SAGE TECHNOLOGIES
Inspiring Knowledge

Agile Project Planning and Execution Using **Scrum**

Faculty & Participants
An Introduction

Name, Role, Experience, Current Challenge, Hobby

Working Agreements

- Working Time
- Break Time
- Electronics
- Corner Talk
- Group Exercises
- Participation
- Our Values (for this project)
  - Focus
  - Communication
  - Respect
  - Openness
  - Courage

Everybody

Values

that guide us

Introspection

- What is the meaning of Agile for you?
- What is meaning of Agile for your customer?
- Your challenges in Project
- Meaning of Project Success?
- # of Project you have done [ ]
- # of Successful projects you worked [ ]
- Why Project fails or get challenged

Overall Plan of This Training

Day 1

1. Agile Introduction & Overview
2. The Agile Planning Framework
3. Characteristics of The Agile Team & Team Structure
4. Roles & Responsibilities in Agile Project
5. Product Backlog
6. User Story, Features, Epic, Theme, INVEST Model, Acceptance Criteria

Day 2

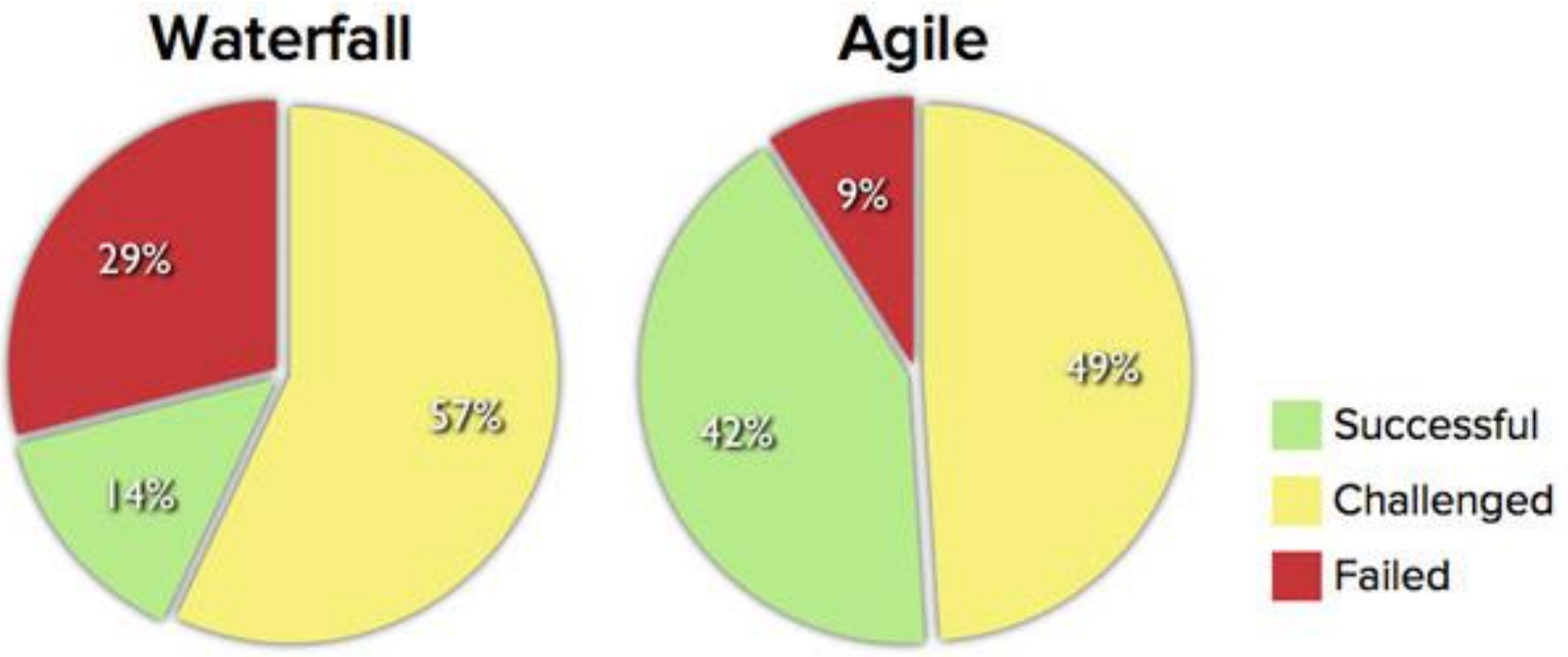
7. Agile Estimation
8. Planning
9. Running a Sprint
10. Closing Out The Sprint
11. Agile Health Checkup
12. Some Important Concepts of Agile Project Management
13. End to end exercise to recap

Why this?

Standish Group Prepares Chaos Reports Regularly.

- 1994: Only 16% of the projects are successful, based on their triple constraints
- 2004: 29%
- 2009: 32%

Process Success with Agile Methods

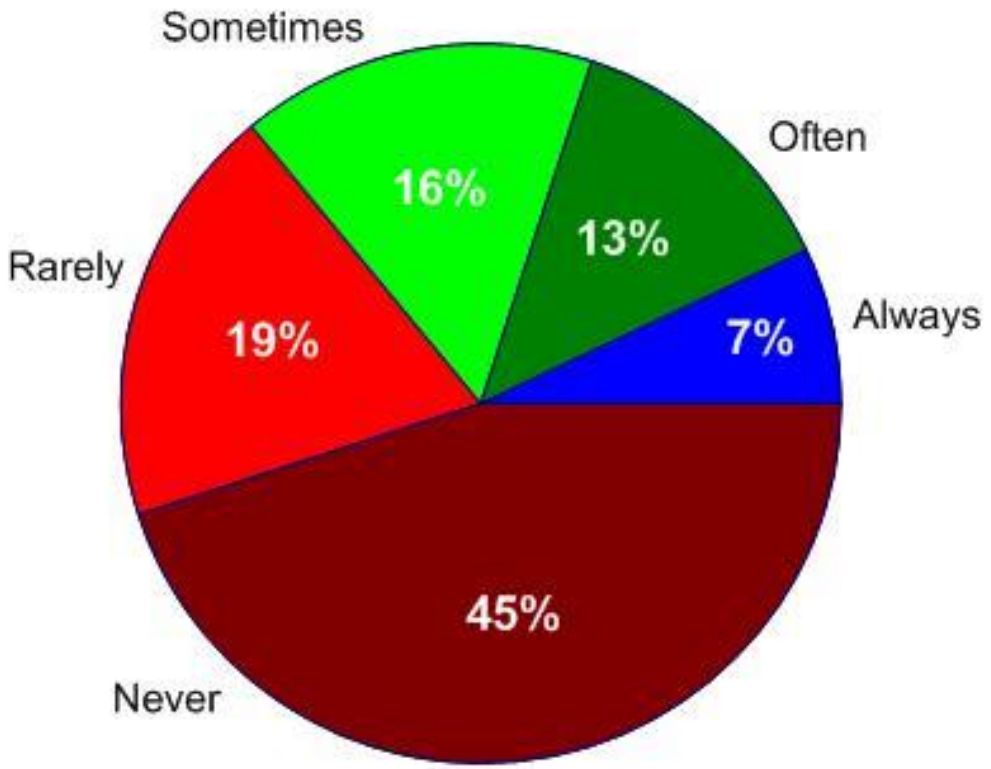


Source: The CHAOS Manifesto, The Standish Group, 2012.

Why this huge change in data?
Definition of Success was left to participants. Only 8% agreed that all three constraints important for them to make a project Success

Value of Delivered and non-used Features

Average percentage of delivered functionality actually used when a serial approach to requirements elicitation and documentation is taken on a "successful" information technology project.



Source: Chaos Report v3, Standish Group.

Cultural aspects of predictive styles

- Any variance is considered as evil
- Focus is on Time , Cost and Scope
- Very often the project teams can be insensitive to the business case of the project

Empirical Process

- Understands the output is not controlled only by controlling input and steps
- Process is defined loosely and at high level
- Process learn and evolve over the period of time (adaptation)
- Adaptive methods are based on empirical processes

Adaptive Methods

- XP (Extreme Programming)
- Scrum
- Crystal
- FDD (Feature Driven Development)
- Many others....

When to go agile? Exploration factor

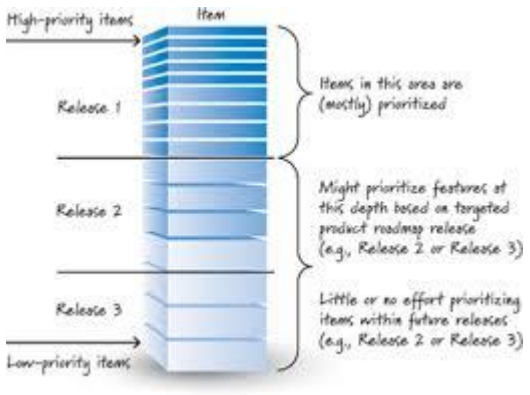
Product Requirements	Bleeding edge	Leading edge	Familiar	Well known
Erratic	10	8	7	7
Fluctuating	8	7	6	5
Routine	7	6	4	3
Stable	7	5	3	1

Agile Framework at a Glance

Product Owner

Product Backlog

Release Planning

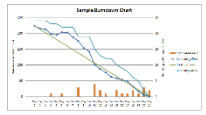


Release Backlog

Sprint Planning

User Story	Task	Total Hours
User Story 1		24 Hrs
	Task1 (10 hrs)	
	Task2 (6 hrs)	
	Task3 (8 hrs)	
User Story 2		21 Hrs
	Task1 (6 hrs)	
	Task2 (15 hrs)	
User Story 3		34 Hrs
	Task1 (13 hrs)	
	Task2 (12 hrs)	
	Task3 (9 hrs)	
User Story 4		21 Hrs
	Task1 (11 hrs)	
	Task2 (3 hrs)	
	Task3 (7 hrs)	
User Story 5		14 Hrs
	Task1 (12 Hrs)	
	Task2 (2 hrs)	

Sprint Backlog



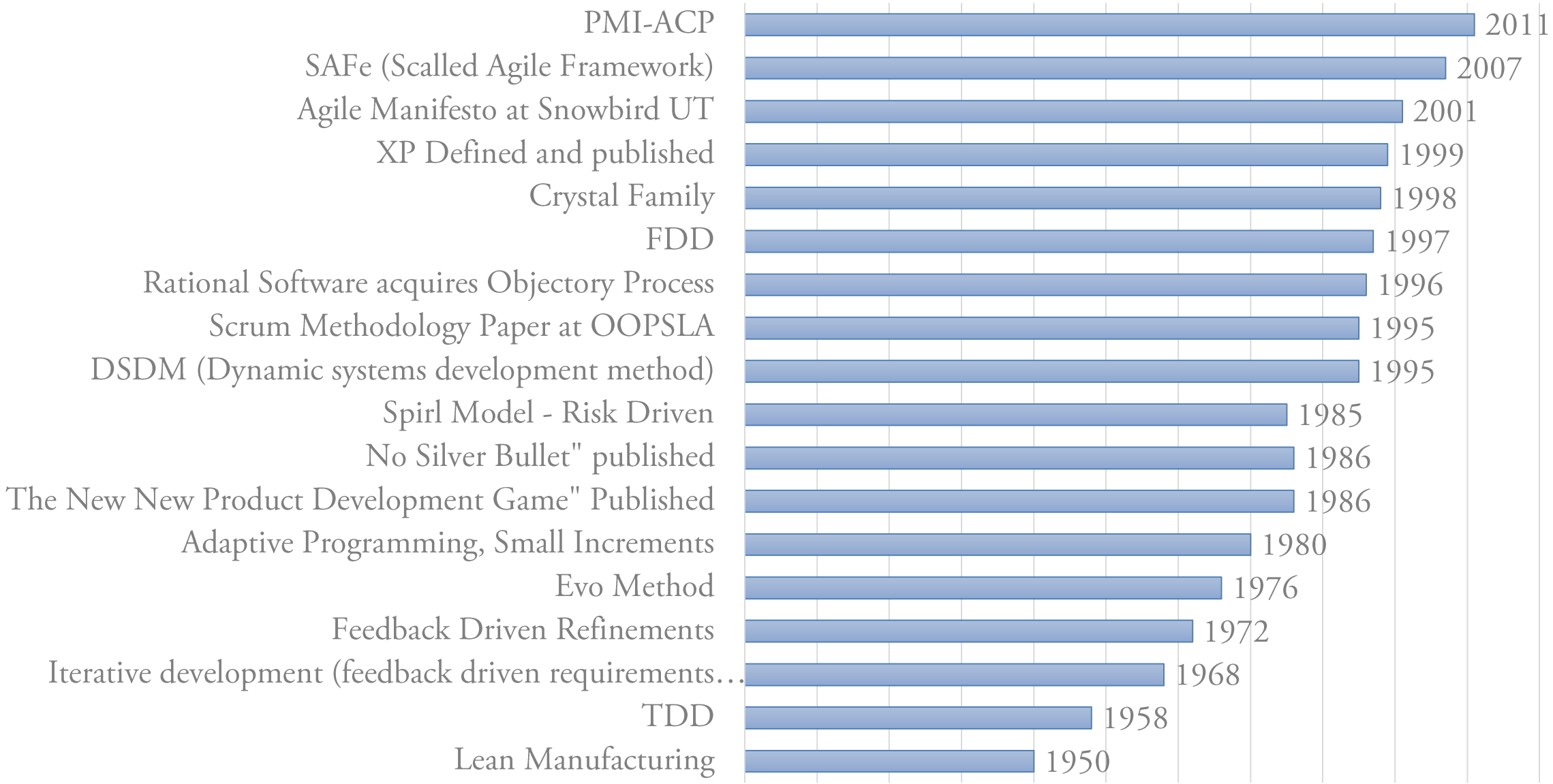
Burndown/Up Chart

Scrum Master

Daily Scrum

Sprint Retrospective

History of Agile Development



Manifesto for Agile Software Development

- **Individuals and interactions** over process 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 12 agile principles

1. Our highest priority is to satisfy the customer through ***early and continuous delivery*** of valuable software.
2. ***Welcome changing requirements***, even late in development. Agile processes harness change for the customer's competitive advantage.
3. ***Deliver working software frequently***, from a couple of weeks to a couple of months, with a preference to the shorter time scale.
4. ***Business people and developers*** must work together daily throughout the project

The 12 agile principles

5. Build projects around ***motivated individuals***. Give them the environment and support they need, and ***trust them*** to get the job done.
6. The most effective method of conveying information to and within a development team is ***face-to-face conversation***.
7. ***Working software*** is the primary measure of progress.
8. Agile processes promote ***sustainable development***. The sponsors, developers and users should be able to maintain a ***constant pace*** indefinitely.

The 12 agile principles

9. Continuous attention to ***technical excellence and good design*** enhances agility.
10. Simplicity – the art of ***maximizing the amount of unwanted work not done*** – is essential.
11. The best architectures, requirements and designs emerge from ***self organizing teams***.
12. At regular intervals, the ***team reflects*** on how to become more effective, then tunes and adjusts it's behavior accordingly.

SCRUM Values

1. Commitment

Be willing to commit to a goal. Scrum provides people all the authority they need to meet their commitments

2. Focus

Do your job. Focus all your efforts and skills on doing the work you have committed to doing. Don’t worry about anything else

SCRUM Values

3. Openness

Scrum keeps everything about a project visible to everyone

4. Respect

Individuals are shaped by their background and their experiences. It is important to respect the different people who comprise a team

5. Courage

Have the courage to commit, to act, to be open and to expect respect

Agile methods (including SCRUM) are

value/principle based, rather than rule-based

Scrum is..

- Founded on empiricism
- Believes knowledge comes from experience
- Believes in making decisions based on what is known
- Iterative incremental approach
- Risk control
- Transparency
- Adaptation

Command & Control Systems (Exercise)

- Break into teams, and develop a team charter based on the agile principles and values
- Should be documented on an A4 sized document
- Should reflect the agile principles and values
- All the team members should sign on it

Characteristics of Agile Team

- Agile team is value and principle driven
- Agile team is self organizing
- Agile team is self motivating team
- Agile team is delivery driven and not task driven
- Agile teams are not department driven
- Agile team do not promote heroism

Sprint

- In XP it is also called “Iterations”
- Fixed duration for which team commits a potentially shippable product to Product Owner
- Product is understood, designed, coded, tested and demo and delivered in a sprint.
- The duration can vary from 1-6 weeks depending on agile method chosen and project characters
- In Scrum ideal duration is 4 weeks.
- The duration is NOT variable and it is decided at the start of project. What if it vary??
- If work committed cannot be done in a sprint work duration cannot be extended but work is moved into next sprint
- No extra time for testing and bug fixing

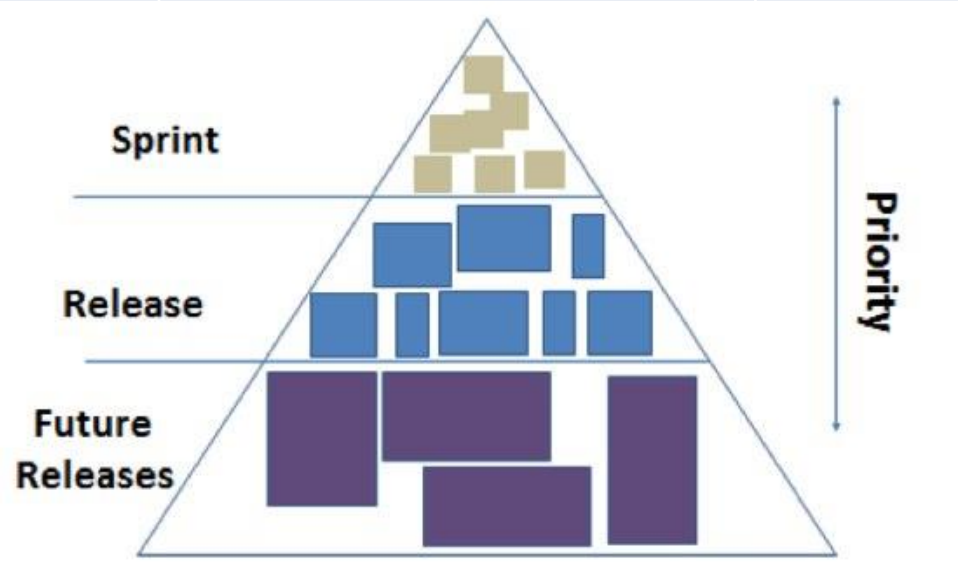
Product Backlog

What is a product backlog?

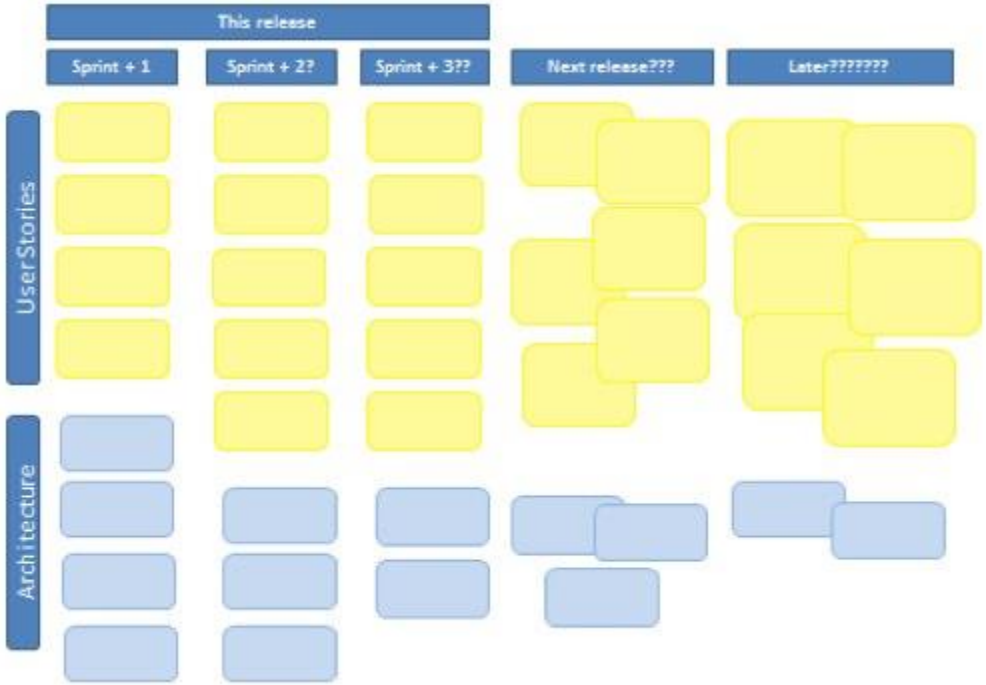
- A wish list of features (user stories)
- Continuously growing (product backlog grooming)
- Owned by the product owner
- Prioritized (based on risk, cost, value)
- Tells you what need to be tested?
- When stops growing – indication that the product is reaching end of life !

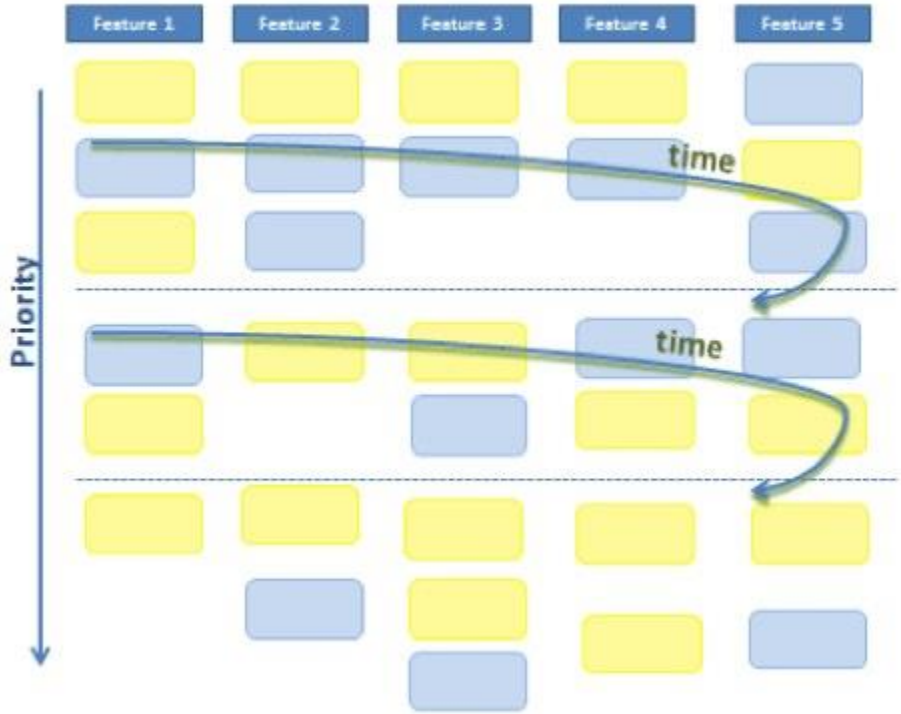
Product backlog

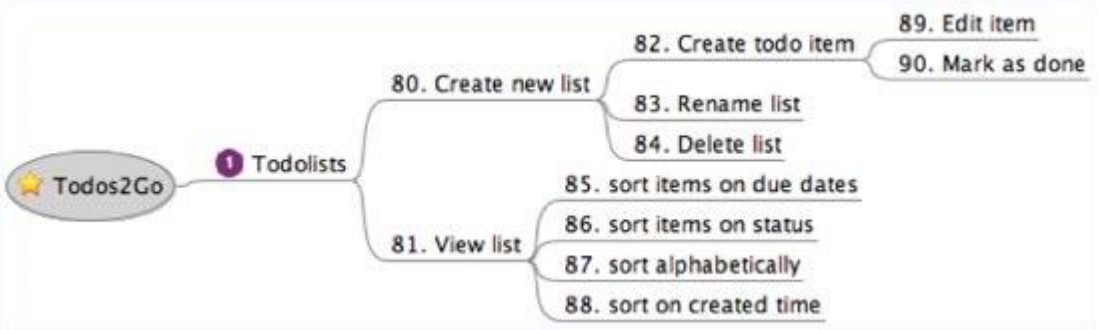
ID	Theme	As a/an	I want to...	So that...	Notes	Priority	Status
2	Deposits	Cachier	credit customer account for the deposited with me	Customer account reflects the new balance		1	Done
3	Deposits	Cachier	update the wrongly entered deposit amount	Customer account reflects the correct amount		2	Done
4	Deposits	Auditor	know whether amount is credited into customer's account	I can tell customer the status		6	Todo
5	Non-Functional	Manager	Any page which is marked as super confidential should not be printed on unauthorized printer			3	Done
6	Non-Functional	Cachier	My cash entry page should loa			8	Done
7	Archive	IT Admin					Done
8	Security	Cachier					Done
9	Security	Manager					Done



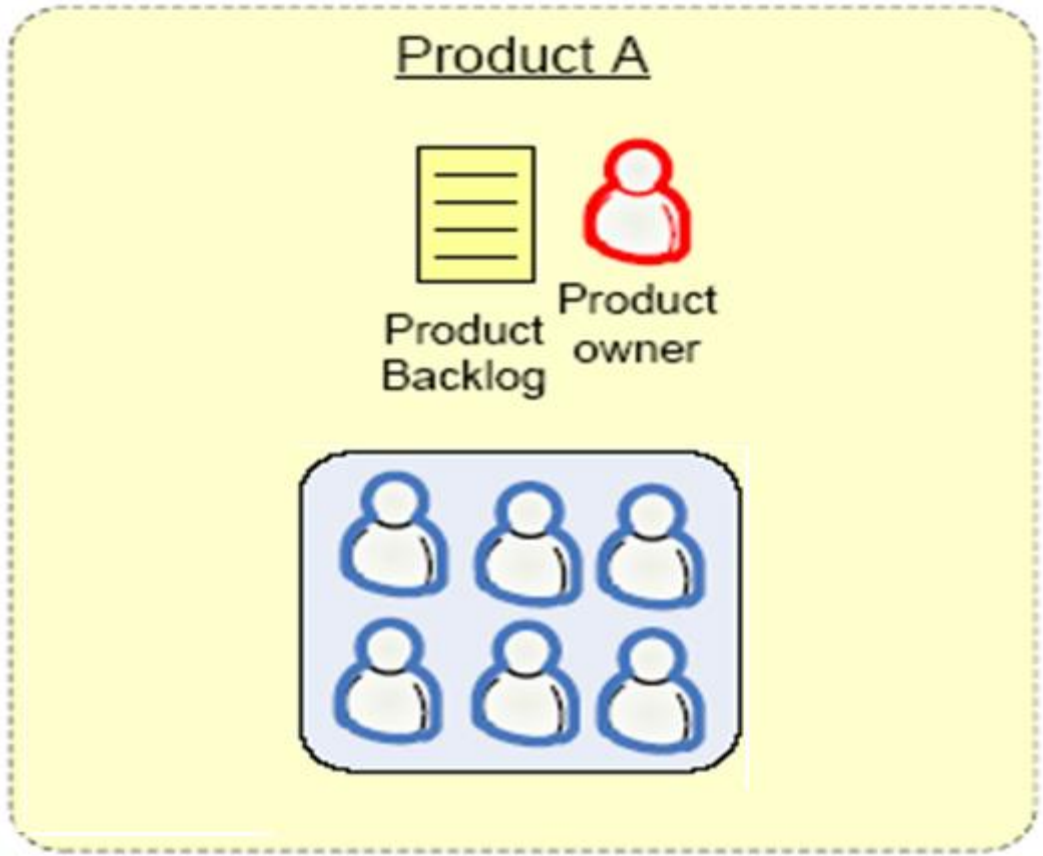
Visualize Your Product Backlog





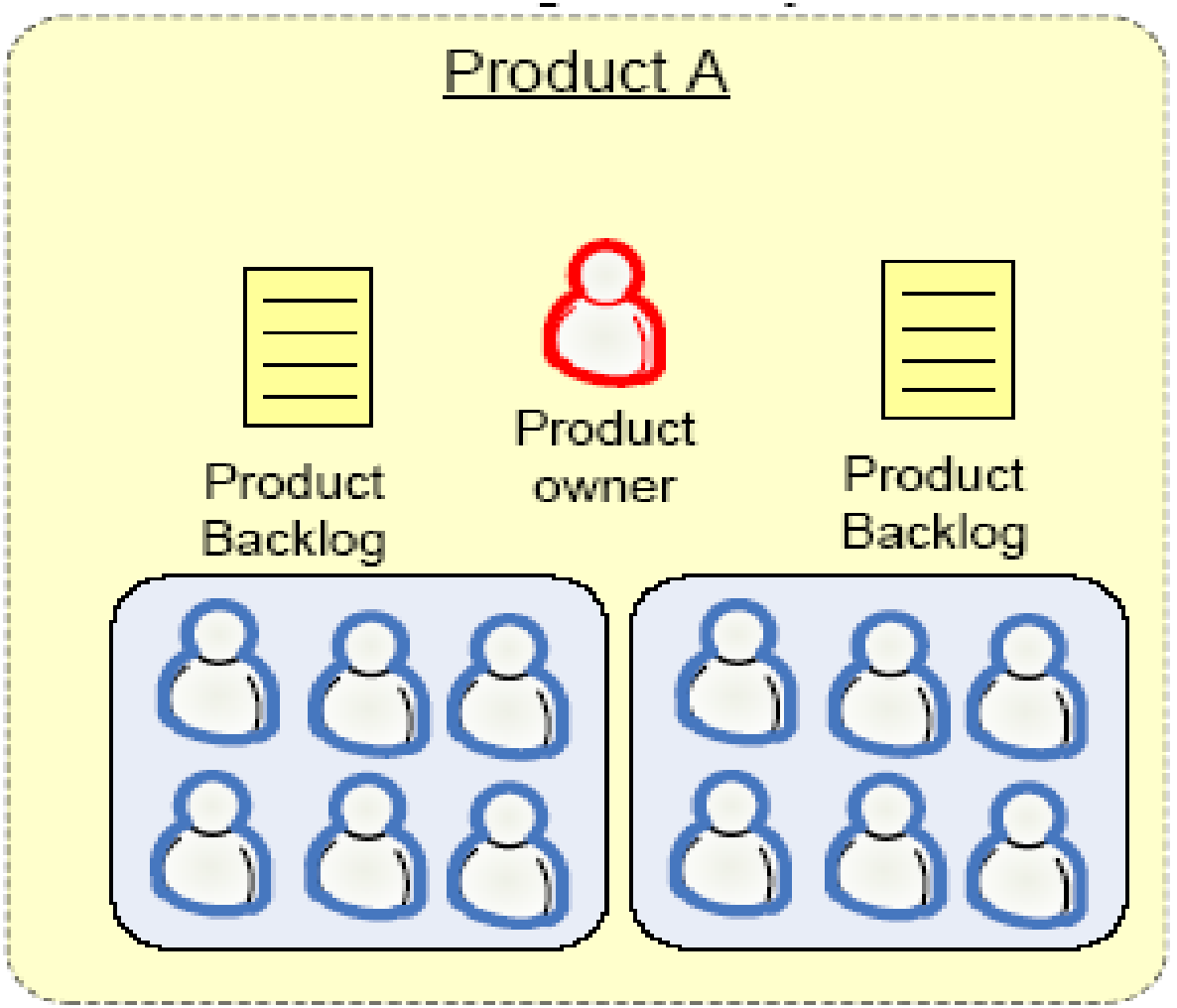


1 Product Owner – 1 Product Backlog

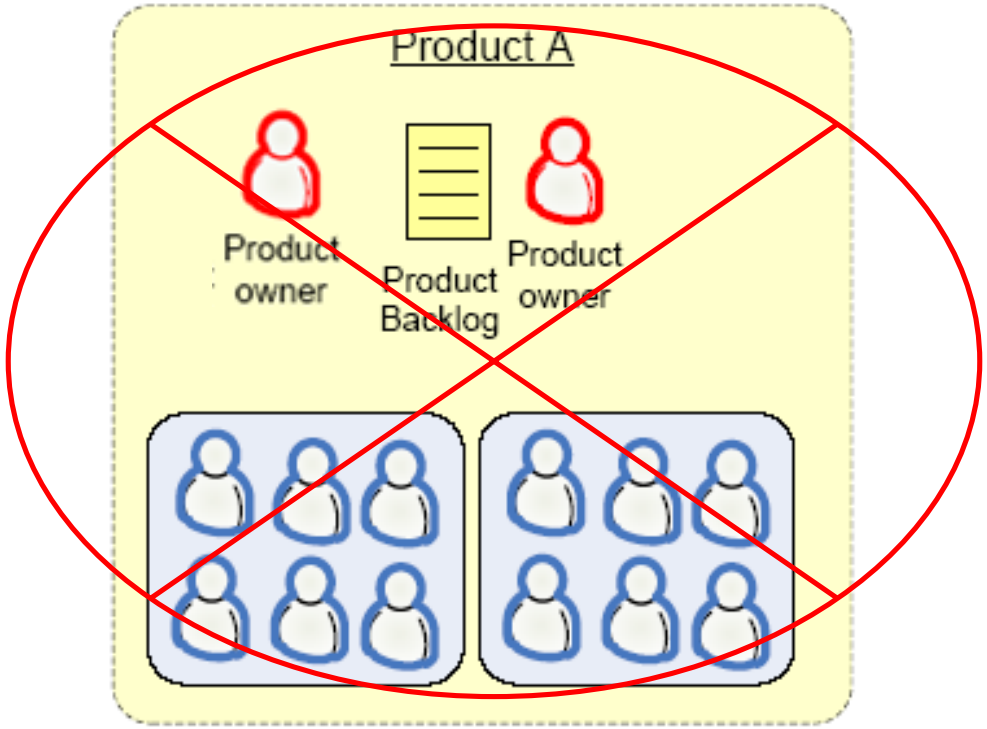


Ideal

1 Product Owner – 2 Product Backlogs



Multiple Product Owners, 1 Product Backlog



User Story

User story

- A user story concept is kernel of Agile Project Management
- A user story is work which a user want system to accomplish because it meets some of her/his objectives.
- A user story is not functional specification document. It is a promise of product owner to the team that he will explain the requirements in details when the team is working on this
- User story template : “As a user I want to accomplish something so that business value”
- Every user story must have acceptance test cases

3C of User story

- Card
- Conversation
- Confirmation

Types of User Stories

- Business user story
  - "As a class teacher I want to mark attendance of student so that we can issue them certificate"
- Bug user story
  - "An error message is displayed whenever I try to save file in pdf format"
- Technical user / Technical Spikes story
  - "Research a search component in .NET3.5 which is fit for our application"
- Non-functional user story
  - "The Student Affairs Information System is up and running 99.9% during the registration time period defined in the Academic Calendar."
- Documentation user story
  - "Develop a user manual for teachers to use teacher module"

Theme, Epic, Feature, Story, Task

- Epic is a collection of features. An epic is typically 1-3 months in duration
- Feature is collection of stories. A feature is typically 2-4 weeks in duration
- User-story is smallest unit of requirement created from features. A user-story is typically less than a week in duration
- Task are smallest unit of executable items which team members assign to themselves to complete a user story. A task is typically of 8 hours in duration
- A group of related user stories is known as theme

I.N.V.E.S.T Model of User Story

I – Independent

N – Negotiable

V – Valuable to users / customers

E – Estimatable

S – Small

T – Testable

Techniques to Identify User Stories

- User interviews
- Questionnaires
- Observation
- Story writing workshops

Exercise (User Story)

- Accounting Application
- In a group write user stories.
- A story in a card.

Estimation

The Estimation Scale

- Complexity assessment is never in arithmetic progression
- 0,1,2,3,5 and 8 (Fibonacci/Hemchandra series)
- 1,2,4 and 8 (Geometrical increase)
- Fruit Size
- T Shirt Size

Most Common Techniques

- Wideband Delphi
- Relative Size Estimation
- Planning Poker
- Mute Mapping

You Use either or any combination of below techniques to above methods.

- Expert Judgement
- Analogy
- Bottom Up
- Top Down

Output of Estimation Workshop

- Estimation of work happens in story points.
- **User Story # Story Points**
- Do not estimate durations but complexity
- Visualize complexity in terms of # of steps, volume, # of dependencies etc.
- Duration cannot be estimated till the time we know the detail of user story. It happens in sprint planning.
- Simple things can take more time and vice versa.

Story points to schedule

Understand

- Velocity,
- Duration,
- Elapsed time,
- Productivity,
- Availability of resource,
- IEH (ideal engineering hours)

Before you commit the schedule

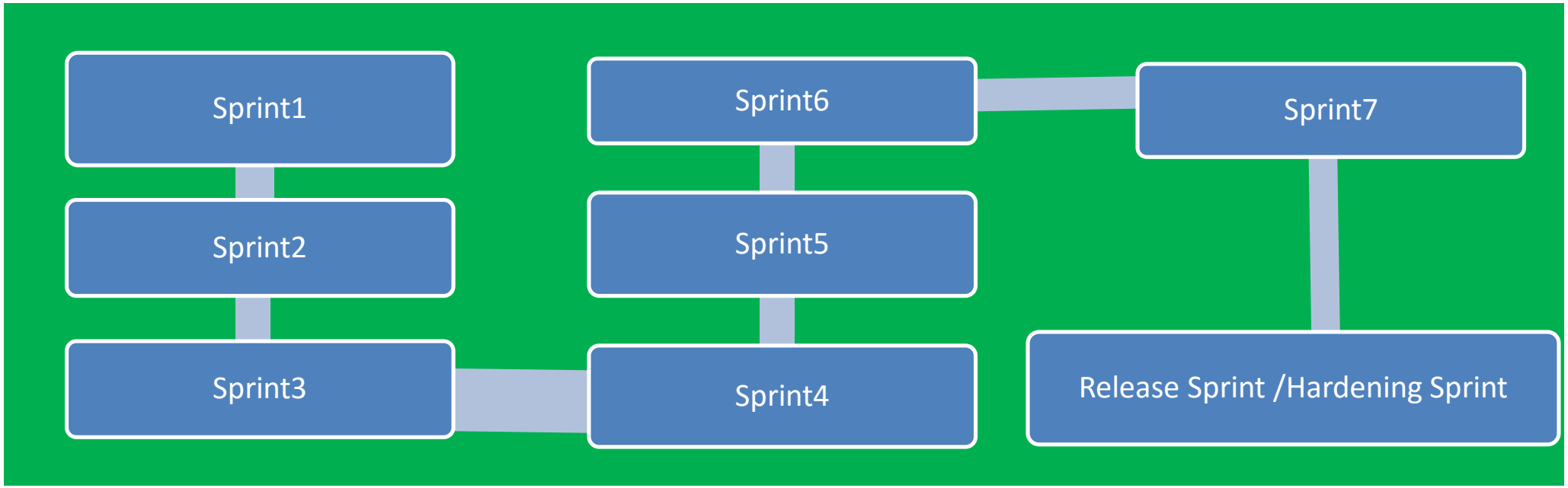
Mute Mapping

Relative Size Estimation

Release Planning

- After Completing Product Backlog Estimation
- Define Release Goal
- Determine MMF (Minimum Marketable Features)
- Guess a velocity (velocity get stabilized after 4-6 sprints)
- Determine Number of Sprints Required

Release Sprint



During a "release sprint"

- No extra feature is added
- Team prepares a product for release
- Final system testing and fixing system issues
- Documentation

Scrum

Sprint Planning

Sprint Planning

- What is this?

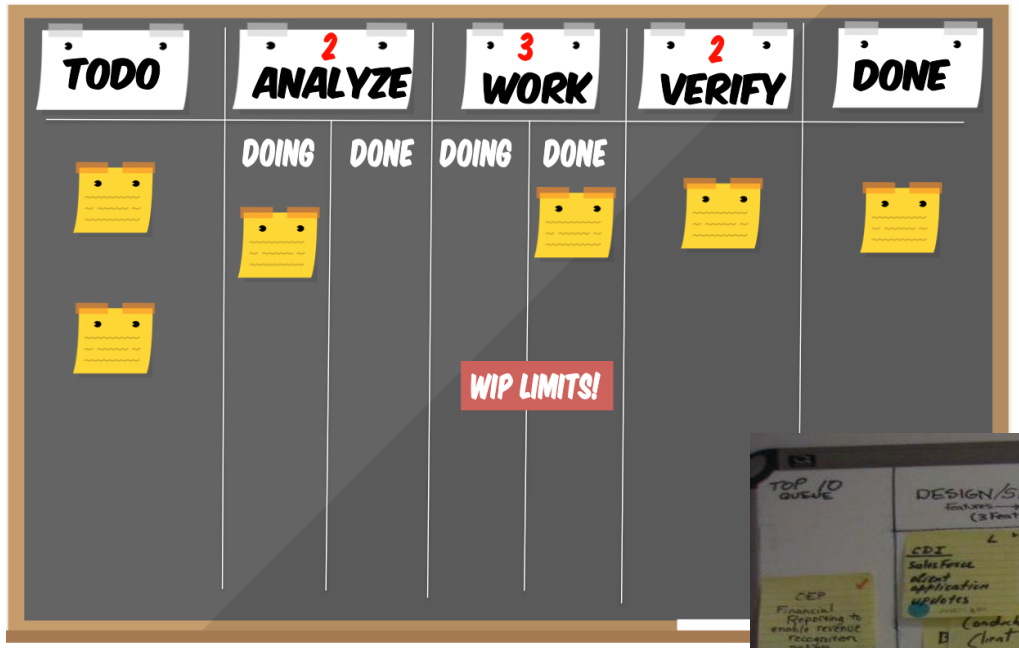
- First define goal of the sprint before you start sprint planning

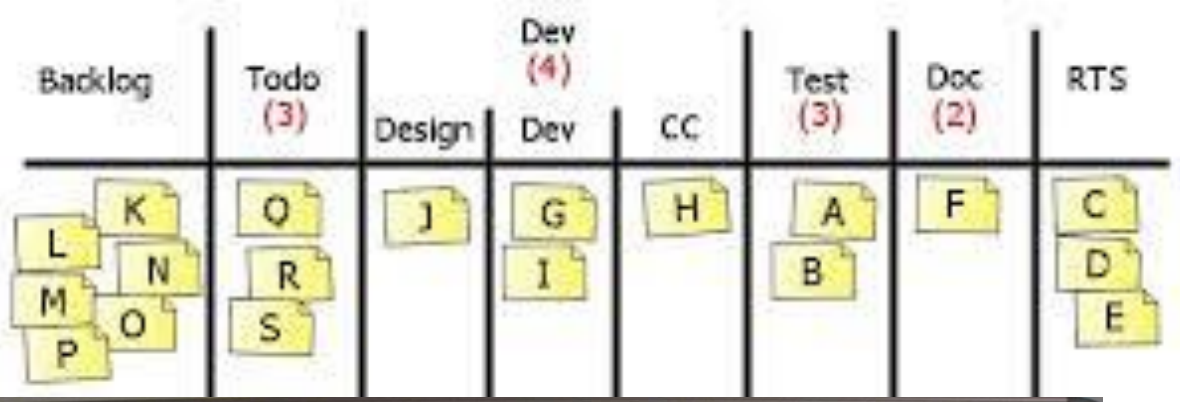
- When ?
- Who can participate?
- How much time is required?
- What is the outcome?

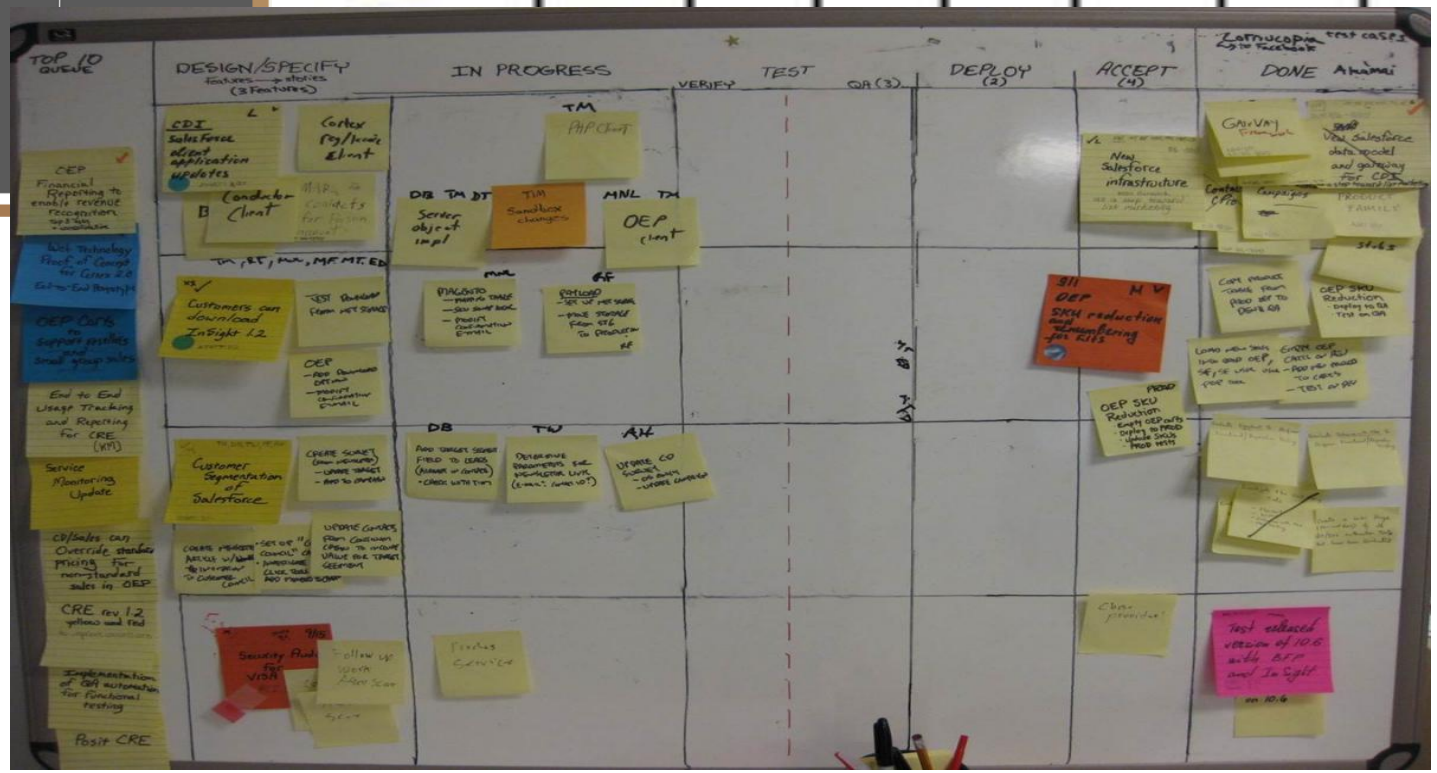
Output of sprint planning meeting

- A sprint goal
- A list of team members, and their commitment levels
- A sprint backlog
- A defined sprint demo date
- A defined time and place for the daily Scrum

Scrum Board







- Scrumban/Kanban
- Yellow: Stories
- Green: Tasks we created at Sprint planning
- Red: Tasks that were created after Sprint planning

Non programming tasks in Sprint

Example of non-programming tasks that often need to be done in a sprint

- Set up a test environment
- Clarify requirements
- Discuss deployment details with operations
- Write deployment documents (release notes, RFC, or whatever your organization does)
- Contact with external resources (GUI designers for example)
- Improve build scripts
- Further breakdown of backlog items during the sprint.
- Identify key questions from the developers and get them answered

Scrum Roles

Product owner

- Controls the product backlog
- Product Backlog grooming
- Prioritizes the features to be developed
- Conducts demo
- Business value justification
- Can cancel sprint
- Participate in sprint planning, daily scrum, retrospective
- Helps team in understanding requirements

Scrum Master

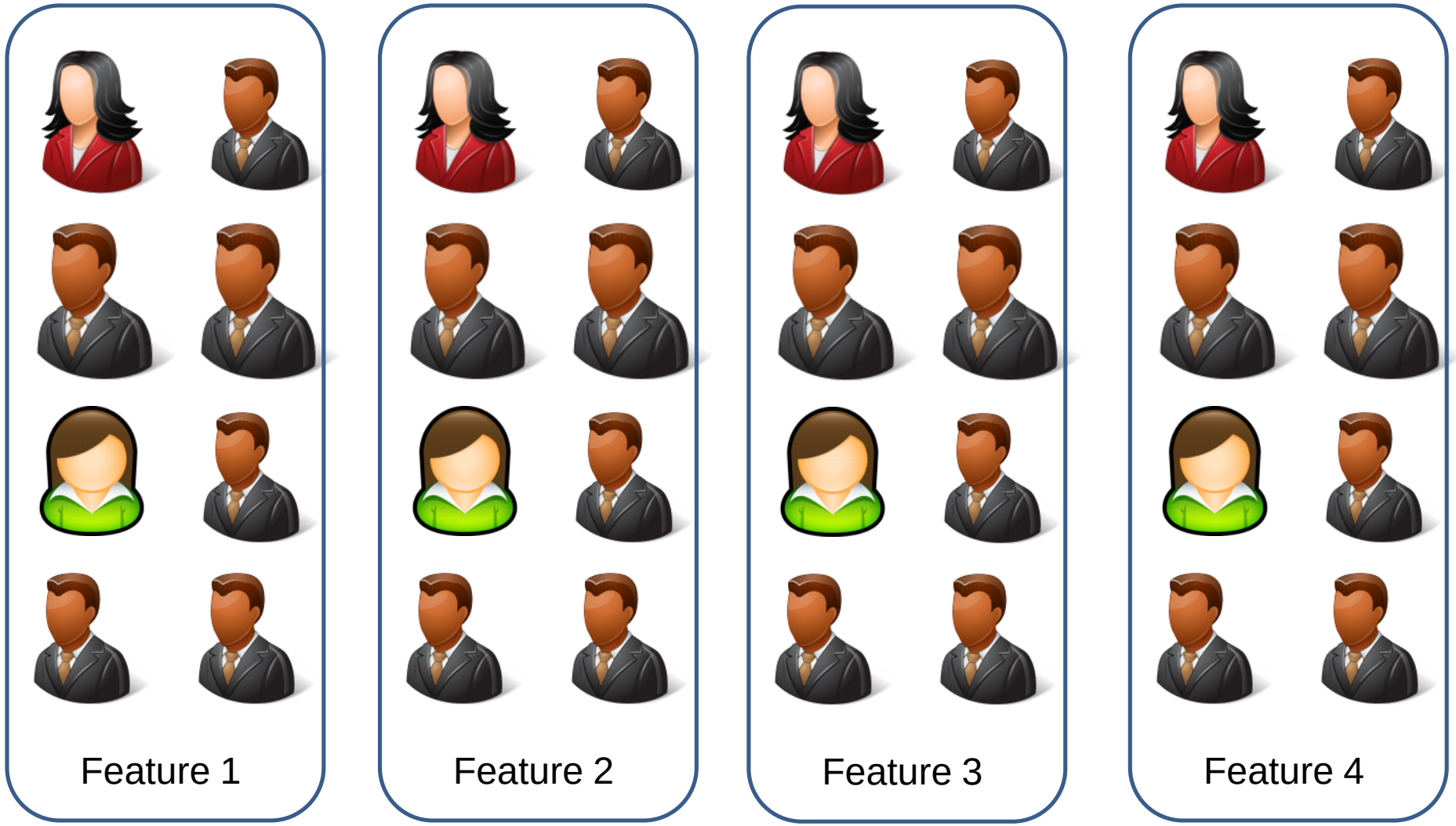
- Responsible for the success of scrum
- Scrum values, practices and rules are enacted and enforced
- Represents the management and the team to each other
- Conducts all daily scrums
- Removes impediments

Scrum Teams

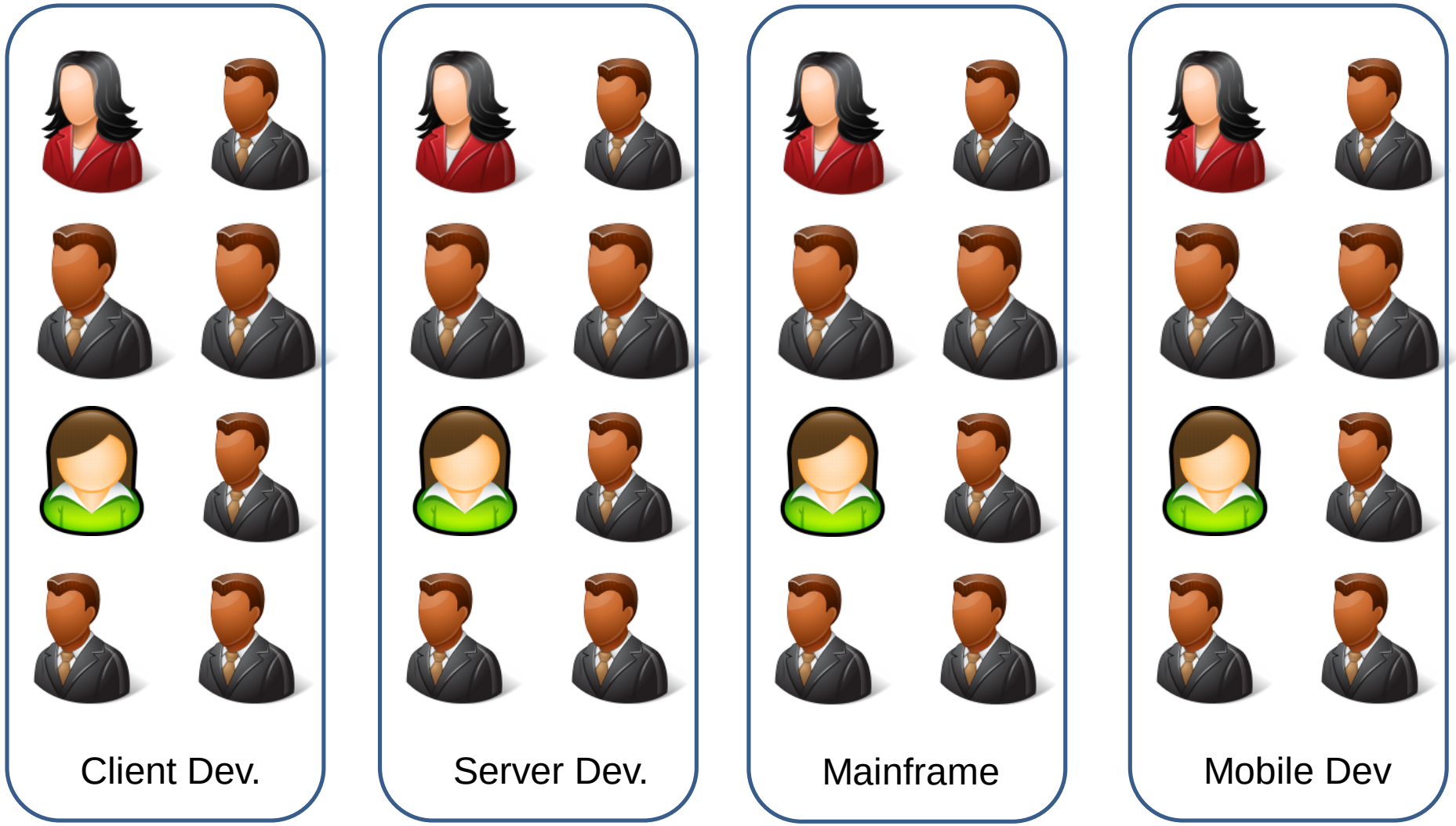
- A team commits to achieving a sprint goal. The team is accorded full authority.
- Team size is 7+/- 2
- Cross functional
- No titles
- No hierarchy

It does not mean seniority should not be respected. Local culture, value system, organization culture impacts the structure a lot

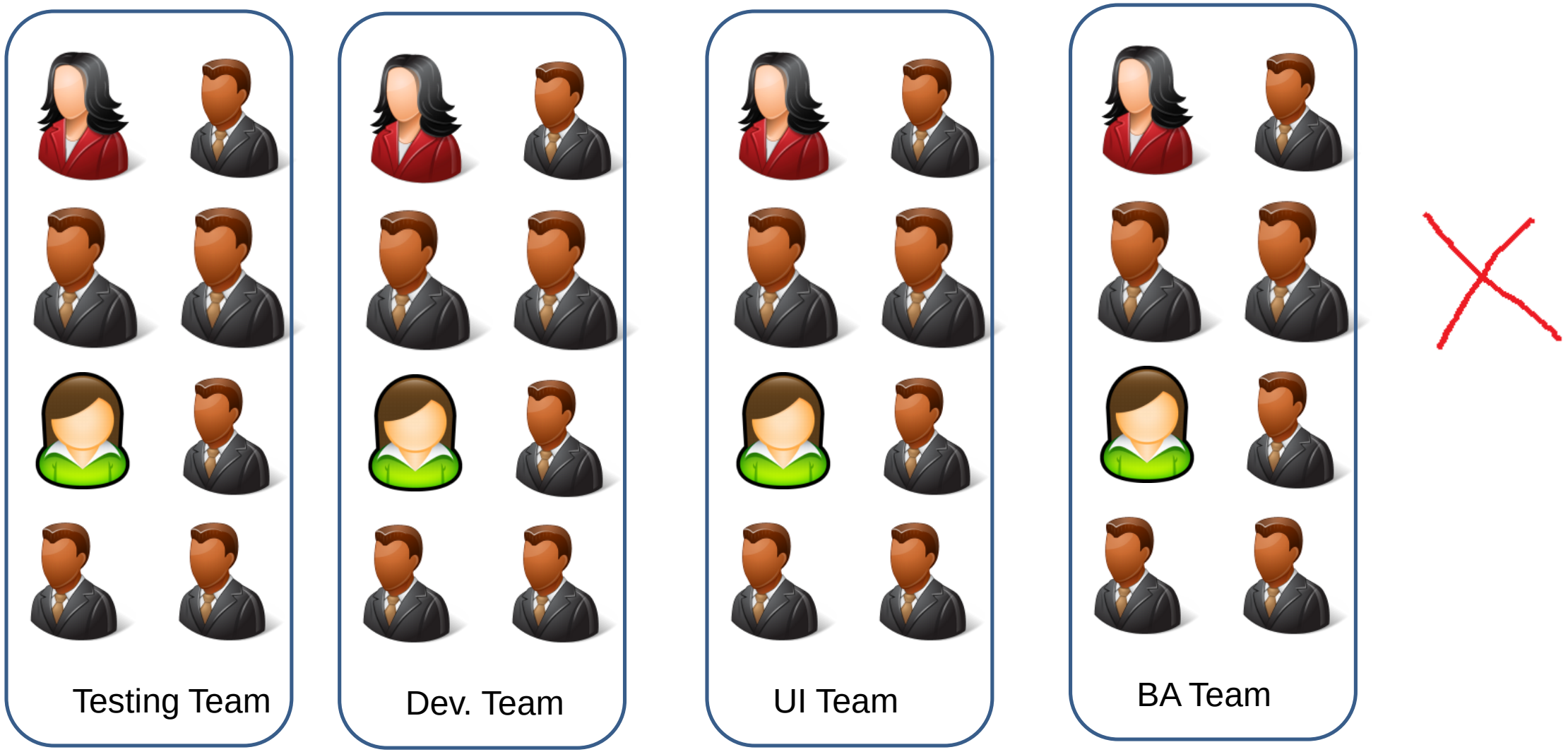
Team Strategies



Team Strategies



Team Strategies



Introspection

- Half way through the sprint, the product owner wants to change the features taken up for sprint. What should be done?
- Half way through the sprint the team wants to remove a feature from the sprint. What should be done?

Introspection

- One of the team members is unable to complete his tasks on time. What can be done?
- The team is lagging behind. As a scrum master what should you do?
- A tester wants to do coding. What needs to be done?.

Daily Standup

Daily Standup

- Information sharing session
- Re-planning and risk assessment session
- Duration: 15 min
- One person max 2 min
- One person at a time
- No status update
- No problem resolution
- Information sharing with 3 question
- No Q/A rounds
- No management interruption
- Management people may be invited to avoid other unnecessary meetings
- Only Scrum Team can talk

3 Questions of Daily Scrum

1. What did you do yesterday
2. What will you do today?
3. What obstacles are in your way?

During the scrum meeting ….

- If required the teams can revise their estimates
- No explanation (Just information Sharing)
- Update your burn down chart before scrum meeting start
- Must discipline yourself to finish the update in structured way in 2 mins.

Questions about Scrum meetings?

Why daily?

"How does a project get to be a year late?"

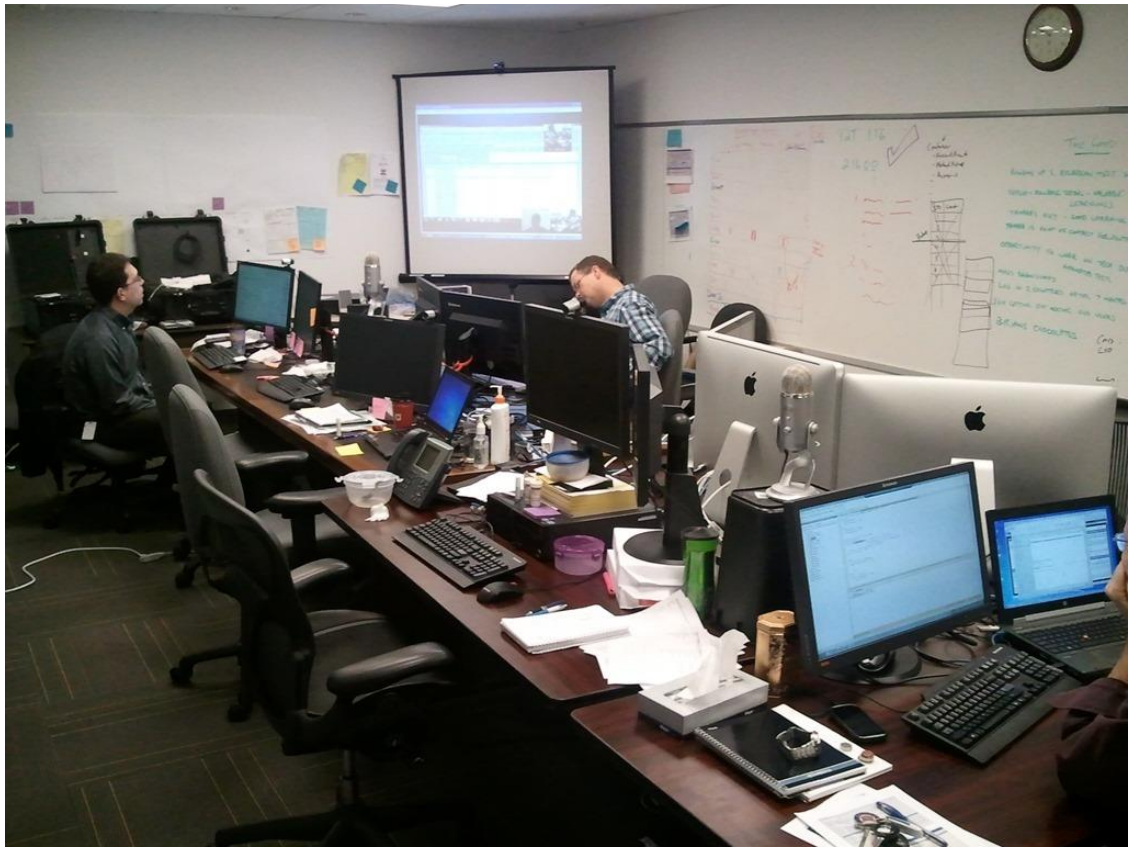
"One day at a time."

Fred Brooks, *The Mythical Man-Month.*

Can Scrum meetings be replaced by emailed status reports?

1. No
2. Entire team sees the whole picture every day
3. Create peer pressure to do what you say you'll do

Agile Team Space



Agile Team Space

Scrum Meeting Room

Introspection

- You are supposed to attend a scrum meeting at 10.a.m. The time now is 9.45 a.m and you are held up in a traffic jam. What will you do?

Exercise

- A video of Daily Standup
  - Learning from clip?

Sprint reviews

- Four hour informational meeting
- Demo of the product increment that is built during the sprint
- Attended by
  - Management
  - Customers
  - Users
  - Product owner
  - Scrum master is responsible for organizing and coordinating

Product Demo

Sprint retrospectives

- What went right
- What went wrong

Techniques for Retrospective

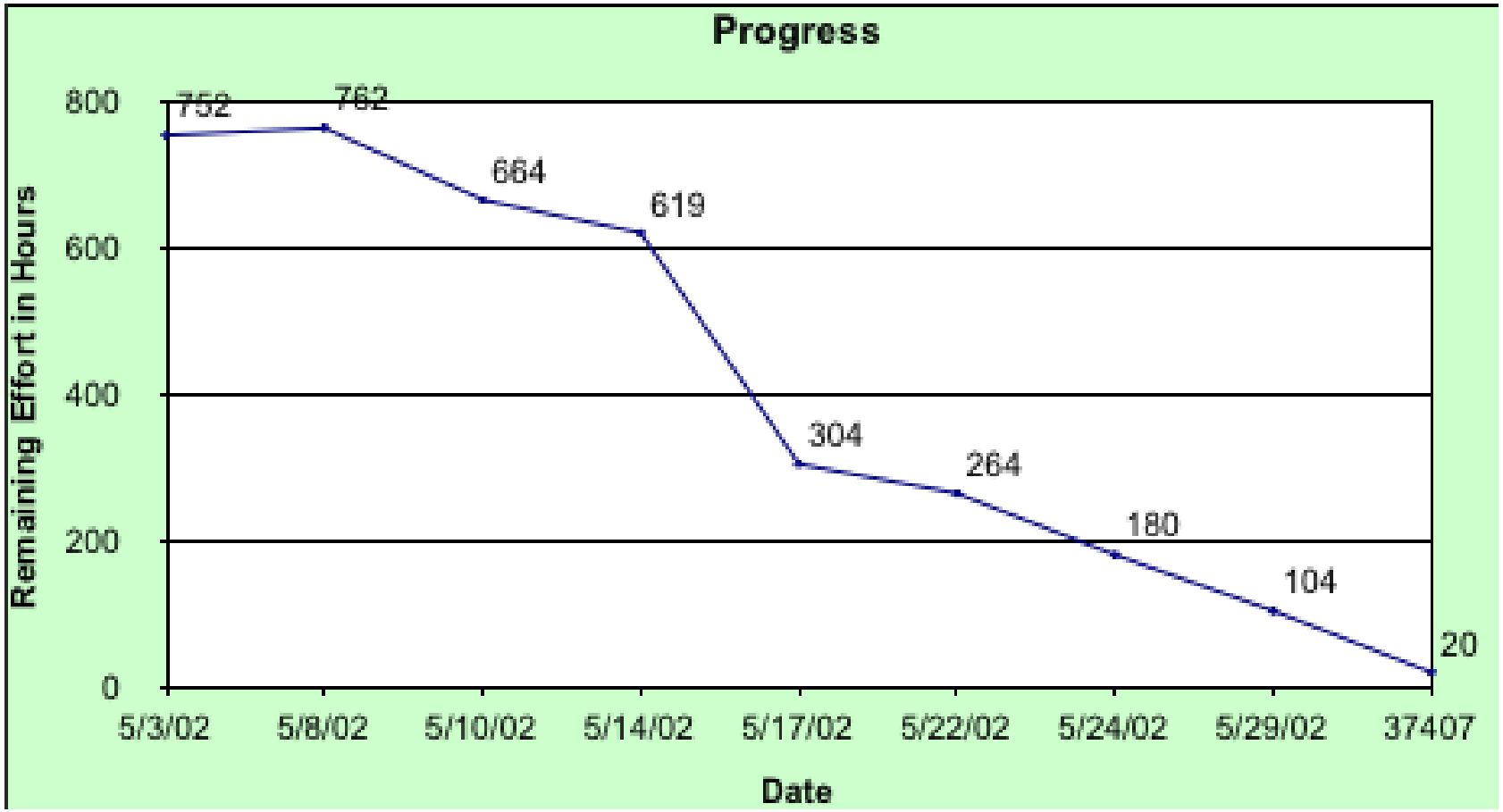
- Force Field Analysis
- Pomodoro Retrospective
- Start-Stop-Continue
- Sailboat

Exercise

- Retrospective Video
- Learning from Clip?

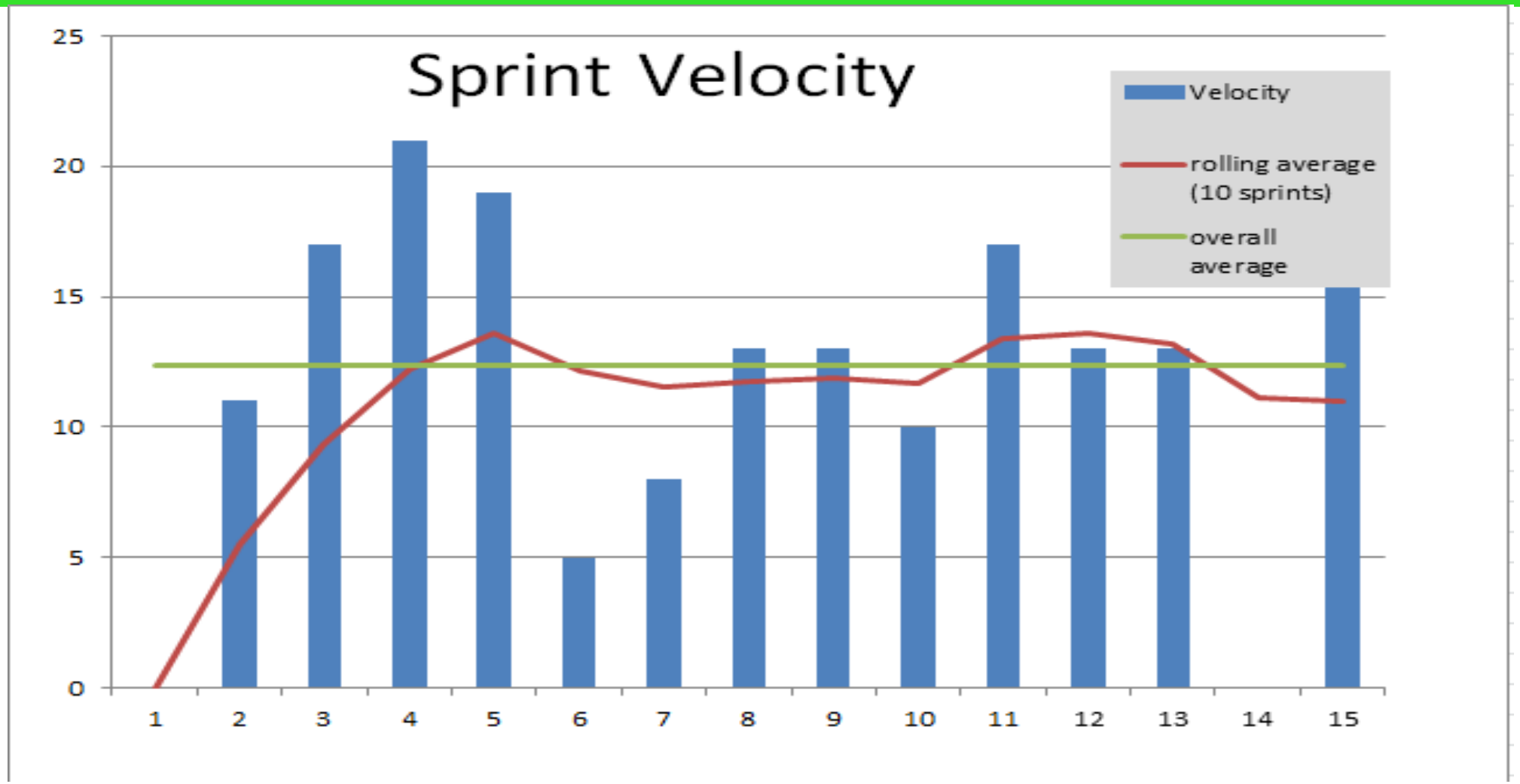
Useful Metrics

Sprint Burn down Chart



Sprint Burnup Chart

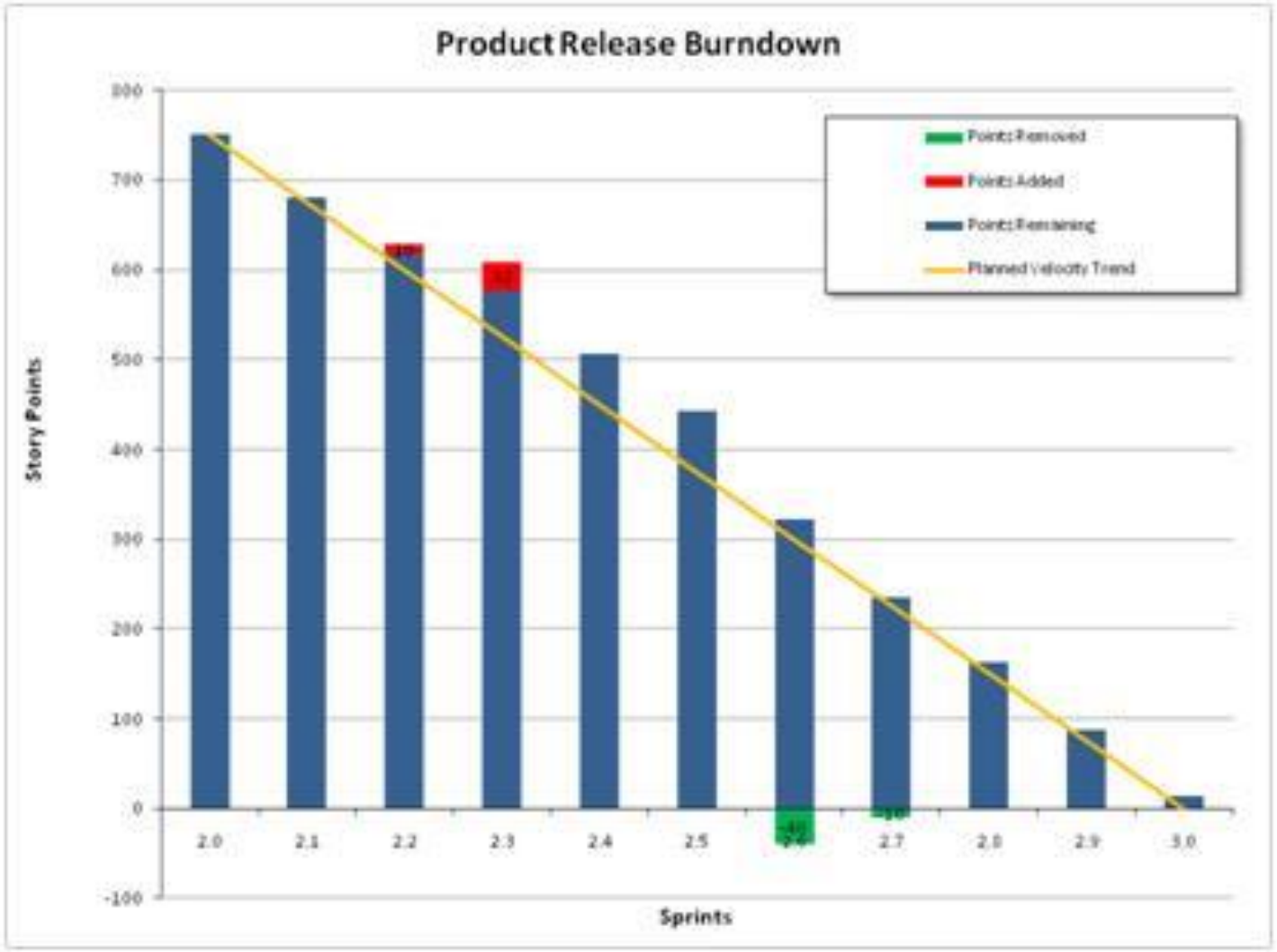
Velocity Chart



Using Velocity badly can cause Technical Debt

Release Burnup Chart

Product Release Burndown



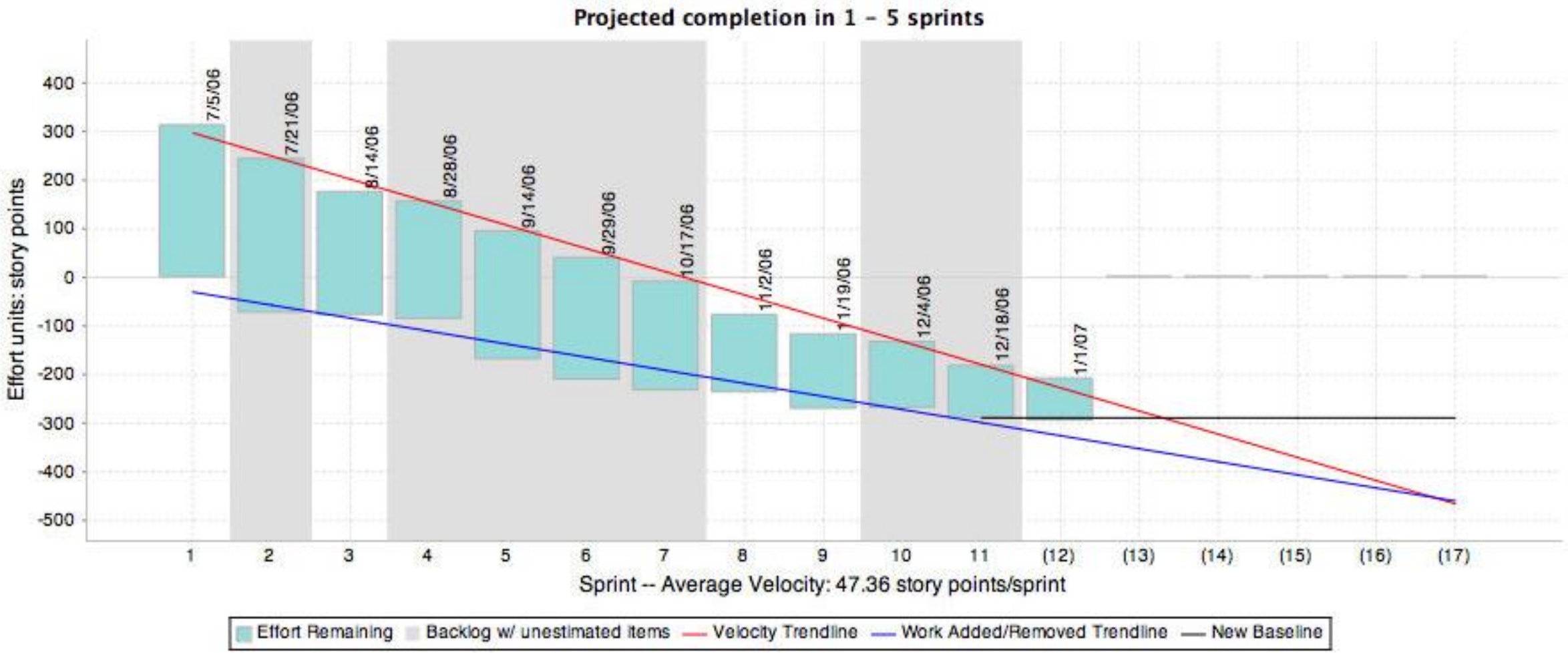
Management

- Visualize the overall project timeline and calculate total project costs.
- Management can also see from a release burndown chart when extra sprints were added because remaining work has been re-estimated or work has been added.
- They can also see the affect of other factors (such as changes in the team's velocity or membership) on the long-term project outlook.

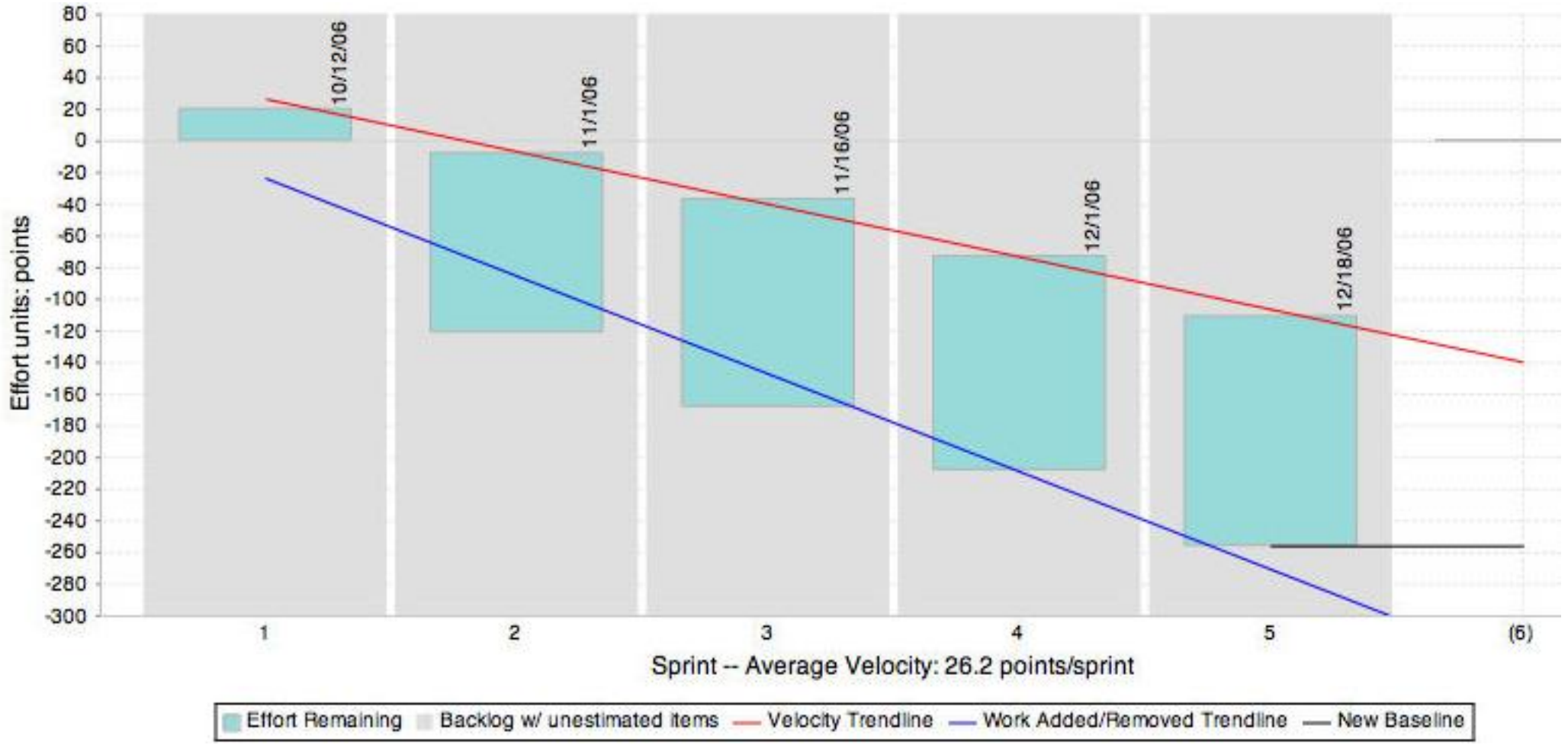
Product Owner

- Team is progressing faster than planned, they may have the option of pulling in lower priority work from the backlog.
- Team is not completing as much as planned in each sprint, the product owner may opt to de-scope some of the project, re-prioritize the backlog, or look to project sponsors to increase the length of the project or add project resources.

Enhanced Burndown (Converging)

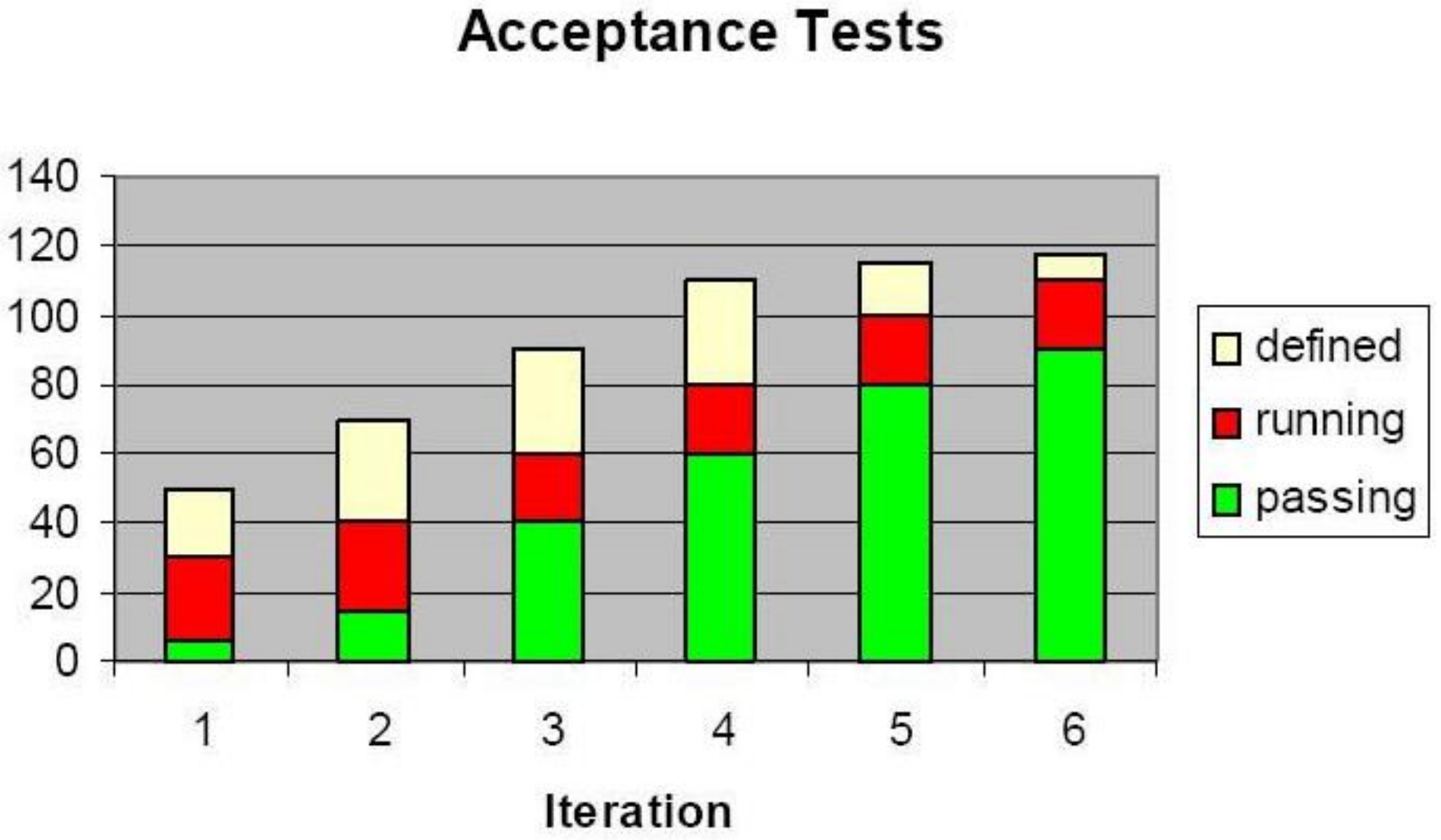


Enhanced Burndown (Non-converging)

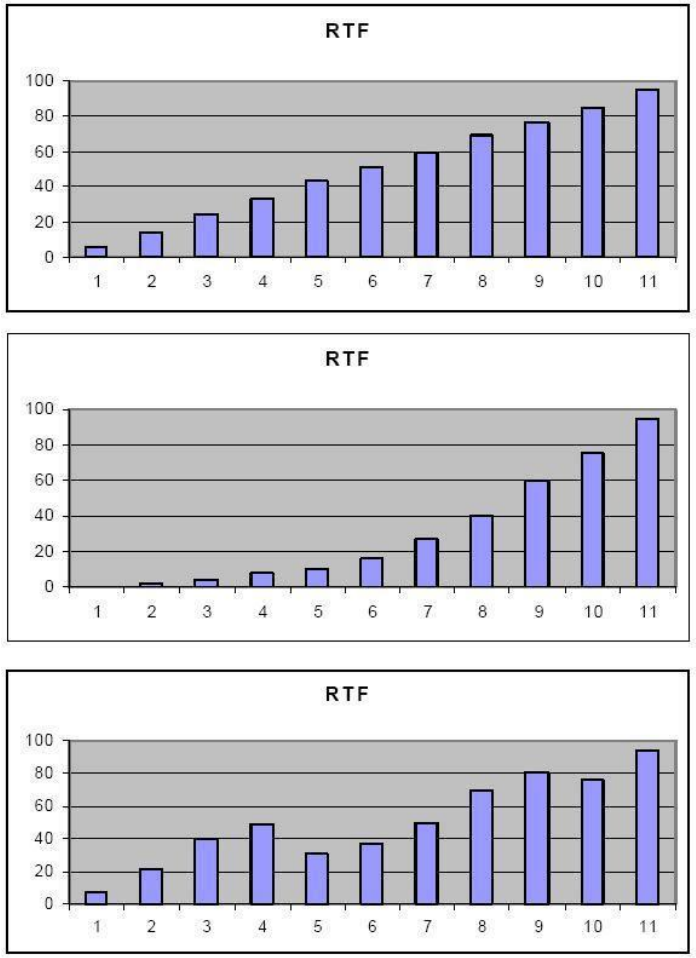


Value Burndown/Up Chart

Product Status



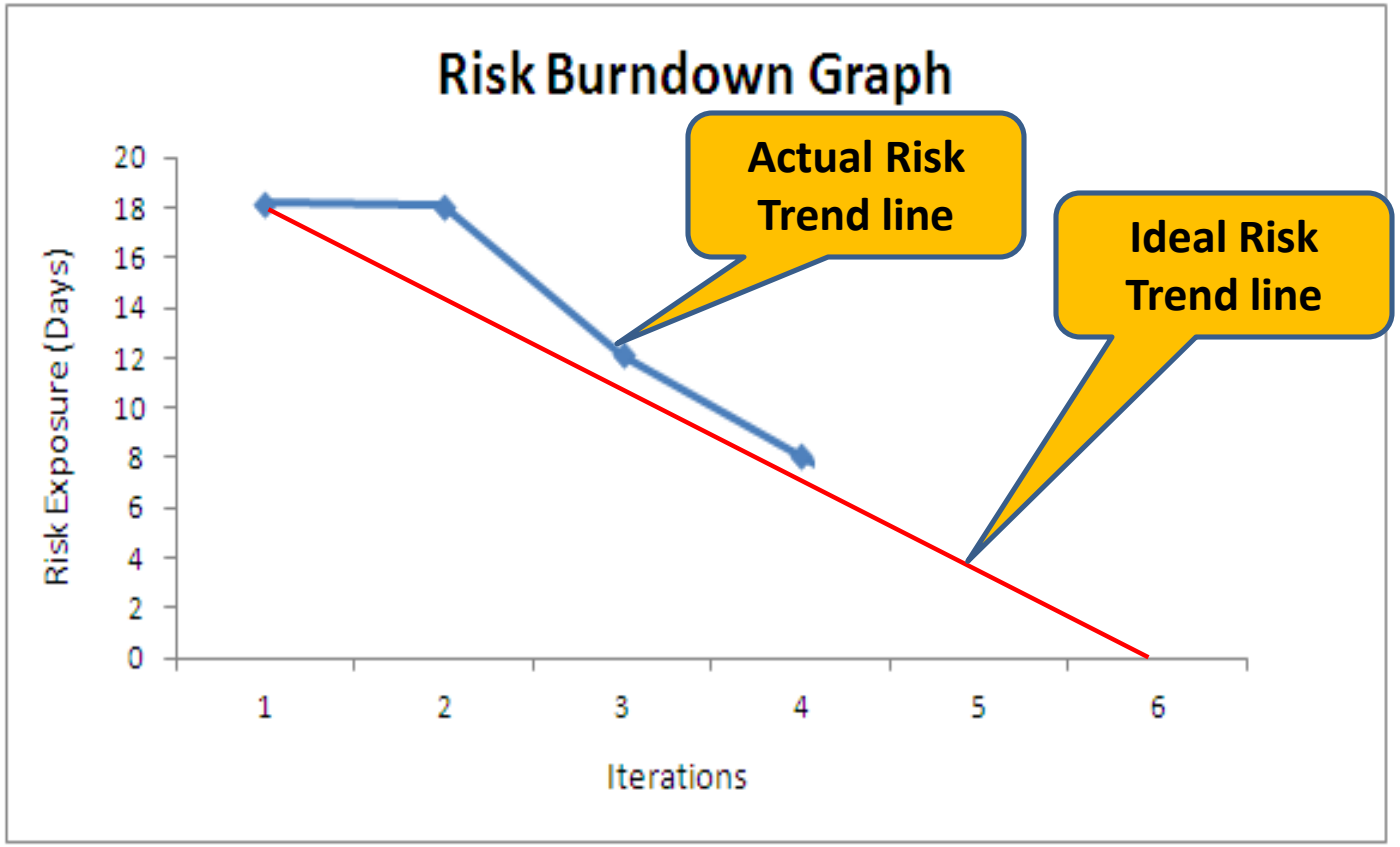
Running Tested Features



- The RTF graph should look like , as the team should be adding features early and often and consistently
- If it looks like , there might have been too much waterfallish behaviour going on – merits a conversation
- This behavior indicates that things broke, and tests were failing. Questions *must* be asked

- A story is said to “pass” if all of its acceptance tests pass Otherwise, it fails
- It treats all stories as “equal”; that is, no story is worth more than another – they all provide equally to the count

Risk Burn-down Graph



If actual risk trend line is above than ideal trend line then it means risks are not coming down at the appropriate rate

Tools for Agile Project Management

Tool Support for Scrum

- Doing Scrum requires no tools
- Using a tool will not make you do Scrum
- Identify your metrics and reporting needs first
- Find tools that work with you
- Start simple and stay that way
- Find tools with unobtrusive touch points
- Assign Tool monitor to handle tool usage

Type of Tools Used by Agile Teams

- Collaboration Tools
  - Whiteboards, wiki, NetMeeting
- Communication Tools
  - Talking, eMail, IM, video conferencing
- Management Tools
  - Big charts, spreadsheets, tracking tools
- Development Tools
  - Version control, build (Ant), automated testing (xUnit, Fit), integration (Cruise Control), IDE w/refactoring support (Eclipse, RefactorIT)

Important Concepts in Agile

Agile Product Build

- Build complete product, all the time
- One button should produce needed documentation, build the product executables, create installation materials, produce test results and tested components
- Build should also work from command line
- Everyone in team should use the same build process

Agile Documentation

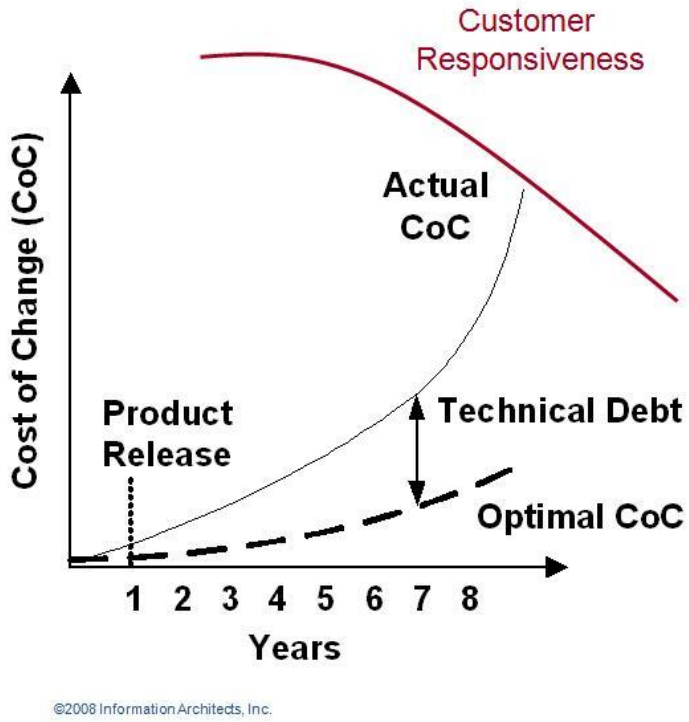
- Maximize stakeholder investment. Produce document only when
  - It is needed by a stakeholder
  - Needed to define contract model
  - You need to think something in many iterations involving multiple groups
  - It is needed for external communication
- Document only those things which are least likely to change
- First identify the specific customer of the document
- The document facilitates in estimating
- Sufficiently index, details, accurate and consistent

Agile Testing

- All code must have test cases, ideally they should be created earlier
- Unit tests should be executed during automated build
- A build should be performed many times a time, ideally whenever anything is checked-in or committed to configuration server
- If anybody's code causing crash he should be informed immediately and that person should fix that problem first
- All unit test must pass before code can be released

Refactoring

- Agile programmer writes simple and bare minimum code they do not complicate the code
- Down the line structure the code without changing its behavior. It helps in improving the quality (maintainability, readability) of code



- Once on far right of curve, all choices are hard
- If nothing is done, it just gets worse
- In applications with high technical debt, estimating is nearly impossible
- Only 3 strategies
  - Do nothing, it gets worse
  - Replace, high cost/risk
  - Incremental refactoring, commitment to invest

When does Agile Methodologies Works Best?

- When problems you are solving have
- following characteristics
- – Going to **change** while solving
- – **Speed** of development cannot be determined
- – **Turbulence** in environment
- – Customer **doesn’t know** how exactly it will look
- like

Agile Health Checkup

To know the agile maturity in your project you can perform following Agile Health Checkup. Discuss in group and ask team to rate these parameters on the scale of 1-5

1. **Frequent Delivery**
2. **Reflective Improvement**
3. **Close Communication** (does it take less than 1 min to get you question answered by a person who know the answer?)
4. **Focus** (everybody understand the goal and desired outcome of the delivered software?)
5. **Personal Safety** (can you give bad news to your boss?)
6. **Easy Access to Outside Experts**
7. **Strong Technical Environment** (SVN used? Test Automation?)
8. **Sunny Day Visibility** (Does everyone on the team understand the rate of progress being made on the product?)
9. **Regular Cadence or Rhythm** (Is heartbeat of the system on?)

Technical Debt

- While writing code agile programmer do not pay attention to structure, duplication etc but to functionality and make sure that code is passing all unit test cases
- In this process if they do not clean the code by structuring (refactoring) it will become unreadable and un-maintainable and over a period of time this cause increases response time to fix the problems, adding new feature and deteriorate the quality.
- The concept of unclean code is called technical debt. Technical debt keep increasing over the period of time therefore Agile team need to pay this debt back by putting efforts in refactoring.

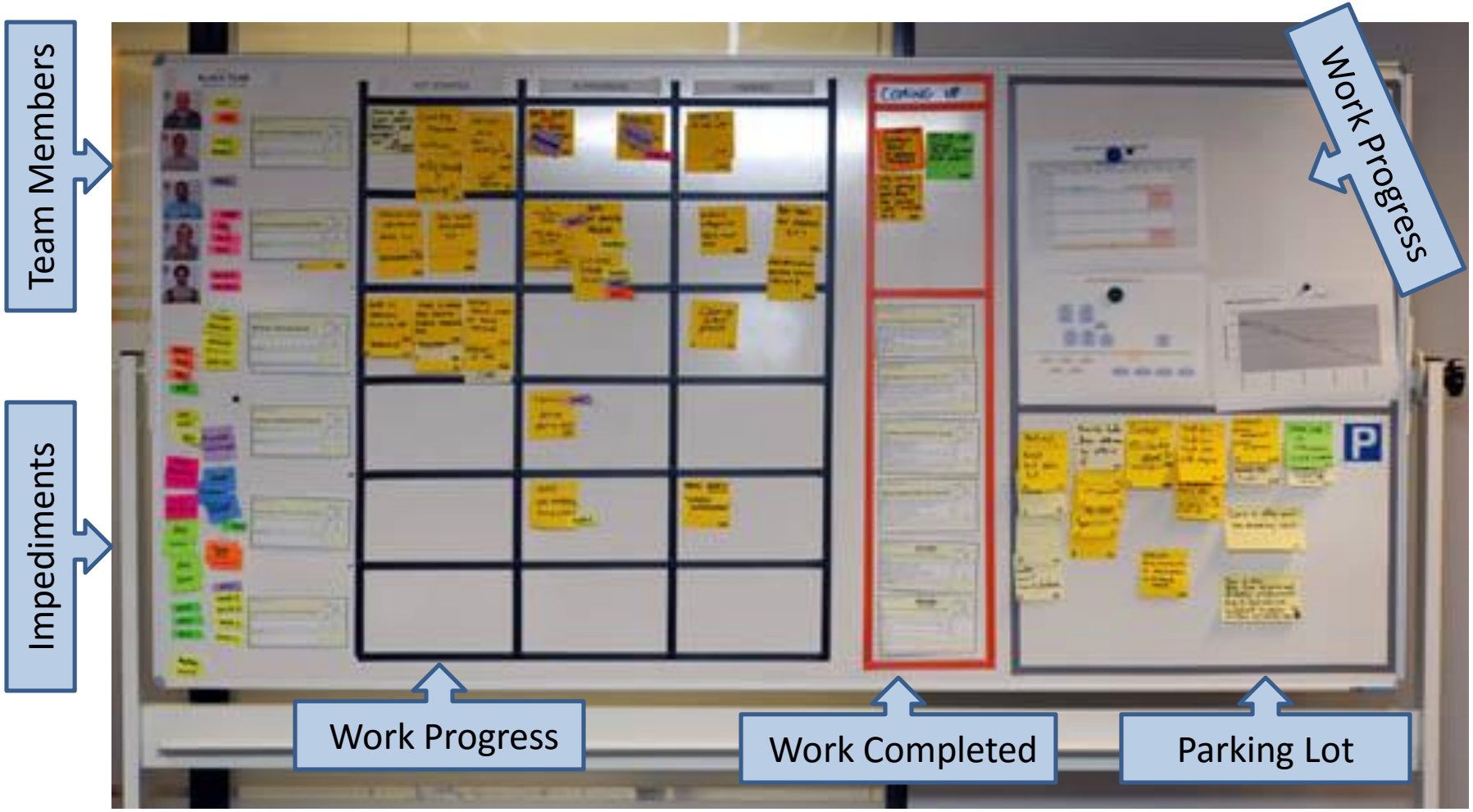
Travel Light

- Simple Design (Do Simple Things). Simplify Vigorously
- You Aren’t Gonna Need It (YAGNI),
- Once And Only Once (DRY),
- TAGRI (They aren't gonna Read It)

Osmotic Communication

- Agile project relies on collocation, minimum documentation, least reporting, maximum constructive engagement. This can be achieved if frequently sought important information is available in published form in team space
- Whoever need the information can go and get the information without wasting time in requesting, making, sending, receiving the information
- Benefits
  - Least cost, effort and time waste in communication
  - Updated information is always available without making new reports
  - People can get whatever particular information they are looking for
  - Happens at the same time
  - Feedback loop is quick
  - Those people who are left in regular reporting also get benefitted
  - No junk, old, repetitive information, but fresh and useful.
- Dis-benefits
  - Some people get extra information which they do not need
  - It is left to individual's interpretation

Information Radiator



Time-boxing

- Iteration length is time-boxed.
- Following sequence of activities takes place in any time-boxed iteration
  - Grooming product backlog (done by product owner, in parallel to iteration work)
  - Iteration Planning (1 hour for every week of iteration)
  - Daily stand-up (15 min, max 2 min for one person, typical team size of agile team is 7±2)
  - Regular iteration work
  - Iteration Review (1 hour for every week of iteration)
  - Iteration Retrospective (1 hours for every week of iteration)

Definition of Done (DoD)

- The DoD changes over time. Organizational support and the team's ability to remove impediments may enable the inclusion of additional activities into the DoD for features or sprints.
- Continuous Integration (CI) helps you validating the "Doneness"
- There are 3 level of DoD
  - Story DoD
  - Iteration DoD
  - Release DoD

Definition of Done (DoD)

Story "Done"

- Unit test should provide 60-70% test coverage
- Story is either written in pair or reviewed by peer
- All code checked in
- All unit code passed
- All acceptance test case passed
- Story accepted by owner

Definition of Done (DoD)

Iteration "Done"

- Iteration should have defined Iteration Goal
- All acceptance test cases should run for all user stories in Iteration
- All stories completed must be accepted by the product owner
- Defects identified are fixed or planned for future
- Code performance is tested and accepted
- If database is involved then database script should be available, automated and tested
- Backup of iteration work product is taken

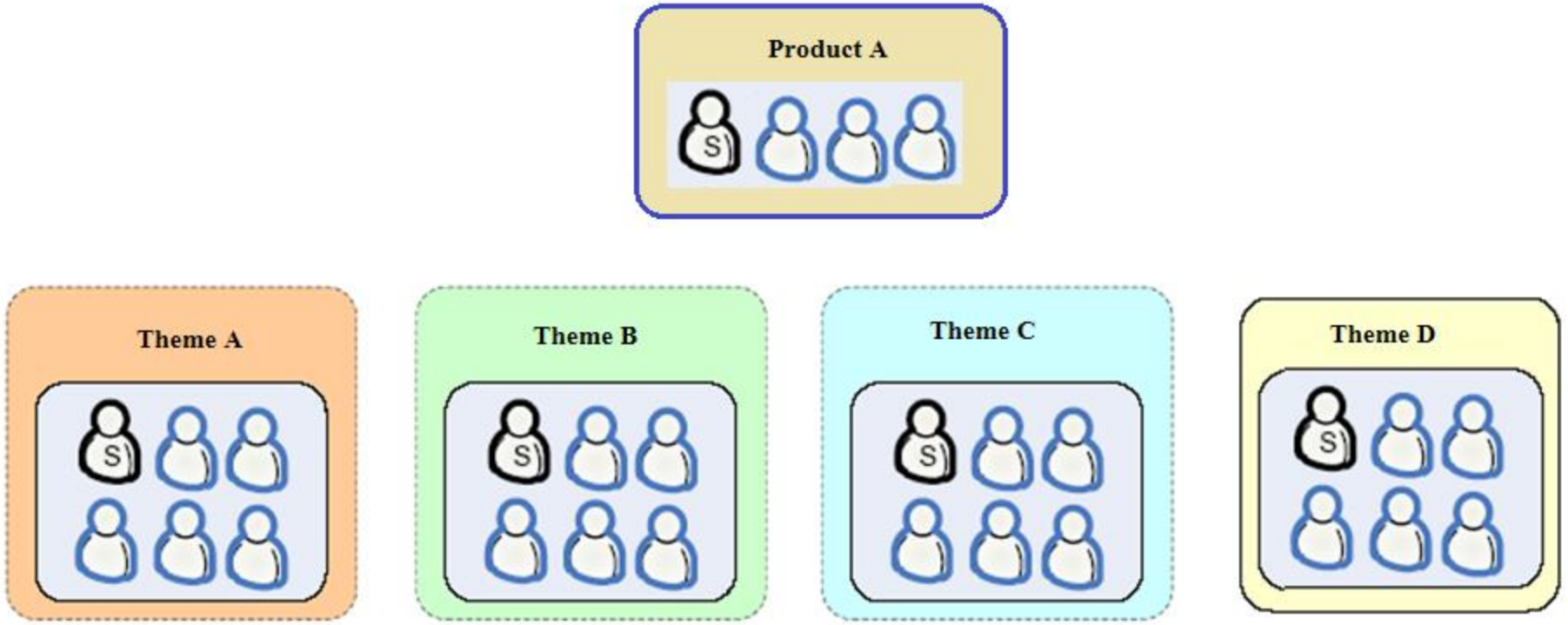
Definition of Done (DoD)

Release "Done"

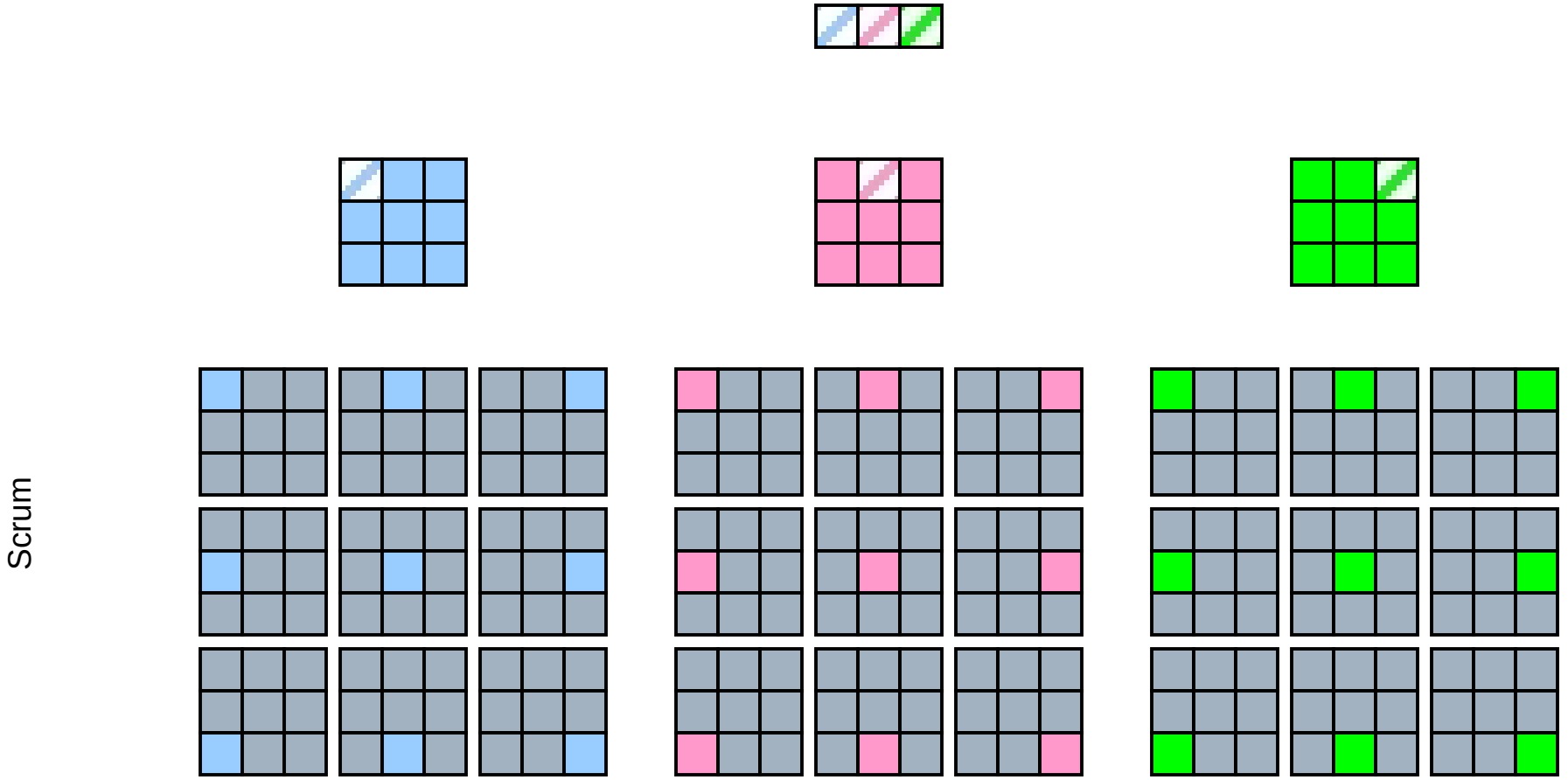
- Release should have defined Release Goal
- Product has formal release date
- Product is deployed on staging area
- Stress testing done and results accepted
- All non-functional requirements are tested and results accepted
- Required documentation is available
- Release should not have any known bug

Scaling Scrum

SCRUM of SCRUMS



Scrum of Scrums



Scrum Smells When....

Scrum Smells !

Loss of Rhythm

- Symptom: Sprints are not always the same length.

Talking Chickens

- Symptom: Chickens attending the daily Scrum are allowed to ask questions or make observations.

Specialized Job Roles

- Symptom: A project team has highly specialized job roles or descriptions such as Architect, Designer, DBA, or Tester

Scrum Smells !

Missing Pigs

- Symptom: Not all pigs attend the daily Scrum Meeting

Persistent Signatures

- Symptom: The wild fluctuations shown on a team's initial sprint burndown charts continue to be seen in much later sprints

ScrumMaster Assigns Work

- Symptom: Work is assigned by the ScrumMaster rather than signed up for by developers.

The Daily Scrum is For the ScrumMaster

- Symptom: The Daily Scrum Feels like it is a status update from the team members to the ScrumMaster

Agile Agreement

- The Product owner promises the team that she/he will supply an initial product backlog
- The product owner promises the team that he/she will prioritize the product backlog when needed
- The product owner promises that an empowered "voice of customer" will be provided to answer business domain question promptly (minutes/hours, not days)
- The ScrumMaster promises to keep the team healthy by focusing on the removal of impediments, both internal and external
- The ScrumTeam promises that its work will be transparent, that it will make decisions and solve problems as a group, and that no individual team member will be left behind
- Each member of the scrum team promises that they will bring issues, problems, impediments and realities encountered to the ScrumTeam.

Signed By:
Organization

Signed By:
Scrum Team

Agile Agreement

- The ScrumTeam promises the stakeholders that there is product owner on the ScrumTeam driving the ScrumTeam based on stakeholder interests.
- The ScrumTeam promises to use the stakeholders' time wisely, by focusing on questions that are relevant to the work being done now.
- The ScrumTeam promises to deliver demonstrable product at the end of every sprint for review and validating by the stakeholders
- The ScrumTeam promises that they will do quality work the best way they know how within the constraints set forth by the organization.
- The organization promises the ScrumTeam that there are stakeholders (including SME) who will help when needed
- The organization promises that they will help the scrum master in the removal of impediments to the scrum team's progress
- The organization promises the ScrumTeam that they will not change prioritize or constraints in the middle of a sprint without ScrumTeam's consent.
- The organization promises that being on a Scrum Team will not hurt of members careers.

Signed By:
Organization

Signed By:
Scrum Team

Hari P Thapliyal,

PMP, PMI-ACP, MCITP, PRINCE2 Practitioner, CSM, MBA, MCA, CIC, PGDFM

PMO Architect & Project Management Evangelist

Profile: http://in.linkedin.com/in/harithapliyal

For content related queries please contact me via

hari.Prasad@vedavit-ps.com