Agile-SCRUM Implementation Under CMMI Framework

By

Dr. Bin Cong

Certified SCRUM Master

SEI CMMI Instructor

SEI CMMI HM LA

Professor and Director

Software Engineering Program at Cal State University

SCRUM…or SCRUM-BUT

"The implementations of Scrum out there are so crippled it is amazing! Yet most of them still say they are doing better they were doing better than old process (if they had any).

We have a lot of work to do to get people to do Scrum basics."

- Jeff Sutherland

Workshop Objectives

- Reinforce correct understanding of key concepts/practices of SCRUM
- Understand how to adopt SCRUM under CMMI Framework
- How to serve effectively as SCRUM role

Product Backlog

Sprint 1: Goal: Understand the Process, People, and Tools in SCRUM

Sprint 2: Goal: Scrum your Scrum

Sprint 3: Goal: Plan and Estimation in SCRUM

Sprint 4: Goal: Understand Advanced Scrum Topics

Product Backlog

Sprint 1: Goal: Understand the Process, People, and Tools in SCRUM

Sprint Backlog:

- SCRUM Origins and Shu Ha Ri
- SCRUM Practices : Framework, Roles, Product Management, Agile thinking
- 7 wastes in product development ● Read your Scrum chart
- Exercise 1

Sprint 2: Goal: Scrum your Scrum

Sprint 3: Goal: Plan and Estimation in SCRUM

Sprint 4: Goal: Understand Advanced Scrum Topics

The Paper

HBR
JANUARY–FEBRUARY 1986

The New New Product Development Game

Hirotaka Takeuchi and Ikujiro Nonaka

The rules of the game in new product development are changing. Many companies have discovered that it takes more than the accepted basics of high quality, low cost, and differentiation to excel in today's competitive market. It also takes speed and flexibility.

This change is reflected in the emphasis companies are placing on new products as a source of new sales and profits. At 3M, for example, products less than five years old account for 25% of sales. A 1981 survey of 700 U.S. companies indicated that new products would account for one-third of all profits in the 1980s, an increase from one-fifth in the 1970s.[1]

This new emphasis on speed and flexibility calls for a different approach for managing new product development. The traditional sequential or "relay race" approach to product development—exemplified by the National Aeronautics and Space Administration's phased program planning (PPP) system—may conflict with the goals of maximum speed and flexibility. Instead, a holistic or "rugby" approach—where a team tries to go the distance as a unit, passing the ball back and forth—may better serve today's competitive requirements.

Under the old approach, a product development process moved like a relay race, with one group of functional specialists passing the baton to the next group. The project went sequentially from phase to phase: concept development, feasibility testing, product design, development process, pilot production, and final production. Under this method, functions were specialized and segmented: the marketing people examined customer needs and perceptions in developing product concepts; the R&D engineers selected the appropriate design, the production engineers put it into shape, and other functional specialists carried the baton at different stages of the race.

Under the rugby approach, the product development process emerges from the constant interaction of a hand-picked, multidisciplinary team whose members work together from start to finish. Rather than moving in defined, highly structured stages, the process is born out of the team members' interplay (see *Exhibit 1*). A group of engineers, for example, may start to design the product (phase three) before all the results of the feasibility tests (phase two) are in. Or the team may be forced to reconsider a decision as a result of later information. The team does not stop then, but engages in iterative experimentation. This goes on in even the latest phases of the development process.

Exhibit 1 illustrates the difference between the traditional, linear approach to product development and the rugby approach. The sequential approach, labeled type A, is typified by the NASA-type PPP system. The overlap approach is represented by type B, where the overlapping occurs only at the border of adjacent phases, and type C, where the overlap extends across several phases. We observed a type B overlap at Fuji-Xerox and a type C overlap at Honda and Canon.

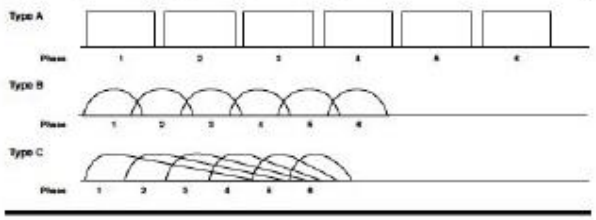
This approach is essential for companies seeking to develop new products quickly and flexibly. The shift from a linear to an integrated approach encourages trial and error and challenges the status quo. It stimulates new kinds of learning and thinking within the organization at different levels and functions. Just as important, this strategy for product development can act as an agent of change for the larger organization. The energy and motivation the effort produces can spread throughout the big company and begin to break down some of the rigidities that have set in over time.

In this article, we highlight companies both in Japan and in the United States that have taken a new approach to managing the product development process. Our research examined such multinational companies as Fuji-Xerox, Canon, Honda, NEC, Epson, Brother, 3M, Xerox, and Hewlett-Packard. We then analyzed the development process of six specific products:

- ☐ FX-3500 medium-sized copier (introduced by Fuji-Xerox in 1978)
- ☐ PC-10 personal-use copier (Canon, 1982)
- ☐ City car with 1200 cc engine (Honda, 1981)
- ☐ PC 8000 personal computer (NEC, 1979)
- ☐ AE-1 single-lens reflex camera (Canon, 1976)
- ☐ Auto Boy, known as the Sure Shot in the United States, lens shutter camera, (Canon, 1979)

We selected each product on the basis of its impact, its visibility within the company as part of a "breakthrough" development process, the novelty of the product features at the time, the market success of the product, and the access to and availability of data on each product.

EXHIBIT 1
Sequential (A) vs. overlapping (B and C) phases of development



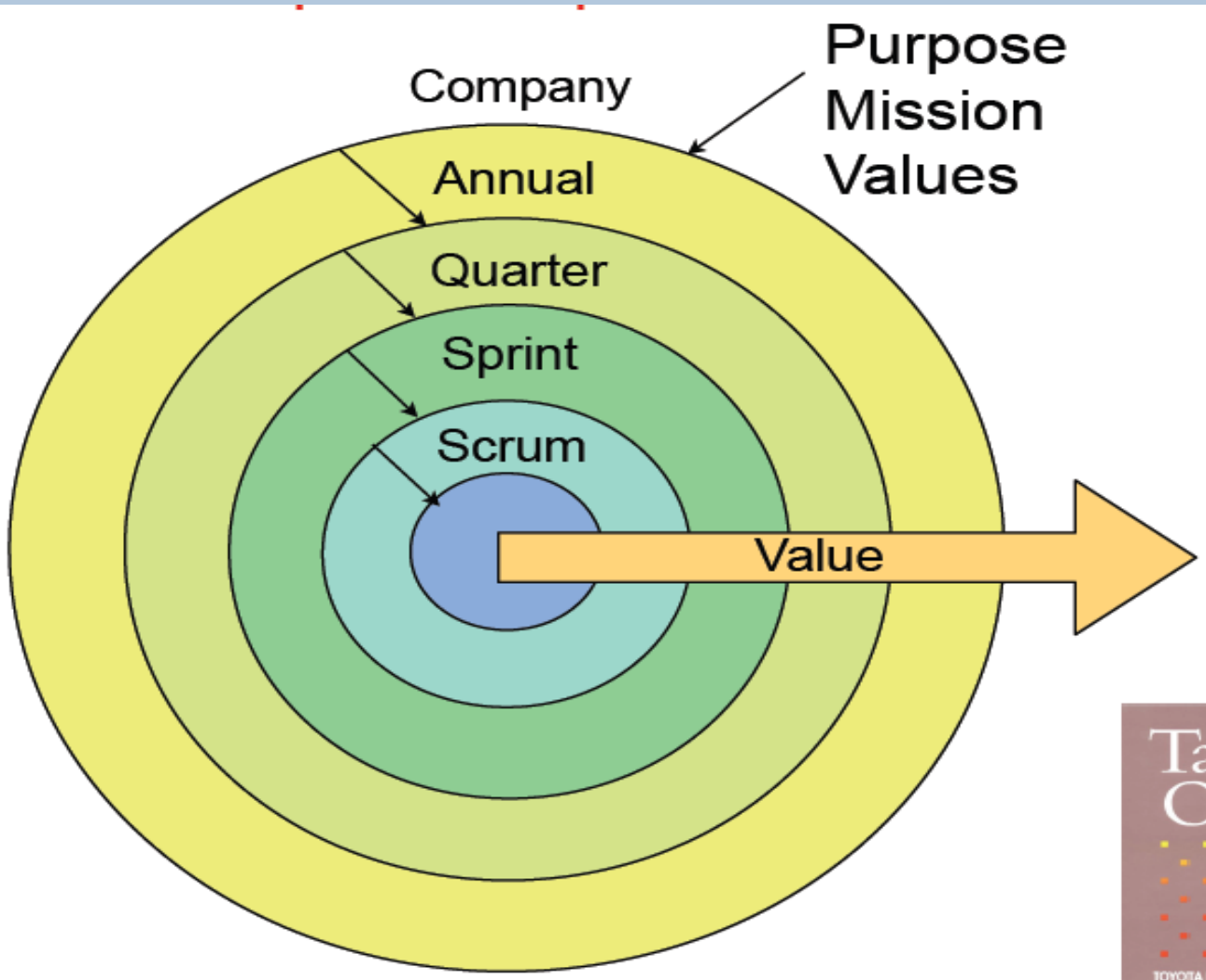
In today's fast-paced, fiercely competitive world of commercial new product development, speed and flexibility are essential. Companies are increasingly realizing that the old, sequential approach to developing new products simply won't get the job done. Instead, companies in Japan and the United States are using a holistic method—as in rugby, the ball gets passed within the team as it moves as a unit up the field.

This holistic approach has six characteristics: built-in instability, self-organizing project teams, overlapping development phases, "multilearning," subtle control, and organizational transfer of learning. The six pieces fit together like a jigsaw puzzle, forming a fast, flexible process for new product development. Just as important, the new approach can act as a change agent: it is a vehicle for introducing creative, market-driven ideas and processes into an old, rigid organization.

Mr. Takeuchi is an associate professor and Mr. Nonaka, a professor at Hitotsubashi University in Japan. Mr. Takeuchi's research has focused on marketing and global competition. Mr. Nonaka has published widely in Japan on organizations, strategy, and marketing.

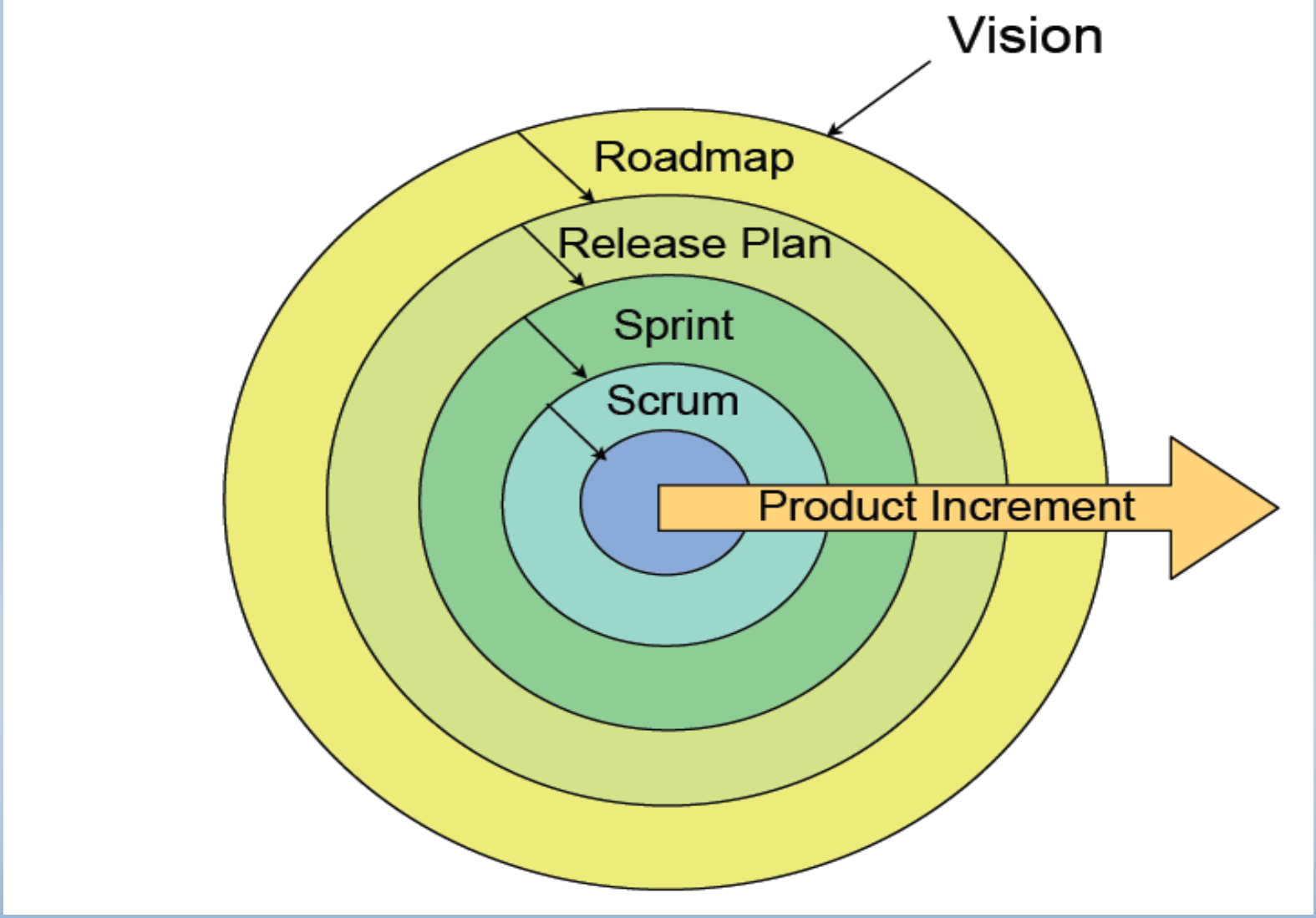
Authors' note: We acknowledge the contribution of Kenichi Imai in the development of this article. An earlier version of this article was coauthored by Ken-ichi Imai, Ikujiro Nonaka, and Hirotaka Takeuchi. It was entitled "Managing the New Product Development Process: How Japanese Companies Learn and Unlearn" and was presented at the seventy-fifth anniversary Colloquium on Productivity and Technology, Harvard Business School, March 28 and 29, 1984.

SCRUM Org is a Lean Org



Train managers in Lean so they understand Scrum

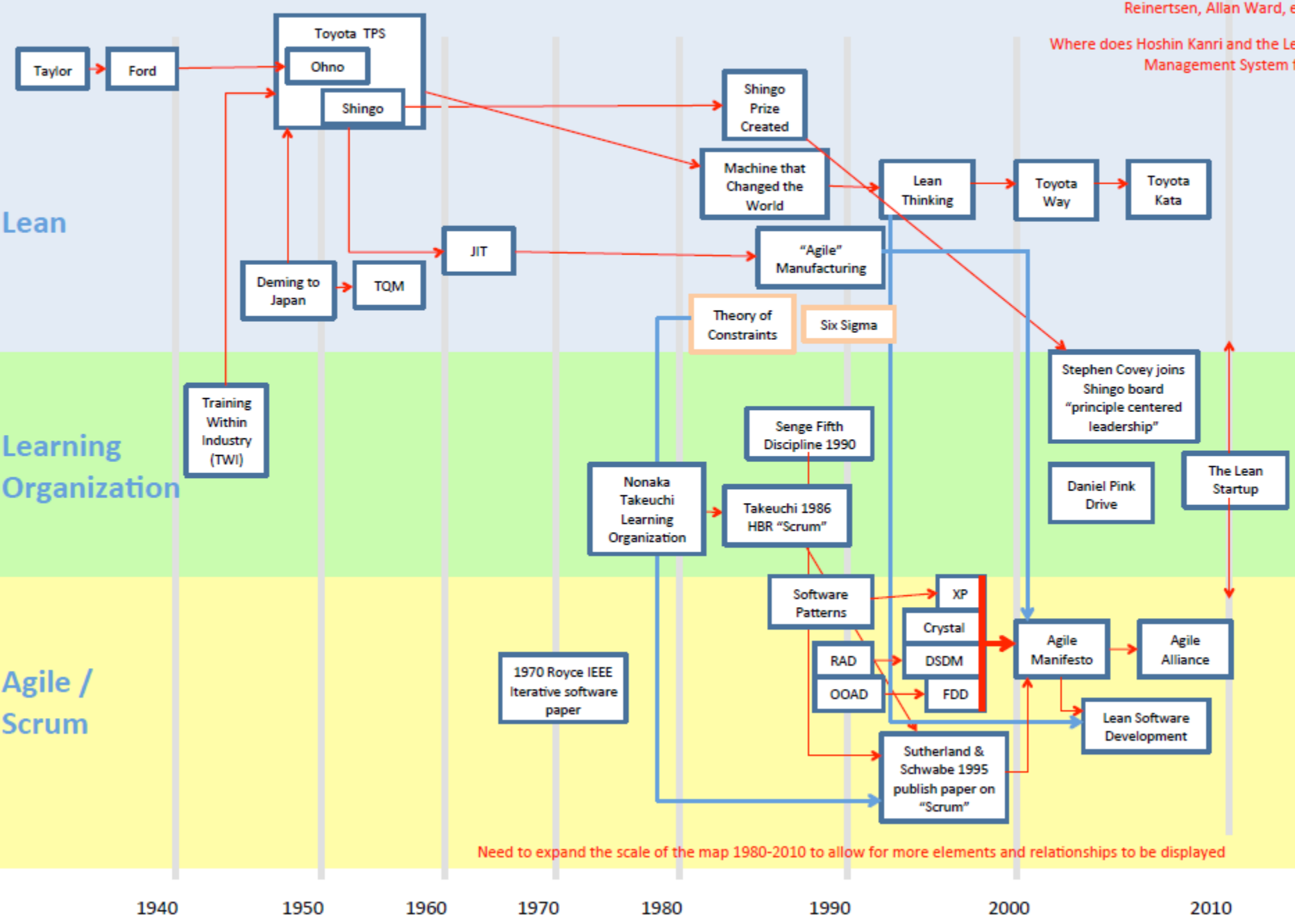
Product Owner's View



Lean and Agile/Scrum Lineage: a visual work in progress

Consider how to fit Lean Product Development in here: Toyota, Reinertsen, Allan Ward, etc.

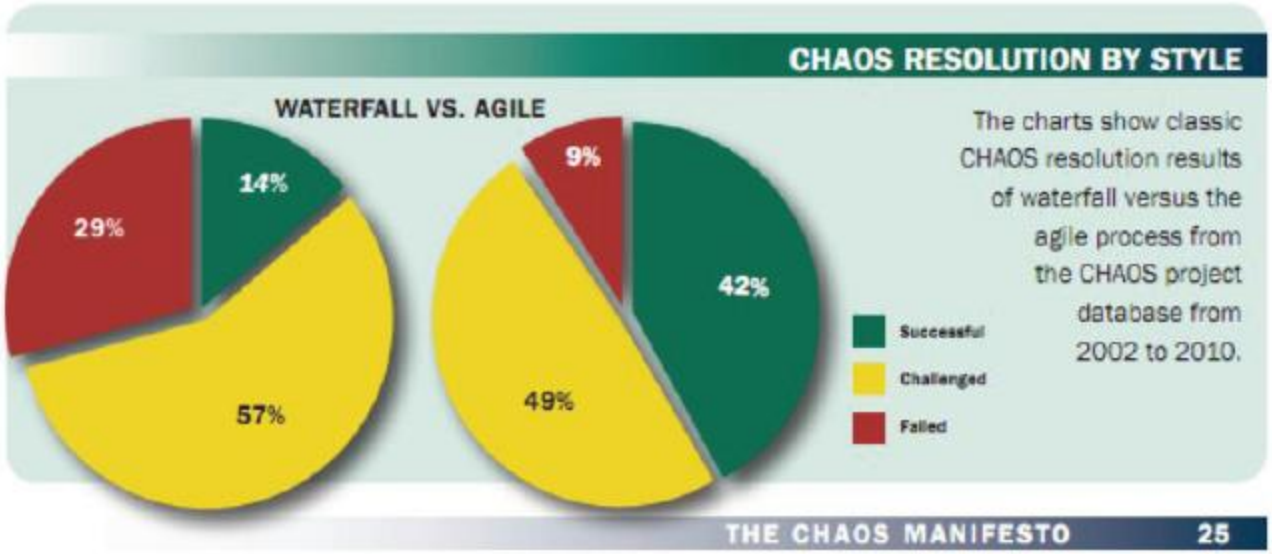
Where does Hoshin Kanri and the Lean Management System fit?



Agile Process

Chaos Manifesto 2011, Standish Group International, Inc.

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 aad delivery of the iterative process.



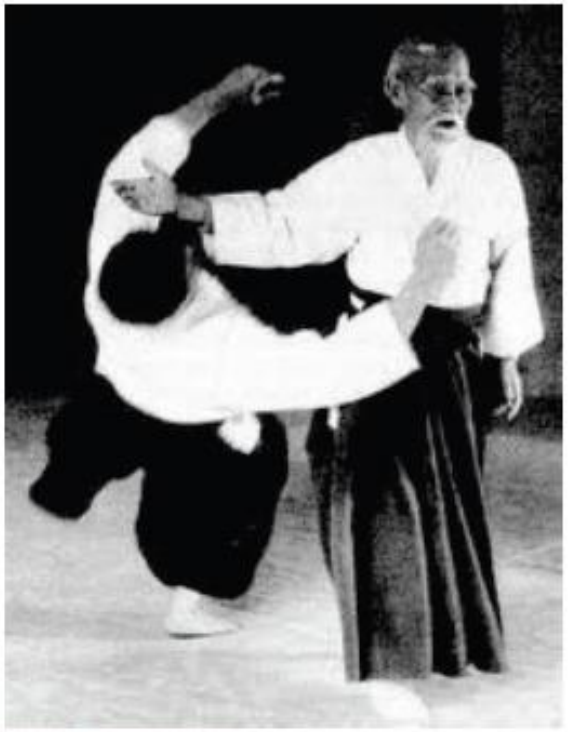
Source:

SCRUM is easy to understand but hard to implement

- A systematic approach to implementing SCRUM will give you the most benefit fastest for the least effort.
- Shu Ha Rui Concept:

Aikido

The Way in Harmony with the Spirit



Shobu Aikido

Shu State

- SCRUM Master sets up process as in the SCRUM guide.
- Team has a known velocity at a sustainable pace.
- Retrospective is used to enhance the performance.

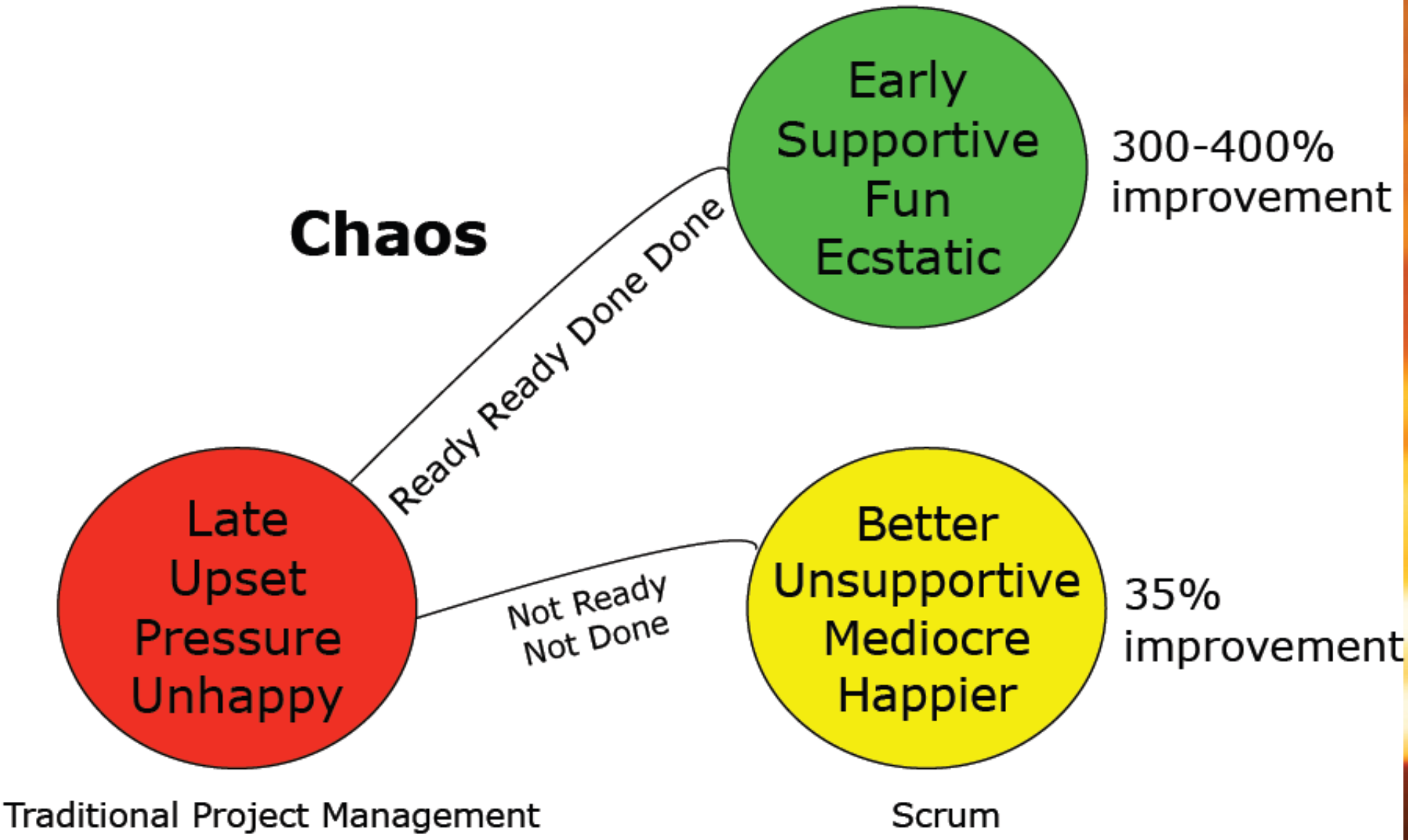
Ha State

- Team has software done with all features tested and no major bugs at the end of a sprint.
- Product owner has READY backlog at beginning of a sprint (for example: good user stories)
- Team has data showing velocity and quality has least doubled.
- Team is positioned to work towards hyperproductivity, the GOAL of SCRUM

Ri State

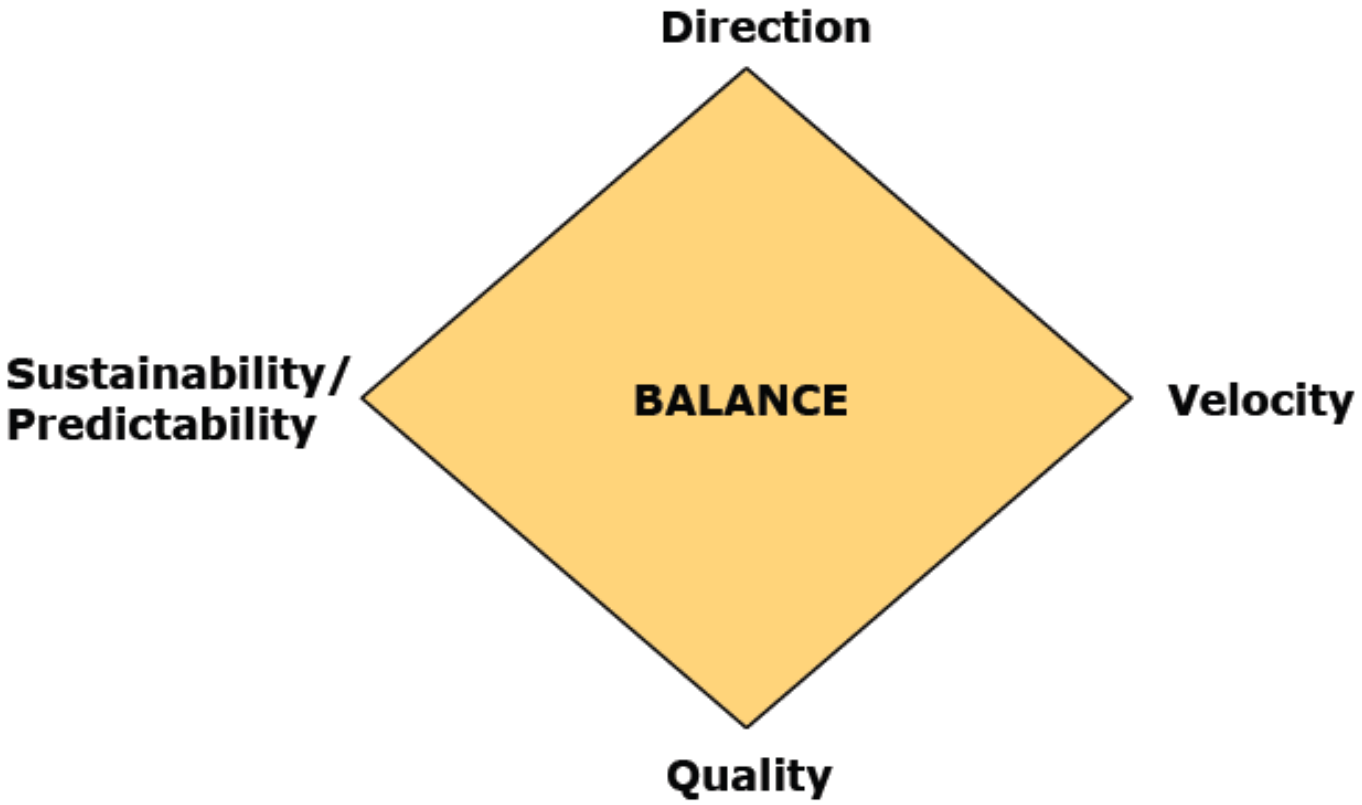
- Team needs to be hyperproductive.

After Introducing Scrum ...



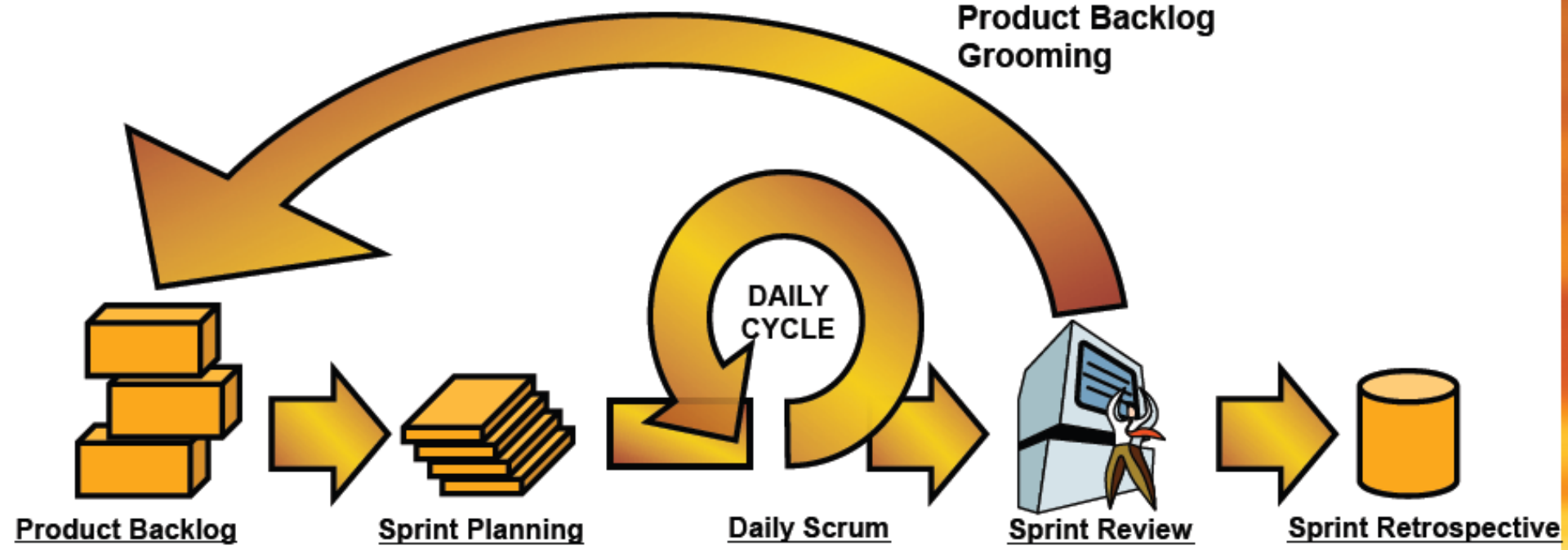
Scrum Radar Chart

Part of successfully implementing Scrum is realizing that there are four key components that need focus. *Major challenges emerged when the team focused on just a single area.* J. Sutherland and I. Altman, "Take No Prisoners: How a Venture Capital Group Does Scrum," in *Agile 2009*, Chicago, 2009.



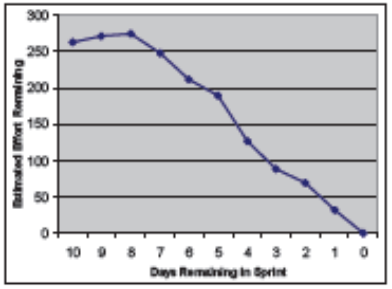
Describe SCRUM components?

Scrum view of Product Management



Product Backlog Item Name	Item Priority	Low	High
gency Portal - Changes to webADMIN Tran History layout display T&A transactions on screen (separate from	1	100	120
gency Portal - Changes to Client page in webADMIN cluding messages - see VI)	2	80	90
gency Portal - Changes to Adjustment page	3	100	120
gency Portal - New page to show checked in at that point	4	-	-
ges (Search page, results, links from Reminder/provider info			
Agency Portal - Link to searches - Hide checked in	5	3	12
gency Portal - Web Support for T&A	6	20	30
gency Portal - New webADMIN screen to show what rvices they have in eligibility according to the State	7	150	200
	8	-	-
gency Portal - Help Files	9	40	50
Agency Portal - Tutorial	10	80	120
	11	20	25
rdholder Portal - Add additional page for Time and ndance transactions, separate from PS and Cash	12	60	80
rdholder Portal - Add new screen for checking in and cking out of the provider	13	80	100

		Days Remaining	10	9	8	7
			12/17	12/20	12/21	12/22
Task #	Task Name	Hours	280	260	210	221
110	Configuration Document - Assessment		18	18	18	17
120	Configuration Document - Draft		15	15	15	15
130	Configuration Document - Final		10	10	10	10
140	WRD Documentation		8	8	8	8
150	Process Definition Documentation		8	8	8	8
200	IDT Centers- WID Setup in Dev		2	2	1	0
220	IDT- Services Setup Scripts		8	8	1	1
230	QA for IDT Centers Setup		4	4	4	4
310	Client Suspend/Unsuspend QA Testing		0	0	0	0
320	BRAT Express Email options QA Testing		2	0	0	0
410	Research to identify current ADA Calls use		0	0	0	0
420	Research for removal of dummy person		0	0	0	0
430	Create ADA- for Proxy changes		16	10	4	0



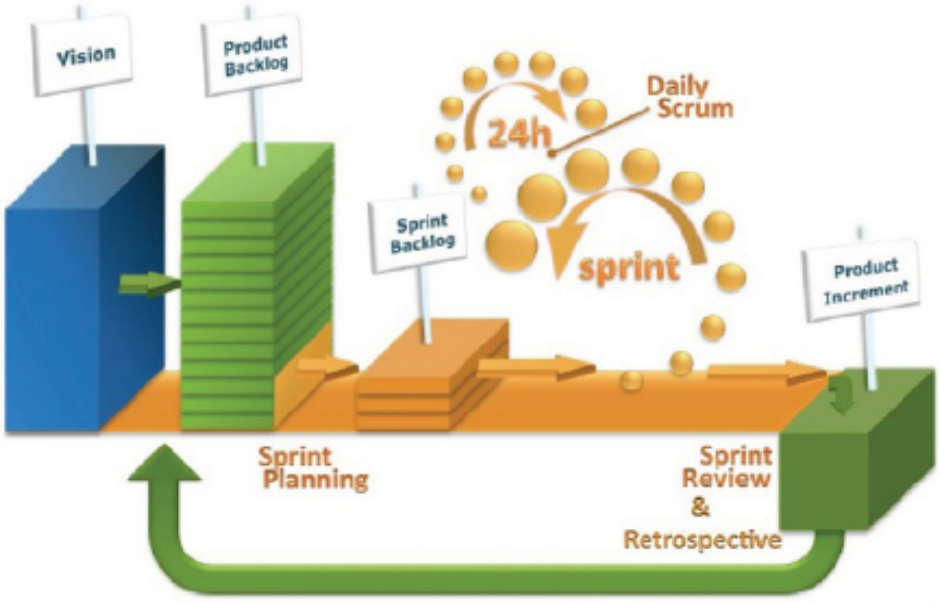

```
' Define the variables.
Dim rng As Excel.Range
Dim ValueAxisMinValue As Double
Dim sht$name As String
Dim Title As String, mChart As Chart
' Store the location of the data as a string.
rng = Selection.Address()
' Store the start date for the chart.
ValueAxisMinValue = Selection.Cells(2, 2).Value
' Ask user for the Chart title.
Title = InputBox("Please enter the chart title")
' Store the sheet name.
sht$name = ActiveSheet.Name
' Turn off screen updating.
Application.ScreenUpdating = False
' Create a chart located on a chart sheet.
Set mChart = Charts.Add
With mChart
    .ChartWizard Source:=Sheets(sht$name).Range(rng), _
     Gallery:=xlBar, Format:=3, PlotBy:=xlColumns, _
     CategoryLabels:=1, SeriesLabels:=1, HasLegend:=1, _
     Title:=Title, CategoryTitle:="", ValueTitle:="", _
     ExtraTitle:=""
    'Remove the legend.
    .Legend.Delete
    'Create and format the series.
    With .SeriesCollection(1)
```

PRODUCT BACKOG | SPRINT BACKLOG | BURNDOWN CHART | DEMO | LESSONS LEARNED

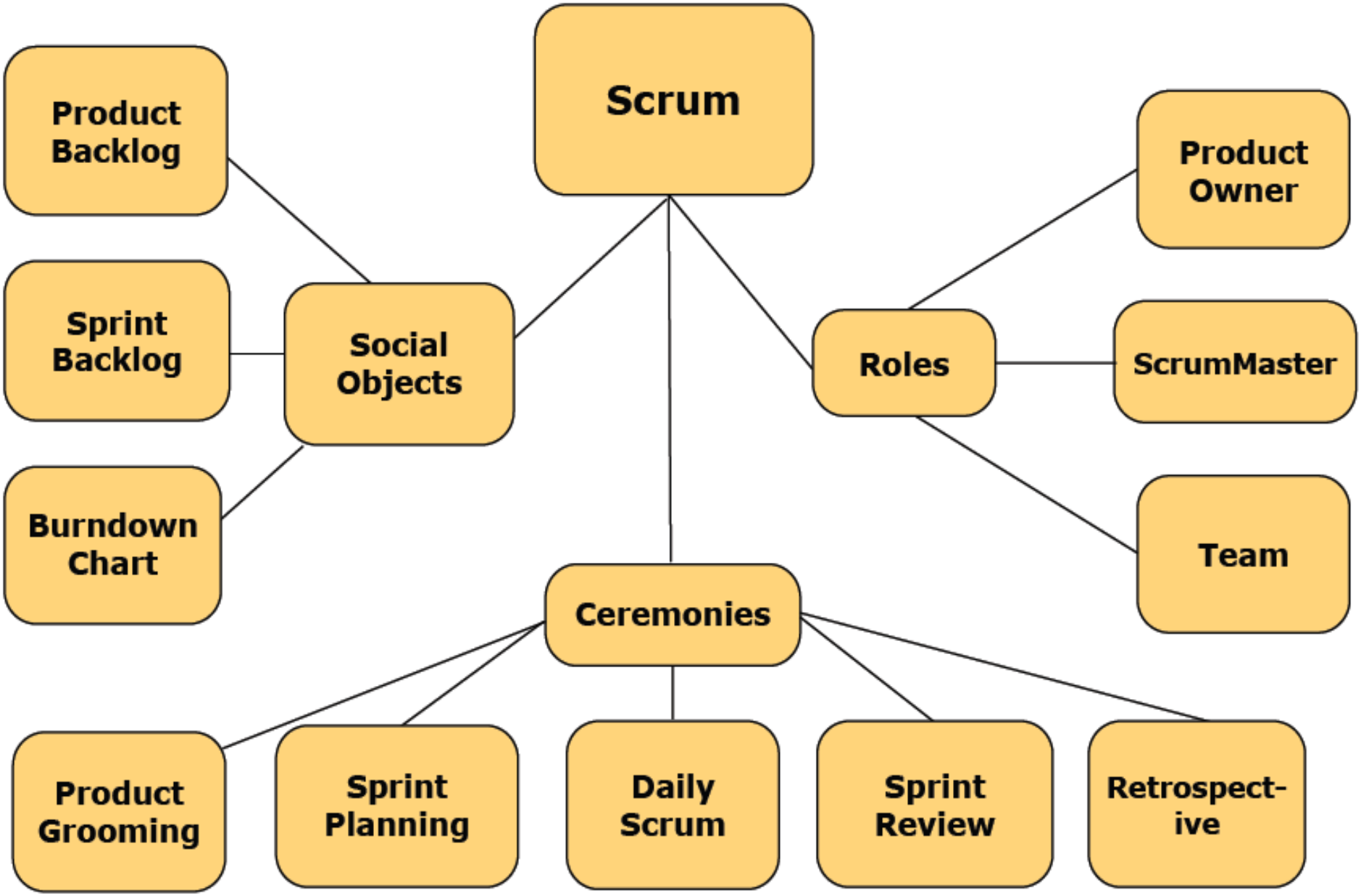
Graphics: ScrumInc Partner SolutionsIQ

Scrum Framework in 30 seconds

- A product owner creates a prioritized wish list called a product backlog.
- During sprint planning, the team pulls a small chunk from the top of that wishlist, a sprint backlog, and decides how to implement those pieces.
- The team has a certain amount of time, a sprint, to complete its work - usually two to four weeks - but meets each day to assess its progress (daily scrum).
- Along the way, the ScrumMaster keeps the team focused on its goal.
- At the end of the sprint, the work should be potentially shippable, as in ready to hand to a customer, put on a store shelf, or show to a stakeholder.
- The sprint ends with a sprint review and retrospective.
- As the next sprint begins, the team chooses another chunk of the product backlog and begins working again.



Scrum Framework



Principles of Scrum

- Key values of SCRUM
  - Transparency
  - Inspection
  - Adoption
- Uniqueness of SCRUM
  - Empirical Process control
  - Succinct work cadences
  - Reorientation based on work done so far

SCRUM Terminology

- Sprint
- Product Owner
- Scrum Master
- Scrum Team
- Product Backlog
- Sprint Backlog
- Burn-down chart
- Daily Scrum
- Scrum Retrospective

Key Roles and Responsibilities

Product Owner

Goals
- Get the correct product
- Develop features with most value

Responsibilities
- Create a product vision
- Release planning
- Facilitate real user feedback
- Determine business value
- Define acceptance criteria

ScrumMaster

Goals
- Customer gets value for money
- Protect team from outside interference
- Product owner and team fulfill their responsibilities

Responsibilities
- Nurture Team's self-management
- Support team to be productive
- Support product owner in determining business value
- Eliminate discovered Impediments
- Facilitate the Scrum process
- Ensure communication
- Manage project environment

Development Team

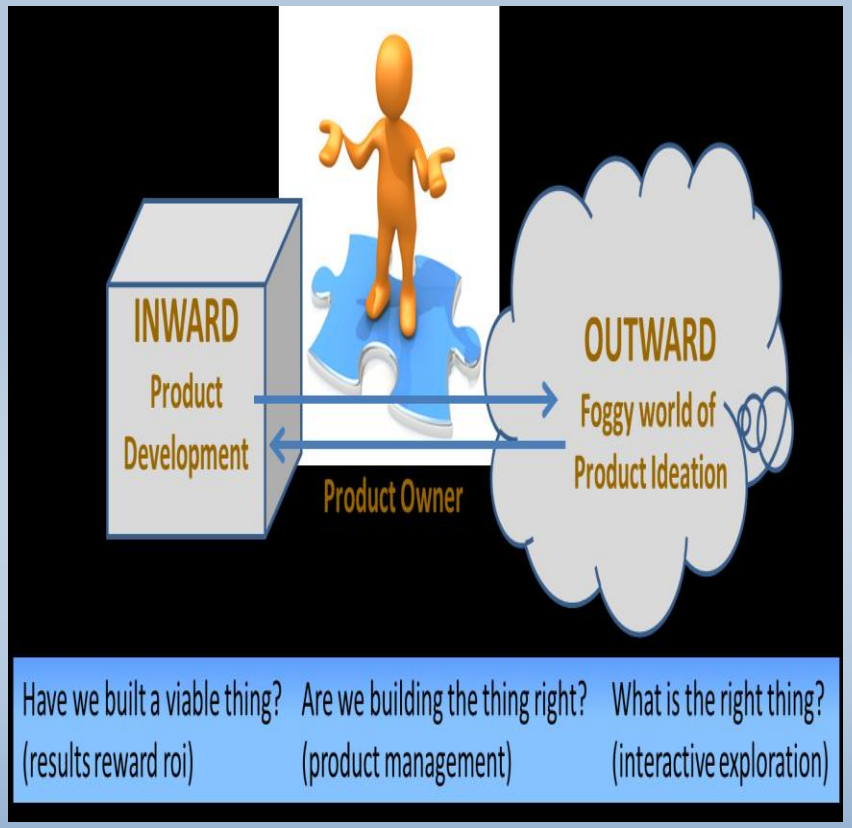
Goals
- Develop most value to customer
- Maintain ability to deliver change

Responsibilities
- Estimate effort and dependencies
- Plan iteration execution
- Develop regular deliveries of ready functionality

SCRUM Frame work

Roles of SCRUM

- Product owner
  - Represents stakeholder or customer
  - Responsible for ROI
  - Defines features of product
  - Decide on release date and content



SCRUM Framework

Roles of SCRUM

What should be on your biz card?

A title?

- SCRUM Team
  - Represents developing team , no sub teams!
  - Typically group of 7 members
  - Responsibility of satisfying customers and stakeholders needs
  - Should be self-organized and self-led

SCRUM Frame work

Roles of SCRUM

- SCRUM Master
  - Ensure that Scrum process is stick to its principles and used as intended
  - Responsible for establishing SCRUM practices
  - Also eliminates different impediment issues
  - Enables good interaction

SCRUM Framework Contd..

Scrum Artifacts

- Product backlog
  - Owned by Product Owner
  - All tasks are prioritized and sorted
  - “User Stories”
  - Provides complete information of tasks
  - Re-prioritized at end of each sprint

	Item #	Description	Est	By
Very High				
	1	**Finish database versioning**	16	KH
	2	**Get rid of unneeded shared Java in database**	8	KH
	-	**Add licensing**	-	-
	3	Concurrent user licensing	16	TG
	4	Demo / Eval licensing	16	TG
		Analysis Manager		
	5	File formats we support are out of date	160	TG
	6	Round-trip Analyses	250	MC
High				
	-	**Enforce unique names**	-	-
	7	In main application	24	KH
	8	In import	24	AM
	-	**Admin Program**	-	-
	9	Delete users	4	JM
	-	**Analysis Manager**	-	-
	10	When items are removed from an analysis, they should show up again in the pick list in lower 1/2 of the analysis tab	8	TG
	-	**Query**	-	-
	11	Support for wildcards when searching	16	T&A
	12	Sorting of number attributes to handle negative numbers	16	T&A
	13	Horizontal scrolling	12	T&A
	-	**Population Genetics**	-	-
	14	Frequency Manager	400	T&M
	15	Query Tool	400	T&M
	16	Additional Editors (which ones)	240	T&M
	17	Study Variable Manager	240	T&M
	18	Haplotypes	320	T&M
	19	**Add icons for v1.1 or 2.0**	-	-
	-	**Pedigree Manager**	-	-
	20	Validate Derived kindred	4	KH
Medium				
	-	**Explorer**	-	-
	21	Launch tab synchronization (only show queries/analyses for logged in users)	8	T&A
	22	Delete settings (?)	4	T&A

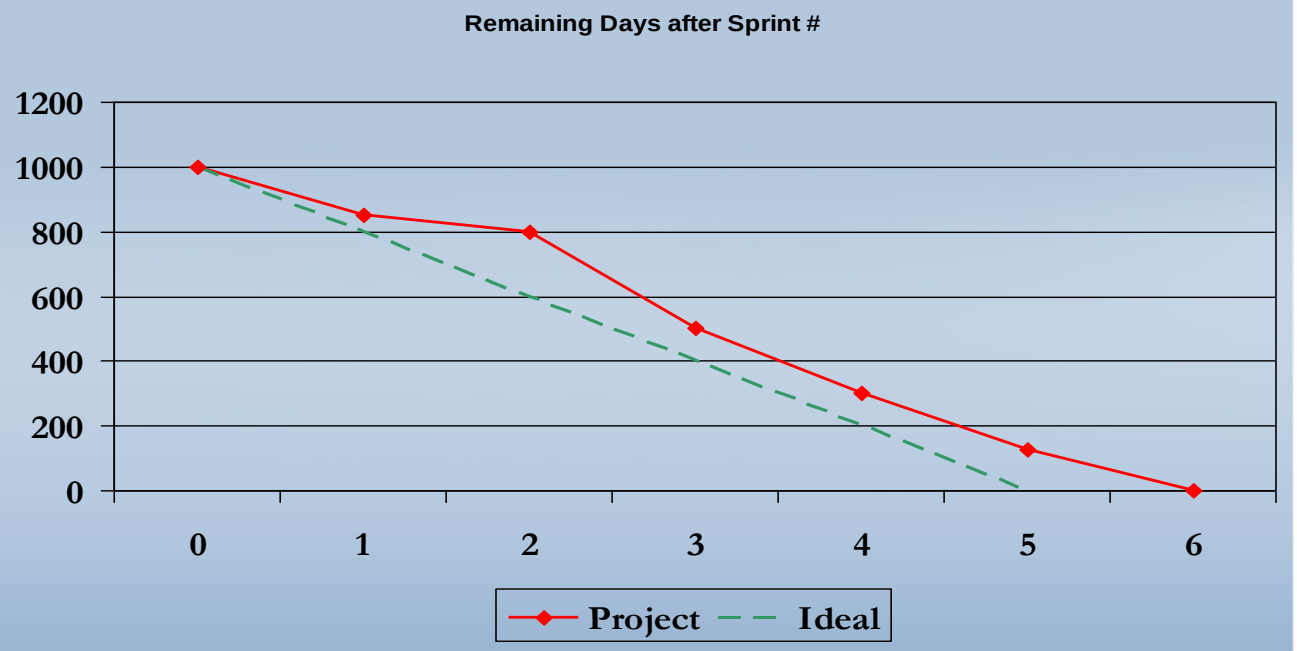
SCRUM Framework Contd..

- Sprint backlog
  - List of tasks to be achieved in particular sprint.
  - Owned by the scrum team
  - Includes all details
  - Updated periodically

	Days Left in Sprint	15	13	10	8	
						F
Who	Description	7/22/2002	7/24/2002	7/26/2002	7/31/2002	
	Total Estimated Hours:	554	458	362	270	0
-	**User's Guide**	-	-	-	-	-
SM	Start on Study Variable chapter first draft	16	16	16	16	
SM	Import chapter first draft	40	24	6	6	
SM	Export chapter first draft	24	24	24	6	
	Misc. Small Bugs					
JM	Fix connection leak	40				
JM	Delete queries	8	8			
JM	Delete analysis	8	8			
TG	Fix tear-off messaging bug	8	8			
JM	View pedigree for kindred column in a result set	2	2	2	2	
AM	Derived kindred validation	8				
	Environment					
TG	Install CVS	16	16			
TBD	Move code into CVS	40	40	40	40	
TBD	Move to JDK 1.4	8	8	8	8	
	Database					
KH	Killing Oracle sessions	8	8	8	8	
KH	Finish 2.206 database patch	8	2			
KH	Make a 2.207 database patch	8	8	8	8	
KH	Figure out why 461 indexes are created	4				

SCRUM Framework Contd..

- Burn-down Chart:
  - Visual representation of team progress in a sprint
  - Work remained in the Sprint



SCRUM Meetings

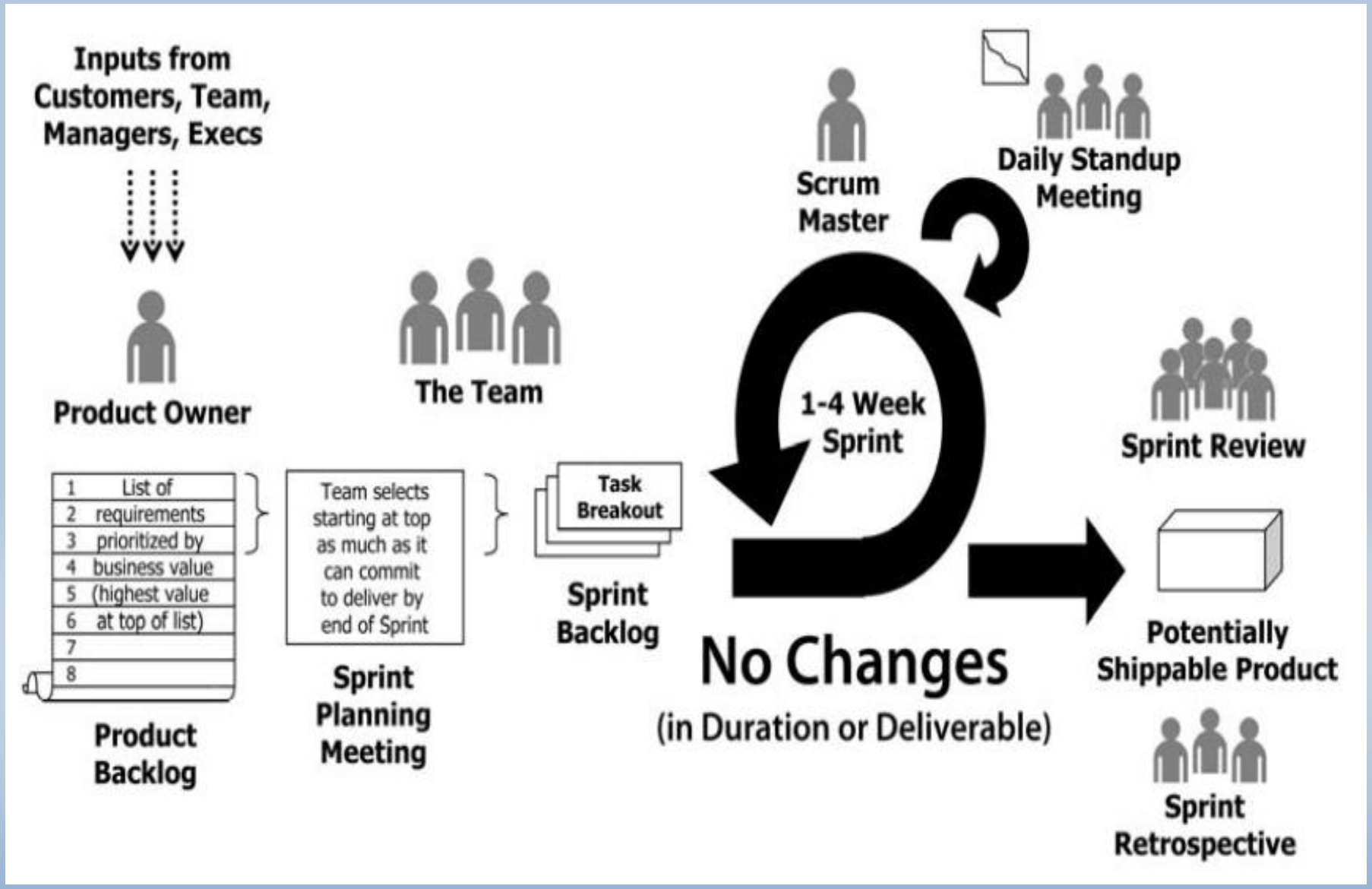
- Sprint planning meeting
  - Time-boxed and lasts for maximum of 6 to 7 hours.
  - Product owner and team form this meeting
  - Prepares sprint backlog by Scrum master and team.

- Daily Scrum
  - *What has he done since their last meet?*
  - *What will he do before the next meet?*
  - *What are the impediments he came across during his part of work?*

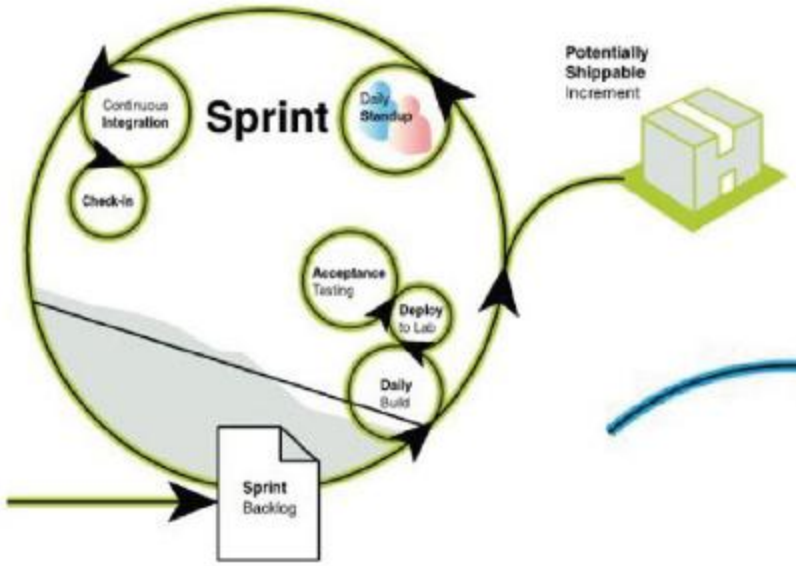
Contd..

- Sprint review meeting
  - Time boxed to four hours.
  - Team demonstrates product increment to product owner, stake holders and customers.
  - Main agenda is to review the tasks done in particular sprint
- Sprint retrospective
  - Time boxed to three hours.
  - Team, Scrum Master, and (optionally) Product Owner review the last Sprint
  - What went well? What didn't go well ? and What can be improved?

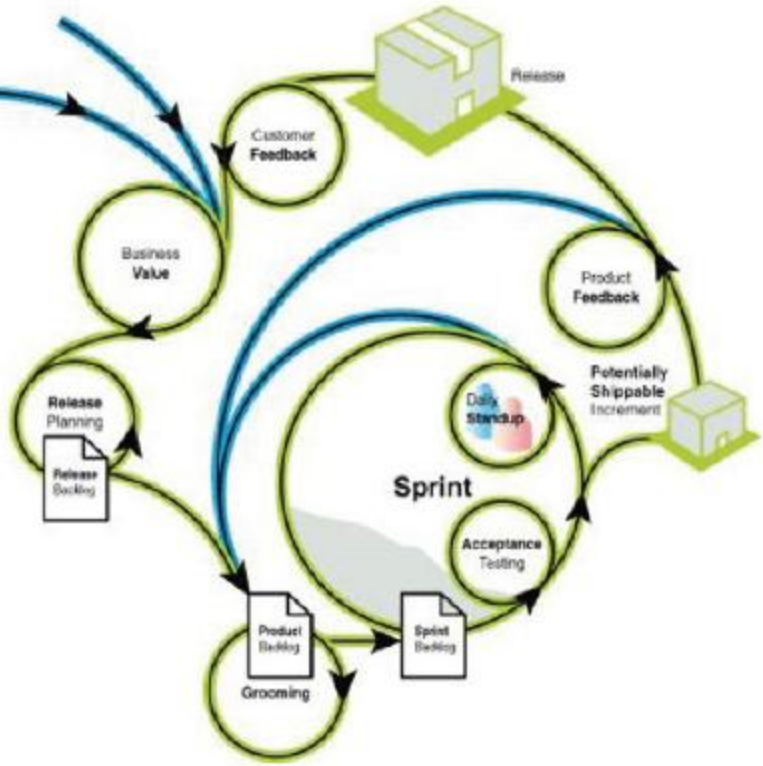
Complete SCRUM



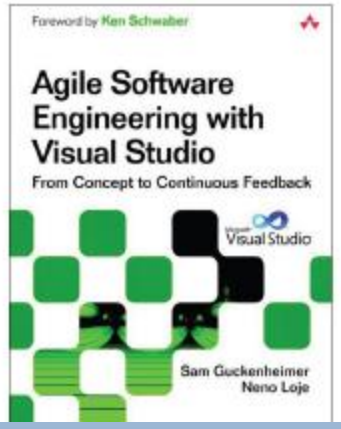
Expanded Scrum Cycle



Product Owner



Engineering Practices



Product Backlog

Sprint 1: Goal: Understand the Process, People, and Tools in SCRUM

Sprint 2: Goal: Scrum your Scrum

Sprint Backlog:

- Agile 10 Year Retrospective
- Toyota's A3 Process
- Improve your Scrum Process
- Exercise 2
- The whole story of "User Story"
- Exercise 3 (Part 1)

Sprint 3: Goal: Plan and Estimation in SCRUM

Sprint 4: Goal: Understand Advanced Scrum Topics

Resolving Difficult Impediments

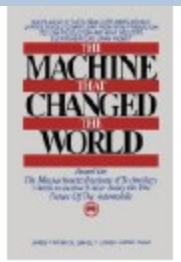
Taxonomy of Waste

- Lack of focus on removing impediments
- Inadequate testing (no working software)
- Priorities and specifications (product backlog issues)
- Interruptions
- Waterfall tradition (mindset)
- Technical Debt
- Dependence on key personnel
- Source code branches (merging/refactoring)
- Lack of team rooms
- Handling organizational issues
- Lack of customer focus
- Too many discussions without action

Agile 10 Year Retrospective

Snowbird 2011

- Demand technical excellence
- Promote individual change and lead organizational change
- Organize knowledge and improve education
- Optimize the whole value chain



9th Hidden Turning Point in History

U.S. News and World Report, 21 Apr 1991 - see also J. Womack, D. Jones, D Roos, "The Machine that Changed the World: The Story of Lean Production." Harper Perennial, 1991.

- W. Edwards Deming taught the Japanese the PDCA cycle.
- PLAN: In Scrum, the Product Owner has a business plan and needs to execute it in a way that maximizes stakeholder value.
- DO: The ScrumMaster owns the process and facilitates the team that executes the plan.
- CHECK: The Product Owner inspects the results of team work in short cycles.
- ACT: The ScrumMaster facilitates a retrospective where the team discovers how to produce better results in the next cycle.
- Continuous improvement is the outcome of the PDCA cycle. Continuous improvement and respect for people is the Toyota Way as well as the Scrum Way.

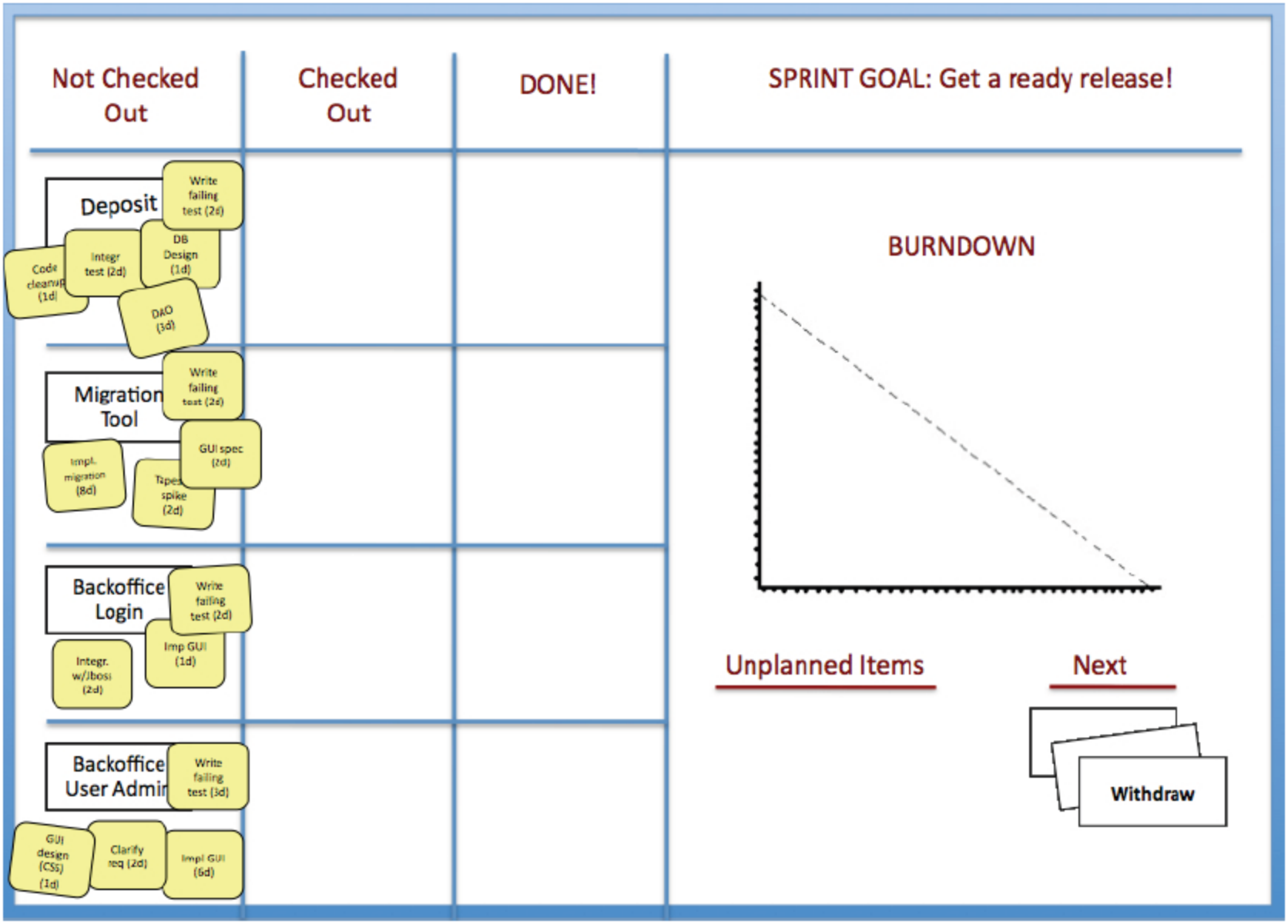
Daily Scrum Meeting

15 minutes - try this experiment

- What did you do yesterday to get the top priority story done (tested)
- What will you do today to get the top priority story done
- What are impediments that prevent the top priority story getting done today

Photo: Henrik Kniberg

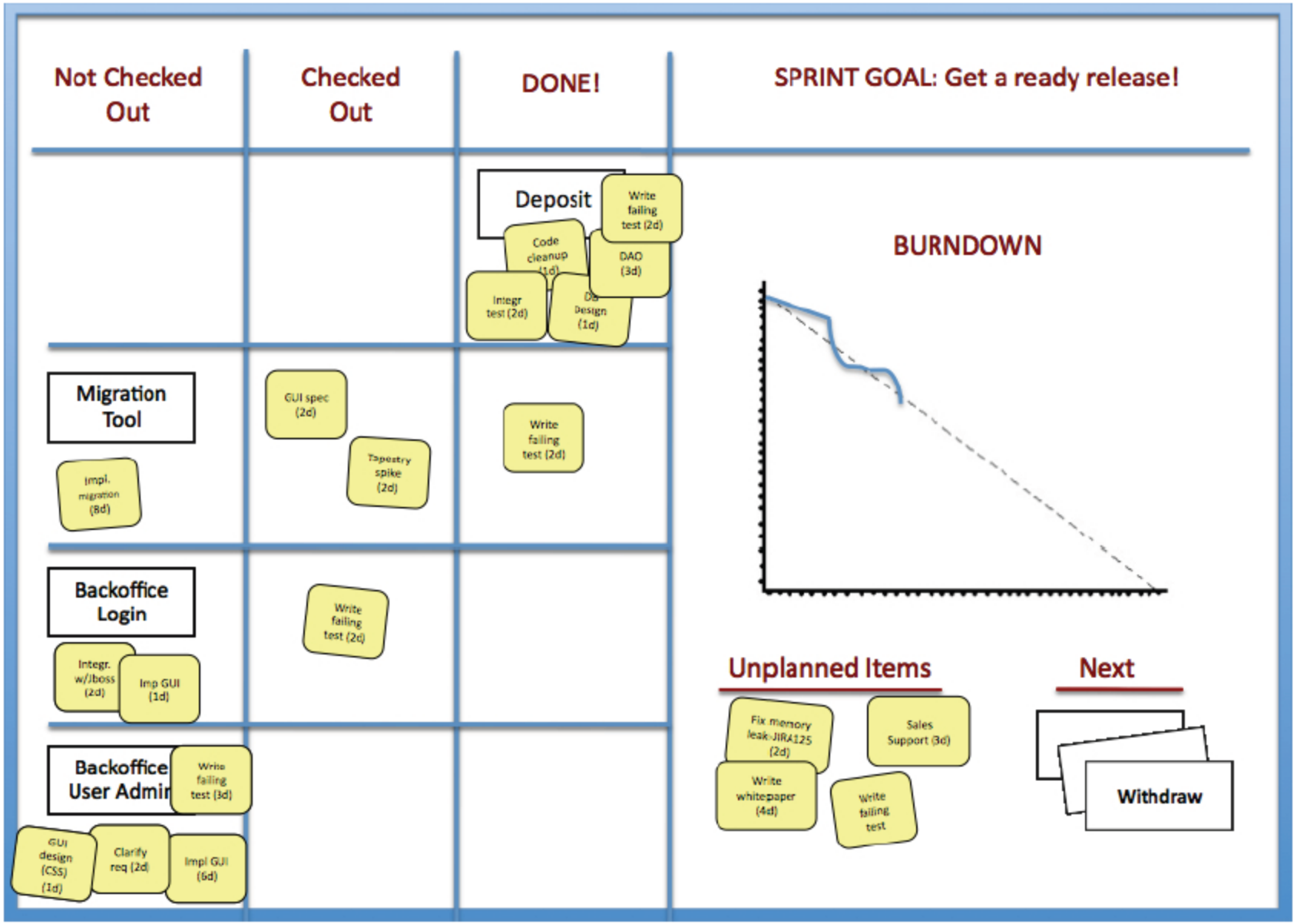
Sprint backlog – day 0



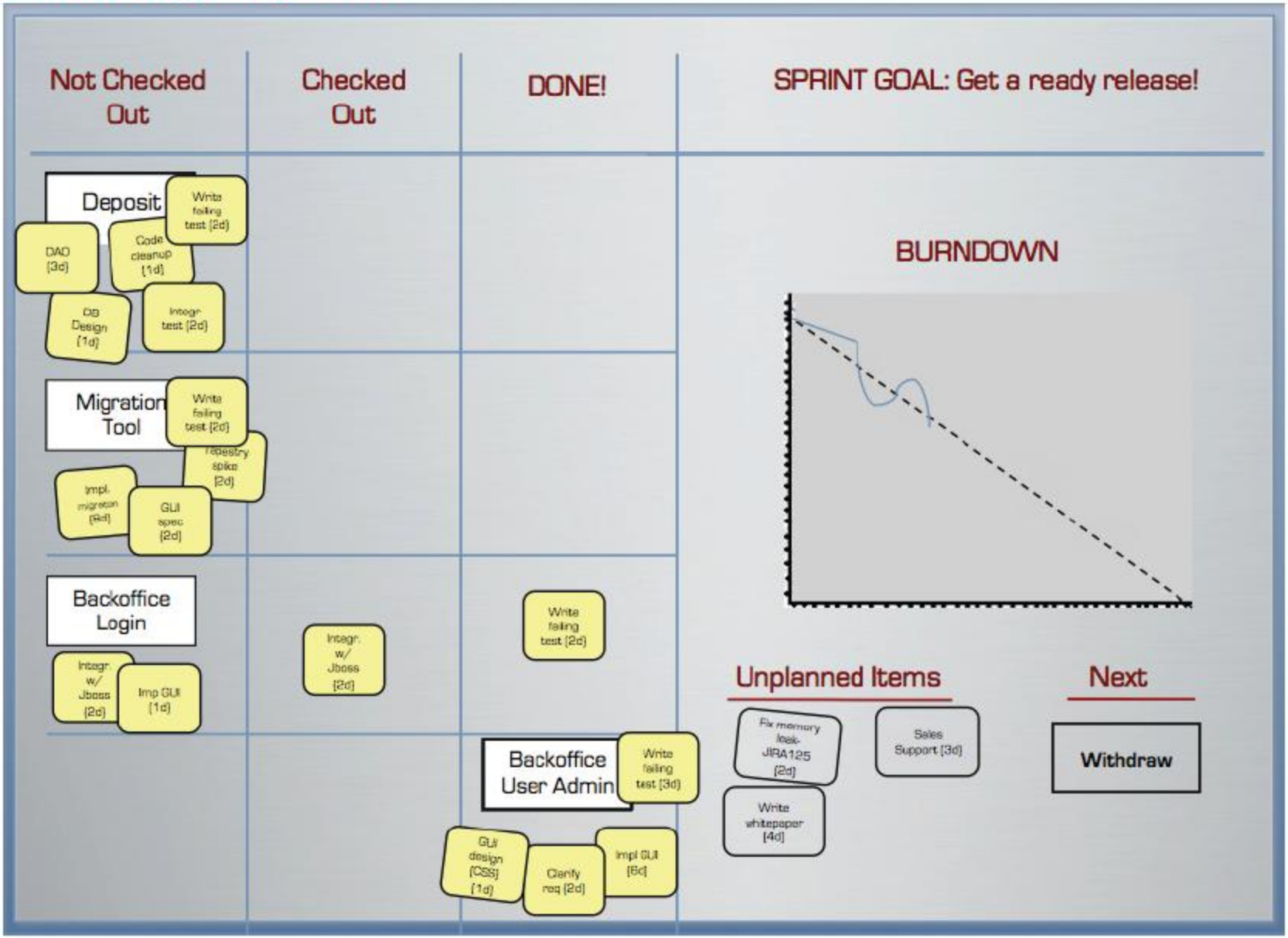
Sprint backlog – after 1st meeting

Not Checked Out	Checked Out	DONE!	SPRINT GOAL: Get a ready release!
Deposit Code cleanup (1d) Integr test (2d) DAO (3d)	Write failing test (2d) DB Design (1d)		BURNDOWN
Migration Tool GUI spec (2d) Impl. migration (8d) Tapestry spike (2d)	Write failing test (2d)		
Backoffice Login Write failing test (2d) Imp GUI (1d) Integr. w/Jboss (2d)			Unplanned Items Next Withdraw
Backoffice User Admin Write failing test (3d) GUI design (CSS) (1d) Clarify req (2d) Impl GUI (6d)			

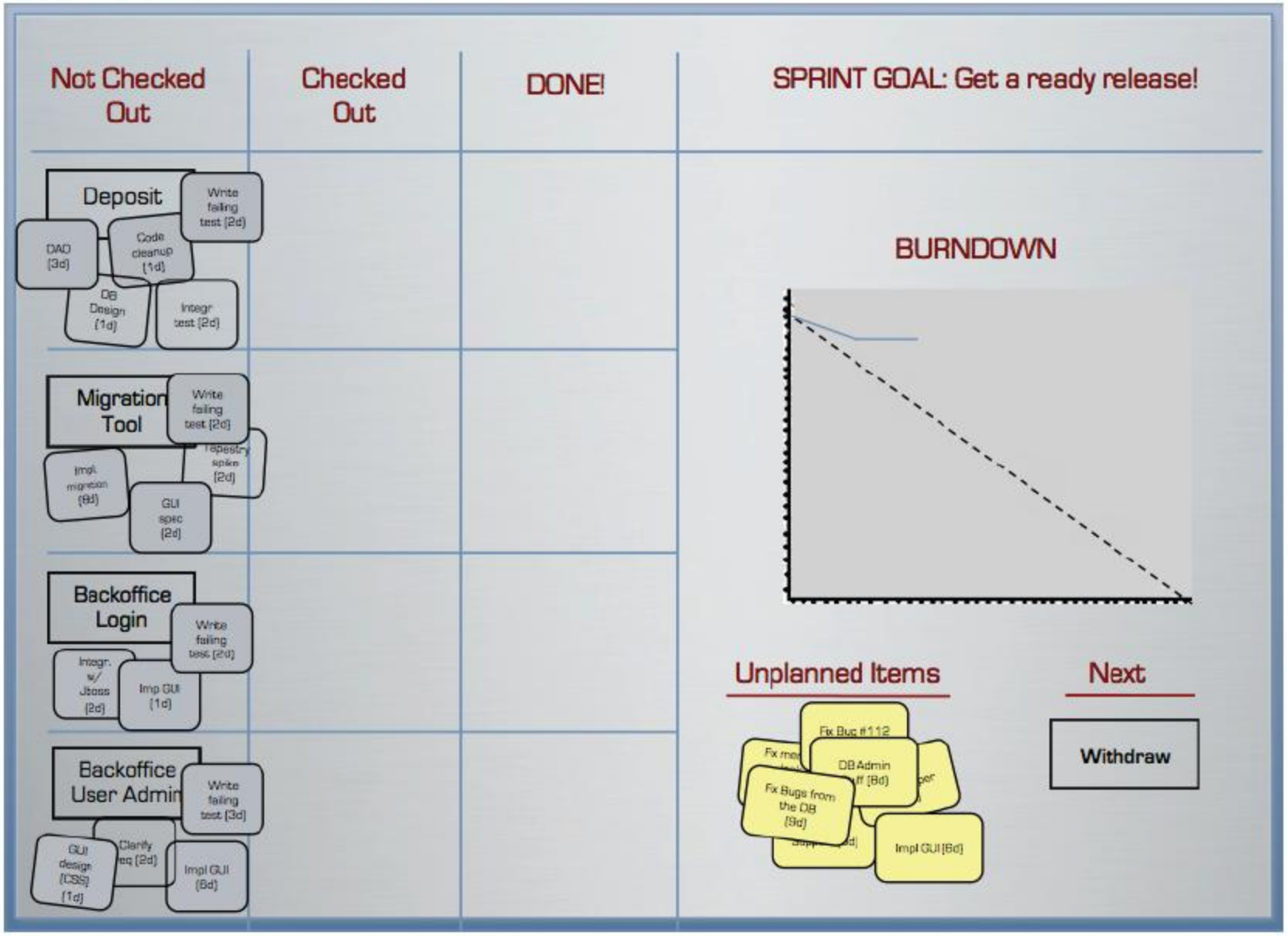
Sprint backlog – day X



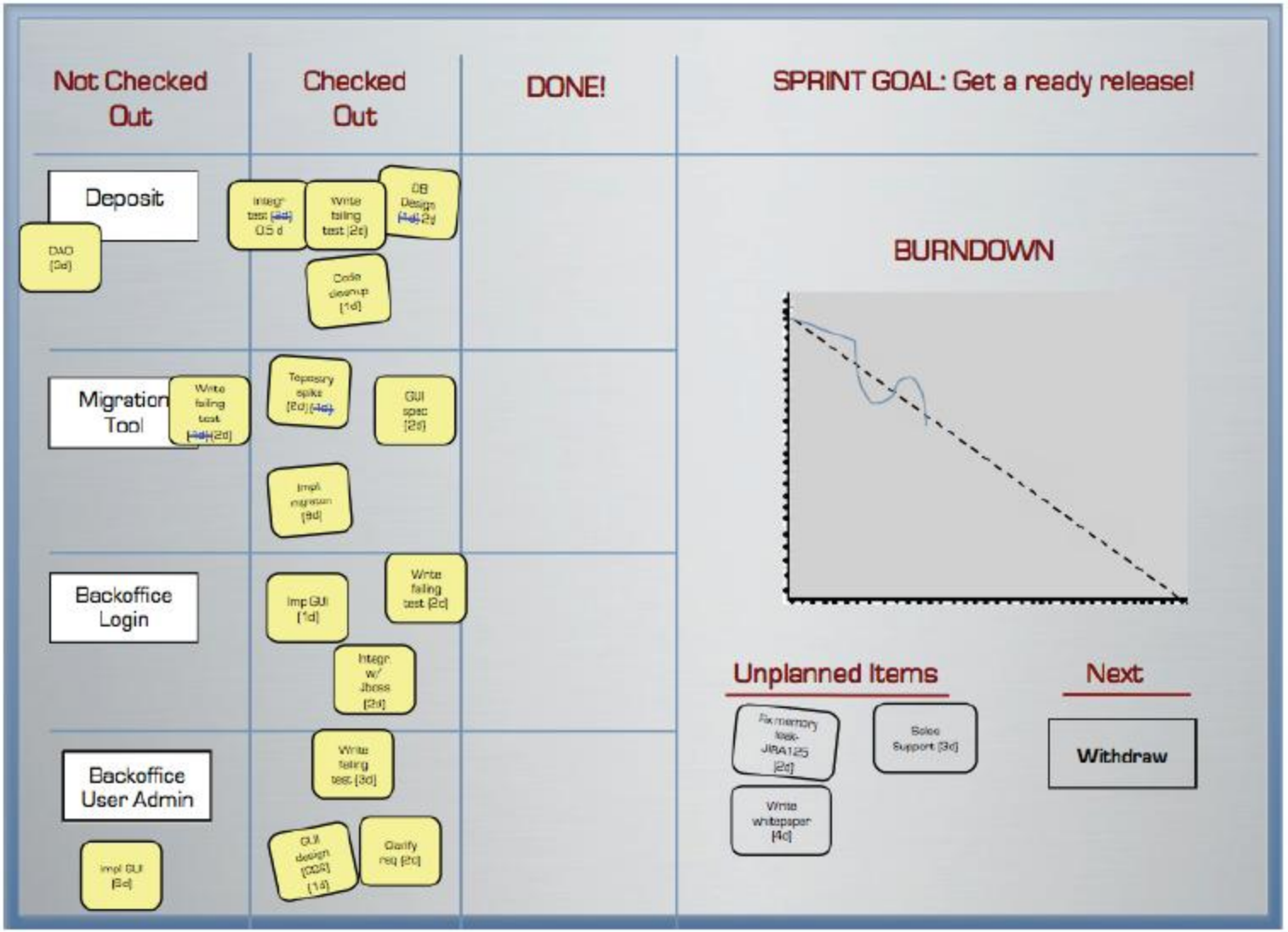
Warning sign #1



Warning sign #2



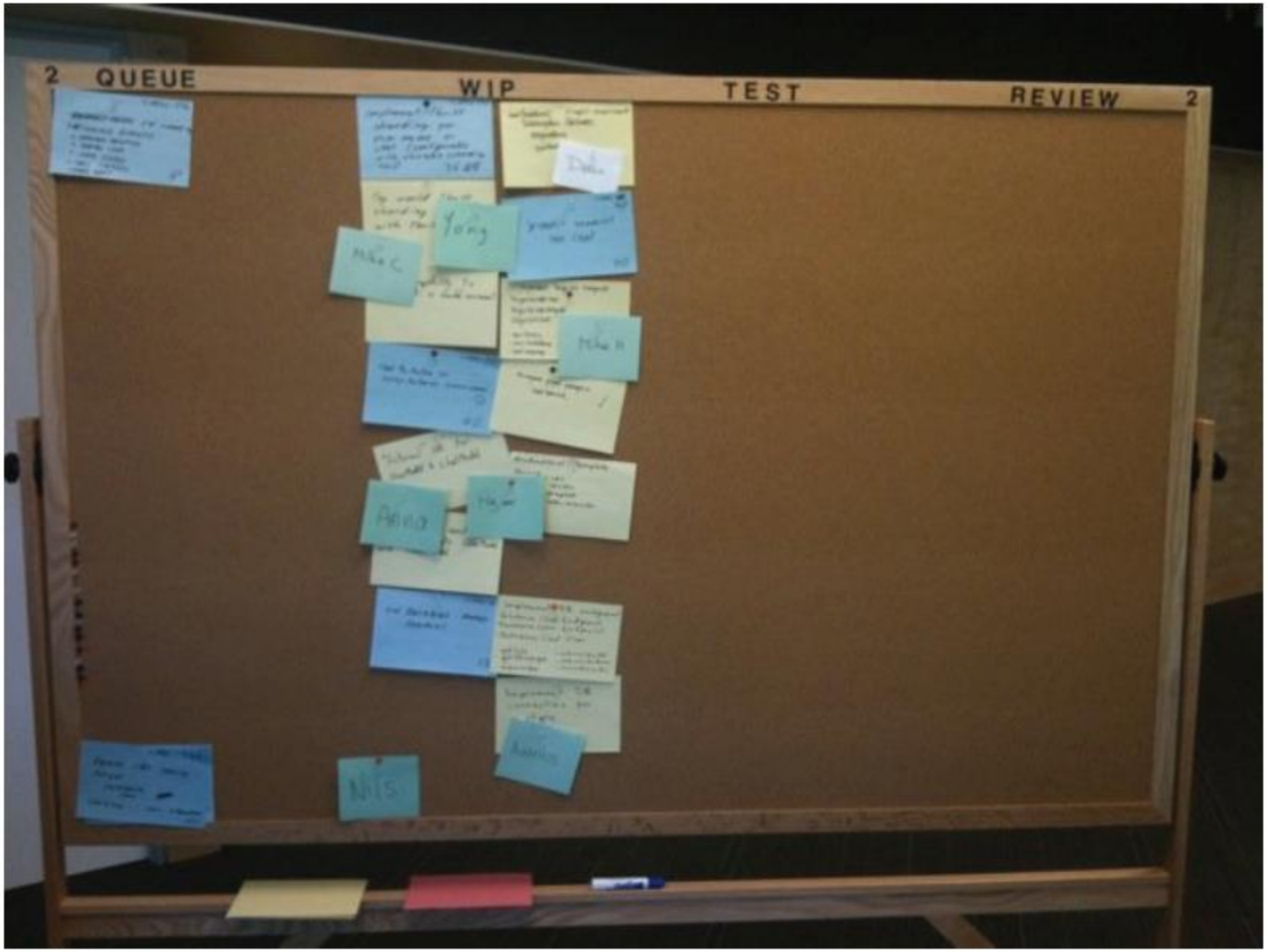
Warning sign #3



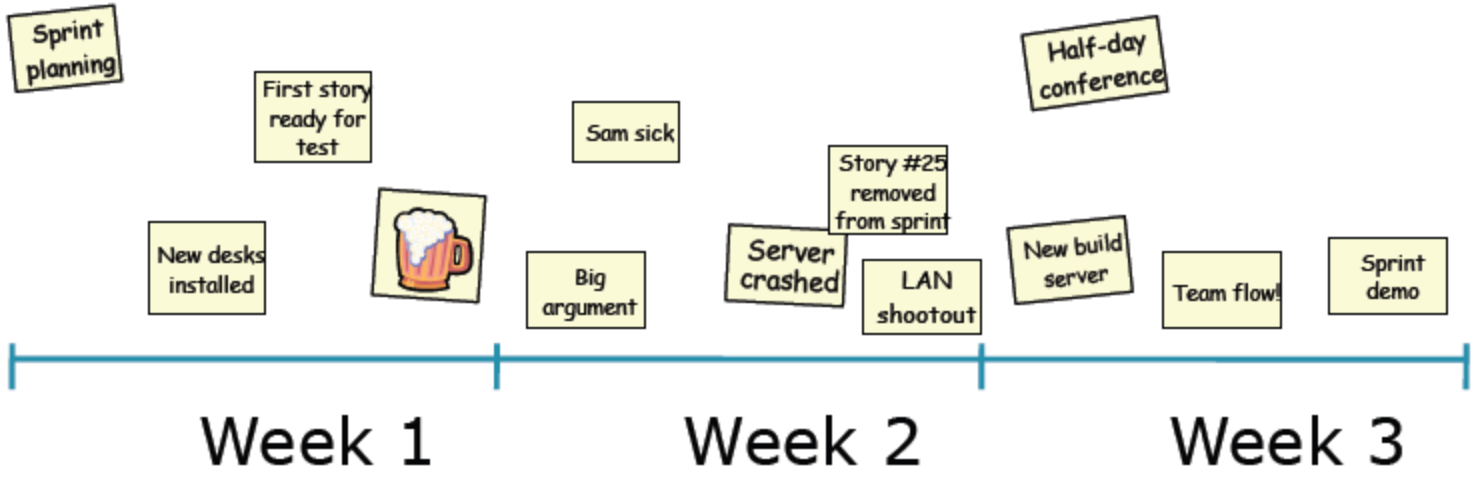
WAIT A SEC

How is that burndown calculated?

Swimlane Scrum



Sprint Retrospective



Sprint Retrospective

3 roles
- Product owner
- Scrum master
- Team

3 artifacts
- Product backlog

Sprint backlog
Sprint burndown

activities

Sprint planning
Daily scrum
Sprint review
- Demo
- Retrospective

"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

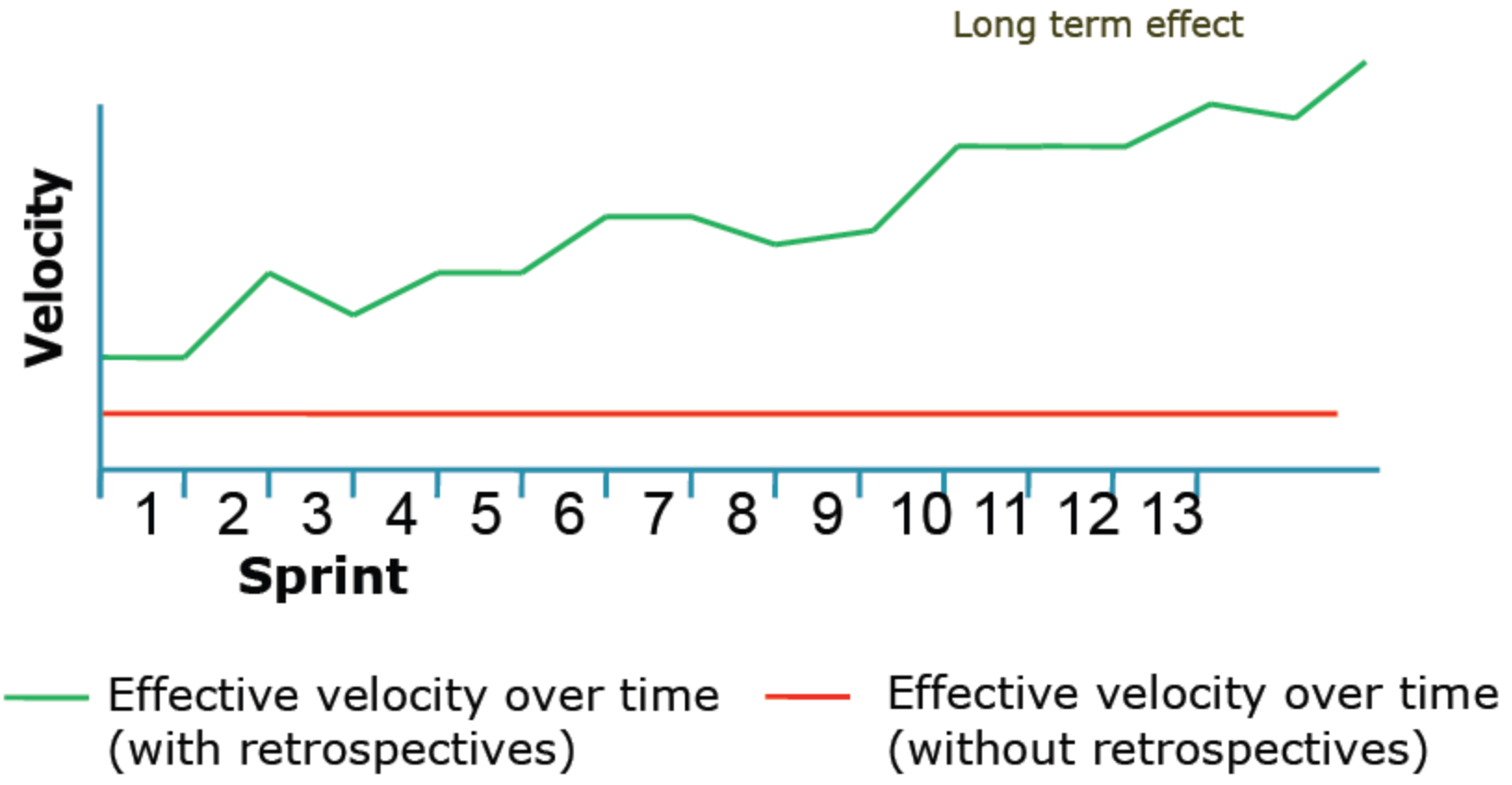
- And the hardest rule: Use common sense.

Question Asked in Retrospectives

- Rank your own work? 1-5; Rank the org? 1- 5; How can I do better?
- As a team how should we improve next sprint?

Scruming the Scrum!

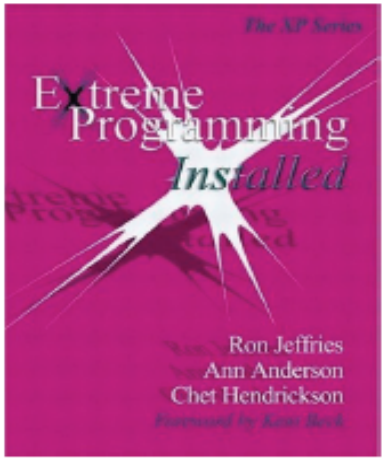
Sprint Retrospective



PRODUCT BACKLOG AND USER STORIES

Increase Granularity of Features

- User Stories derive from XP
  - now widely used for all agile processes
- Card
  - template, notes
- Conversation
  - face to face communication is part of the specification
- Confirmation
  - acceptance tests



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 a <user role> I can <story> so that <benefit>
- As a <person name> can < story > so that <benefit>
- As a <user> I want to <goal> so that <value to attain>

The “so that” line is generally considered optional, but used as a default

Break Epics into Stories

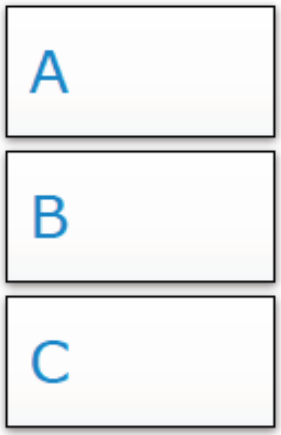
As a frequent flyer I want to book flights customized to my preferences, so I save time

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

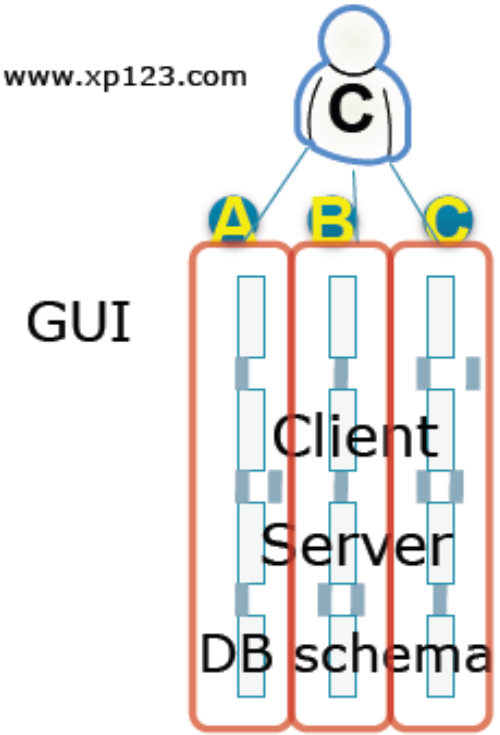
As a premium frequent flyer I want to request an upgrade So I can be more comfortable

User Story - Guidelines



Immediately actionable
Negotiable
Valuable
Estimable
Sized to fit
Testable

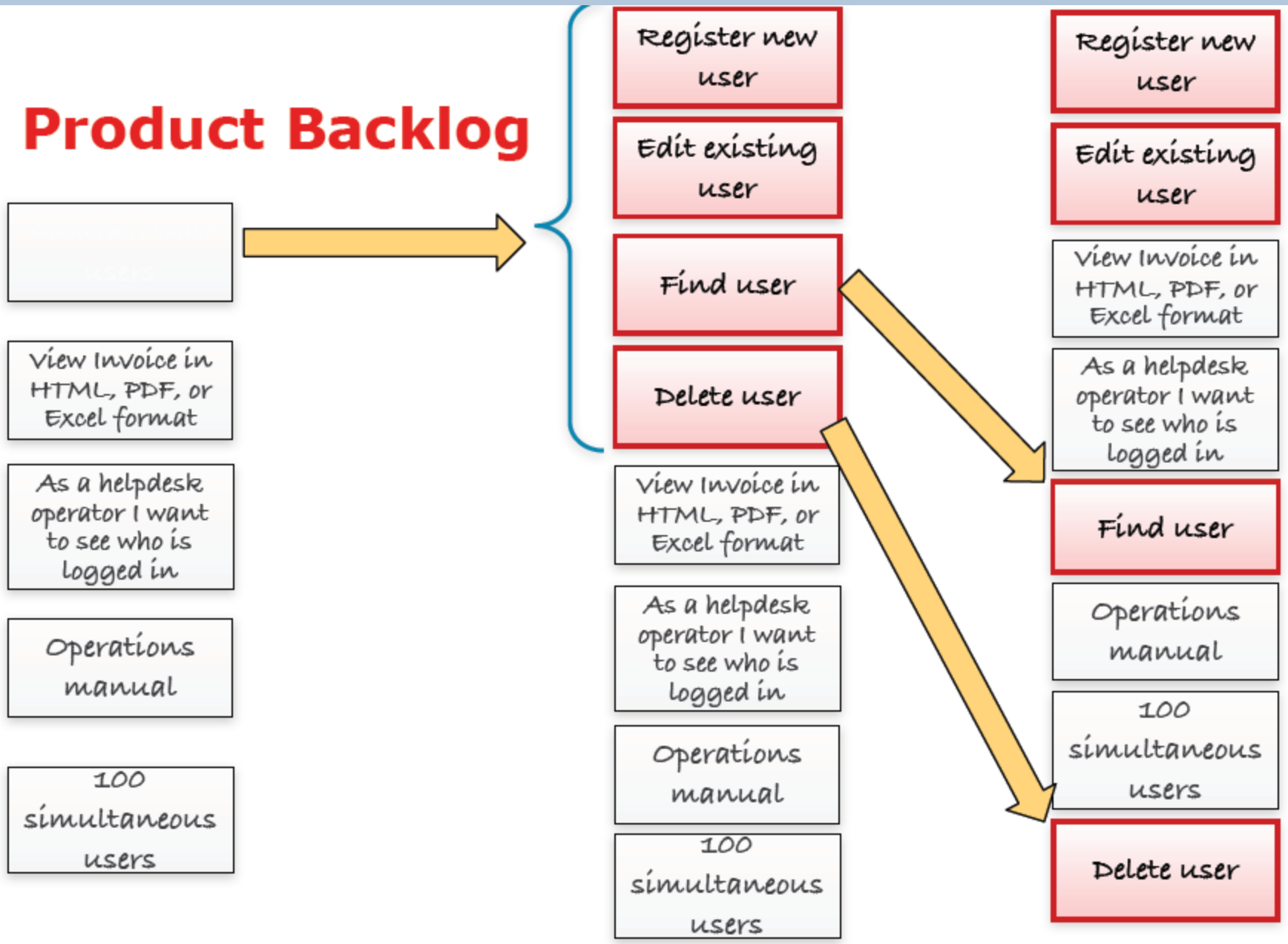
Modified from Bill Wake – www.xp123.com



Modern Agile Acceptance Model

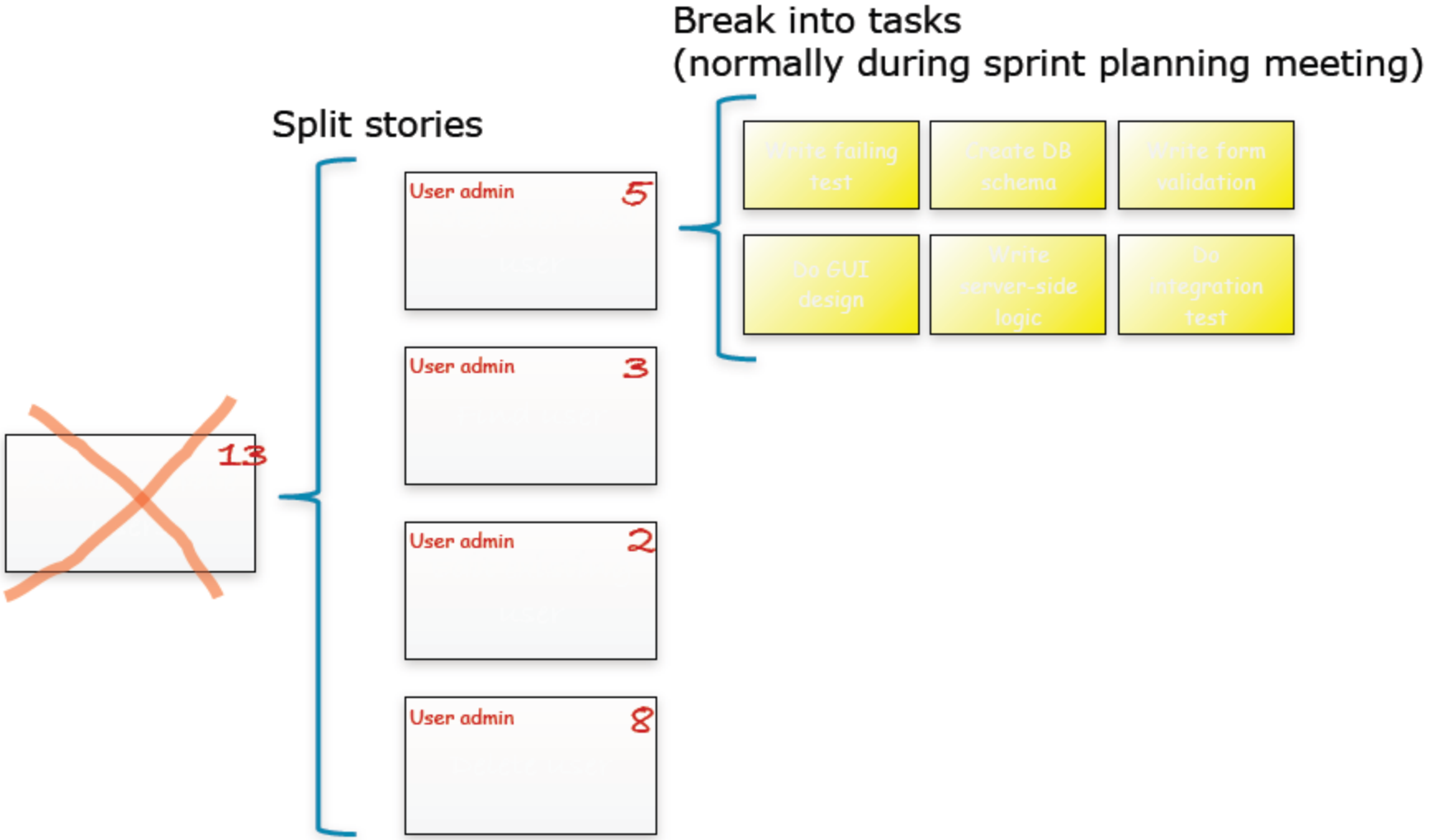
- The move toward Acceptance Test Driven Development requires a more complete model of progressing requirements acceptance
- Example model (sharper definition of done)
  - Conditions of Satisfaction - broad terms
    - Acceptance Criteria - further refined
      - Examples - actual scenarios or data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

Product Backlog



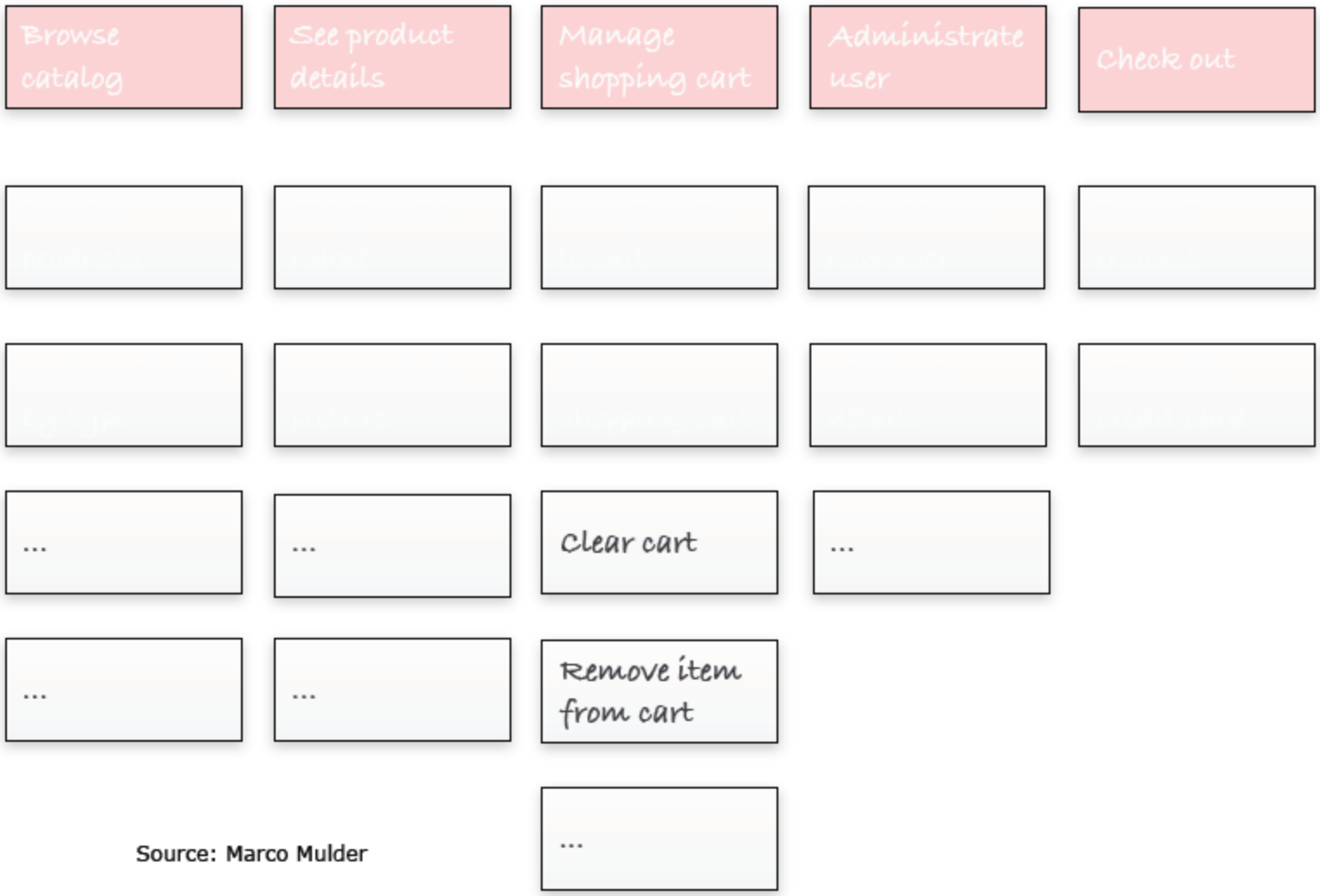
Source: Henrik Kniberg

Splitting Stories and Breaking Out Tasks



Source: Henrik Kniberg

Story Map

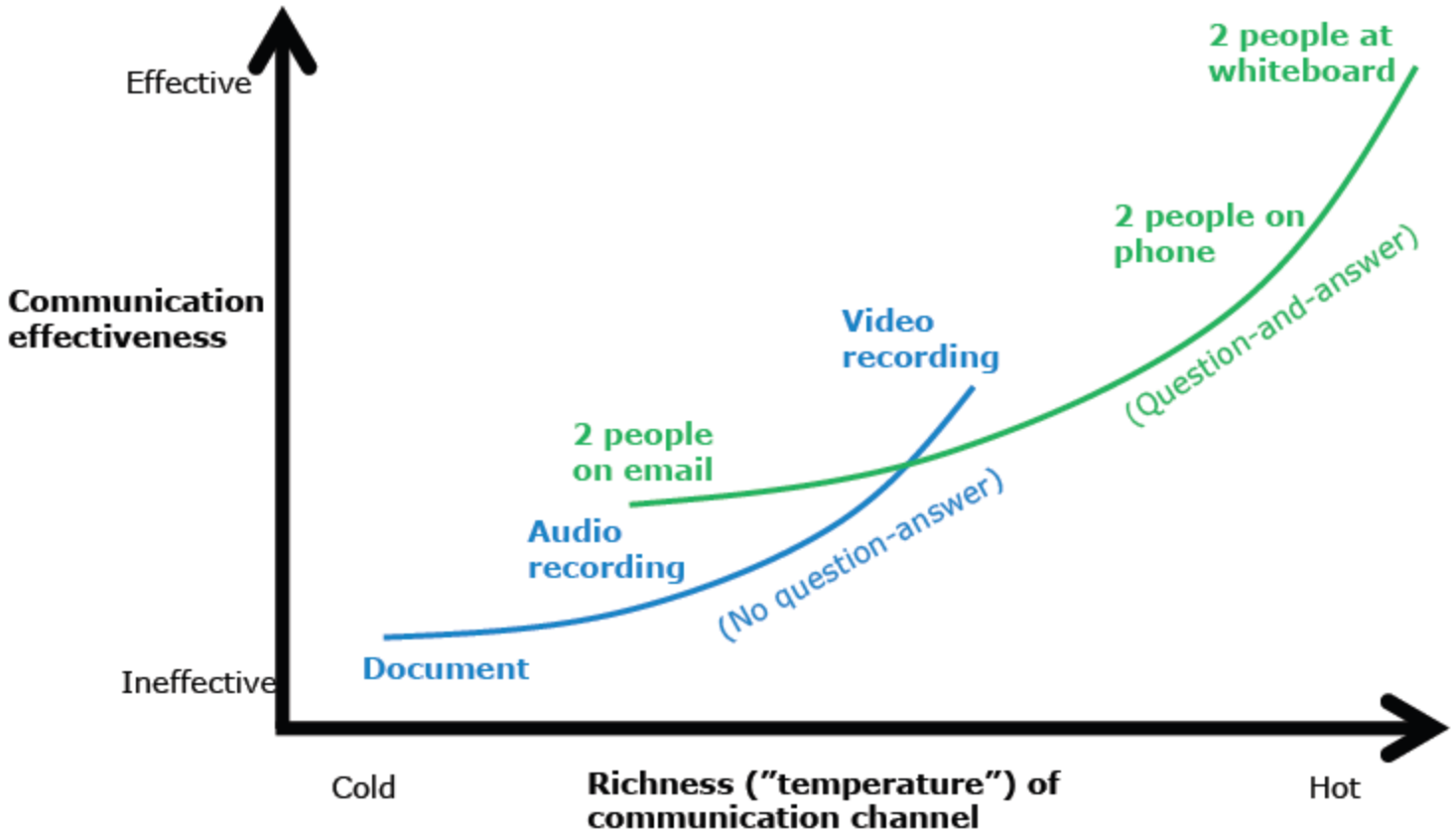


Source: Marco Mulder

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

Communication Effectiveness



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: research from McCarthy and Monk (1994)

Product Backlog

Sprint 1: Goal: Understand the Process, People, and Tools in SCRUM

Sprint 2: Goal: Scrum your Scrum

Sprint 3: Goal: Plan and Estimation in SCRUM

Sprint Backlog:

- Agile Estimation
- Exercise 2 (Part 2) Releasing Planning
- Story of the “Velocity”
- “Roles of Responsibility
- The whole story of “User Story”
- Managing a Sprint
- Fast but Clean (Velocity and Technical Debt)

Sprint 4: Goal: Understand Advanced Scrum Topics

Agile Estimating Strategy

- Don’t estimate time
  - Estimate relative size of stories
  - Measure velocity per sprint
  - Derive release plan
- Estimates are done by the people who are going to do the work
  - Not by the people who want the work done
- Estimate continuously during the project, not all up front
- Prefer verbal communication over detailed, written specifications

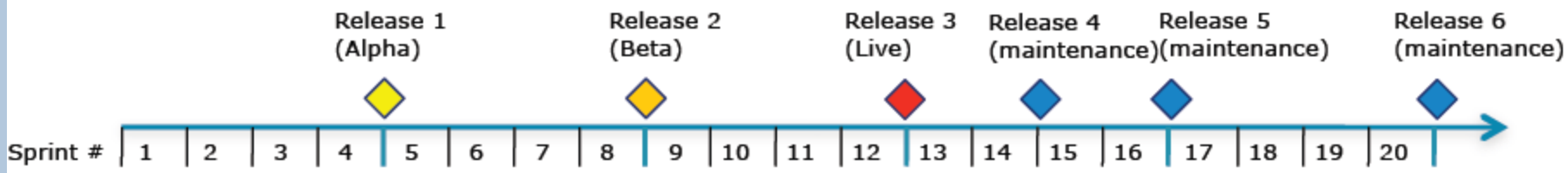
Planning Poker

http://planningpoker.crisp.se

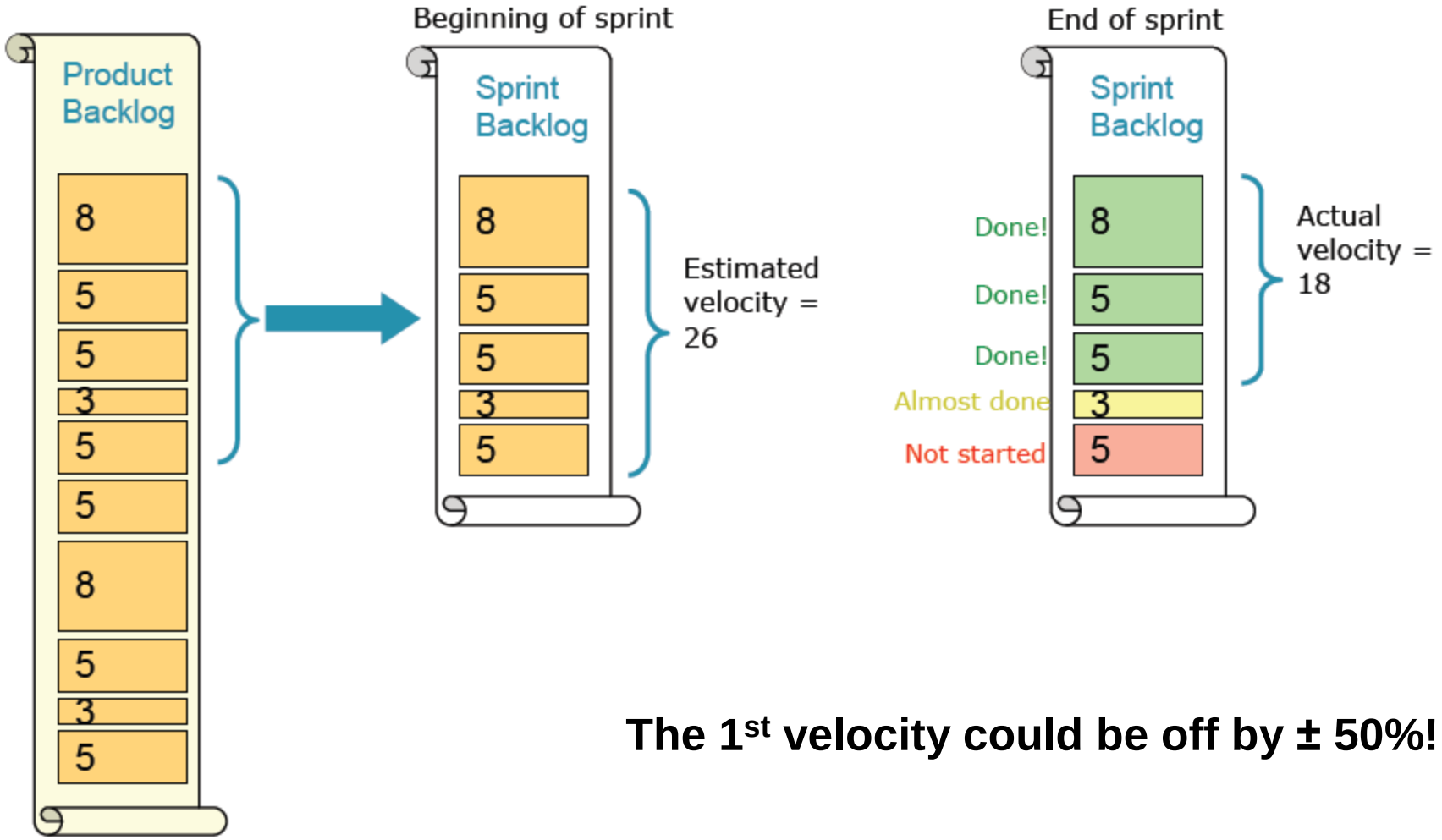
PRODUCT RELEASE PLANNING

Release Cycles

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

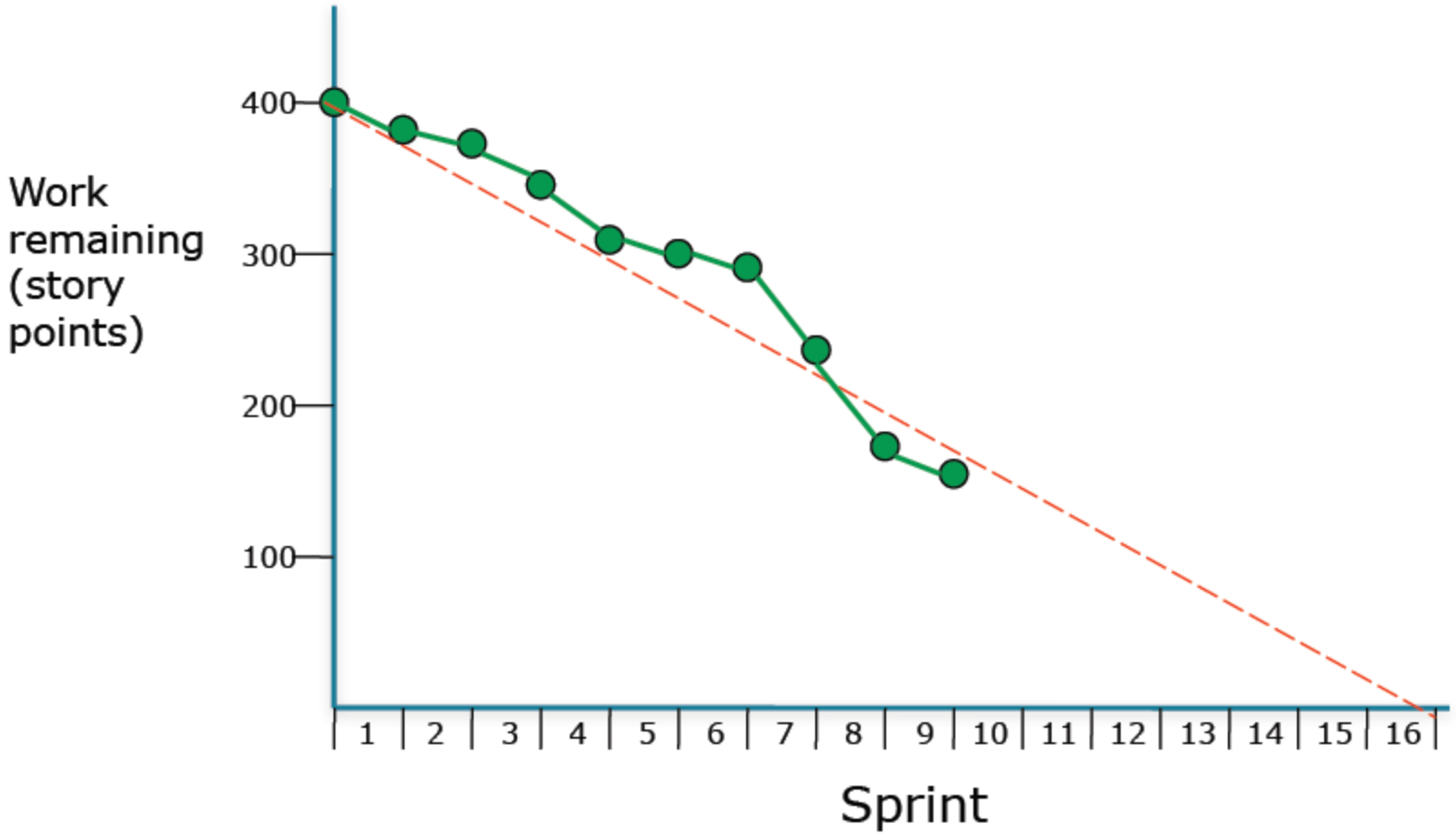


Measuring Velocity

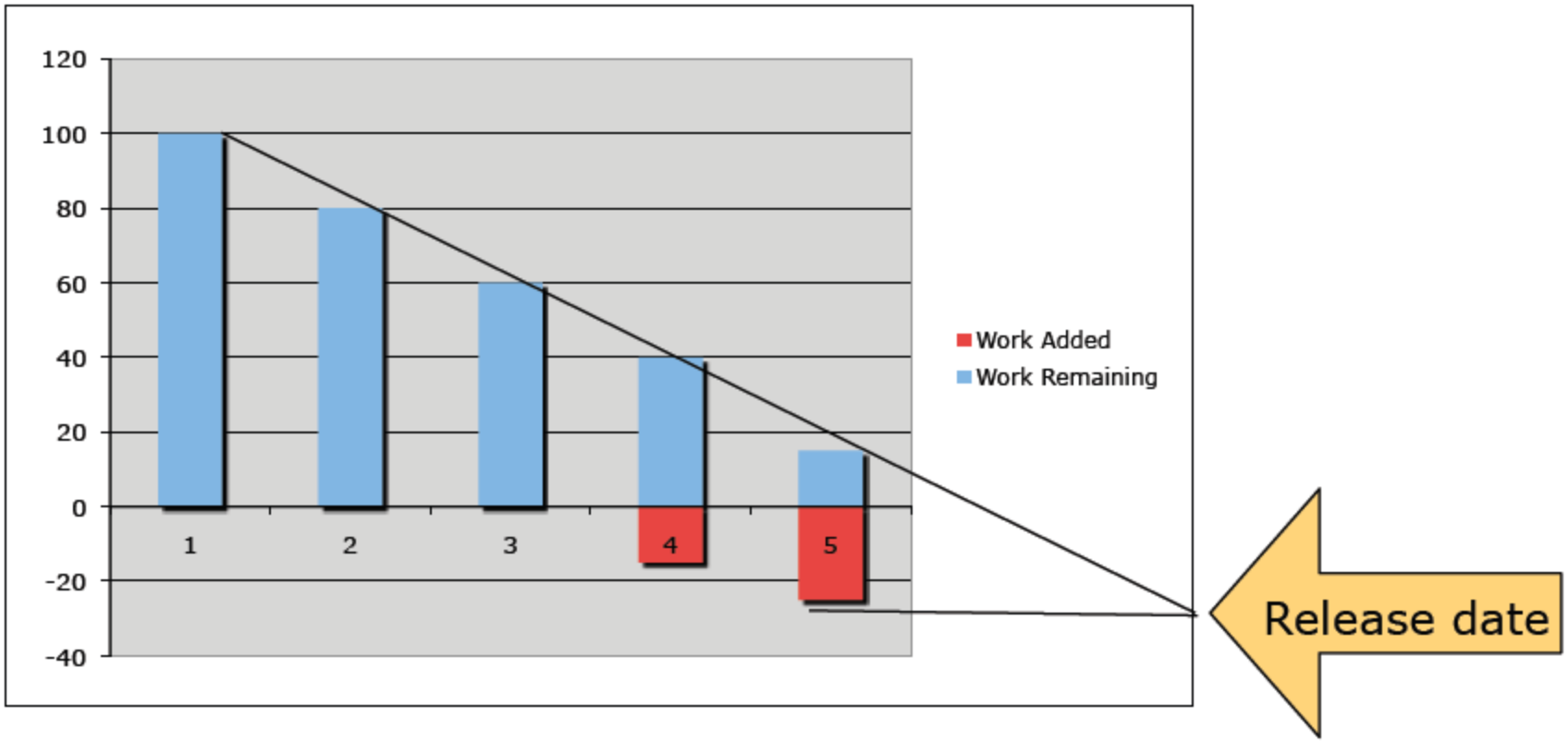


The 1st velocity could be off by ± 50%!

Release Burndown



Release Burndown with Changing Scope



What if this is the first sprint for the team?

- Good rule of thumb is 40% burnup cost.
- 40% of your story points will be used to get the team operational.
  - Team Forming, Storming, Norming
  - Team not understanding Scrum
  - Team thinking they can do more than they can do

Product Owner Responsibility

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

Product Owner

- Represents all stakeholders
- Decides where the team should go
  - Not how they get there
  - Not their speed
- Defines scope / vision / roadmap
- Prioritizes
- Owns product backlog

Does not estimate stories

Usually not the line manager

Who is a Product Owner?

- Mishkin Berteig has suggested the following definition:

  *”It is the Product Owner who is **ultimately responsible for defining business value, priority and details** for all the work done by the development team. Product Owner’s authority stems from a deep understanding of the project’s goals and a respected position among stakeholders.”*

- Typically a project or product manager in a customer organization

- The link between an Agile development team and stakeholders
  - PO has to be in direct contact to both directions

Central Success Criteria

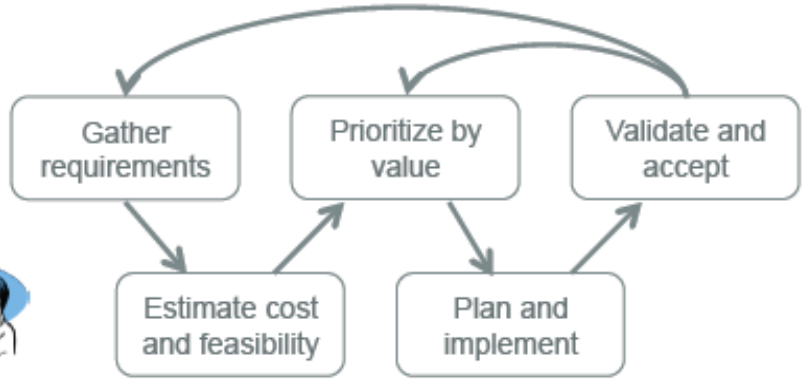
- One PO per project
  - When there are multiple PO's, conflicts regarding requirements and priorities are easily developed
- Ability to represent all stakeholders and user groups
  - All important needs are heard
  - Can meaningfully prioritize different needs
- Authority to make business decisions regarding project's features and priorities
- An active participation in the project
  - Requirements development and prioritization throughout the whole project
- Ability to increase Agility in the project's environment
- Focus on business value and features, not on micromanaging the team

Primary Responsibilities

- Define requirements / features
  - Business point-of-view
  - User point-of-view
  - Communication to the team
- Prioritization
  - Defining business value
  - Value / cost determination
  - Risk management
- Planning the project scope
- Approval of team's deliveries
- Evaluates project progress and results
- Collects feedback from
  - Stakeholders
  - User groups

Team and Product Owner

Clear definition of responsibilities



Product Owner

- Plans overall project schedule
- Defines requirements
- Defines business value and implementation priority

Team

- Provides estimates on effort and technical risk
- Provides information on development speed
- Commits to agreed sprint scope and delivering it

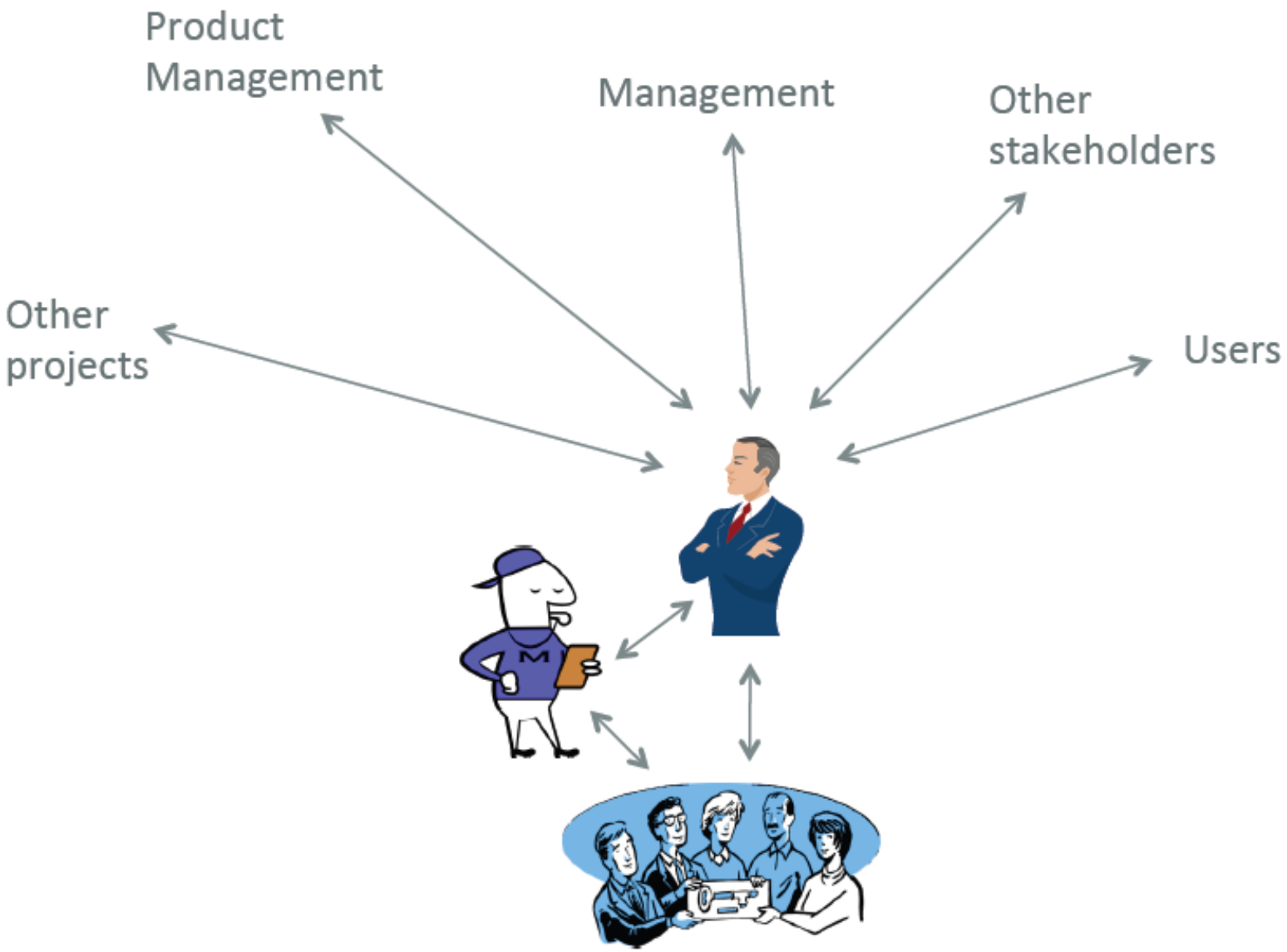
Direct communication absolutely necessary

- Team <-> Product owner
- Team <-> Users and other stakeholders

Regular meetings to set sprint scope and to demonstrate results for feedback

- Also other communication as necessary to clarify requirements

Stakeholders



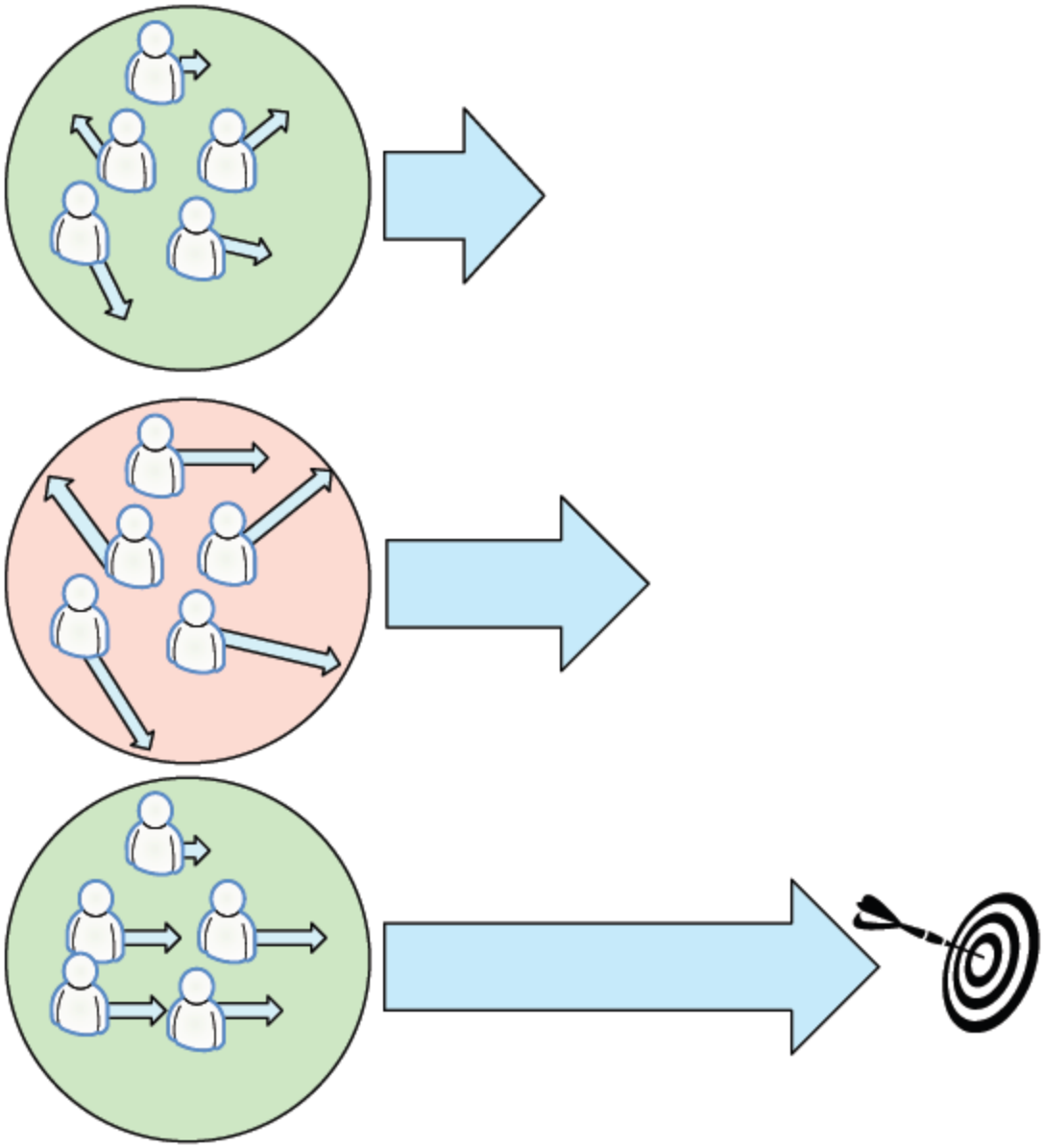
Basic Truths about Teams

- 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

Team

- 7 +/-2 full-time individuals
- Cross-functional
- Sits together
- Shared responsibility
- Self-organizing

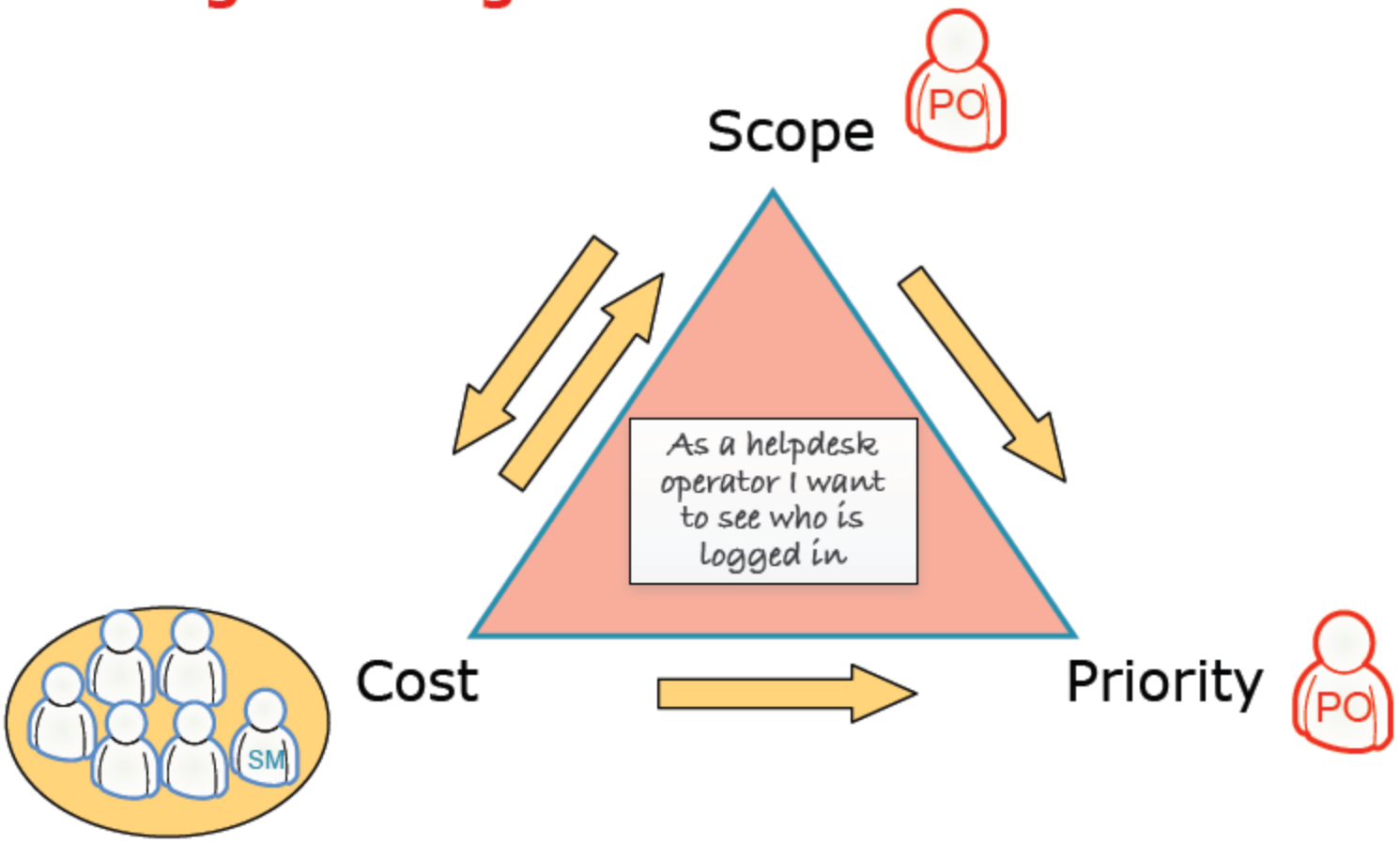


Source: Henrik Kniberg

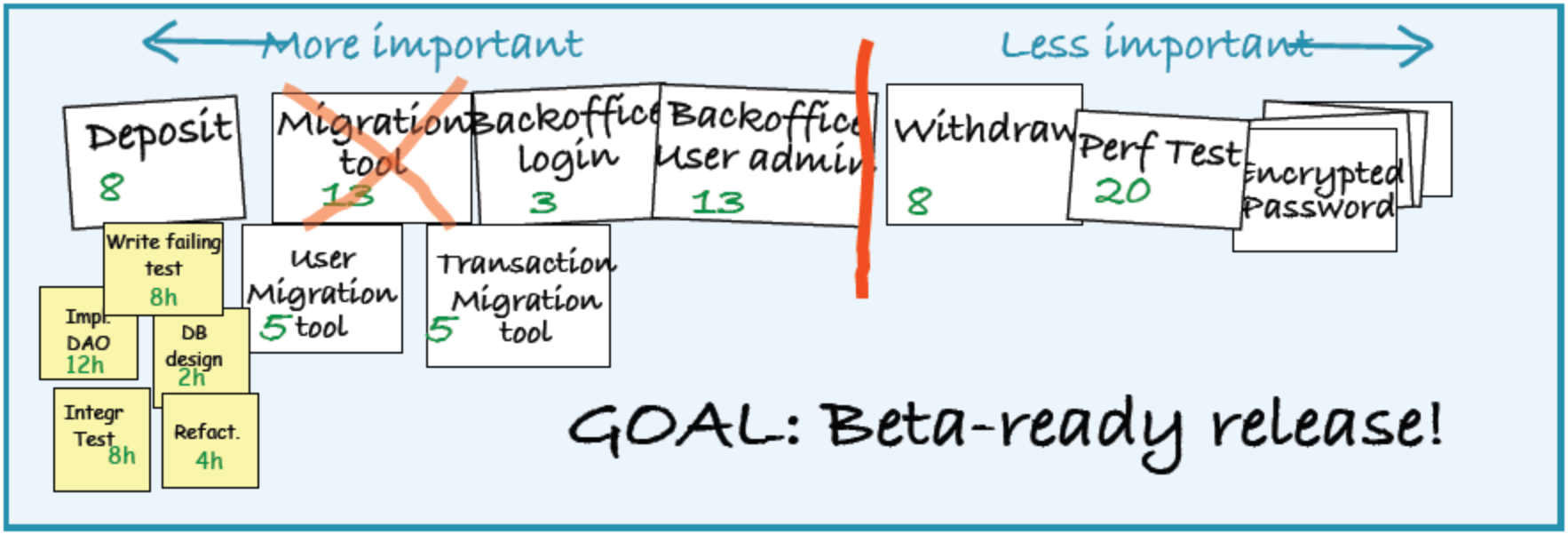
Management Responsibilites

1. Provide strategic vision, business strategy, and resources (business model).
2. Remove impediments surfaced by Scrum teams that the teams cannot remove themselves.
3. Ensure growth and career path of employees.
4. Challenge teams to move beyond mediocrity.

Why is it important that the team AND product owner attend the sprint planning meeting?



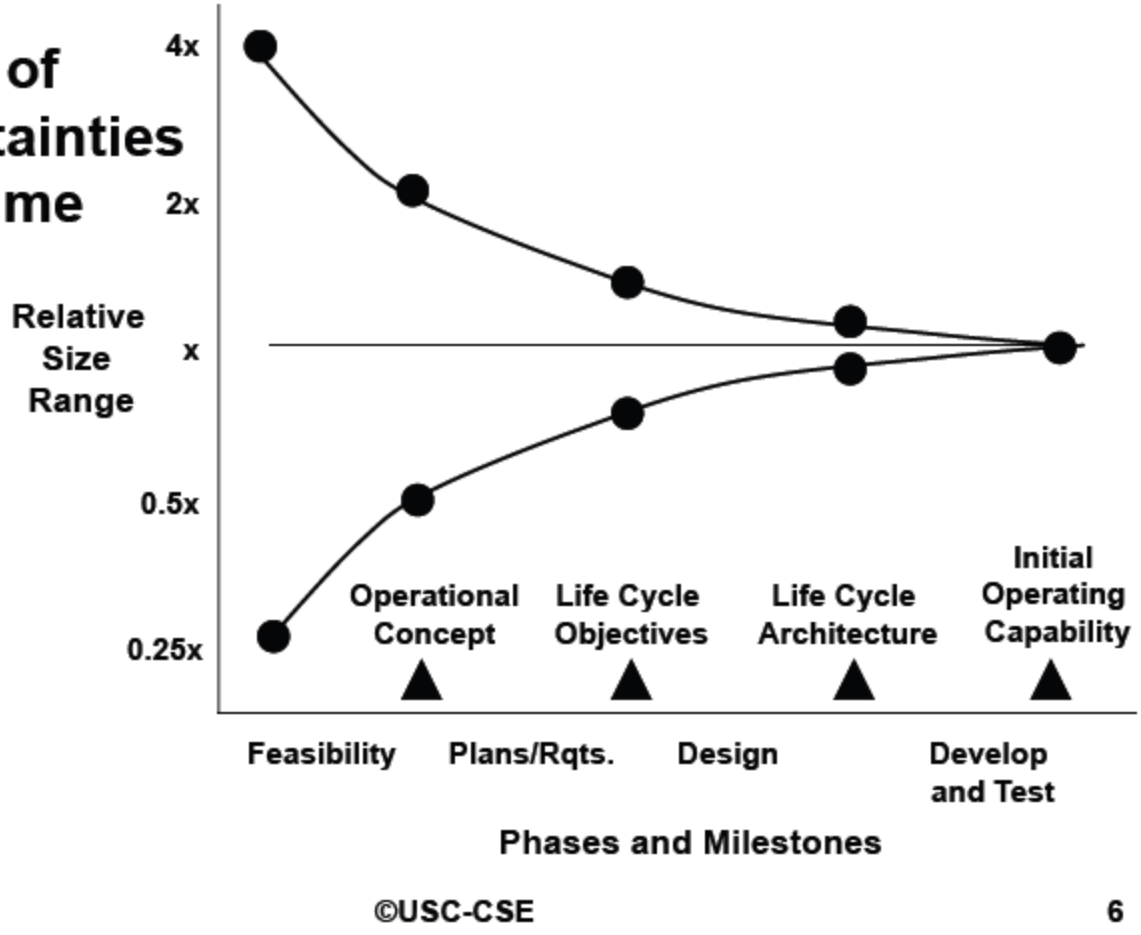
Sprint Planning Meeting - Example



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

Software Estimation Accuracy

- Effect of uncertainties over time



Barry Boehm. Cost Estimation with COCOMO II. CS 577a, University of Southern California, Center for Software Engineering. Fall 2006

Sprint Demo

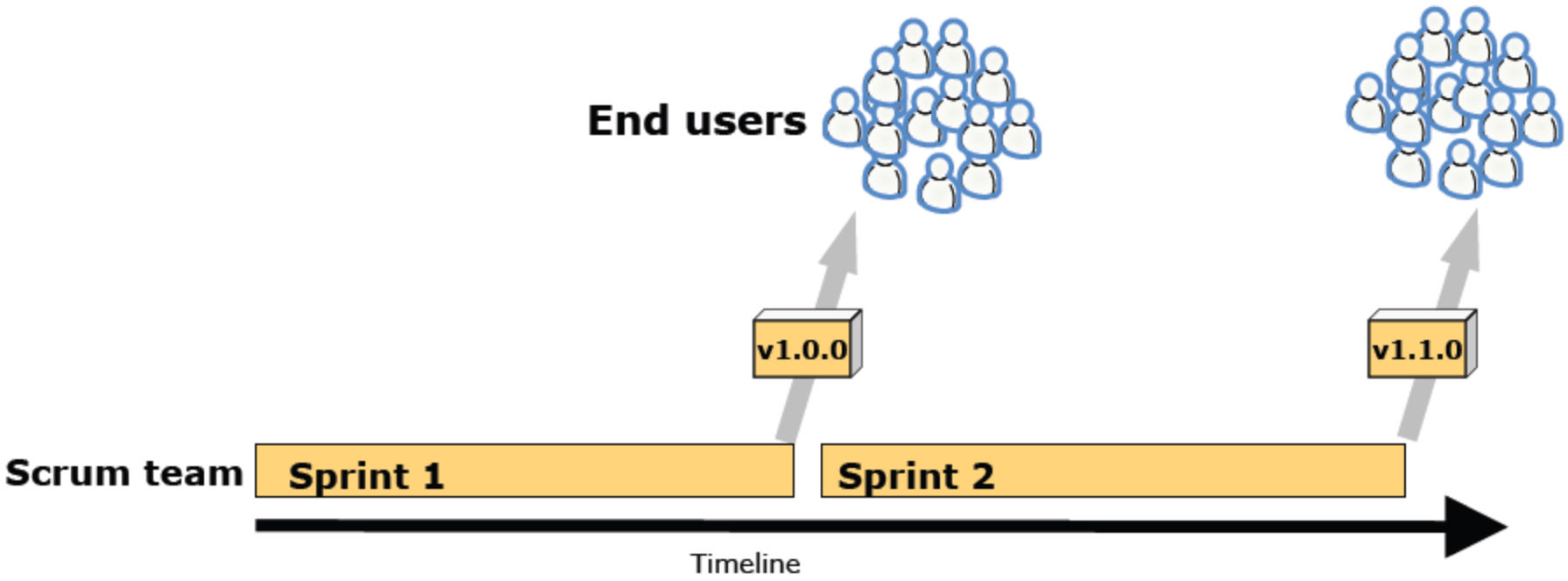
What have we accomplished?

- Team demonstrates working code to stakeholders
- Only 100% completed stories are demonstrated
  - Partially complete stories are ignored
- Direct feedback from stakeholders
- Feedback incorporated into the product backlog

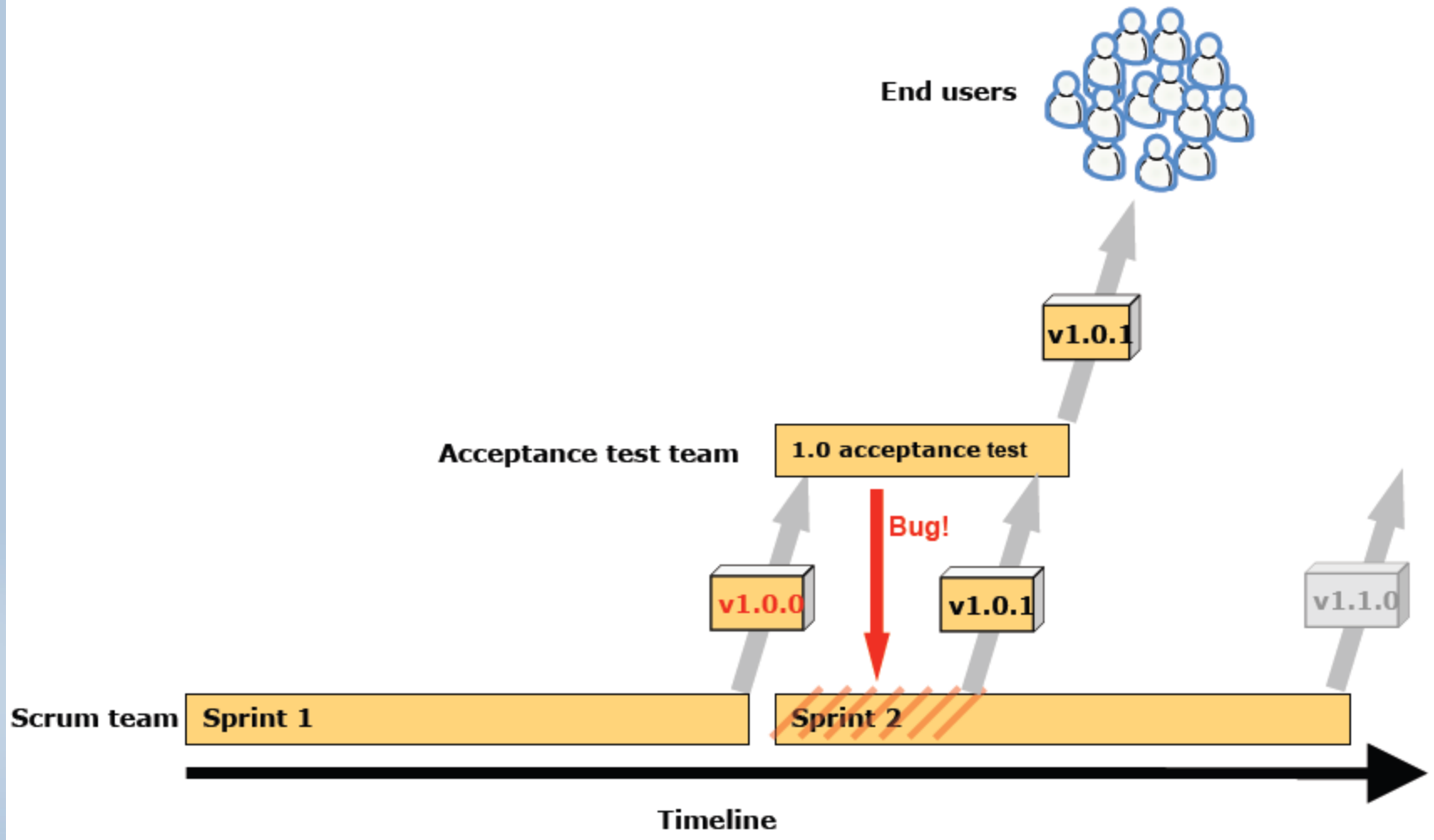
Can We Talk About Other Things?

- Yes.
  - But talk is cheap (and easy to misunderstand)
- Yes.
  - Velocity and its meaning
  - Meaning of progress to date
  - New Release plan
  - Etc, etc, etc

Testing – Ideal Case



Testing – Common Alternative



Product Backlog

Sprint 1: Goal: Understand the Process, People, and Tools in SCRUM

Sprint 2: Goal: Scrum your Scrum

Sprint 3: Goal: Plan and Estimation in SCRUM

Sprint 4: Goal: Understand Advanced Scrum Topics

Sprint Backlog:

- Scrum Implementation ● Nokia Test ● Team Scaling
- The land that Scrum forgot
- CMMI: Safety Net for Scrum
- SCRUM and CMMI:

  Engineering. Project Mgt, Support, Process Mgt , and High Maturity

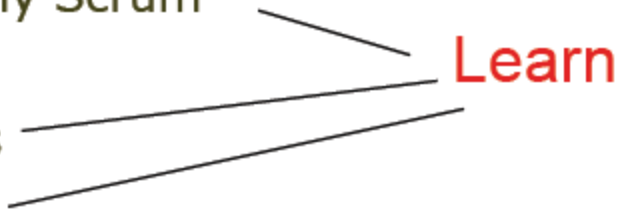
- Q&A ● Exercise 4: XP Game

Implementing SCRUM in an Org

- SCRUM is superimposed and encapsulates whatever engineering practices already existing. This is the easy part.
- The roles of managers and engineers change.
- SCRUM challenges practices, culture and structures that get in the way of focused work. This is the hard part.
- Engineering practices are improved with each Sprints.
- You still need org support.
- You still need to continue improving you SCRUM process and other processes.
- You still need conduct the support work too.
- You need to align with your existing process.
- You still effectively conduct OPP, QPM, CAR, and OPM.

Apply Scrum Now - A 12 Step Program

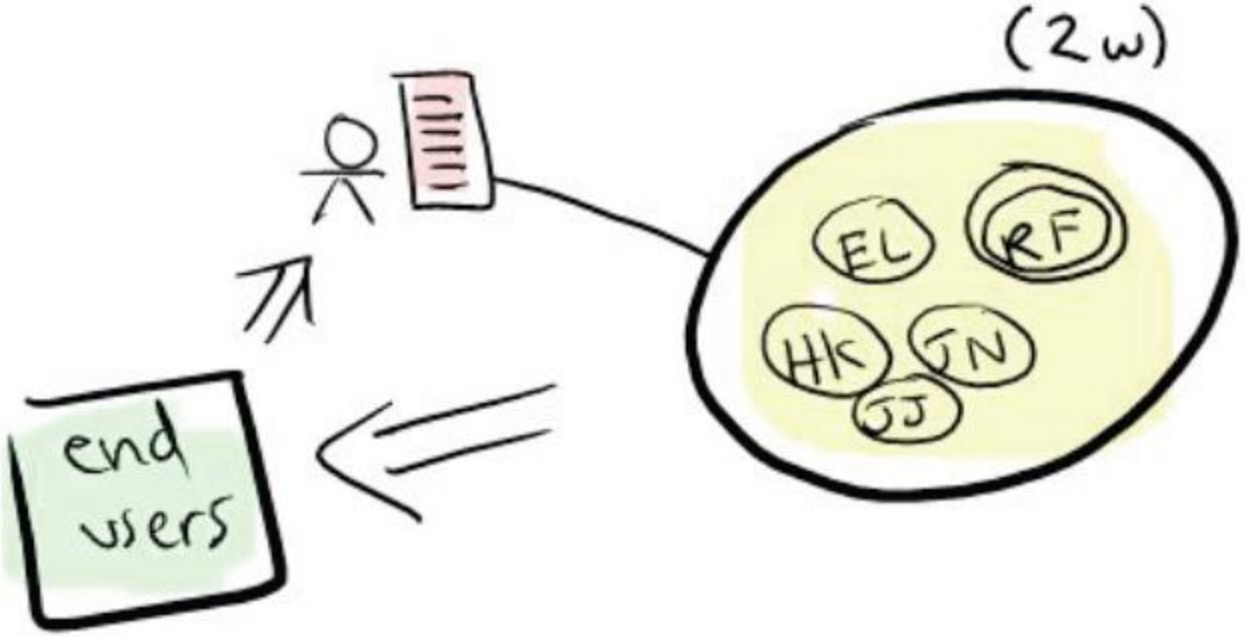
1. Agree on a PO, SM, and full Team. And on a Product goal, theme.
2. Set a date now for the Sprint Review in 2 weeks (+ time to get backlog ready) and send out invites.
3. Review/define a ranked PB of features
4. Estimate the PB items
5. Conduct Sprint Planning with Team and Stakeholders. Complete Sprint Backlog
6. Commit as a team to the Sprint
7. Track status and obstacles daily via the Daily Scrum
8. Track progress using the Sprint Burndown
9. Conduct a Sprint Review; demo done items
10. Conduct a team Retrospective
11. Take action on top impediment (put the kaizen in the backlog)
12. GOTO 2



Source: Hubert Smits & Jean Tabaka

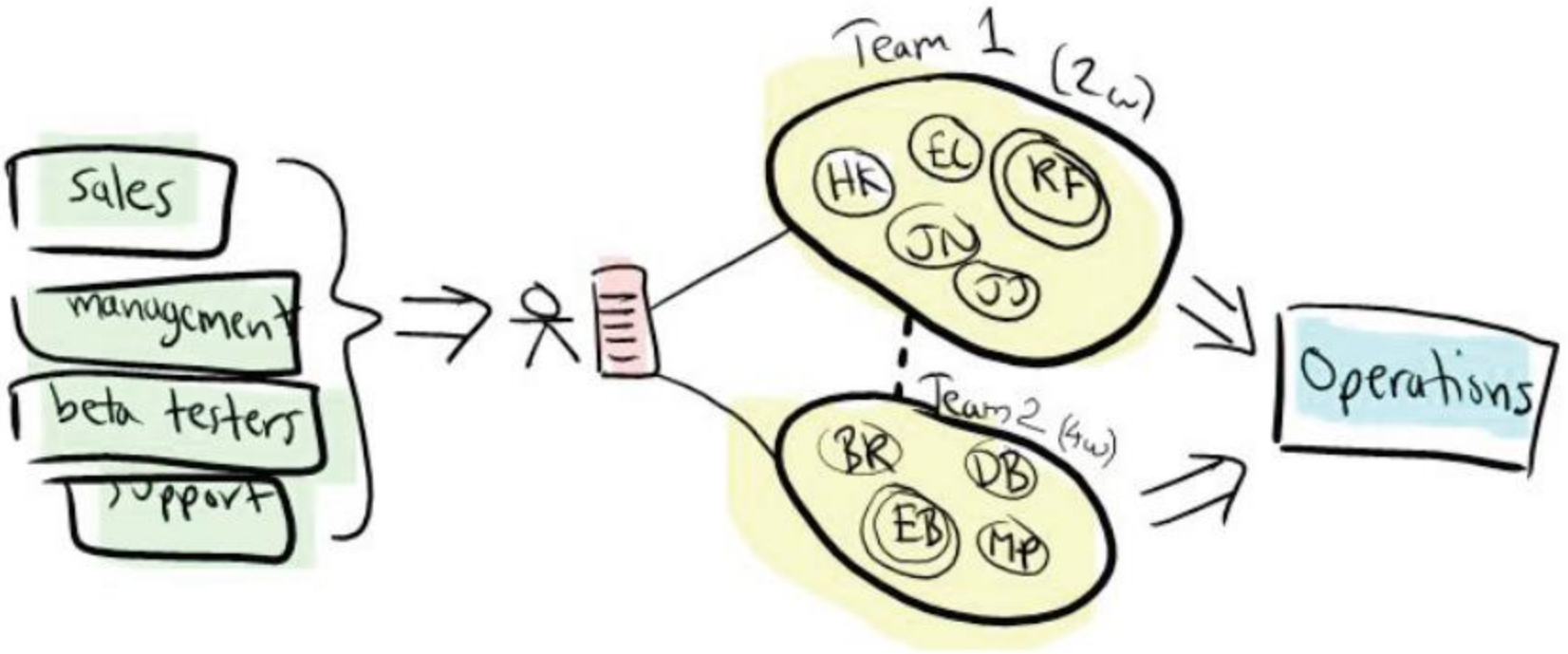
SRRUM SCALING

Example: Simplest possible Scrum organization

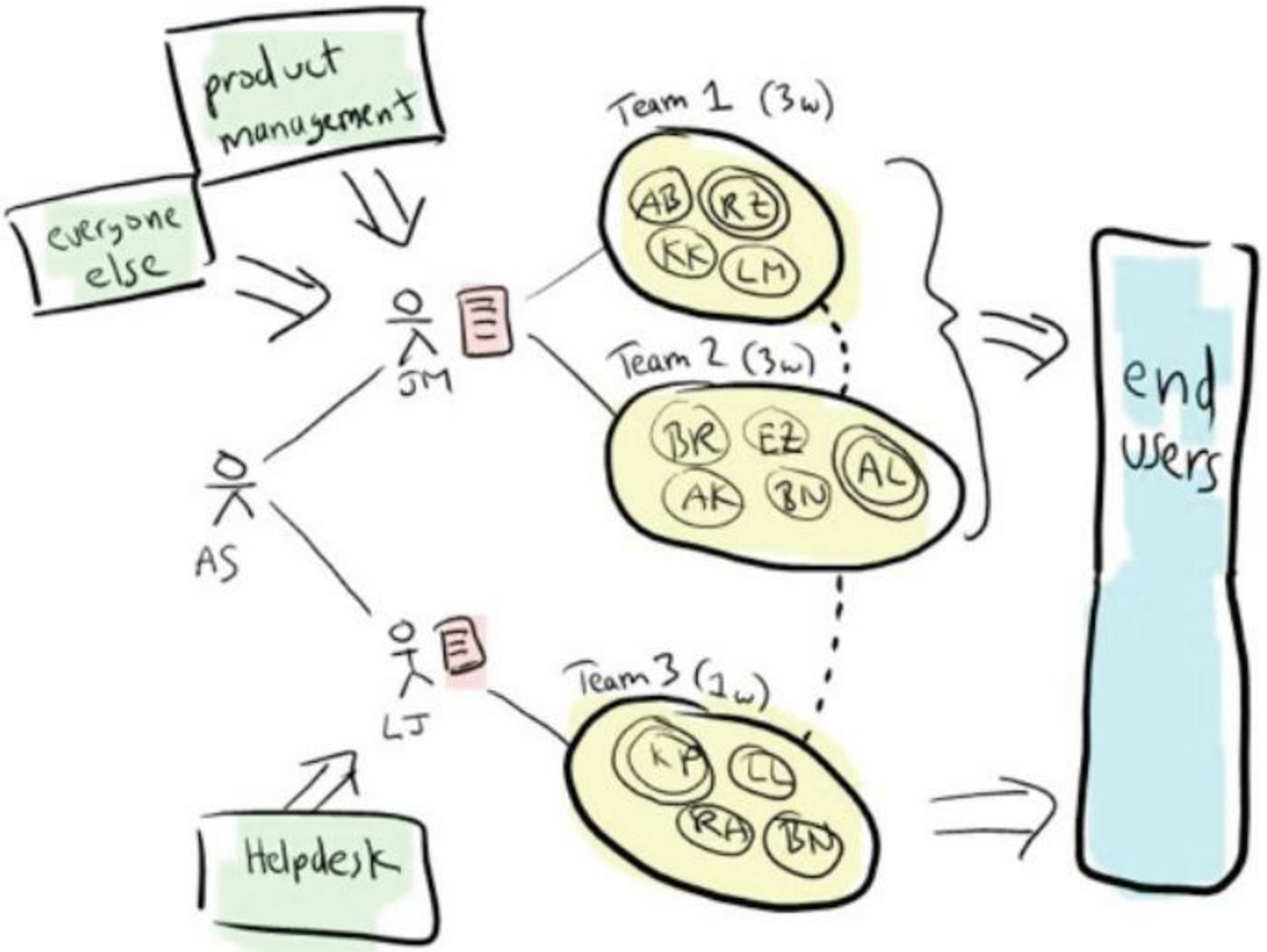


ScrUML
(unofficial Scrum Modeling Language)

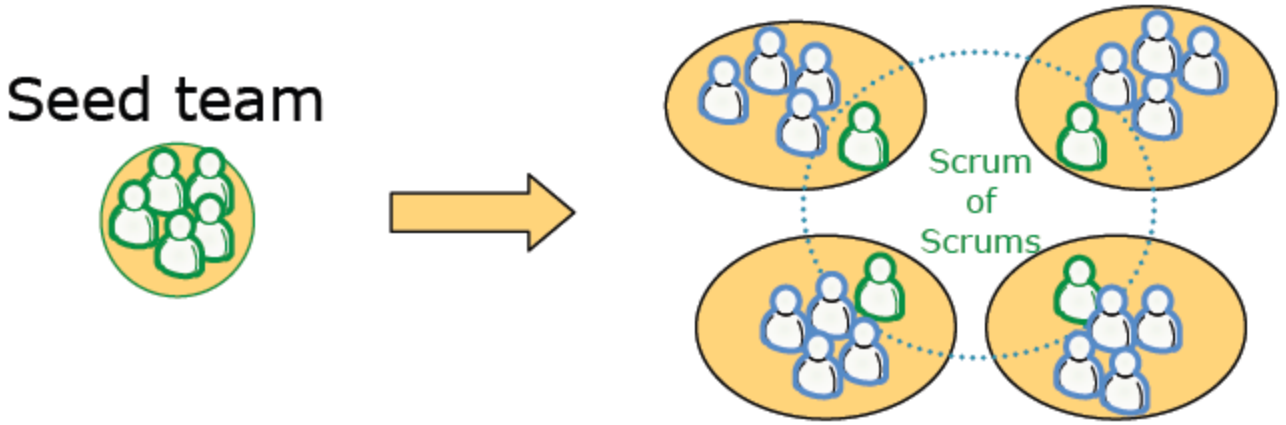
Example: Multiple Teams



Example: Multiple Product Owners

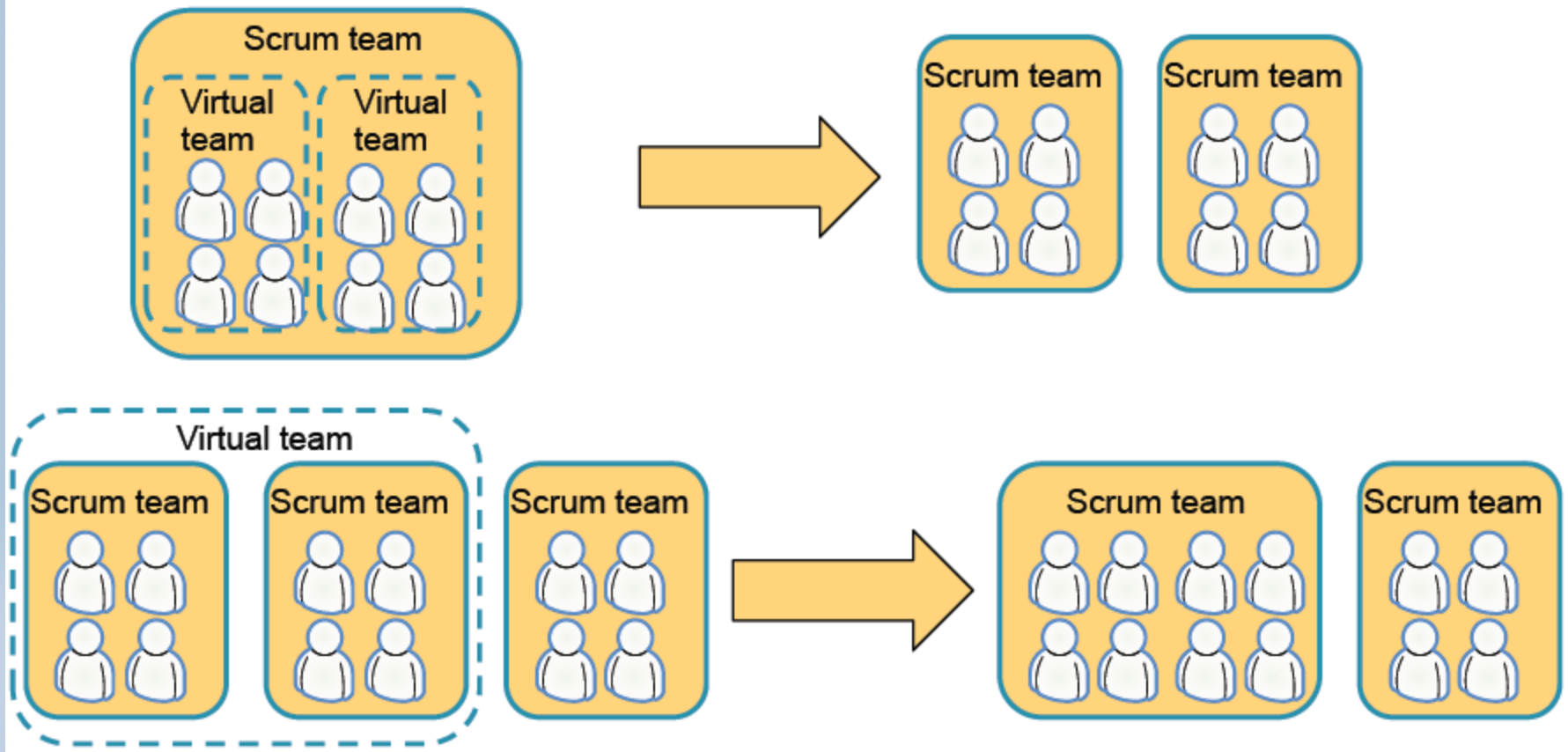


Scaling - Seed Teams

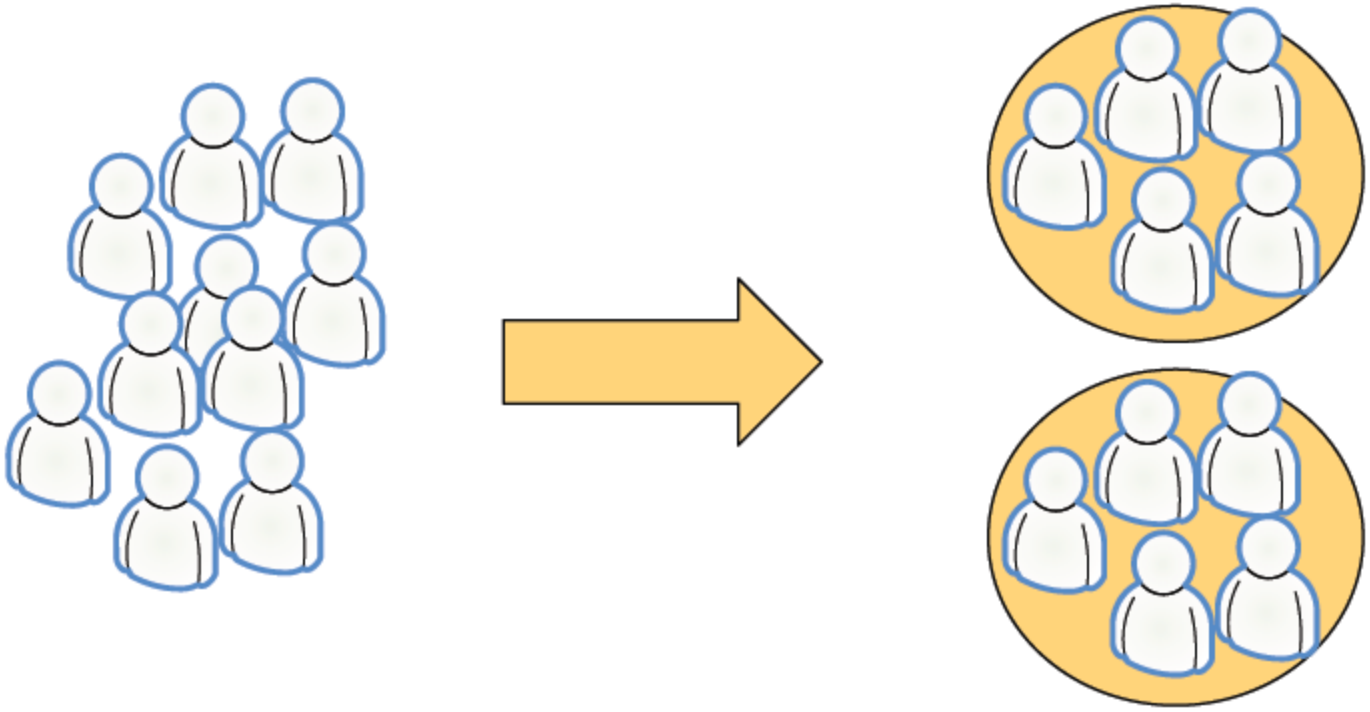


- Start with single "seed team" that creates baseline architecture, prototypes, and development environment.
- "Seed team" members spawn new teams.
- Scrum of Scrums = virtual scrum team responsible for integration and coordination
- DON'T: break up a hyper-performing team!

Team Subdivision



How To Form Teams?



Who defines the teams?

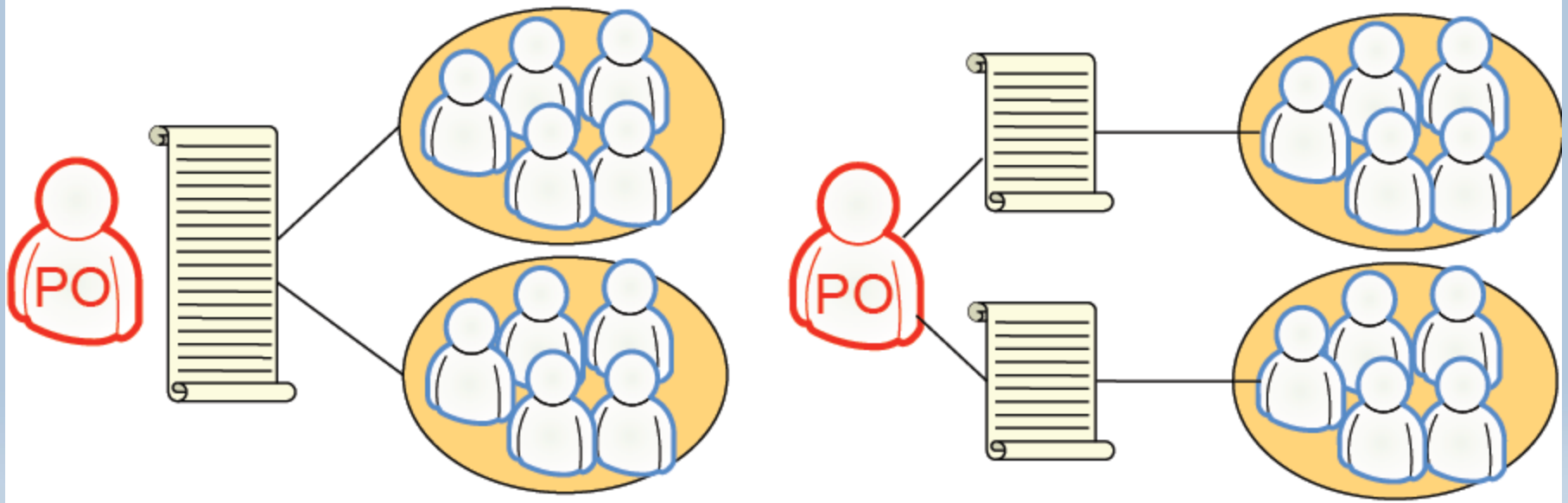
- Option 1: Teams defined centrally
  - **+ Works**
  - **+ Fast**
  - **- Lack of buy-in**
  - **- Doesn't harness collective knowledge**
- Option 2: Teams form themselves from scratch
  - **+ Harnesses collective knowledge**
  - **+ Buy-in**
  - **- Slow**
  - **- Might not work**
- Option 3: Combination of 1 + 2.
  Preliminary teams defined centrally,
  teams then allowed to reform themselves
  - **+ Works**
  - **+ Harnesses collective knowledge**
  - **+ Buy-in**
  - **+ Faster than option 2**
  - **- Slower than option 1**

Self-Organizing to Form New Teams

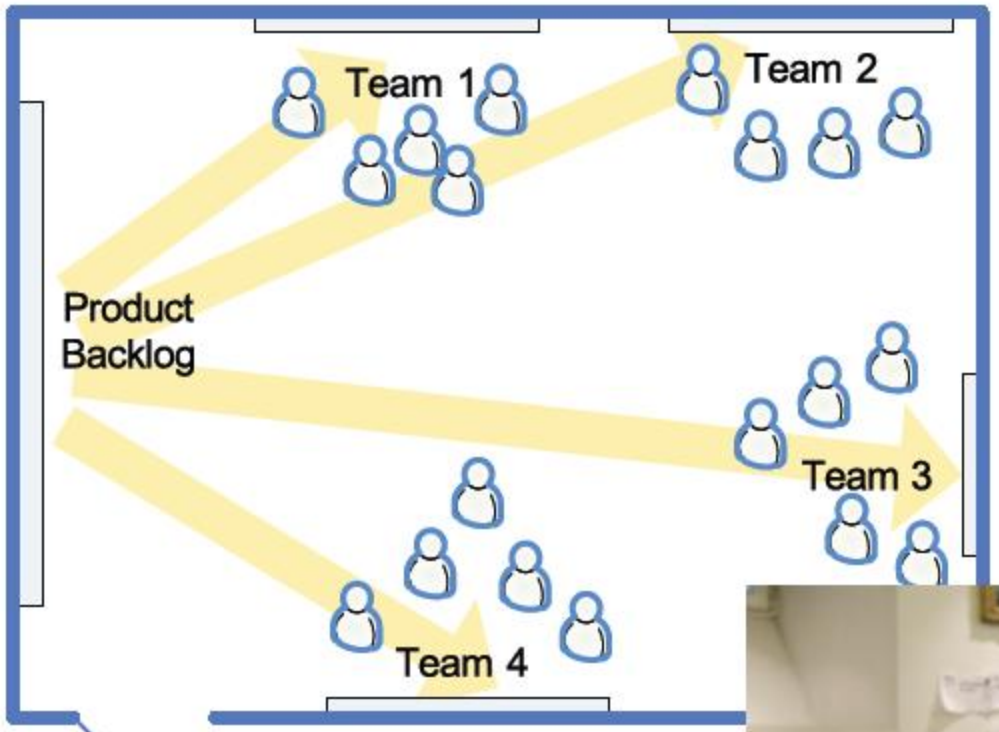
Preliminary team allocation

After a week in the kitchen

1 Backlog or 2 Backlogs?

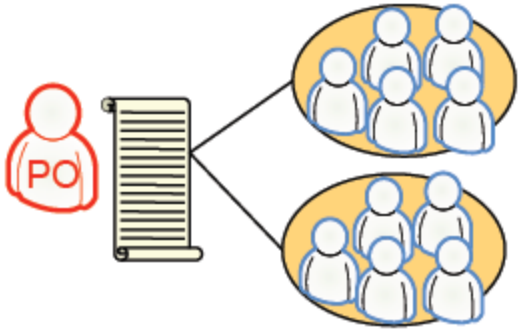


Multi-Team Sprint Planning



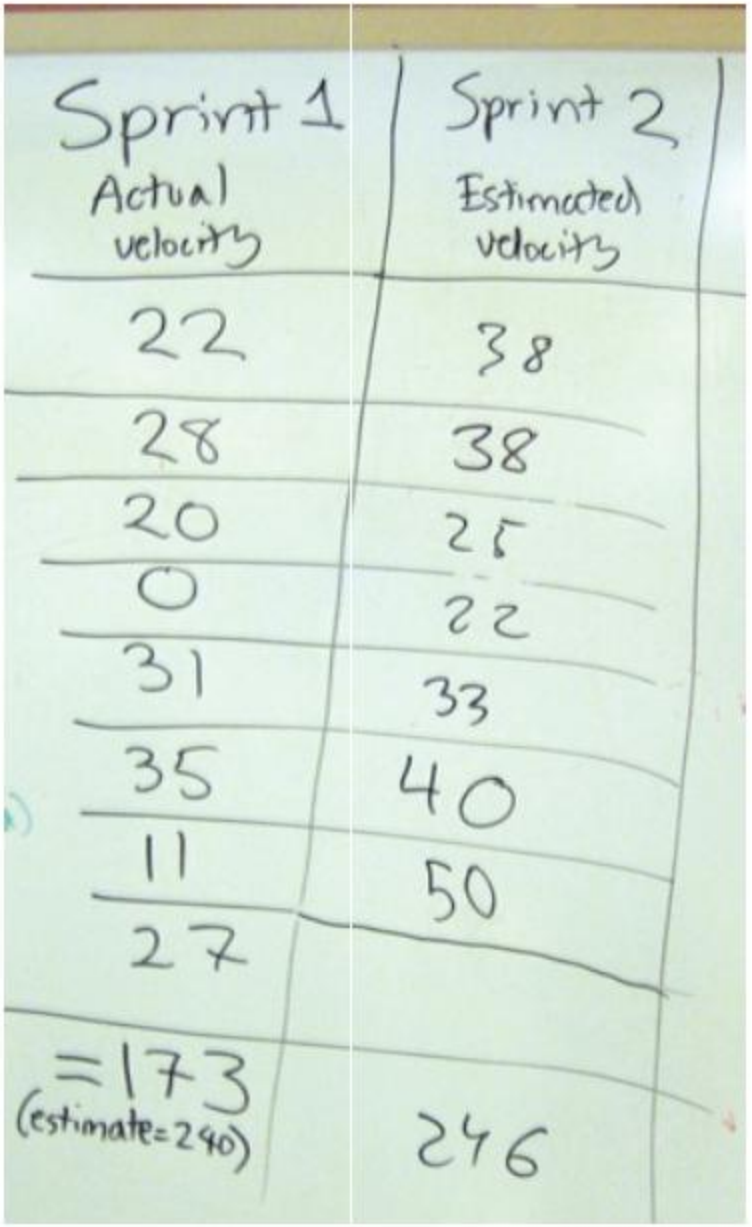
Source: Henrik Kniberg

Multi-Team Sprint Planning Agenda



- 9:00 Welcome, sprint review & retrospective
- 10:00 Goals & priorities for next sprint
- 11:00 Preliminary backlog allocation
- 12:00 Lunch
- 13:00 Preliminary commitment per team
- 14:00 Task breakdown & final commitment per team
- 16:00 Wrap up

Multi-Team Sprint Planning Wrap up



Sprint 1 Actual velocity	Sprint 2 Estimated velocity
22	38
28	38
20	25
0	22
31	33
35	40
11	50
27	
=173 (estimate=240)	246

Multi-Team Sprint Planning - Typical Problems

- People getting bored or feeling inefficient
- Priority and dependency problems discovered
- Fatigue
- Shared resources and specialists
- Running out of time

A Big Perception

CMMI and Agile are at odds with each other!!!

Factors that Affect Perception

- Misuse

  - treat CMMI as a standard (note that Process Area is not a Process.)

  - misusing appraisal ratings

  - CMM’s initial customer base

Factors that Affect Perception

- Lack of Accurate Information from Each Camp

  - M*uch of the impetus of Agile Manifesto and Agile Principles for their collective philosophy* and methods were a direct counter-point to what they (often justifiably) perceived as heavy handed, wasteful, over-burdened and ceremonial processes. “Oil and water” is not an uncommon expression used to refer to the relationship between Agile methods and CMM/CMMI in the writings and postings of Agile supporters.

  - Agile methods were largely ignored by CMMI users. Discussion of Agile methods in CMM/CMMI settings was frequently peppered with anecdotes equating Agile with “no discipline.”

  - Until 2006, the Agile and CMMI communities don’t mix much. Few attendees of Agile conferences also attend CMMI conferences and vice versa. Even more surprising, rarely do the thought leaders in either community publish in the same sources.

Factors that Affect Perception

- Terminology Difficulties

  For example, the word "*predictable.*" Many in the Agile community believe that software projects cannot be predicted with any great precision. "Perfect is the enemy of good enough, so don't try to predict, just react and re-factor." However, CMMI uses the word *predictable in a more subtle sense (defined in Level 4).*

  The Agile community might find a lot of benefit and interesting results from experimenting with this higher-level concept of predictability. For example, David Anderson has achieved predictability in sustaining engineering at Corbis with a release every two weeks, but the content of the release is not bound until five days prior to release due to the unpredictable nature and dynamic character of the work scope. Through better process management and improvement, he's been able to meet CMMI rigor for predictability while simultaneously using an Agile approach that adapts to the unpredictability of the work and the market.

The Truth about CMMI

- CMMI IS A MODEL, NOT A PROCESS STANDARD
- PROCESS AREAS, NOT PROCESSES

The Truth about Agile

- The Agile Manifesto reads as follows: *We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value [the following]:*

  - individuals and interactions over processes and tools

  - working software over comprehensive documentation

  - customer collaboration over contract negotiation

  - responding to change over following a plan

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

  The Manifesto does not mean:

  - "Items on the right" have zero value!

  - It is justified for not having processes, for not documenting their work, and for not having plans!

The Truth about Agile

- Attributes of successful Agile implementations include the following:
  - small teams consisting of approximately ten people
  - an involved customer
  - “rolling wave” or continual planning
  - co-located and cross-functional teams
  - organizations that are not in the habit of breaking up teams until each member is individually competent at Agile
- Agile is not successful when the following is true:
  - lack of processes
  - lack of discipline
  - absent a role for plans or planning
- **A big risk when adopting Agile: weakening quality culture, improvement effort, the role of Quality Dept and EPG.**

The Truth about Agile

The Agile Manifesto has a list of general principles attached to it.

We follow these principles:

- *Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.*
- *We welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.*
- *We deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.*
- *Business people and developers must work together daily throughout the project.*
- *We build projects around motivated individuals. Give them the environment and support they need and trust them to get the job done.*

Continue

- *The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.*
- *Working software is the primary measure of progress.*
- *Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.*
- *Continuous attention to technical excellence and good design enhances agility.*
- *Simplicity—the art of maximizing the amount of work not done—is essential.*
- *The best architectures, requirements, and designs emerge from self-organizing teams.*
- *At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.*

Please Remember that

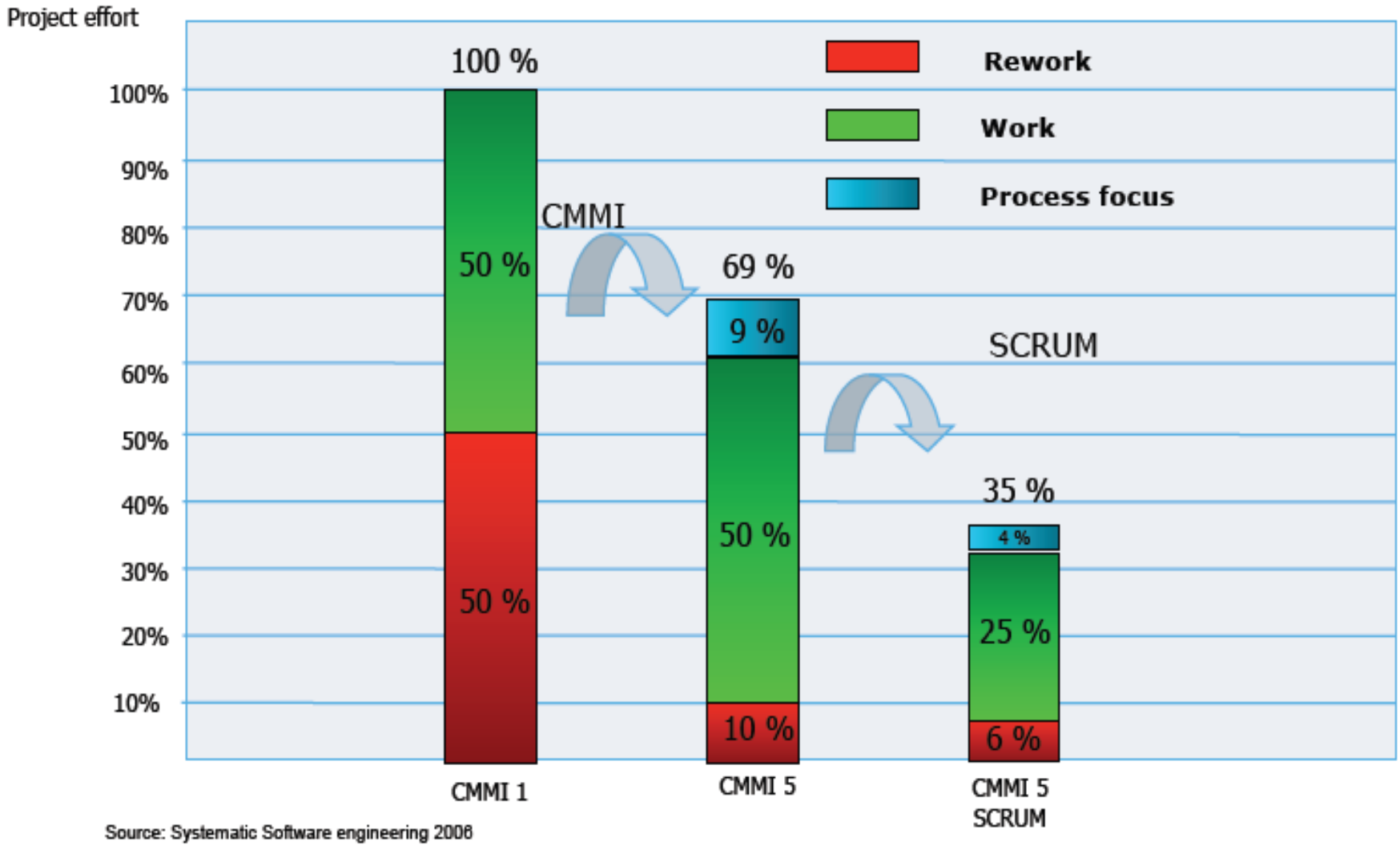
Each word in the *Agile Manifesto and the guiding principles was attained via consensus—a* process familiar to anyone having been on a SCAMPI appraisal. Each word carries meaning. Note the use of *priority, most, primary, sustainable, excellence, and continuous. These are conditional* and qualifying words, not absolutes. Anyone who interprets these words as absolutes are misusing Agile ideas just as they would if they were to interpret of CMMI's *typical work products as absolutes.*

When adopting both CMMI and Agile methods, characterize the situations that are more suitable for the implementation of Agile methods according to the level of trust, customer/end user availability, project scope, scale, expected lifespan of the product, cost of delay, value of early (partial) delivery, and cost of failure present in the situation.

There Is Value in Both Paradigms

- CMMI and Agile are compatible: CMMI focuses on “what” and Agile focuses on “how.”
- CMMI and Agile can complement each other: Agile methods provide software development how-to’s that are missing from CMMI best practices that work well—especially with small, co-located project teams. CMMI provides the systems engineering practices that help enable an Agile approach on large projects. CMMI also provides the process management and support practices that help deploy, sustain, and continuously improve the deployment of an Agile approach in any organization.

CMMI/SCRUM Performance Analysis



Source: Systematic Software engineering 2006

Challenges when Using Agile for Large/Complex Projects

- Keeping the small teams aligned and coordinated for the duration of the project to ensure its success: overall system capabilities to be developed, including non-technical requirements; scope, quality, schedule, cost, and risk tradeoffs; product architecture. (scrum of scrums)

  From a systems engineering perspective, these activities are challenging for large, complex projects to perform well:

  - establishing overall product objectives and a project vision

  - managing the allocation of requirements to teams

  - defining and maintaining interfaces and constraints among teams (for both the product and the process)

  - maintaining effective product integration, verification, and validation strategies for the overall product (or service)

  - coordinating risk management across the project

- CMMI provides a “safety net” for large projects that helps reduce the risk of something going very wrong.

Challenges when Adopting Agile at org Level

Impediments to introducing an Agile approach also include factors common to the introduction of any new technology:

lack of management support and general resistance to change. For example, with the introduction of an Agile method, management often fears a loss of control. Blending a top-down approach (i.e., CMMI) with a bottom-up approach (i.e., Agile) may go a long way in helping to reassure management that its concerns are being addressed in the improvement effort.

The Consensus

- CMMI and Agile each brings something to the table on how to run the business that the other side should listen to and learn from.
- CMMI and Scrum together may be the “magic potion” of software.
- High maturity organizations were able to manage change better and institutionalize the change. Hence, a change to Scrum worked better in a high maturity organization. The transition was faster and institutionalized better.

The Consensus

- Management of complexity requires process discipline while management of change requires adaptability. CMMI provides process discipline and Scrum enhances adaptability.
- When mixing the two, a magic potion emerges, where the mindset from Scrum ensures that processes are implemented efficiently while embracing change, and CMMI ensures that all relevant processes are considered.
- Individually CMMI and Scrum has proven benefits but also pitfalls. An Agile company may implement Scrum correctly but fail due to lack of institutionalization, or inconsistent or insufficient execution of engineering or management processes. CMMI can help Agile companies to institutionalize Agile methods more consistently and understand what processes to address.

CMMI COULD BE THE SAFETY NET FOR SCRUM

Advanced SCRUM Applications

"Scrum works for all projects – projects of all size, projects that involve multiple applications reusing components; projects where extremely high quality is expected, and business projects."

Ken Schwaber and Mike Beedle

Applying Scrum to Multiple Related Projects

- Issues:

  - resources sharing

  - the complexity of the environment is magnified by at least one level of magnitude
- **Never start with multiple projects at once.**
- The 1st Application – Reusability is the focus

What you can do to help with reusability

- Partition the architecture into layers: a layered architecture for an online system will typically look like the following:

  - workflows – units of work linked together.

  - units of work – components that include the front ends and basic mechanics for requesting services and rendering onto presentations.

  - business service layer – reusable strategies.

  - architectural services – logging, security, persistence, distribution, concurrency.
- Partition every layer into smaller packages that might be reused by other application.
- Keep close control on the external interfaces of these packages once the designs start “freezing” – stop changing.

One team for each application and one team for the “shared resources.”

Two Important Practices

- Do not try to reuse anything that is not stable enough.
- The 1st Scrum team will continue to develop and support future releases of the 1st applications.

Can you see the relation with CMMI's Engineering PAs? IPM?

Shared Resources Scrum Team

- May not need this if the shared resources are small.
- Include some developers from the 1st application.
- Try to support and enhance the shared components satisfying the requirements of multiple applications teams.
- Increase the team size incrementally.

Scrum of Scrums

- Weekly or bi-weekly meeting of:

  - Scrum masters of all applications

  - Scrum masters of the support teams: Shared resources, testing, and production support.

Applying Scrum to Larger Projects

- Similar to managing multiple related projects.
- The 1st executable prototype
- The 1st branch of development
- Do not try to do parallel development if the CM, testing, and release processes are not in place.
- Use of Shared Resources Scrum Team
- Scrum of Scrums

Agile Project Management Tools

- Agilefant
- Xplanner
- IceScrum
- Version One
- Agilo
- XPStory Studio
- XPWeb