

PRACTICAL ROADMAP TO GREAT SCRUM

SYSTEMATICALLY ACHIEVING HYPERPRODUCTIVITY

With help from Google, Yahoo, Microsoft, IBM, Oracle, MySpace, Adobe, GE, Siemens, Disney Animation, BellSouth, Nortel, GSI Commerce, Ulticom, Palm, St. Jude Medical, DigiChart, RosettaStone, Healthwise, Sony/Ericsson, Accenture, Trifork, Systematic Software Engineering, Exigen Services, SirsiDynix, Softhouse, Philips, Barclays Global Investors, Constant Contact, Wellogic, Inova Solutions, Medco, Saxo Bank, Xebia, Insight.com, SolutionsIQ, Crisp, Johns Hopkins Applied Physics Laboratory, Unitarian Universalist Association, Motley Fool, Planon, FinnTech, OpenView Venture Partners, Jyske Bank, BEC, Camp Scrum, DotWay AB, Ultimate Software, Scrum Training Institute, AtTask, Intronis, Version One, OpenView Labs, Central Desktop, Open-E, Zmags, eEye, Reality Digital, DST, Booz Allen Hamilton, Scrum Alliance, Fortis, DIPS, Program UtVikling, Sulake, TietoEnator, Gilb.com, WebGuide Partner, Emergn, NSB (Norwegian Railway), Danske Bank, Pegasystems, Kanban Marketing, accelare



Jeff Sutherland, Ph.D.

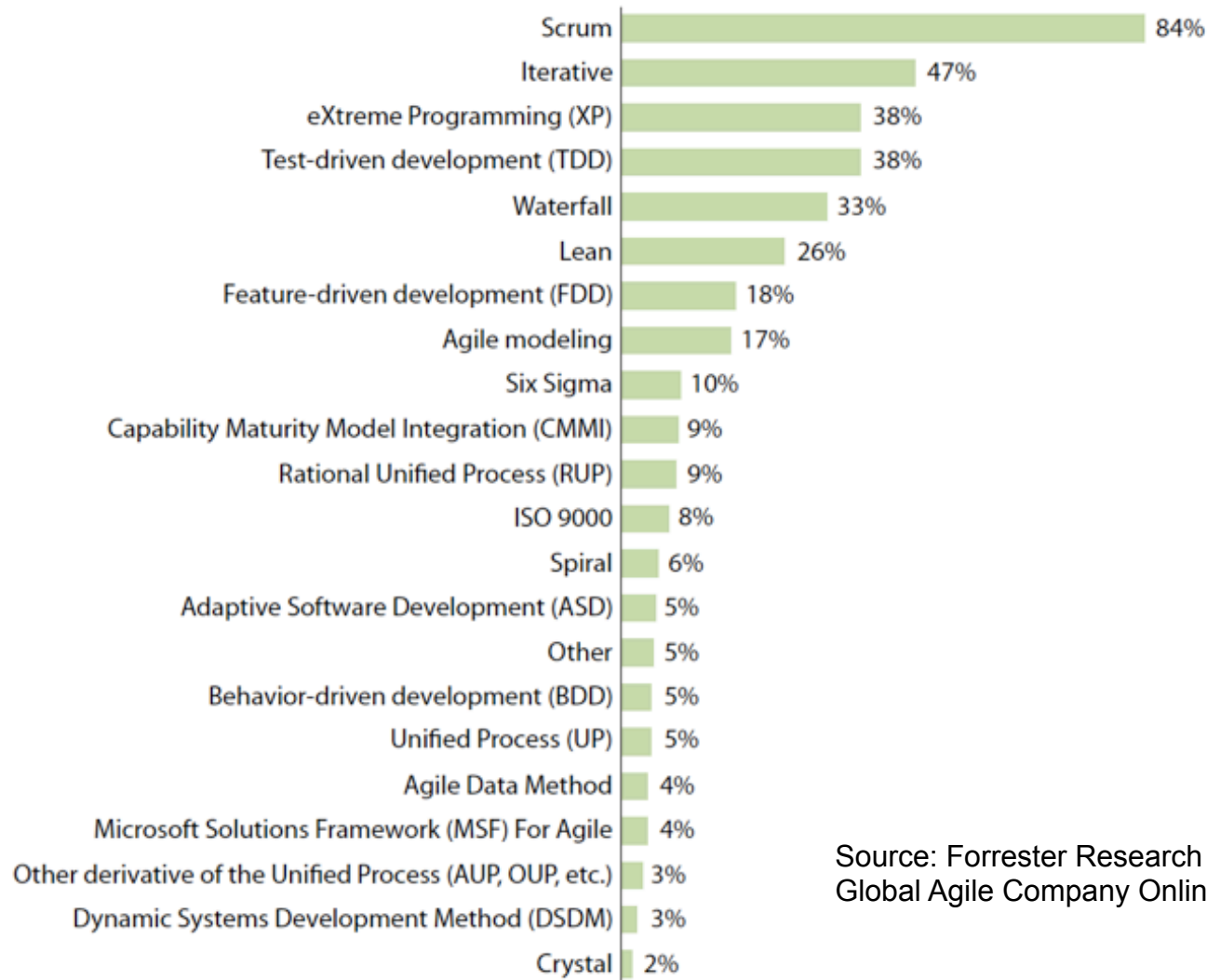


- **Chairman, Scrum Training Institute**
 - **CEO Scrum, Inc. and Senior Advisor, OpenView Venture Partners**
 - **Agile coach for OpenView Venture Partners portfolio companies**
 - **Chief engineer for 11 software companies**
 - **Created first Scrum at Easel Corp. in 1993. Rolled out Scrum in next 5 companies**
 - **Achieved hyperproductive state in all companies. Signatory of Agile Manifesto and founder of Agile Alliance**
-
- <http://jeffsutherland.com/scrum>
 - jeff@scruminc.com



© Jeff Sutherland 1993-2009

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)

© Jeff Sutherland 1993-2009

Venture Capital Strategy: Follow the money

- **Invest only in Agile projects**
 - One hyperproductive company out of 10 might meet investment goals for a venture group
 - Two or more hyperproductive could alter the market
- **Invest only in market leading, industry standard processes – this means Scrum and XP**
- **Ensure teams implement basic Scrum practices**
 - Everyone passes the Nokia test
 - Management held accountable at Board level for removing impediments
 - Implementation of hyperproductive Scrum

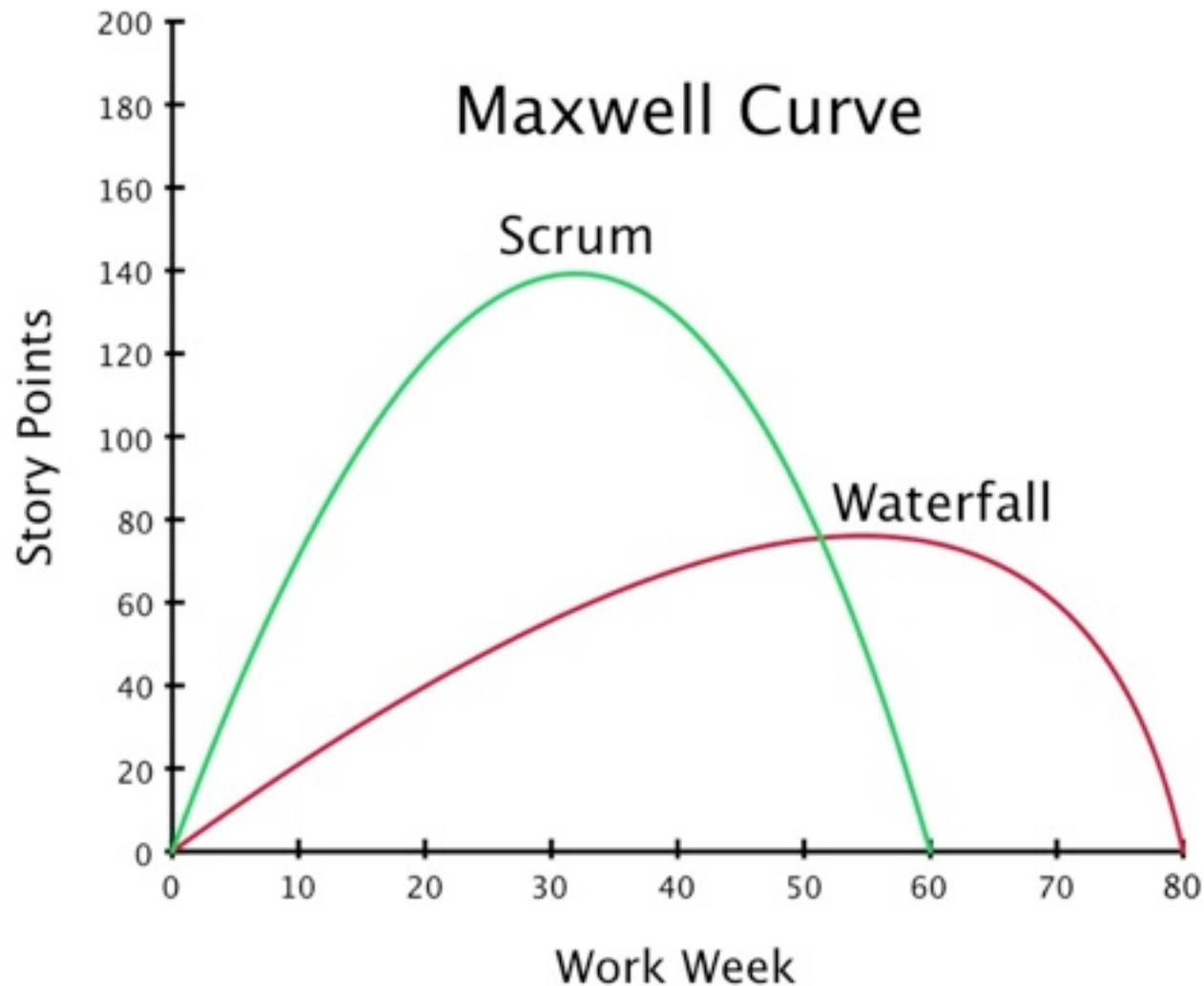


CSM v10.21 © Jeff Sutherland 1993-2009

Basic Truths about Hyperproductive Scrum

- Everyone must be trained in Scrum framework
- Backlog must be READY before taking into Sprint
- Software must be DONE at the end of the Sprint
- Pair immediately if only one person can do a task
- No Multitasking
- Physical Scrum Board
- Short sprints (often 1 week)
- Burn down Story points only
- Everything (including support) is prioritized by PO
- Servant leadership – it's not about you

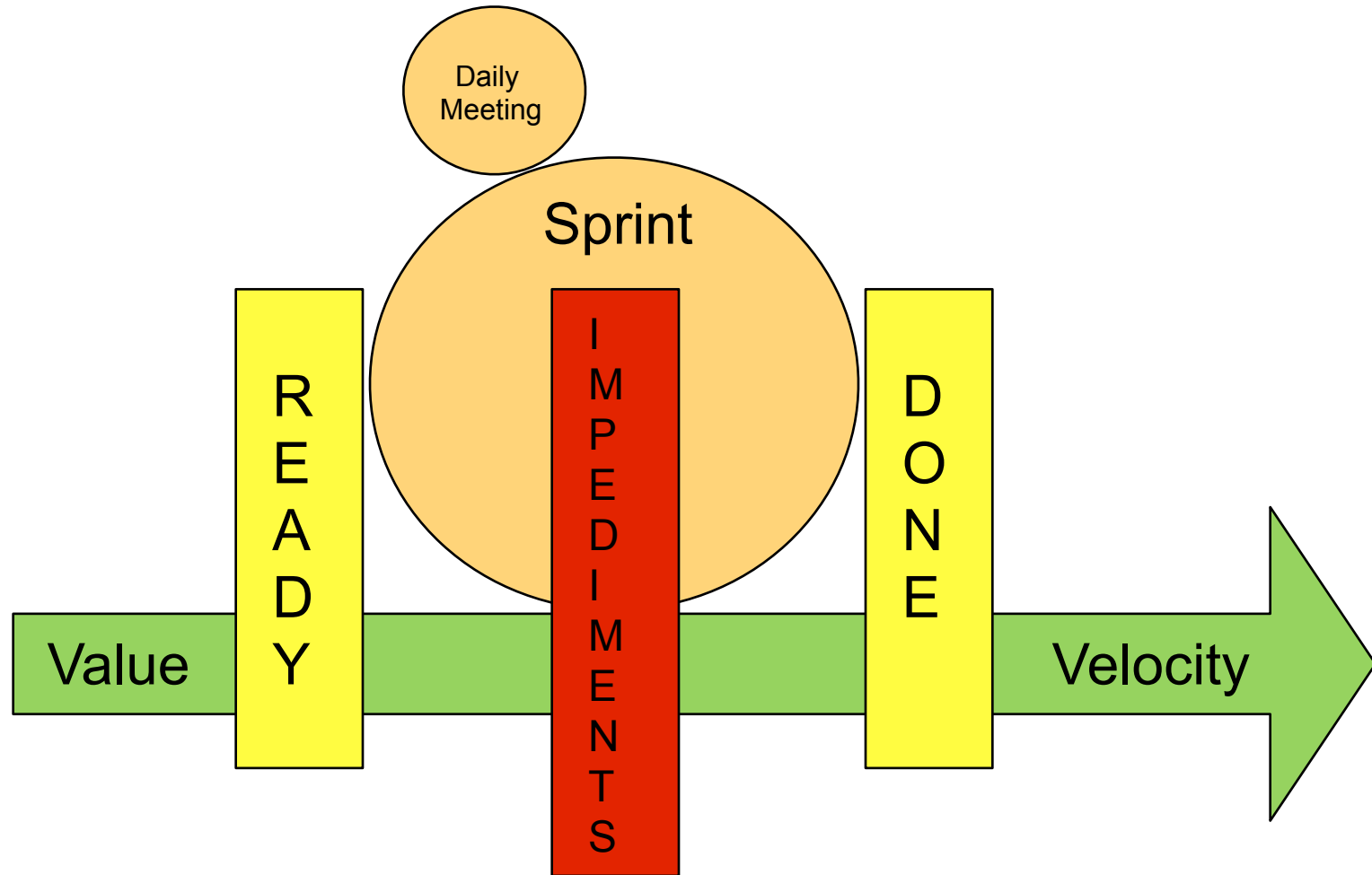
"FÅ GJORT DOBBELT SÅ MYE TIL HALVE PRISEN!"







www.openviewventurepartners.com

© Jeff Sutherland 1993-2009

Keys to high performance Scrum ...



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)**
-  **50% of Scrum teams worldwide don't do this**

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 DONE and double story process efficiency you will be running at four times waterfall performance.**
- **Less than 1% of Scrum teams worldwide do this.**

SELF-ORGANIZATION - the third doubling

- **Individuals self-organize work to maximize team velocity**
- **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**

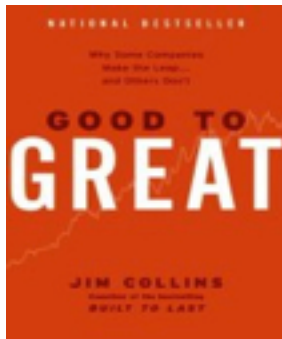
Russian vs. Dutch Velocity

Distributed/outourced teams

	SirsiDynix[2]	Xebia[3]
Person Months	827	125
Lines of Java	671,688	100,000
Function Points	12673	1887
Function Points per Dev/ Mon	15.3	15.1




1. M. Cohn, User Stories Applied for Agile Development. Addison-Wesley, 2004
2. J. Sutherland, A. Viktorov, J. Blount, and N. Puntikov, "Distributed Scrum: Agile Project Management with Outsourced Development Teams," in HICSS'40, Hawaii International Conference on Software Systems, Big Island, Hawaii,
3. J. Sutherland, G. Schoonheim, E. Rustenburg, M. Rijk. Fully Distributed Scrum: The Secret Sauce for Hyperproductive Outsourced Development Teams. Agile 2008, Toronto, Aug 4-8 (submission, preliminary data)

© Jeff Sutherland 1993-2008




Benchmarked Out of the Box



Scrum looked at projects off the chart

-  (IBM Surgical Team) F. P. Brooks, *The Mythical Man Month: Essays on Software Engineering*: Addison-Wesley, 1995.
-  Takeuchi and Nonaka. [The New New Product Development Game](#). Harvard Business Review, 1986
-  J. O. Coplien, "Borland Software Craftsmanship: A New Look at Process, Quality and Productivity," in 5th Annual Borland International Conference, Orlando, FL, 1994.

Scrum: A Pattern Language for Hyperproductive Software Development

-  By M. Beedle, M. Devos, Y. Sharon, K. Schwaber, and J. Sutherland. In *Pattern Languages of Program Design*. vol. 4, N. Harrison, Ed. Boston: Addison-Wesley, 1999, pp. 637-651.

Every team can achieve hyperproductivity

-  J. Sutherland, S. Downey, and B. Granvik, "[Shock Therapy: A Bootstrap for a Hyper-Productive Scrum](#)" in *Agile 2009*, Chicago, 2009.
-  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.



Going from Good to Great with Scrum

Are you READY READY to be DONE DONE?




Carsten Ruseng Jakobsen and Jeff Sutherland

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



Tuesday, October 20, 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.**
-  **C. R. Jakobsen and K. A. Johnson, "Mature Agile with a Twist of CMMI," in Agile 2008, Toronto, 2008.**
-  **J. Sutherland, C. Jakobsen, and K. Johnson, "Scrum and CMMI Level 5: A Magic Potion for Code Warriors!," in Agile 2007, Washington, D.C., 2007.**

Download papers at jeffsutherland.com/scrum
Click on "Jeff Sutherland's Papers"

© Jeff Sutherland 1993-2009

How can we systematically go hyperproductive?



Mission Critical



Systematic Software Engineering A/S

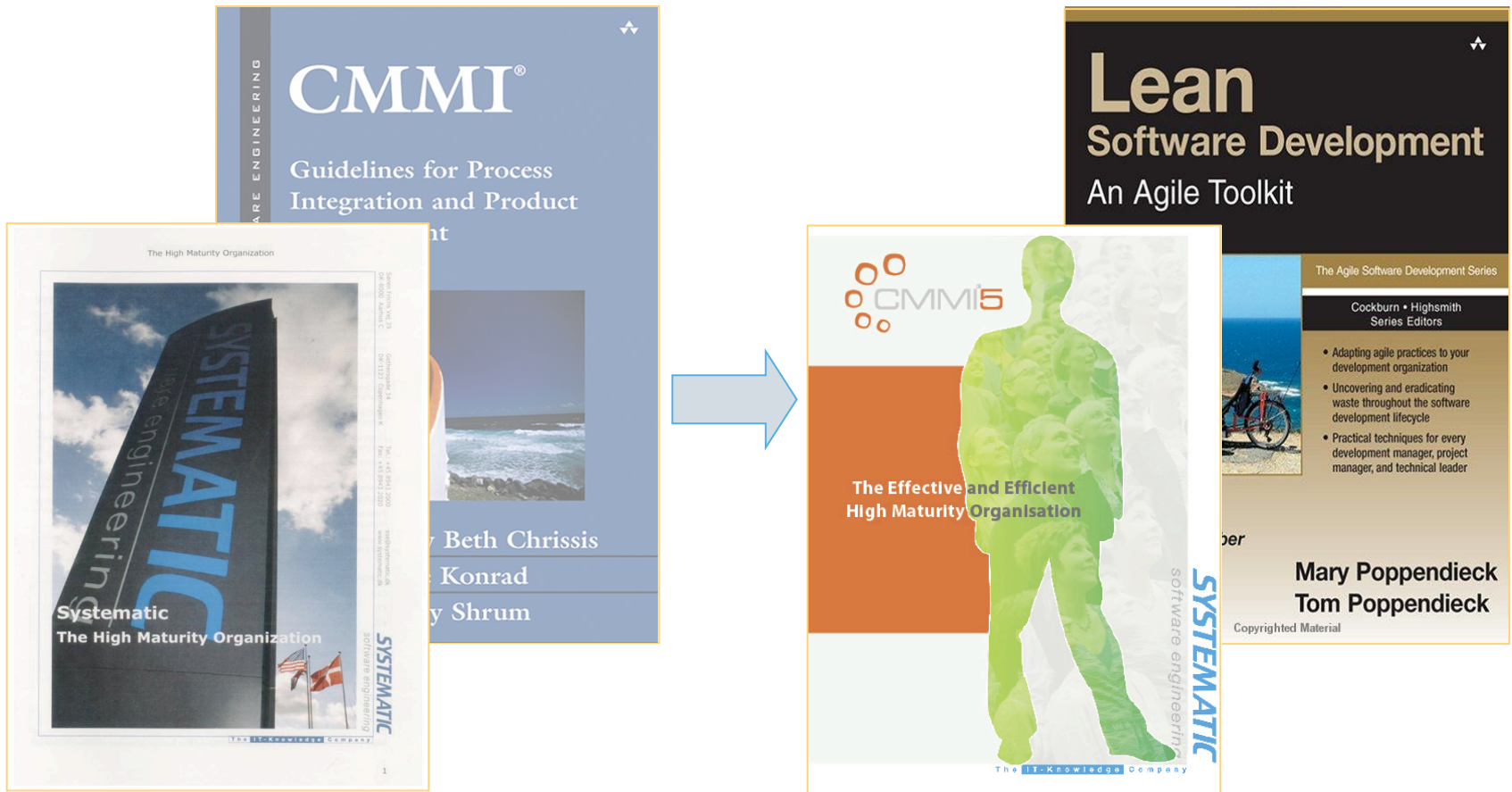
- Established in **1985** and now Denmark's **largest privately-owned** software and systems company
- **500+ employees**; 71% hold a MSc or PhD in software engineering
- High **employee satisfaction** – attractive **workplace** for **ambitious** software engineers
- Dun & Bradstreet credit rating: **AAA**
- **CMMI Maturity Level 5 and ISO 9001:2000 and AQAP 2110 + 150**
- **Supplier of products and projects to more than 27 countries, export share is 60%**

CMMI Background

- **CMMI 1.1 published 2001**
 - CMM sunsetted Dec 2002
- **CMMI 1.2 Aug 2006 - simplification**
- **CMMI 1.3 Nov 2010 - more simplification**
- **CMMI Maturity Level 4**
 - Quantitatively Managed
 - Statistically characterized
- **CMMI Maturity Level 5**
 - Causal Analysis
 - Innovation Deployments

Source: Hillel Glazer

Systematic used Scrum to implement Lean



Directive from Strategic Planning Session in summer 2005:
Future Improvements should be primarily based on Lean

Scrum and CMMI – Agile 2007

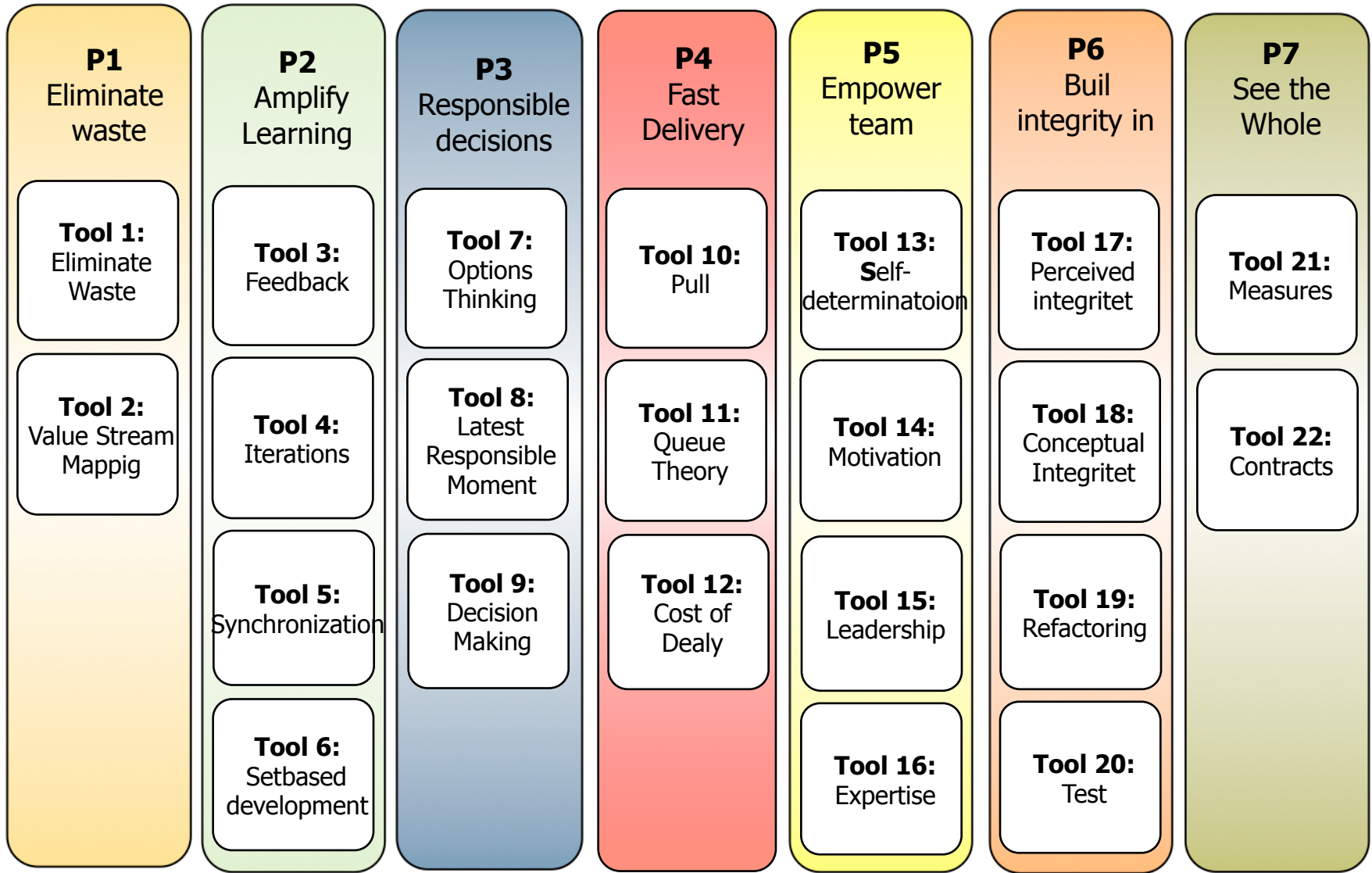
Sutherland, Jacobsen and Johnson

- *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.*

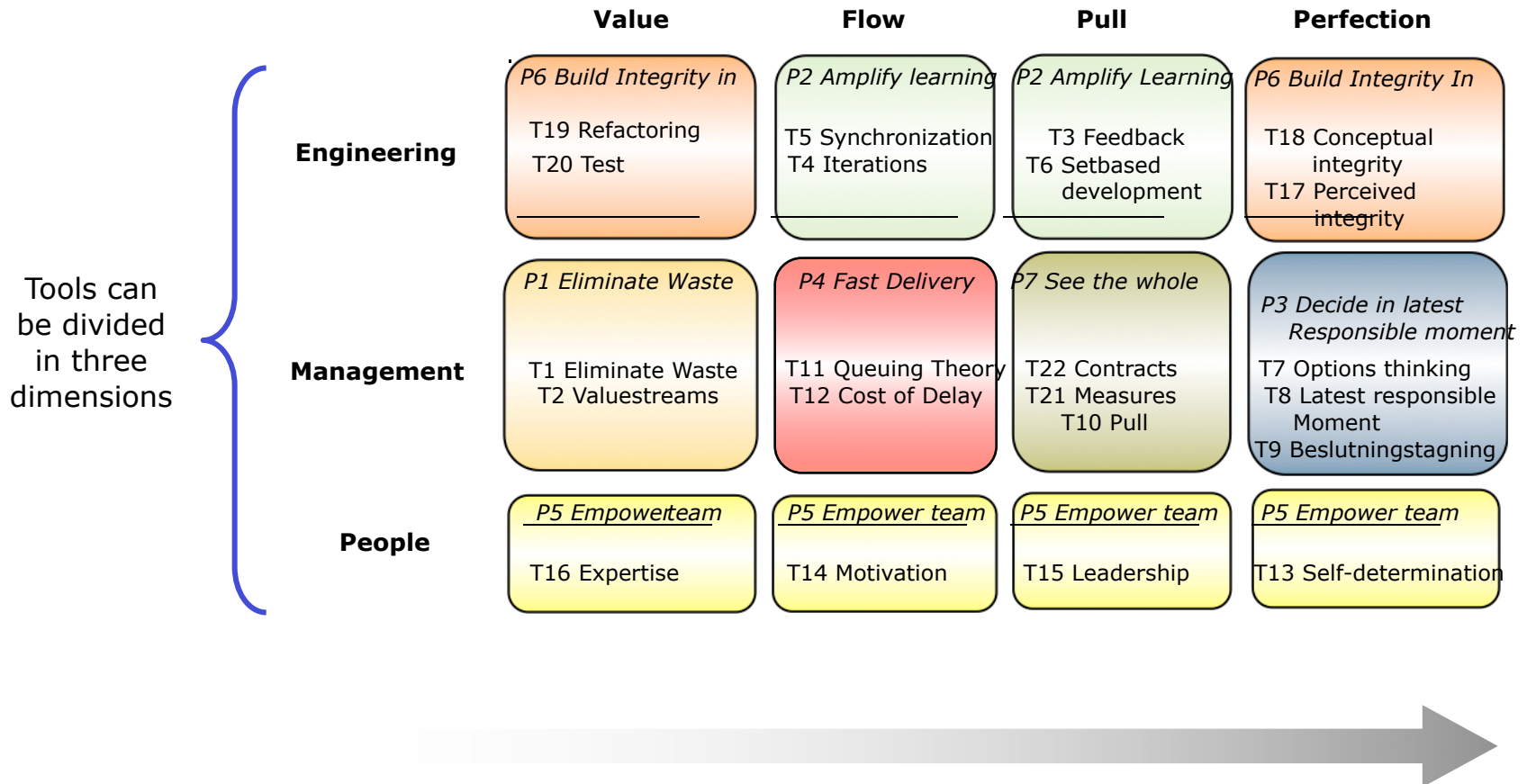
Customers demand more complexity and more speed

- **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



Systematic's new model for Lean SW development



These are thinking tools – Projects and employees knows best how to transform them

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.**

A Larger 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

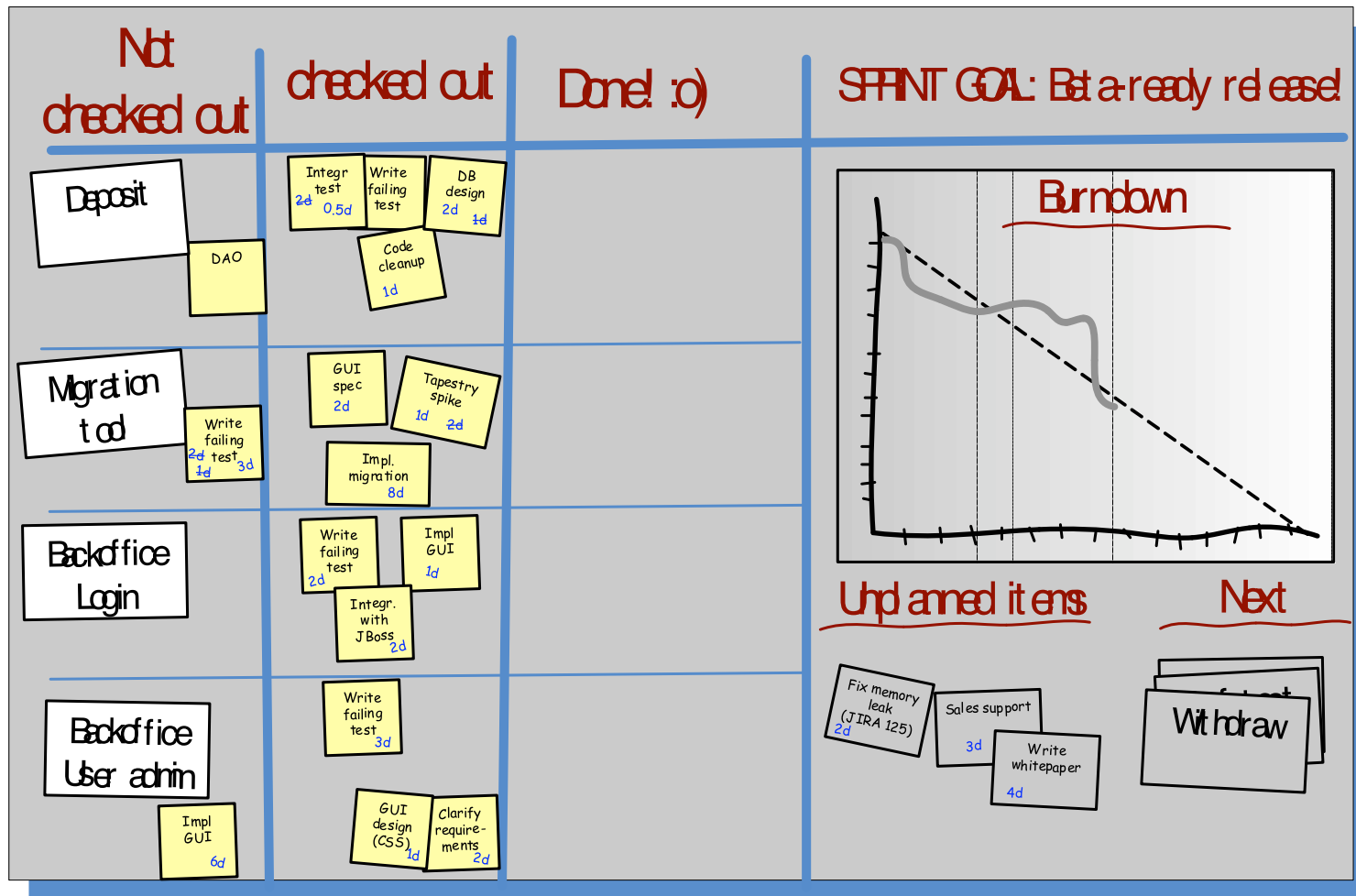
- **Test activities should be an integrated activity through out the projects lifetime.**
 - 🌐 Scrum inherently supports this, through cross-functional teams and frequent deliveries to the customer.
- **Story-based software development method should be the default recommended method for software development 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 often try to optimize for personal efficiency or personal interest and generate repeated Sprint failure**
- **This is not "self-organization"**
- **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

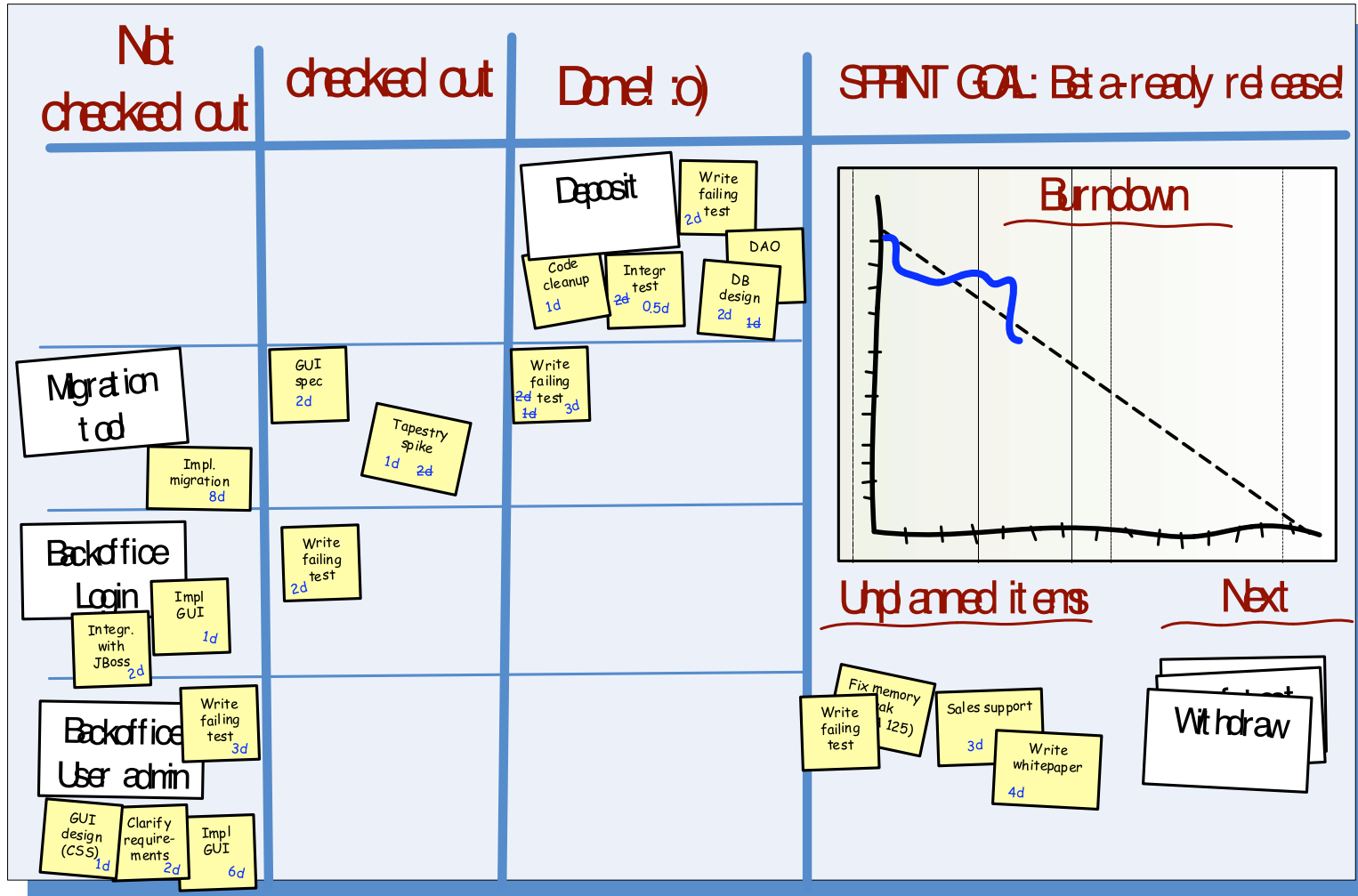
WAIT A SEC

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

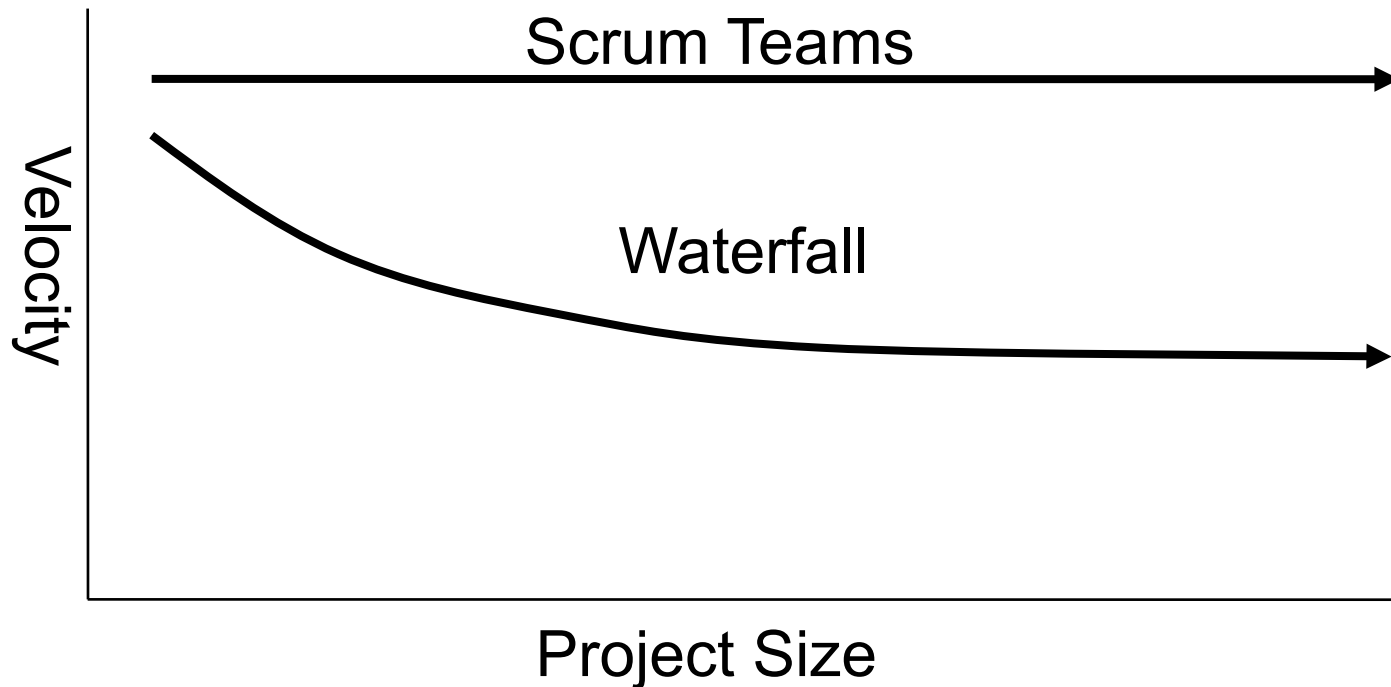
3 activities

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

Source: Henrik Kniberg

© Jeff Sutherland 1993-2009

Systematic noticed linear scalability



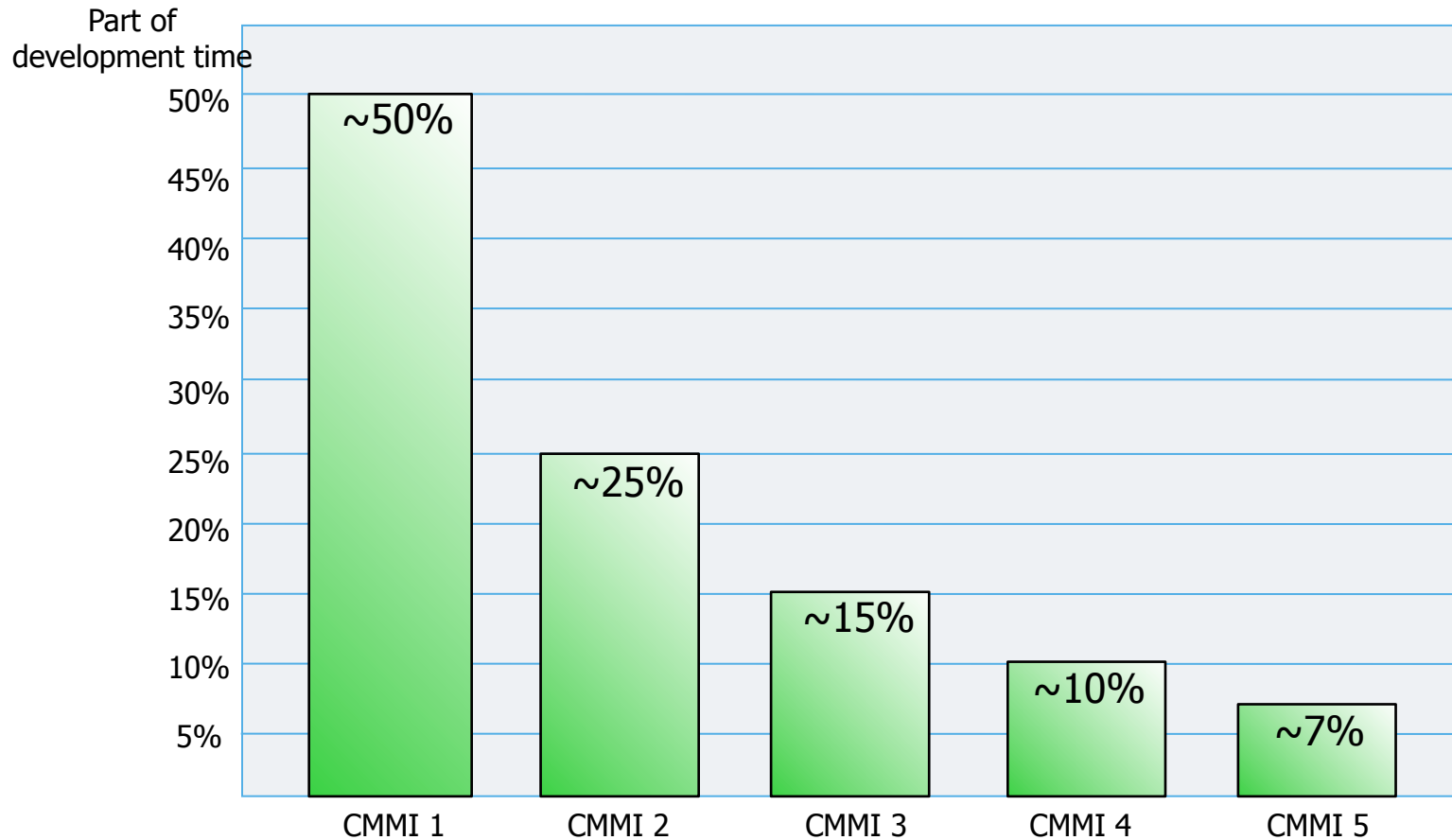
- J. Sutherland, A. Viktorov, J. Blount, and N. Puntikov, "Distributed Scrum: Agile Project Management with Outsourced Development Teams," in HICSS'40, Hawaii International Conference on Software Systems, Big Island, Hawaii, 2007.
- J. Sutherland, C. Jacobson, and K. Johnson, "Scrum and CMMI Level 5: A Magic Potion for Code Warriors!," in Agile 2007, Washington, D.C., 2007.

© Jeff Sutherland 1993-2008

Systematic adoption of Scrum and story based development

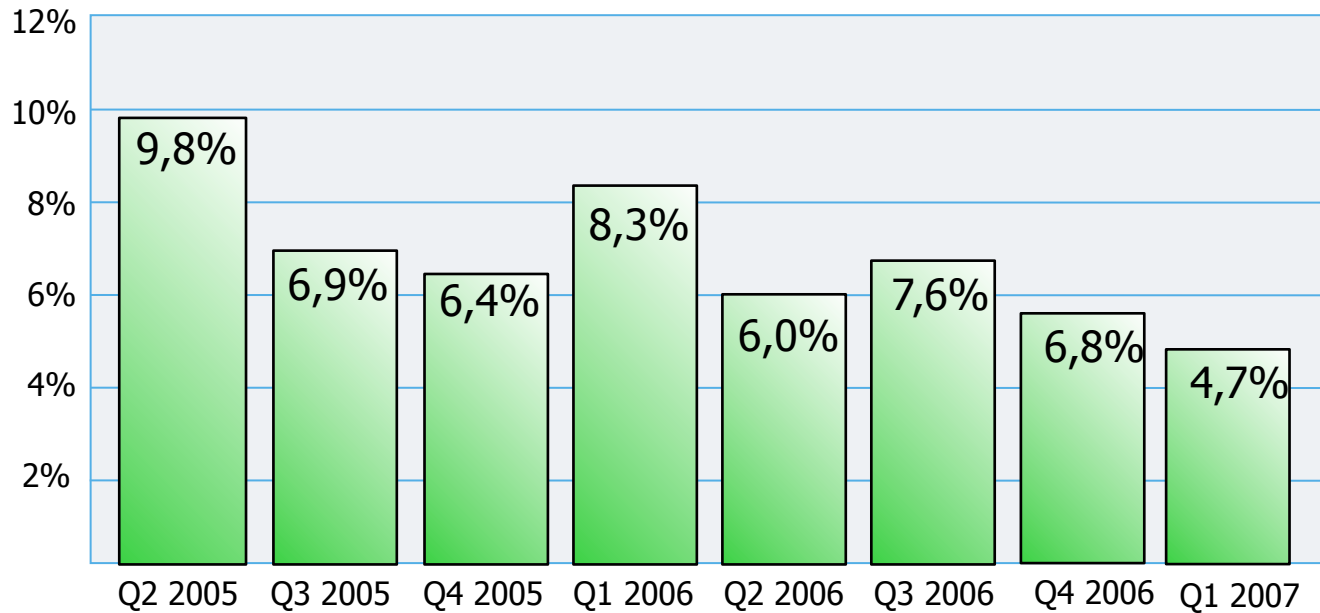
- **Process Action Teams (PATs) were formed to integrate the experience and knowledge gained from the pilots, into the processes shared by all projects in the organization.**
- **The largest change to project planning is that features and work are planned in *sufficient detail* as opposed to a complete 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"



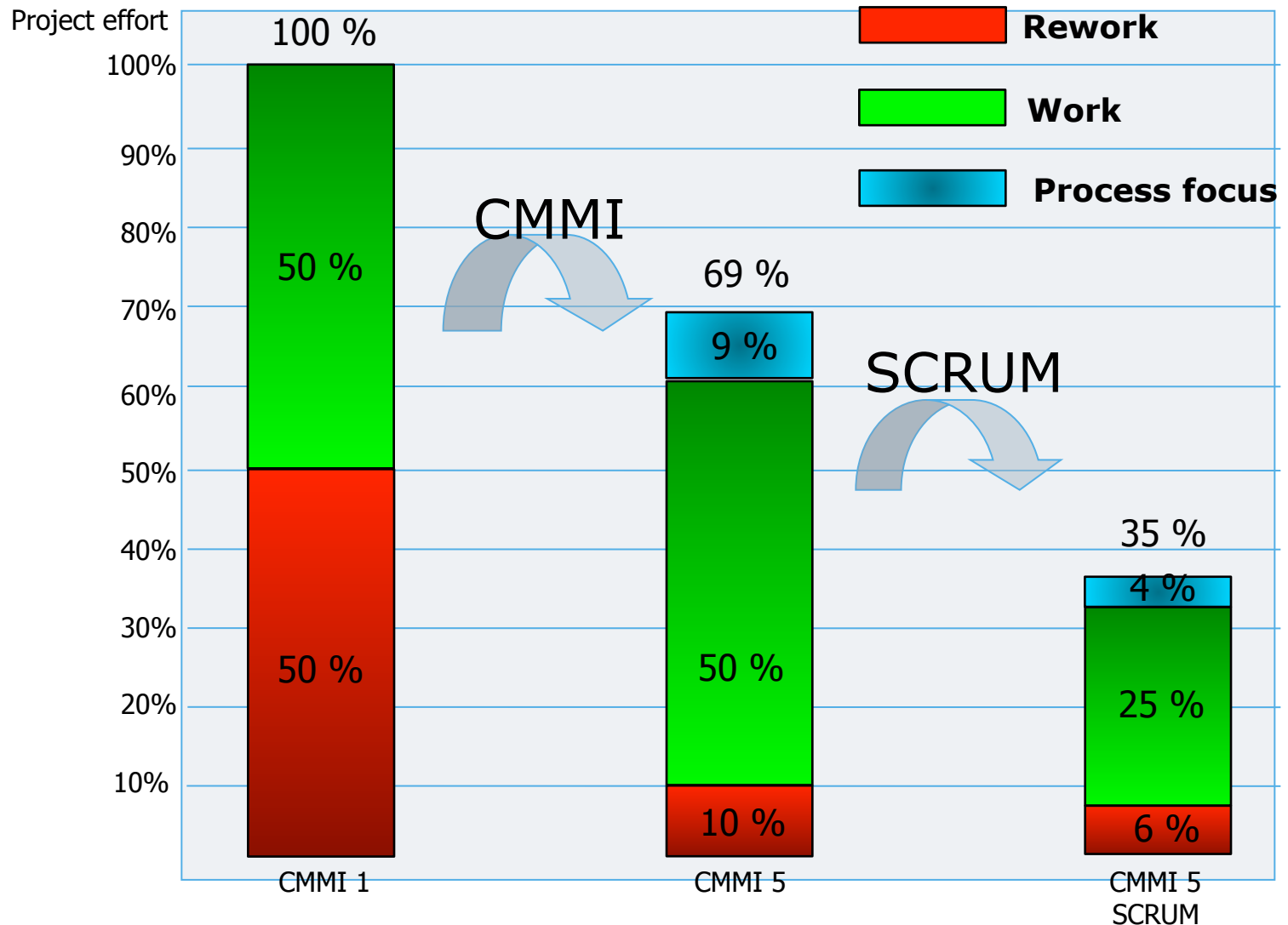
Source: Krasner & Houston, CrossTalk, Nov 1998
Diaz & King, CrossTalk, Mar 2002

Rework at Systematic



Scrum applied to CMMI Level 5 company

— 6 month results



© Jeff Sutherland 1993-2009

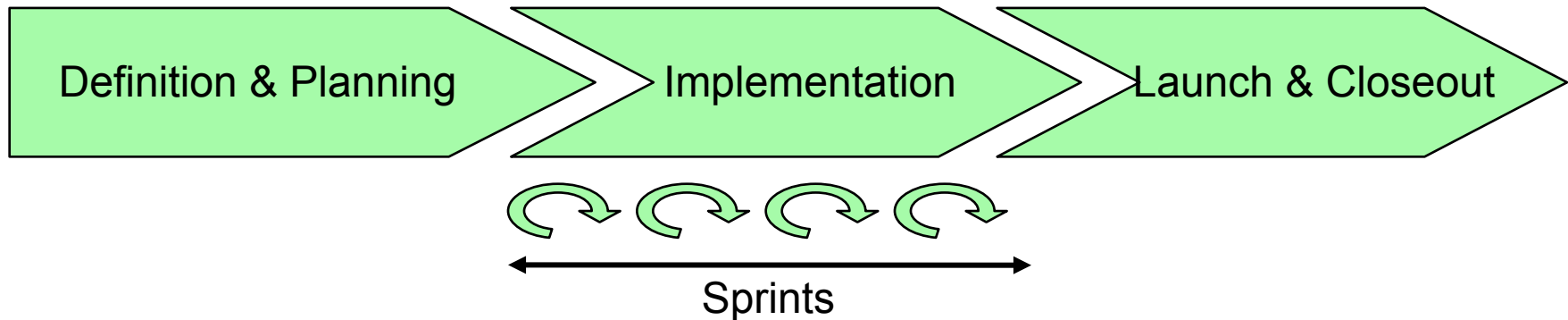
Faser i et projekts livscyklus

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



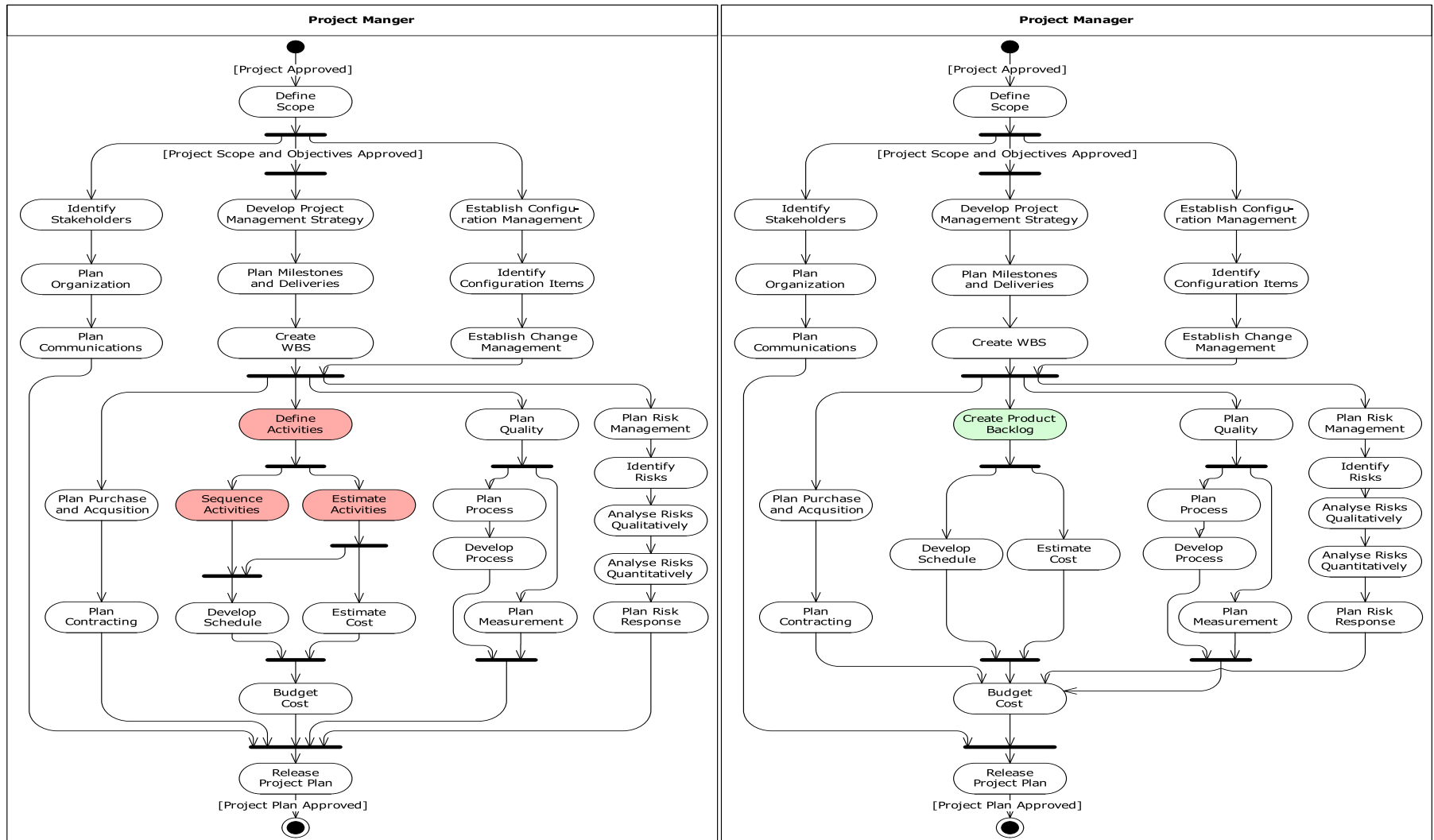
SCRUM: *Create Product Backlog*

Define backlog items
Establish Estimates
Prioritize backlog items
Identify dependencies

SCRUM: *Create Sprint Backlog*

Monitor progress against sprint plan
Remove impediments

SCRUM and PDP-Common

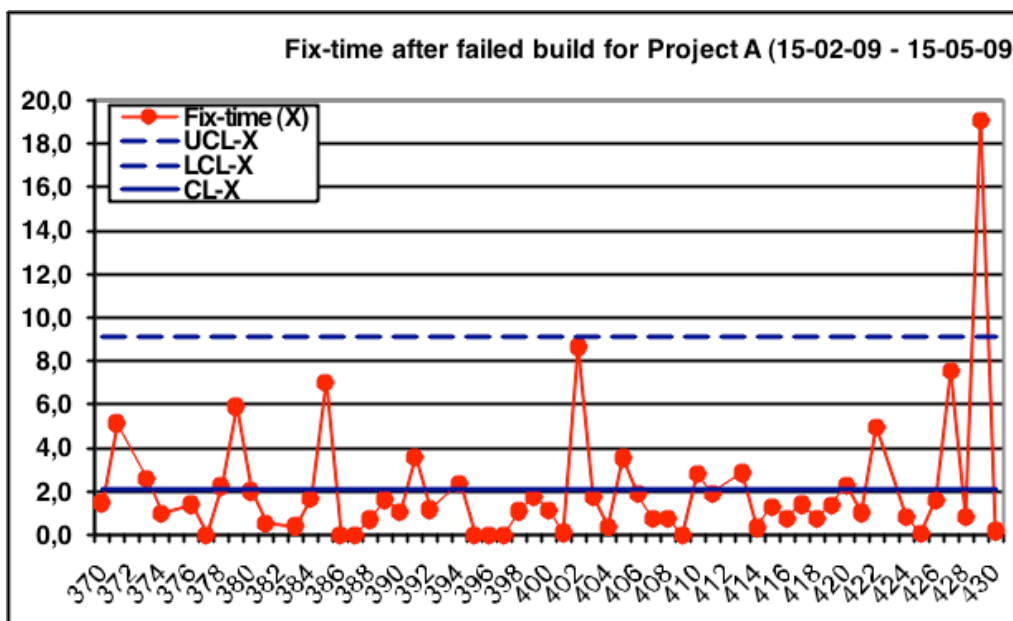


Impediments

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 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



Sutherland, J., C. Jacobson, et al. (2007). Scrum and CMMI Level 5: A Magic Potion for Code Warriors! Agile 2007, Washington, D.C., IEEE.

© Jeff Sutherland 1993-2009

Next steps for Systematic

- **Assure all teams run at 4x performance and 40% fewer defects while maintaining CMMI 5 compliance**
- **Use Function Point Analysis to improve data collection capability to research quality**
- **Execute the second doubling of performance of teams based on Function Point Analysis by focusing on READY state of Product Backlog**

Learn and improve from success

Q2 2008

Project	Performance	Deviation
A	192%	18%
B	76%	64%
C	86%	92%
D	54%	50%
E	258%	48%

Q3 2008

Project	Performance	Deviation
A	140%	44%
B	74%	64%
C	81%	83%
D	70%	59%
E	365%	75%

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

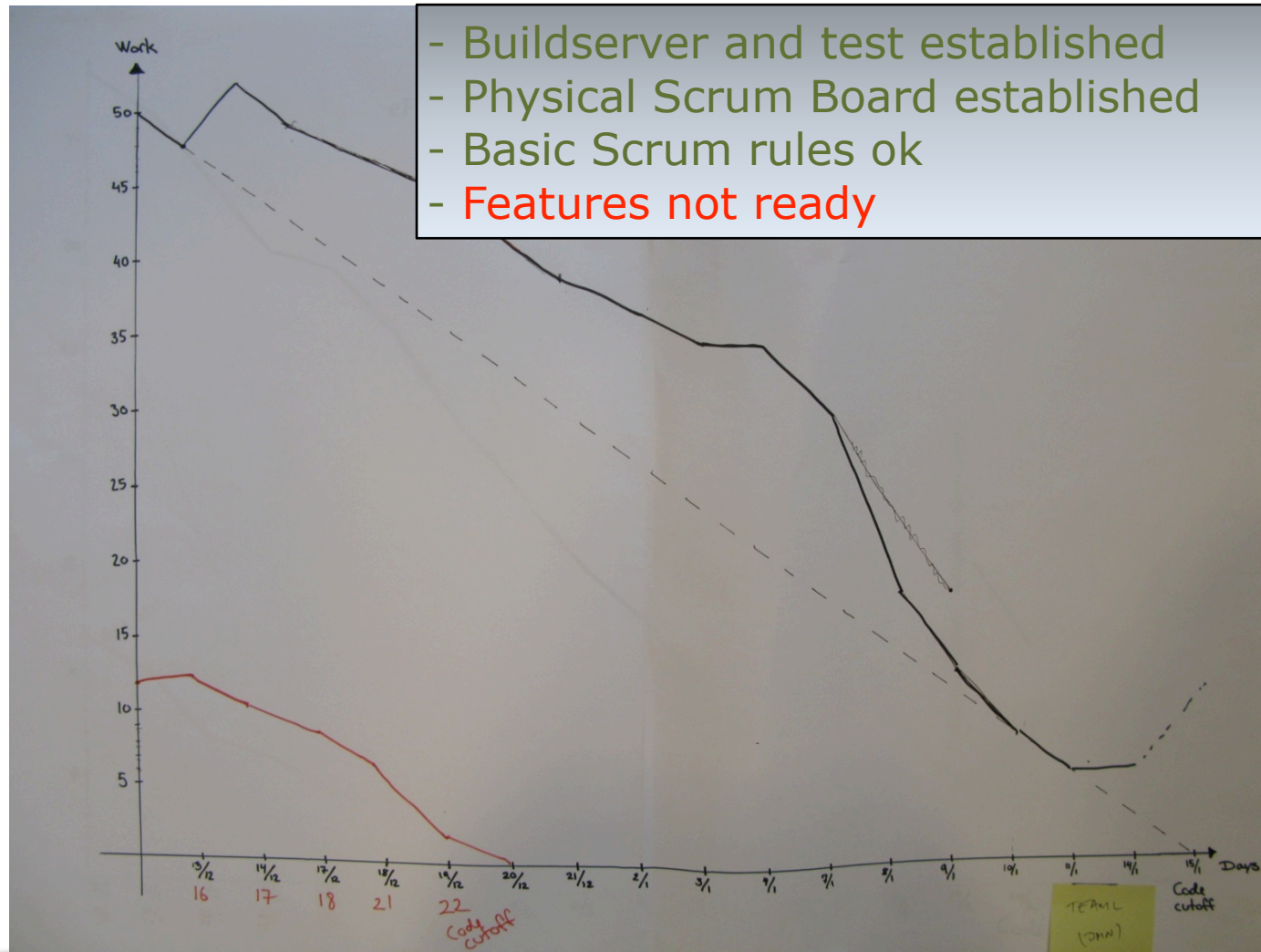
Projects investigated

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%



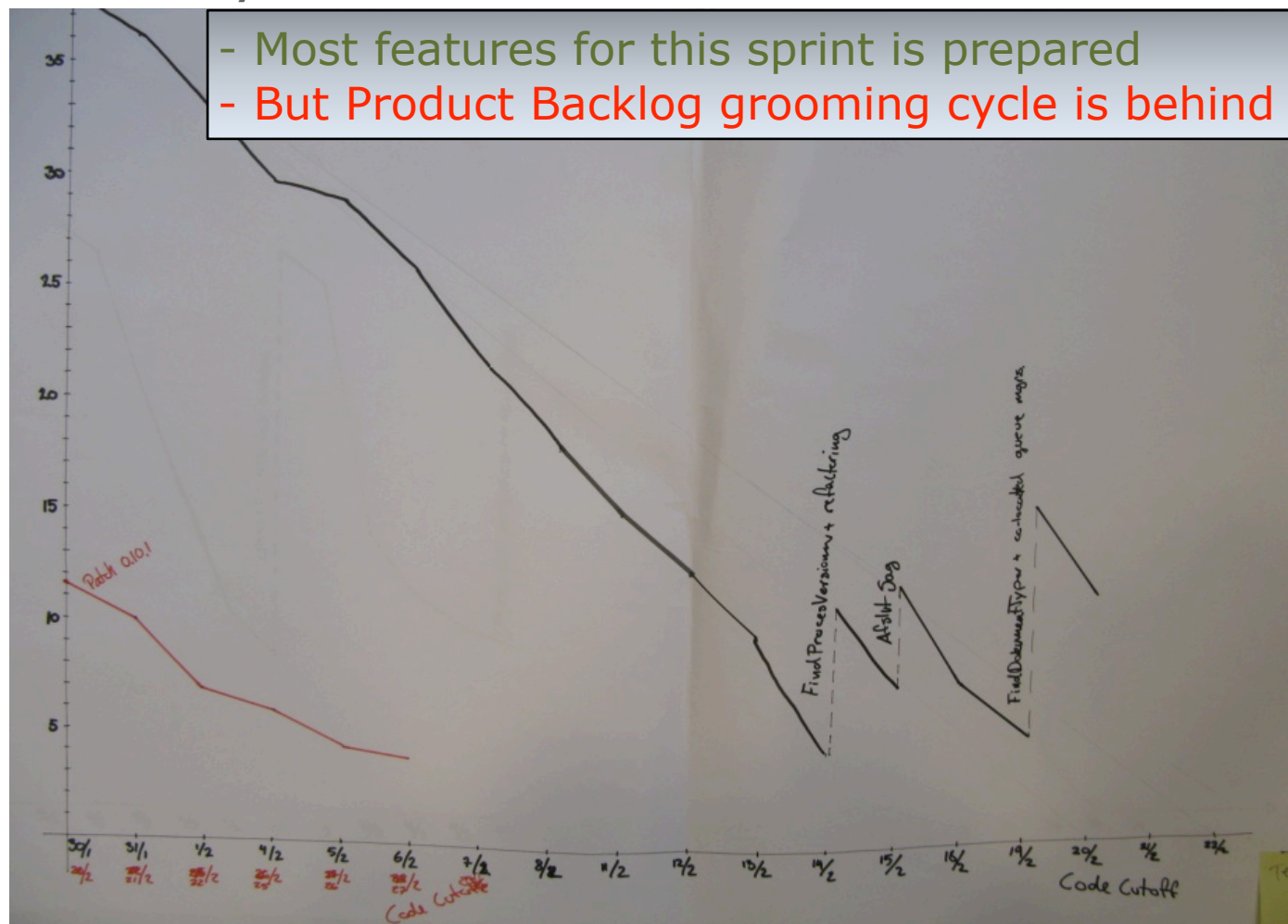
Revision:

Page

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

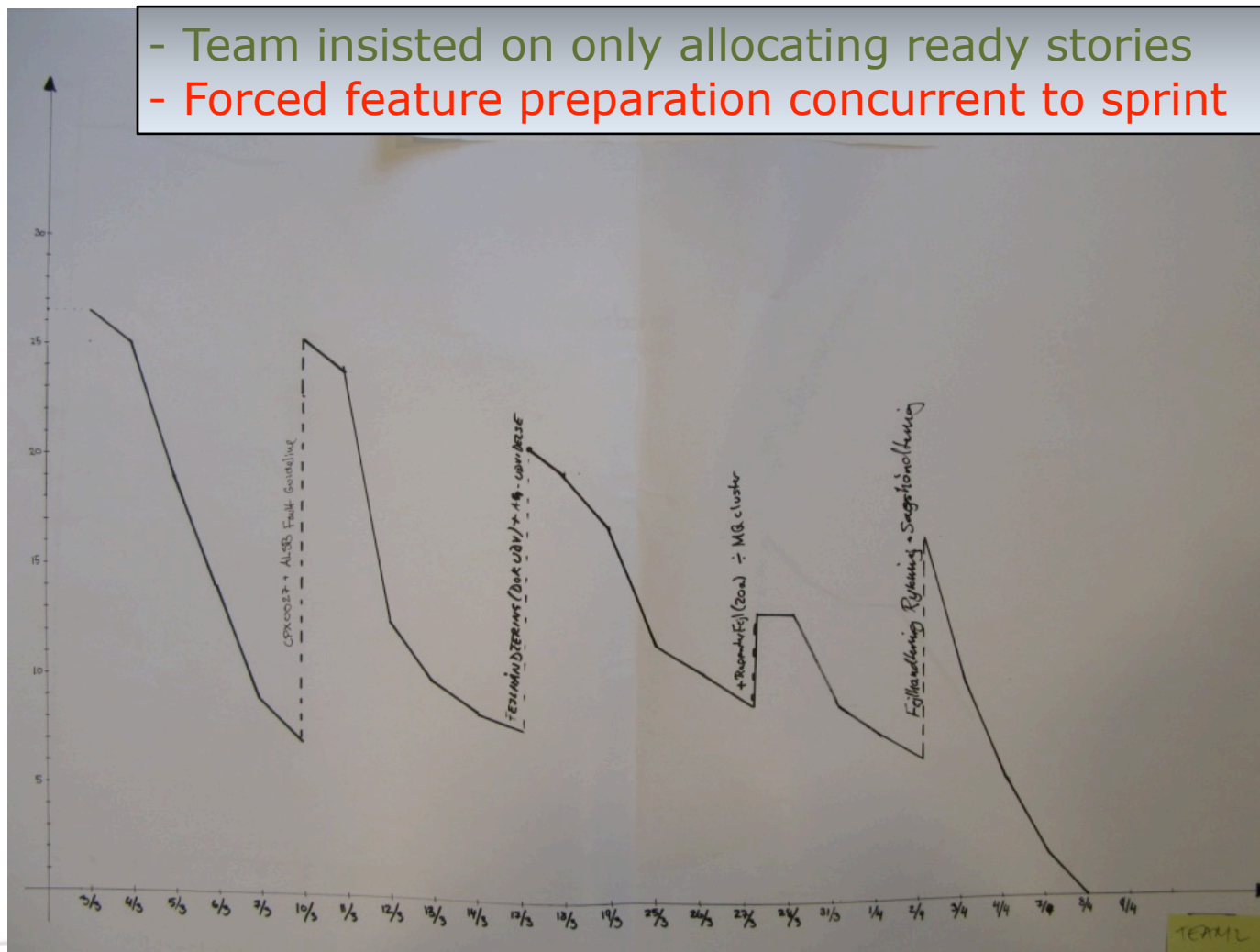


Revision:

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

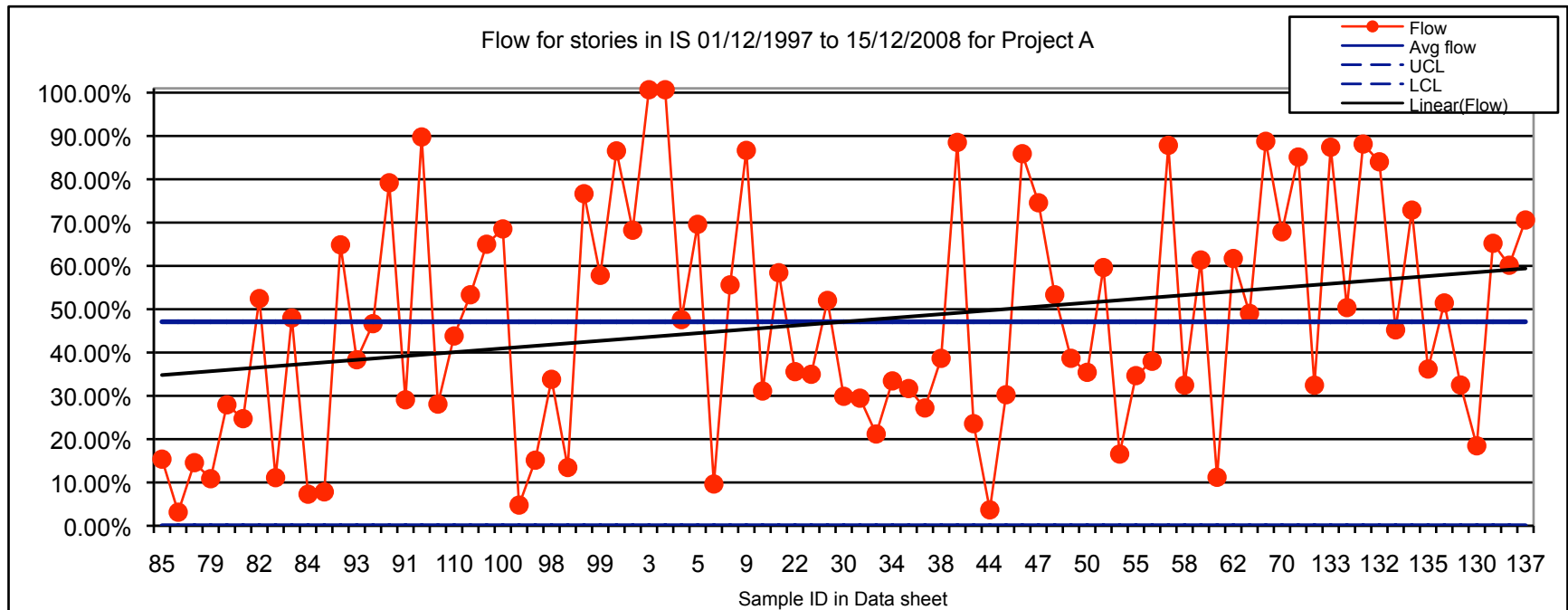


Revision:

Page

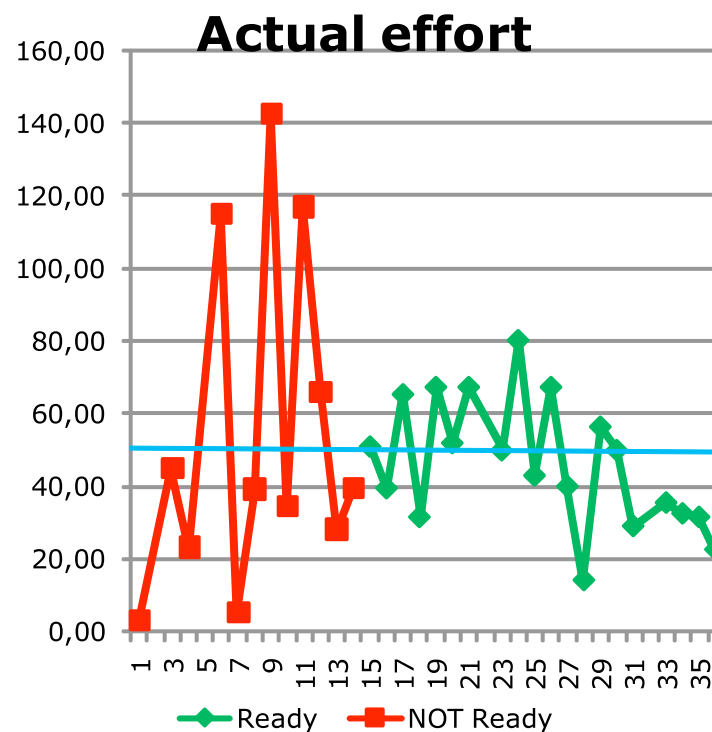
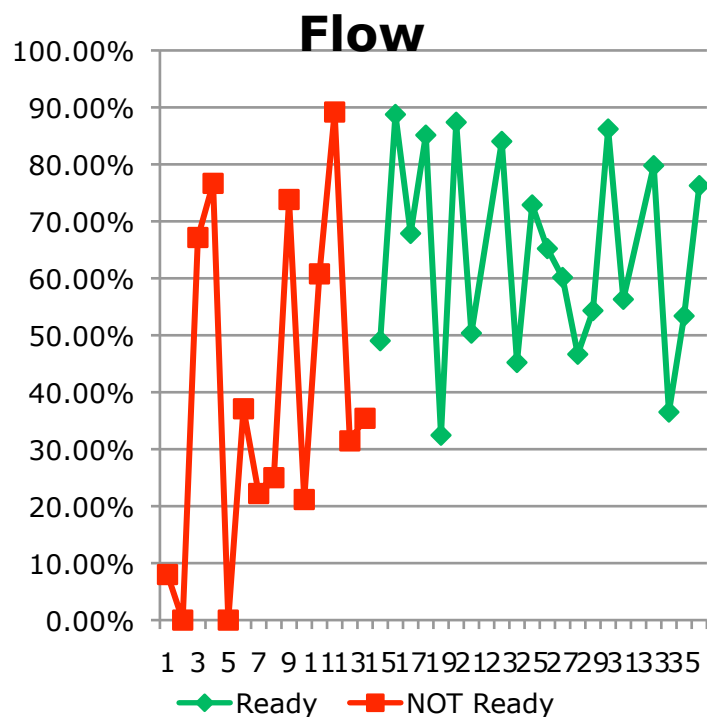
Result

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



Effect

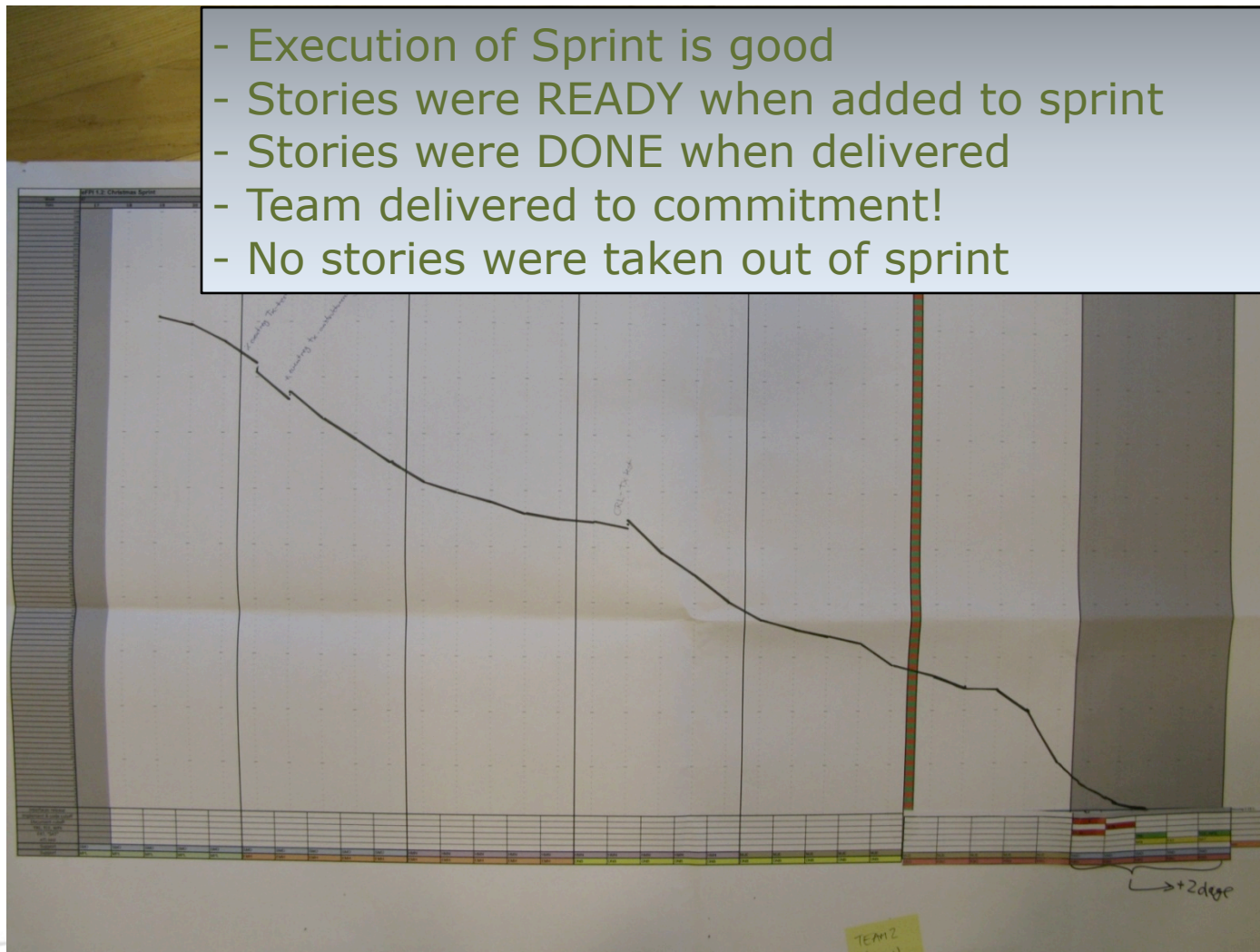
When work allocated to sprint is **READY**, flow and stability is achieved



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

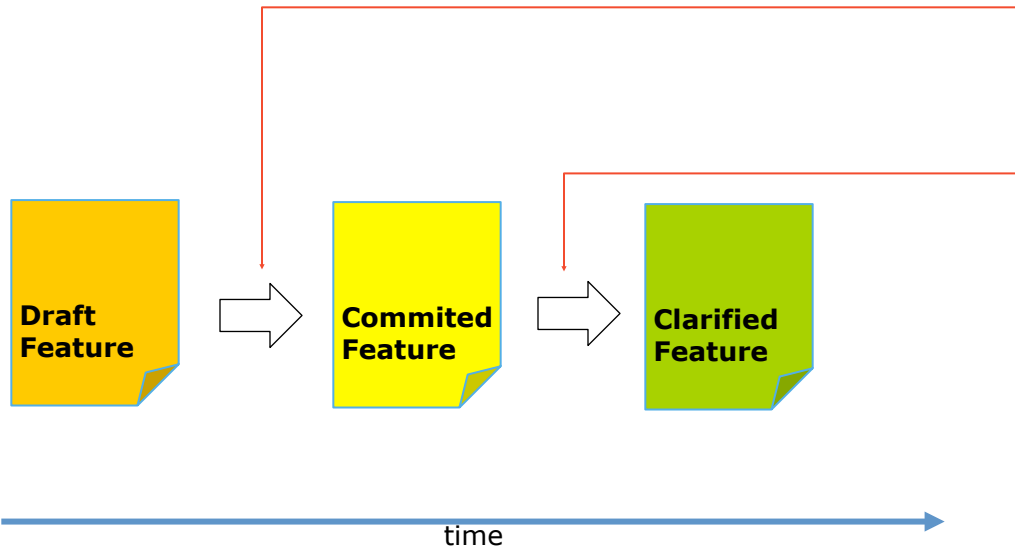


Revision:

Page

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



Ready for Implementation Checklist

Feature: _____
 Product Owner: _____
 Architect: _____
 Lead Developer: _____

Procedure / Primary role	Activity	Work Product(s)	Completed
Prepare Feature for Commitment / Product Owner	Customer requirements approved and baselined	PMA/095	<input type="checkbox"/>
	Customer requirements assigned to the feature	PMA/098, FDD	<input type="checkbox"/>
	Customer requirements sufficiently understood	FDD	<input type="checkbox"/>
	Technical design drafted (focus – feasibility)	FDD, EST	<input type="checkbox"/>
	Risks identified	FDD, EST	<input type="checkbox"/>
	Test design drafted (focus testability)	FDD, EST	<input type="checkbox"/>
	Unknowns, assumptions, constraints, concerns identified	FDD, EST	<input type="checkbox"/>
	ROM (effort, size) established	EST	<input type="checkbox"/>
	Concept review conducted	RER	<input type="checkbox"/>
	FDD approved	DTS	<input type="checkbox"/>
Clarify Feature for Development / Architect	Fit into sprint considered	FDD	<input type="checkbox"/>
	Feature decomposed into fit-to-sprint-features	FDD	<input type="checkbox"/>
	Plan for unknowns/assumptions/concerns/constraints established	FDD, EST	<input type="checkbox"/>
	Estimates (effort & size) updated	EST	<input type="checkbox"/>
	Concept review conducted	RER	<input type="checkbox"/>
Prepare Feature for Implementation / Lead Developer	Unknowns, assumptions, concerns resolved	FDD	<input type="checkbox"/>
	Product requirements developed	PMA/098, FDD	<input type="checkbox"/>
	Test design drafted (no uncertainties)	FDD	<input type="checkbox"/>
	Technical design drafted (no uncertainties)	FDD	<input type="checkbox"/>
	Decomposition into stories performed	FDD	<input type="checkbox"/>
	Stories estimated (effort)	EST	<input type="checkbox"/>
	Concept review conducted	RER	<input type="checkbox"/>
	FDD approved	DTS	<input type="checkbox"/>

SSE/06574/CHK/0007 \$Revision: 1.1 \$ \$Date: 24 Sep 2008 \$

Continue to improve

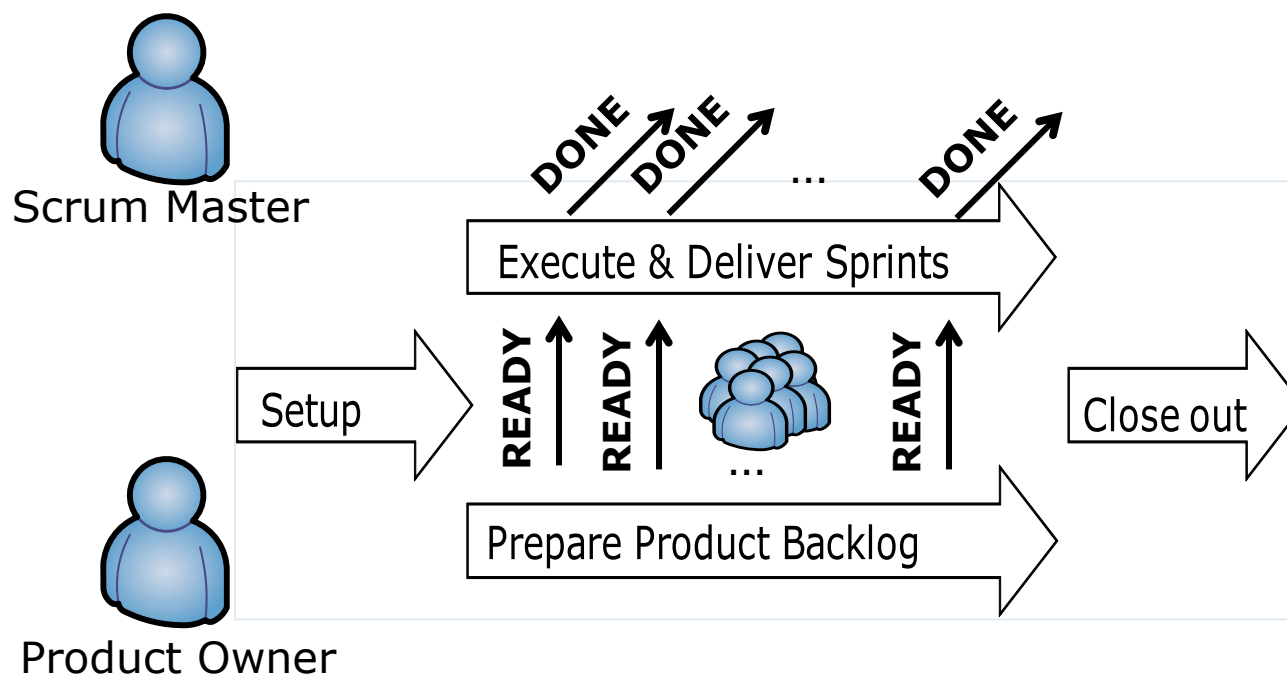
Identifying root causes to stories not achieving desired flow (03/2009)

- **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

Revision:

Understanding Scrum success

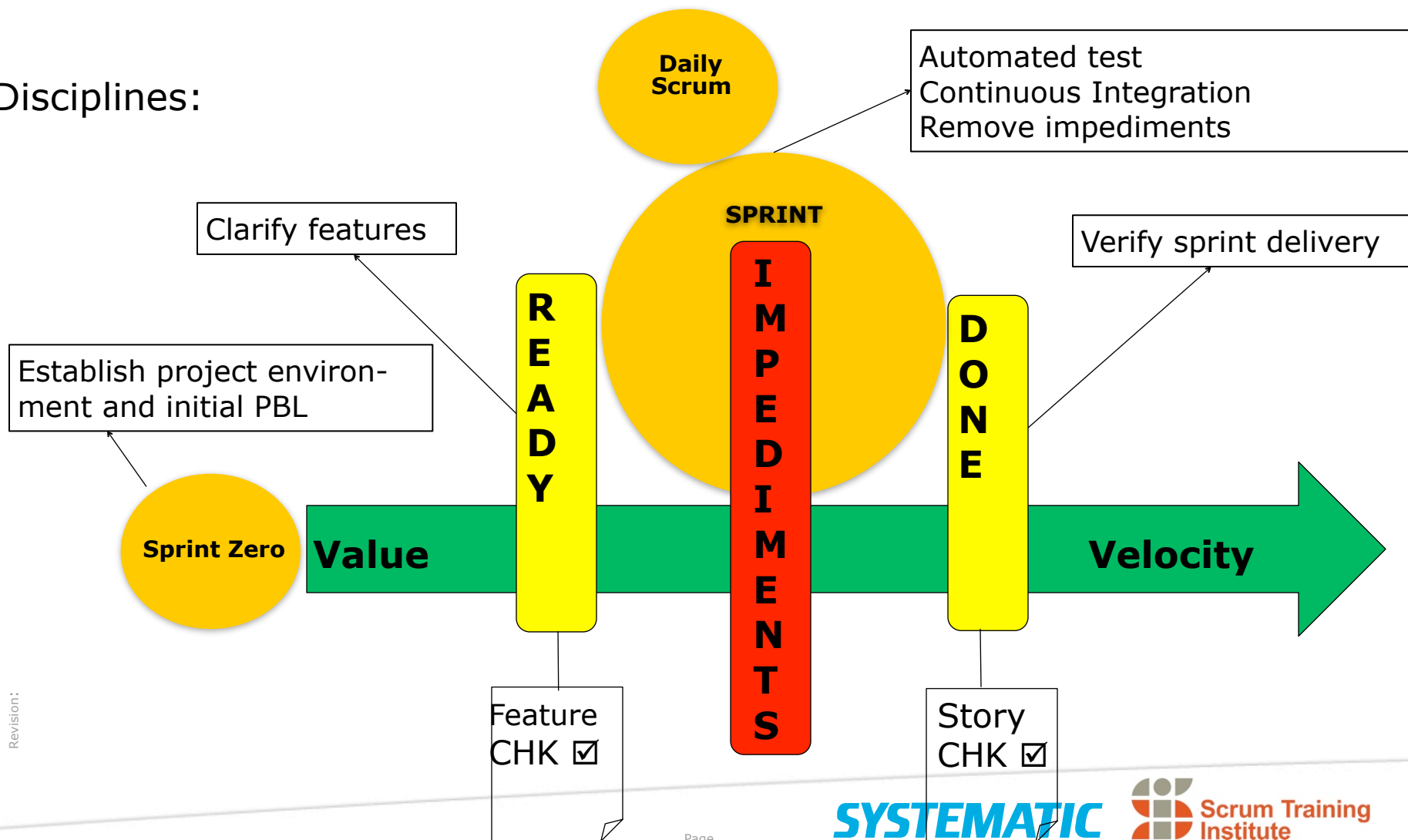
READY and DONE is simple to understand but hard to do



Key is a proper balance between planning and execution activities

The Systematic Scrum model

Disciplines:



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

Revision:



Questions

SYSTEMATIC

 **Scrum Training
Institute**