eliminate Waste</td>
<td>P4 Fast Delivery T11 Queuing Theory T12 Cost of Delay</td>
<td>P7 See the whole T22 Contracts T21 Measures T10 Pull</td>
<td>P3 Decide in latest Responsible moment T7 Options thinking T8 Latest responsible Moment T9 Beslutningstaging</td>
</tr>
<tr>
<td>P5 Empower team T16 Expertise</td>
<td>P5 Empower team T14 Motivation</td>
<td>P5 Empower team T15 Leadership</td>
<td>P5 Empower team T13 Self-determination</td>
</tr>
</tbody>
</table>

These are thinking tools – Projects and employees knows best how to transform them
Lean Thinking:
What specs do developers actually need?

What they get

What they need

PDR

Missing stuff

Reformatted PDR
Toyota Way - Iterative & Incremental

What the Customer Needs

What the Customer Gets

Features

Time

Contract

© Jeff Sutherland 1993-2009
Systematic Pilot Projects

- Selected projects were asked if they would like to pilot improved processes
- Project staff were trained in the Lean mindset
- Scrum and early testing based on story-based development were selected. The pilots were planned and completed.
- The result of the pilots were two-fold
  - it confirmed the general idea of using Lean mindset as source for identification of new improvements
  - it provided two specific successful improvements showing how agile methods can be adopted while maintaining CMMI compliance.
Systematic Pilot – Small Project

First pilot was initiated on a request for proposal
- Systematic inspired by Lean principles suggested a delivery plan with bi-weekly deliveries
- Stated explicit expectations to customer involvement and feedback.
- The project had a team size of 4 and built software for a customer in the Danish Government.

Key reasons for Systematic award:
- commitment to deliver working code bi-weekly
- provided a very transparent process to the customer.
Small Project Success Factors

- Delivery plan and customer involvement resulted in early detection of technology issues.
  - Had a traditional approach been used these issues would have been identified much later with negative impacts on cost and schedule performance.

- Productivity of small project was at the expected level compared to the productivity performance baseline for small projects.

- Another small project with a team size of 5 working for a Defense customer using Scrum shows a similar productivity and the same indicators of high quality and customer satisfaction.
Pilot of Larger Project

- Team of 10 worked on a military messaging system.
  - This project was inspired from the Lean thinking tool “Build Integrity In” to investigate how to do early test, and as a result they invented a story based approach to early testing in software development.
  - The name “Story based” development was inspired from XP, but the approach included new aspects like: short incremental contributions, inspections and was feature driven.

- The idea of story-based development was to subdivide features of work, typically estimated to hundreds of hours of work into smaller stories of 20-40 hours of work.

- The implementation of a story followed a new procedure,
  - the first activity would be to decide how the story could be tested before any code was written.
  - This test could then be used as the exit criteria for implementation of the story.
New Approach to Testing Reduced Defects by 38%

- Many benefits from story-based development were immediately apparent.
  - The combination of a good definition of when a story was complete, and early incremental testing of the features, provided a very precise overview of status and progress for both team and other stakeholders.

- Developing a series of small stories rather than parts of a big feature is more satisfactory
  - Creates a better focus on completing a feature until it fulfills all the criteria for being “done”.

- This project finished early, and reduced the number of coding defects in final test by 38% compared to previous processes.
Another Large Project

Team of 19 working on a module to a electronic patient record system, also worked with early testing.

They ensured that test activities were integrated into development, with a strong focus on “seeing the whole” and understanding how the solution fit into the customers domain.

For each week the project defined a goal to be achieved. The project ensured that test and domain specialists were co-located with the developers.

- This caused discussion and reflection between testers, developers, user experience engineers and software architects, before or very early in the development of new functionality.

As a consequence the amount of remaining coding defects in final test were reduced by 42% compared to previous processes.
Conclusions from Larger Projects

- Test activities should be an integrated activity throughout the projects lifetime.
  Scrum inherently supports this, through cross-functional teams and frequent deliveries to the customer.

- Story-based software development method should be the default recommended method for software development in projects.

- This strategy is commonly known as “Acceptance Test Drive Development”
Challenges:

Developer’s self-interest

- It is against the developer’s self-interest to optimize for team performance
- They will usually try to optimize for personal efficiency or personal interest and generate repeated failure
- ScrumMaster must coach team to move beyond mediocrity
### Typical crash and burn Sprint

#### 3 roles
- Product owner
- Scrum master
- Team

#### 3 artifacts
- Product backlog
- Sprint-backlog
- Sprint burndown

#### 3 activities
- Sprint planning
- Daily scrum
- Sprint review
  - Demo
  - Retrospective

**WAIT A SEC**

How is that burndown calculated?

---

Source: Henrik Kniberg

© Jeff Sutherland 1993-2009
Properly executed Sprint

3 roles
- Product owner
- Scrum master
- Team

3 artifacts
- Product backlog
- Sprint backlog
- Sprint burndown

3 activities
- Sprint planning
- Daily scrum
- Sprint review
  - Demo
  - Retrospective

Source: Henrik Kniberg
Linear Scalability of Scrum Projects

Systematic adoption of Scrum and story based development

- Process Action Teams (PATs) were formed to integrate the experience and knowledge gained from the pilots, into the processes shared by all projects in the organization.

- The largest change to project planning is that features and work are planned in *sufficient detail* as opposed to a complete initial detailed analysis.
  - Result is a Scrum Product Backlog with a complete prioritized list of features/work for the project.
  - All features have a qualified estimate, established with a documented process and through the use of historical data, but the granularity of the features increase as the priority falls.
  - The uncertainty that remains is handled through risk management activities.

- The primary change to project execution processes, is to integrate Scrum as method for completing small iterations (Sprints), on a selected subset of the work with highest priority.
Published experiences with "rework"

Source: Krasner & Houston, CrossTalk, Nov 1998
Diaz & King, CrossTalk, Mar 2002
Rework at Systematic

<table>
<thead>
<tr>
<th>Quarter</th>
<th>2005 Q2</th>
<th>2005 Q3</th>
<th>2005 Q4</th>
<th>2006 Q1</th>
<th>2006 Q2</th>
<th>2006 Q3</th>
<th>2006 Q4</th>
<th>2007 Q1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rework</td>
<td>9.8%</td>
<td>6.9%</td>
<td>6.4%</td>
<td>8.3%</td>
<td>6.0%</td>
<td>7.6%</td>
<td>6.8%</td>
<td>4.7%</td>
</tr>
</tbody>
</table>

Monday, August 24, 2009
Scrum applied to CMMI Level 5 company

- 6 month results

<table>
<thead>
<tr>
<th></th>
<th>Project effort</th>
<th>CMMI 1</th>
<th>CMMI 5</th>
<th>CMMI-5 SCRUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work</td>
<td>50 %</td>
<td>50 %</td>
<td>50 %</td>
<td>25 %</td>
</tr>
<tr>
<td>Rework</td>
<td>50 %</td>
<td>10 %</td>
<td>6 %</td>
<td>4 %</td>
</tr>
<tr>
<td>Process focus</td>
<td>100 %</td>
<td>69 %</td>
<td>35 %</td>
<td>6 %</td>
</tr>
</tbody>
</table>
Faser i et projekts livscyklus

CMMI: *Project Planning*
- SG1: Establish Estimates
- SG2: Develop a Project Plan
- SG3: Obtain Commitment to the Plan

CMMI: *Project Monitor and Control*
- SG1: Monitor Project Against Plan
- SG2: Manage Corrective Actions to Closure

SCRUM: *Create Product Backlog*
- Define backlog items
- Establish Estimates
- Prioritize backlog items
- Identify dependencies

SCRUM: *Create Sprint Backlog*
- Monitor progress against sprint plan
- Remove impediments

Definition & Planning | Implementation | Launch & Closeout

Sprints
SCRUM and PDP-Common
Impediments

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

Examples on causes:

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

Root cause analysis of time to fix automatically generates ScrumMaster’s impediment list.
Systematic CMMI 5 Analysis
First six months of Scrum

- 80% reduction in planning and documentation costs
- 40% reduction in defects
- 50% reduction in rework
- 100% increase in overall productivity
- Systematic decided to change CMMI Level 5 process to make Scrum the default mode of project management
- When waterfall project management is required, they are now need to be contracted for twice the price of Scrum projects
  - Required by some defense and healthcare agencies
  - Results are lower business value
  - Lower customer satisfaction
  - Lower quality
  - Twice the cost

Next steps for Systematic

Assure all teams run at double performance and 40% fewer defects while maintaining CMMI 5 compliance

Use Function Point Analysis to improve data collection capability to research quality

Show a second doubling of performance of teams based on Function Point Analysis by focusing on READY state of Product Backlog
Learn and improve from success

<table>
<thead>
<tr>
<th>Project</th>
<th>Performance</th>
<th>Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>192%</td>
<td>18%</td>
</tr>
<tr>
<td>B</td>
<td>76%</td>
<td>64%</td>
</tr>
<tr>
<td>C</td>
<td>86%</td>
<td>92%</td>
</tr>
<tr>
<td>D</td>
<td>54%</td>
<td>50%</td>
</tr>
<tr>
<td>E</td>
<td>258%</td>
<td>48%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Project</th>
<th>Performance</th>
<th>Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>140%</td>
<td>44%</td>
</tr>
<tr>
<td>B</td>
<td>74%</td>
<td>64%</td>
</tr>
<tr>
<td>C</td>
<td>81%</td>
<td>83%</td>
</tr>
<tr>
<td>D</td>
<td>70%</td>
<td>59%</td>
</tr>
<tr>
<td>E</td>
<td>365%</td>
<td>75%</td>
</tr>
</tbody>
</table>

Performance data from pilot on use of function points were collected. Data are subject to high variance and uncertainty, because it is a new technology used for the first time – However ...

Data could indicate that A and E have good performance, which is also the gut feeling by senior management.

Investigate possible success and practices behind it
Projects investigated

8 interviews of 1 hour with project members

• Questions for project A and E teams:
  • Why high performance?
    • We spent time to prepare and groom our product backlog
    • We ensure that tasks for sprint Planning are READY
  • How can other projects copy your success?
    • We document our practice in a READY checklist
  • Ready state determines process efficiency of a story
    • If story takes 1 ideal day of work and takes 4 calendar days to complete, process efficiency is 25%. We call this FLOW.

• The story of project A ...
First scrum...

13/12-2007 – 22/1-2008 – Flow: 23%

- Buildserver and test established
- Physical Scrum Board established
- Basic Scrum rules ok
- Features not ready
Starting to insist on “well defined”

30/1-2008 – 27/2-2008 – Flow: 48 %

- Most features for this sprint is prepared
- But Product Backlog grooming cycle is behind
Team continues to say NO if task not READY

3/3 - 2008 – 9/4-2008 – Flow: 57%

- Team insisted on only allocating ready stories
- Forced feature preparation concurrent to sprint
Result

Flow increased from appr. 30% to appr. 60% in 2008 for Project A

Flow for stories in IS 01/12/1997 to 15/12/2008 for Project A

<table>
<thead>
<tr>
<th>Sample ID in Data sheet</th>
</tr>
</thead>
<tbody>
<tr>
<td>85 79 82 84 93 91 110 98 3 5 9 22 30 34 38 44 47 50 55 58 62 70 133 132 135 130 137</td>
</tr>
</tbody>
</table>

Flow for stories in IS 01/12/1997 to 15/12/2008 for Project A
Effect

When work allocated to sprint is READY, flow and stability is achieved

Flow

Objective: 60%

Actual effort

Objective: 50h

Monday, August 24, 2009
READY means stable sprints

18/11-2008 – 14/1-2009 – Flow: 54 %

- Execution of Sprint is good
- Stories were READY when added to sprint
- Stories were DONE when delivered
- Team delivered to commitment!
- No stories were taken out of sprint
Feature READY checklist

- Ensure that features are prepared properly before they are decomposed into stories that are committed to a sprint

- Preparation through states:
  - Prepare Feature for Commitment
  - Clarify Feature for Development
  - Prepare Feature for Implementation
Continue to improve

Identifying root causes to stories not achieving desired flow (03/2009)

• READY removed a major impediment
  • Removed disruptions and waste caused by issues being clarified with customer or other

• Data shows more impediments exist:
  • Root causes for 10 stories with flow < 40%
    • Developer was shared between two projects
    • Final inspection completed too late due to support
    • Interrupted by fixing problems with build environment
    • Work on story stopped due to vacation (commitment?)
    • Lead developers typically assist on multiple stories

• It’s about focus, commitment and how to share knowledge
Understanding Scrum success

READY and DONE is simple to understand but hard to do

Key is a proper balance between planning and execution activities
The Systematic Scrum model

Disciplines:

- Clarify features
- Establish project environment and initial PBL
- Daily Scrum
- Automated test
- Continuous Integration
- Remove impediments
- Verify sprint delivery

Sprint Zero

Value

READY

IMPEDIMENTS

DONE

DONE

Sprint

Feature CHK ✓

Story CHK ✓

Velocity

Monday, August 24, 2009
Lessons learned

- **Make features READY before they are DONE**
  - Do not allow a feature to be included in sprint unless it is READY
  - Simple concept, depends on discipline and creates stability in sprint
  - Prepare PBL with at least same speed as sprints

- **Product Owner tasks are not part of sprint plan**
  - Clarification is a disruptive activity by nature
  - Make clear arrangements for how Product Owner activities are supported by team

- **Team both deliver sprints and support Product Owner**
  - Balance is achieved by first ensuring that features and stories are prepared sufficiently using these objectives
    - A feature can be implemented by team in one sprint (<600h)
    - A story can be implemented by 1-2 people within 1-2 days (<50h)
  - Team proactively participated in workshops preparing sprint planning

- **Systematically remove impediments**
  - Sprint retrospective at the core
  - Measure and analyze data, e.g. fix-time for broken builds or flow