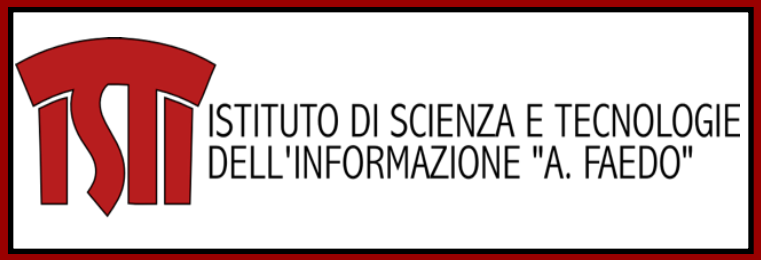


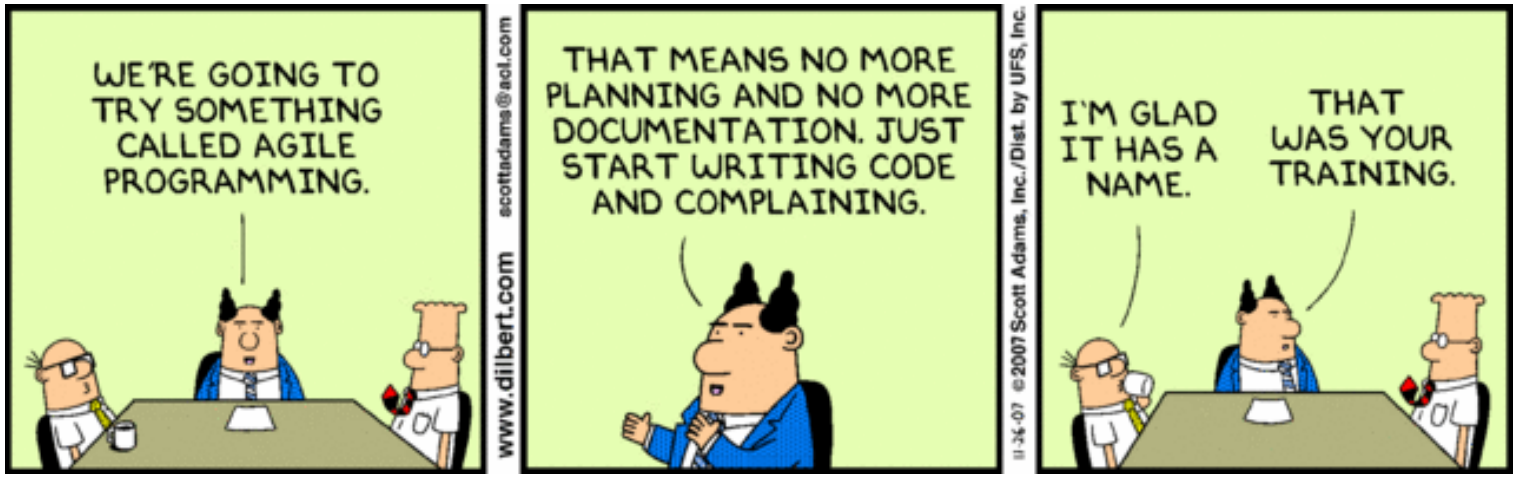
Lean and Agile Development With Scrum (Part 1)

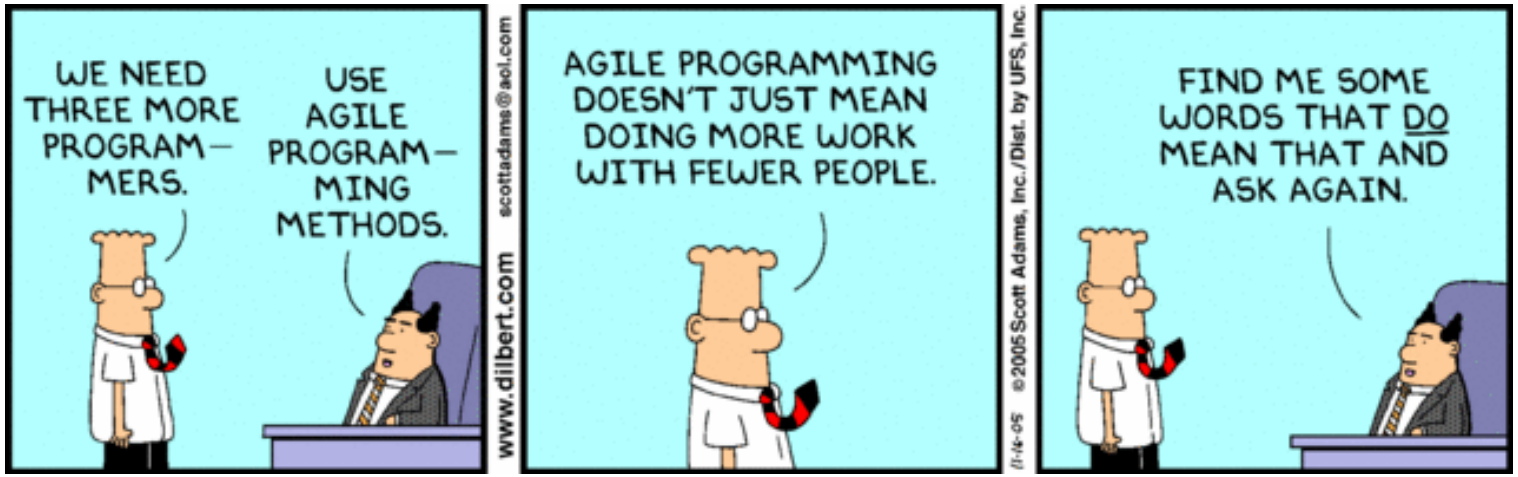
Lucio Davide Spano

lucio.davide.spano@isti.cnr.it
spano@di.unipi.it

3 May 2012

Agile Programming





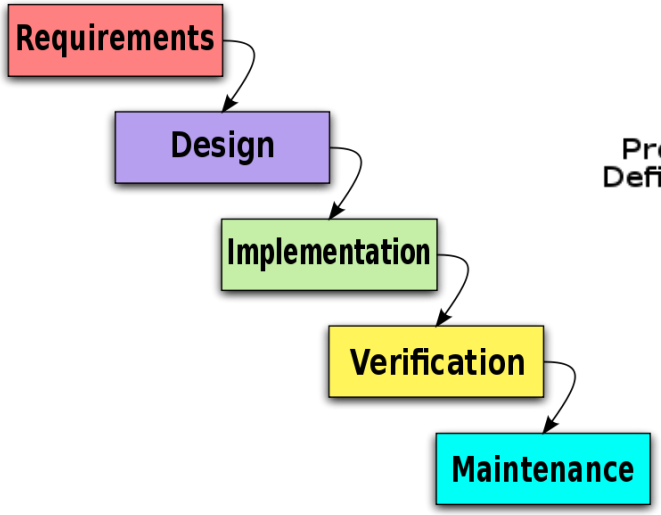
http://www.dilbert.com

HIIS Lab

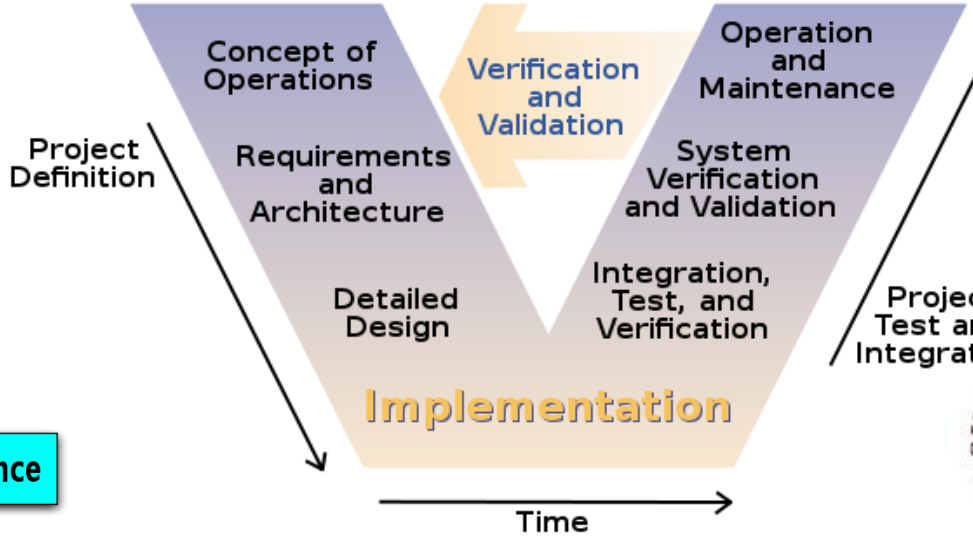
IN SUPREMÆ DIGNITATIS 1343

Traditional Software Development

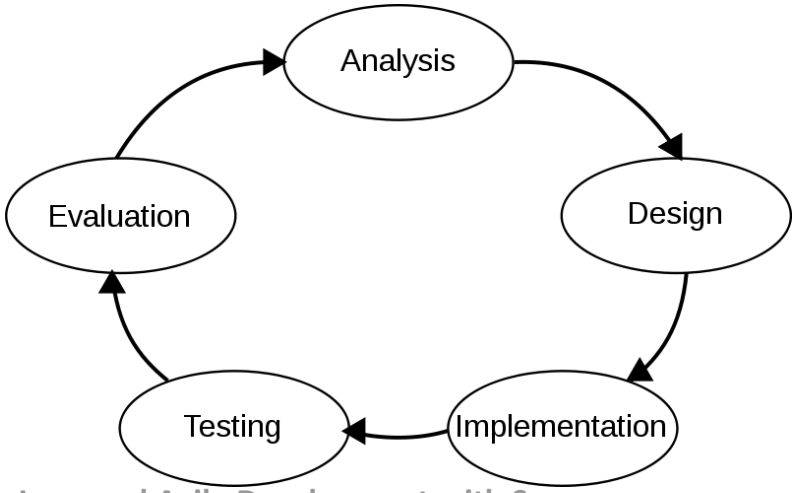
Waterfall



V-Model



SDLC



The steps in practice

- Detailed planning phase at the beginning
- Estimation of work required (Gantt charts)
- Plan reviewed and approved by stakeholders
- Teams start to work…
- Different pieces developed by specialized teams
- Product-line handoffs
- Integration
- Quality Assurance
- Delivery

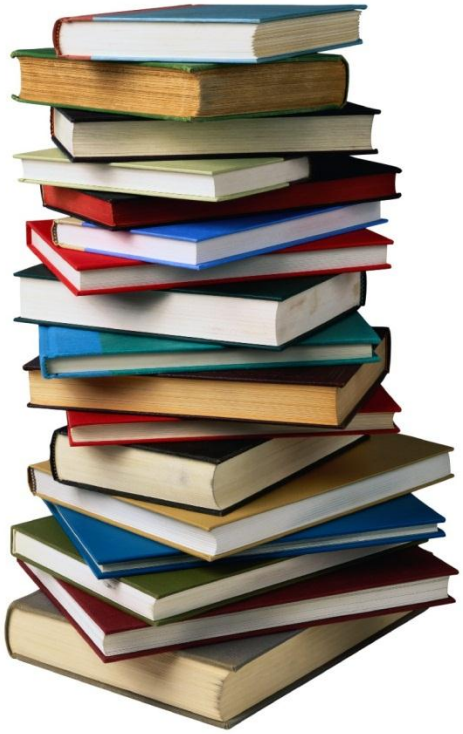
Strengths

- Logical
- Precise plan
- Product-line organization

Weaknesses

- Humans are involved…
- … and in all phases!

Common problems (1)

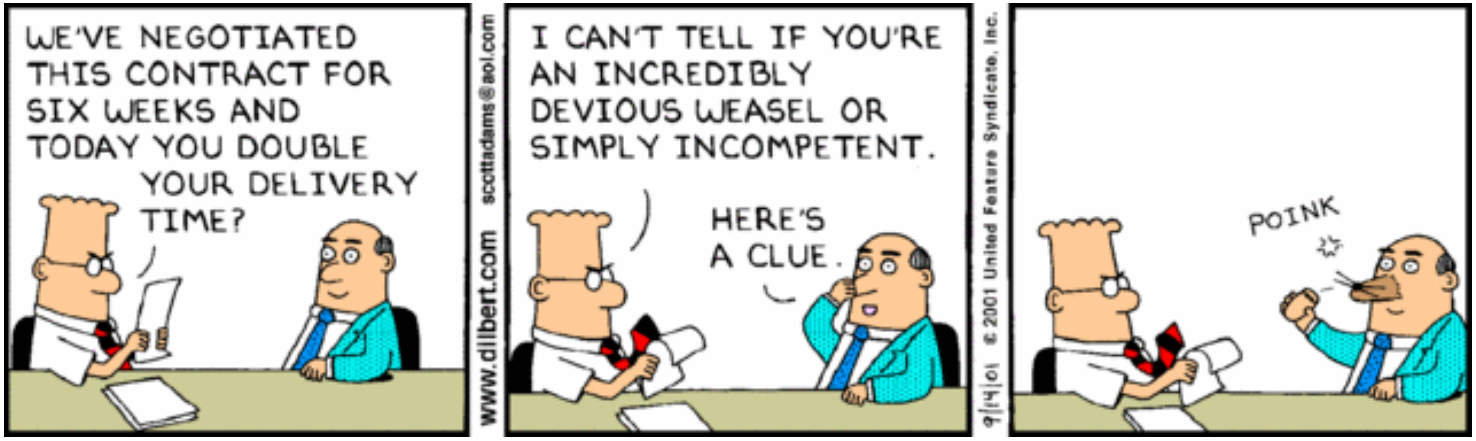


- Good ideas
  - in the middle of the process
  - at the end
- Documentation for handoffs
  - No one reads it
  - Misunderstandings

Common problems (2)

- "He's asking me for something that is not in the specification!" (or in the menu…)

Common problems (3)



- Variability in time estimation

Common problems (4)

- Customer feedback

The Agile Approach

- Things to be considered in software development
  - Learning
  - Innovation
  - Change
- Working software VS end-to-end specifications
- Cross functional teams
- Rapid iterations
- Try to control variability with early feedback
- Agile is for Agile!

Be Agile and not *Do* Agile

- Agile **does not mean**
  - Fewer defects
  - Higher quality
  - Higher productivity
  - No organization
  - Sloppiness
- Agile **means**
  - to be organized to react to changes
- It is a system challenge
- Cannot be ensured simply adopting a practice (value based and not practice based)

Agile Values

1. **Individuals and Interactions** over processes and tools
2. **Working software** over comprehensive documentation
3. **Customer collaboration** over contract negotiation
4. **Respond to change** over following a plan

- *The values on the right are important too.*
  The point is that the ones on the left are more important…

The 12 Agile Principles (1)

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 shorter time scale.

4. Business people and developers must work together daily throughout the project

5. Build projects around motivated individuals. Give them the environment and support they need and thrust 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

The 12 Agile Principles (2)

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
9. Continuous attention to technical excellence and good design enhances agility
10. Simplicity – the art of maximizing the amount of work not done – is essential
11. The best architectures, requirements, and designs emerge from self-organizing (self-managing) teams
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjust its behaviour accordingly

Scrum

Scrum summary

- Transparency
- Inspect
- Adapt

Caveat

"Scrum is not the 'solution'. It assumes that there is no recipe to solve the complex problems of R&D. Rather, it is a framework for self-organizing and cross-functional teams to increase transparency, inspection, and adaptation- creating better feedback and using it well. Scrum exposes weaknesses, does not solve weaknesses."

[Craig Larman]

Big Ideas

- Cross functional teams do everything
- Potentially Shippable Product Increment every 1-4 weeks
- Time-boxes cannot be extended
- The release is steered by the business-side (Product Owner), not by an R&D project manager
- Inspect and Adapt

Scrum Roles

Team

Product Owner

Scrum Master

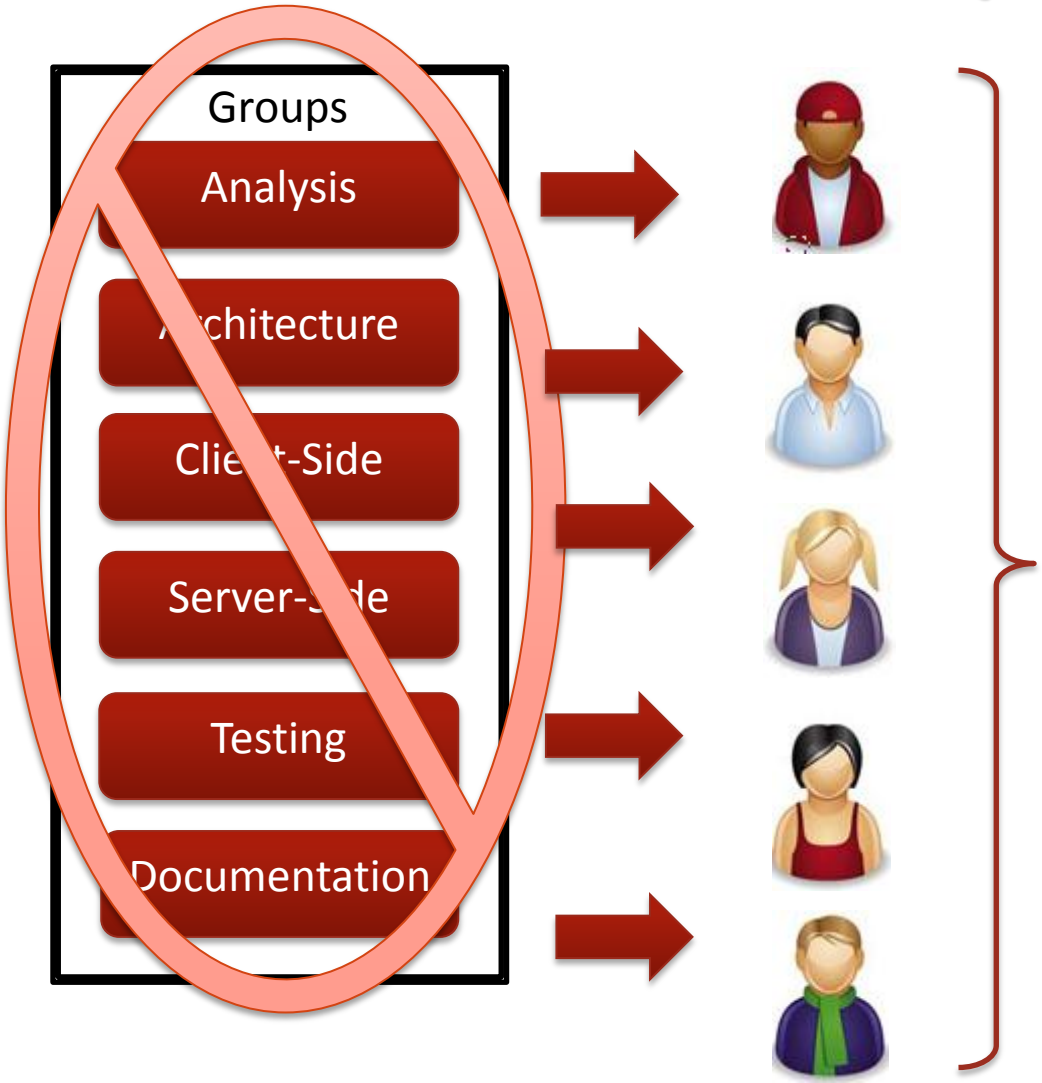
Product Owner

- Responsibilities
  - Product features identification
  - Return of Investment
  - Feature list priority (continuous)
- A single person
  - The customer or the voice of the customer
  - Product Manager or Product Marketing Manager
- Powers:
  - Drives directly the development from a business perspective
  - Accepts or rejects work results
  - Stops a Sprint if necessary

Team

- Responsibilities
  - Build the product
  - Commit what they can do in a Sprint
- Characteristics
  - Cross-functional
  - Self-organizing
  - No project (or team) manager
  - Avoid multitasking
  - Feature teams
- 7 ± 2 persons
  - Ideally co-located

Cross-functional teams (1)



Go where the work is

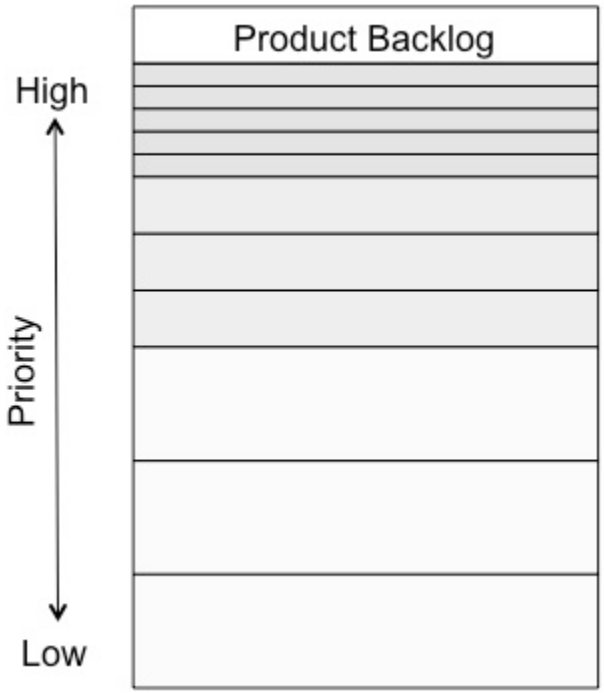
Cross functional teams (2)

- People have different specialities
  - Primary
  - Secondary
  - Tertiary
  - …
- Objective: try to “get the ball across the goal line”!
- Do not **wait** for “someone who can do it faster than us”
- Reduce waste
  - Handoff
  - People waiting while other are overwhelmed
- Increase learning
- No role other than team member inside a team

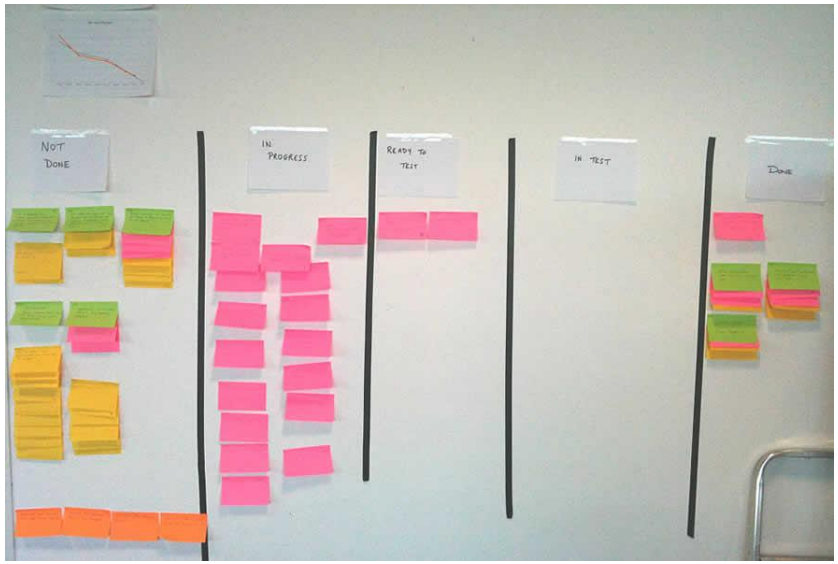
Scrum Master

- Responsibilities
  - Teaches Scrum to all roles
  - Coach for the team
  - Removes barriers for productivity
  - Shields the team from external interferences
- The Scrum Master is not
  - A team manager
  - A project manager
- Does not have authority over the team
- Typical behaviour: "*I observe [thing], what should we do?*"
  - Socratic questioning

Scrum Artefacts



Product Backlog



Sprint Backlog

DONE!

Definition of Done

Increment VS Release

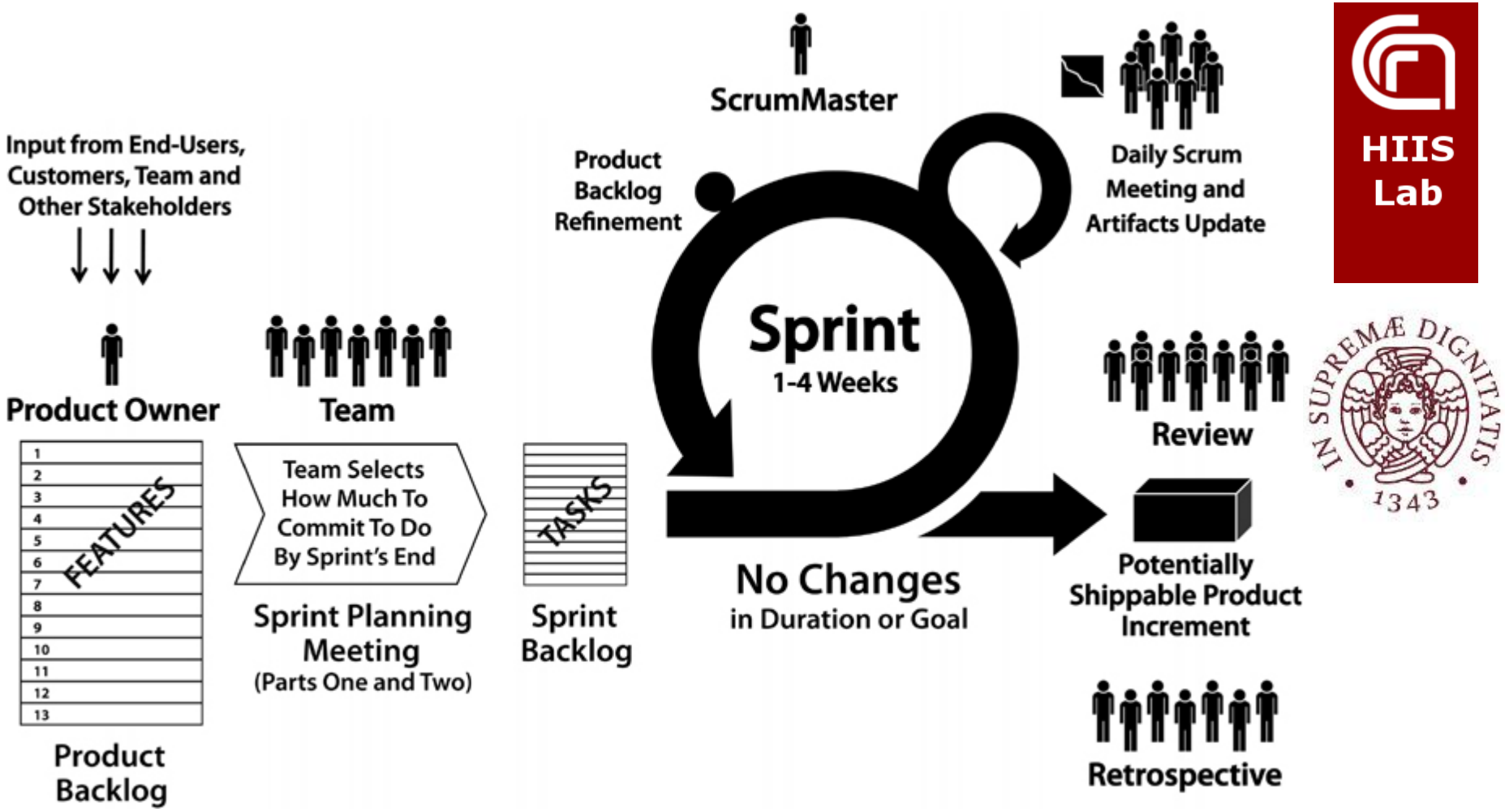
PSPI

- Potentially Shippable Product Increment
- Delivered at the end of each Sprint
- Allows inspect & adapt approach

MVP

- Minimum Viable Product
- Delivered at each product release
- The Product Owner decides its features

Process overview



Product Backlog (1)

- Owned by the Product Owner
- A list of items that could be done by teams:
  - New customer features
    Enable all users to place book in shopping chart
  - Engineering improvement goals
    Rework the transaction processing module to make it scalable
  - Exploratory or research work
    Investigate solutions for speeding up credit card validation
  - Known defects
    Diagnose and fix the order processing script error
- Priority assigned by the Product Owner
- Unique list for all teams
- Continuous evolution for change reaction

Product Backlog (2)

					New Estimates of Effort Remaining at end of Sprint...				
Item	Details (wiki URL)	Priority	Estimate of Value	Initial Estimate of Effort	1	2	3	4	5
As a buyer, I want to place a book in a shopping cart (see UI sketches on wiki page)	...	1	7	5					
As a buyer, I want to remove a book in a shopping cart	...	2	6	2					
Improve transaction processing performance (see target performance metrics on wiki)	...	3	6	13					
Investigate solutions for speeding up credit card validation (see target performance metrics on wiki)	...	4	6	20					
Upgrade all servers to Apache 2.2.3	...	5	5	13					
Diagnose and fix the order processing script errors (bugzilla ID 14823)	...	6	2	3					
As a shopper, I want to create and save a wish list	...	7	7	40					
As a shopper, I want to to add or delete items on my wish list	...	8	4	20					

- Level of detail for the specification
  - Up to the Product Owner and the Team
  - Variable for different items on the list

Definition of Done (1)

- Shared understanding of what “Done” means
- Used for
  - Assessing if a Product Backlog item is done or not
  - Estimation of items for Team commitment
- Precise list of things that can be checked
- It is possible to evolve the definition through the project
- The PSPI should adhere to the this definition

Definition of Done (2)

1. All Acceptance Criteria of the User Story are met
2. Code meets general Coding Standard (e.g. as defined in Checkstyle)
3. Functional tests are performed by team members other than those working on the implementation of that feature
4. Code is either reviewed or produced with a pair-programming method
5. The code is covered by a minimum of 70% Unit Tests and all tests are Green
6. Automated acceptance tests (Selenium) are prepared for the feature and are Green
7. Integration tests of the affected areas are conducted and passed

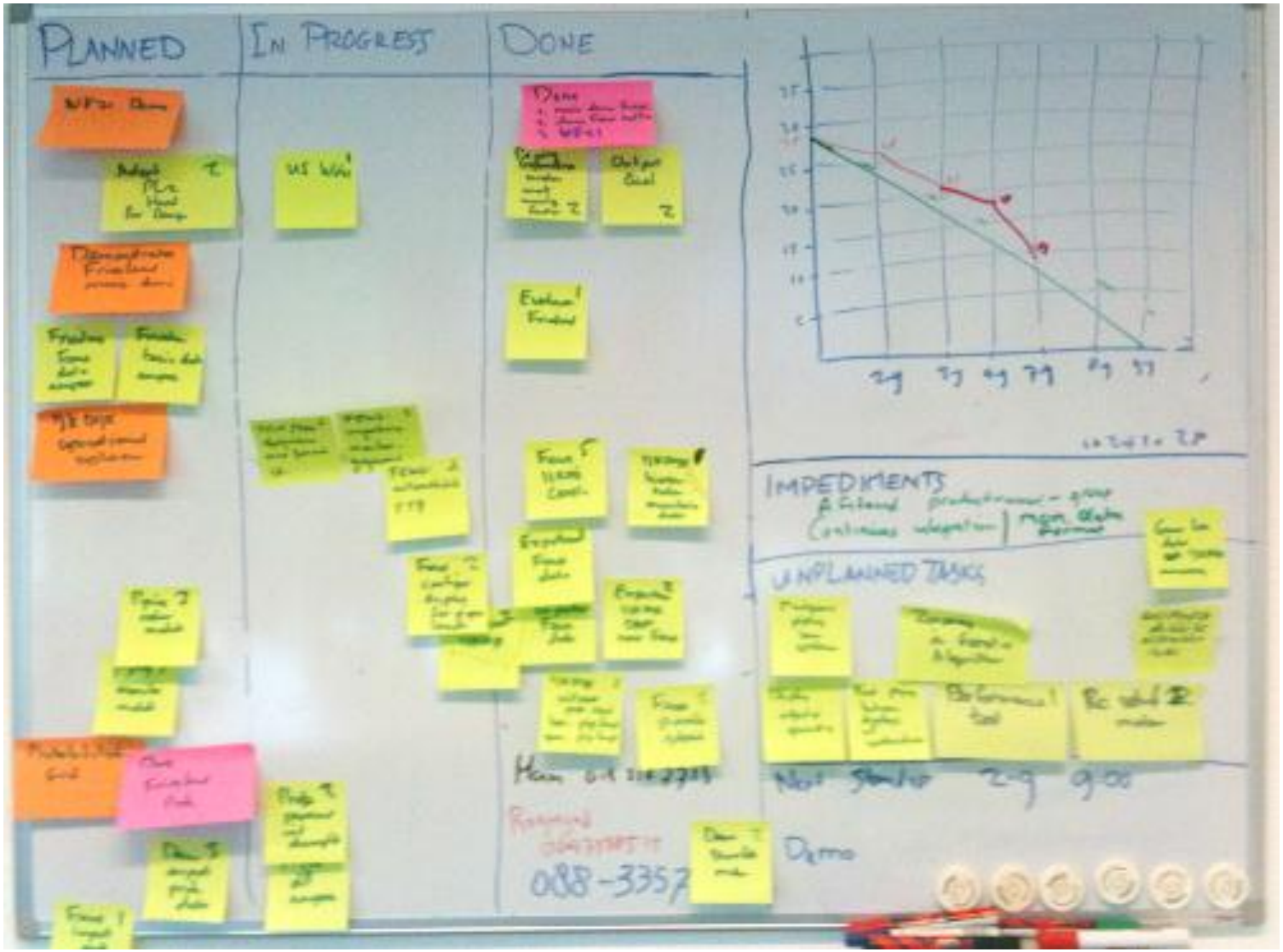
Sprint Planning Part One

- Participants:
  - Scrum Master
  - Product Owner
  - Team
- Decide **What** to do
- Product Owner presents the Goal
- Product Owner offers a **wish list** of items for the Sprint
- Team selects tentative items from the wish list
- Review / Confirm the definition of “done”
- Timeboxed
  - e.g. 2 hours for a 2-week Sprint

Sprint Planning Part 2

- Participants:
  - Scrum Master
  - Team
- Decide **How** to do it
- Pick the items from the wish list
- Split the items into tasks
  - 1-8 person hours maximum
  - Subtasks if higher
- Create the Sprint Backlog
  - Effort evaluation
  - Determine the available time for the sprint
- Communicate the forecast to Product Owner

Sprint Backlog



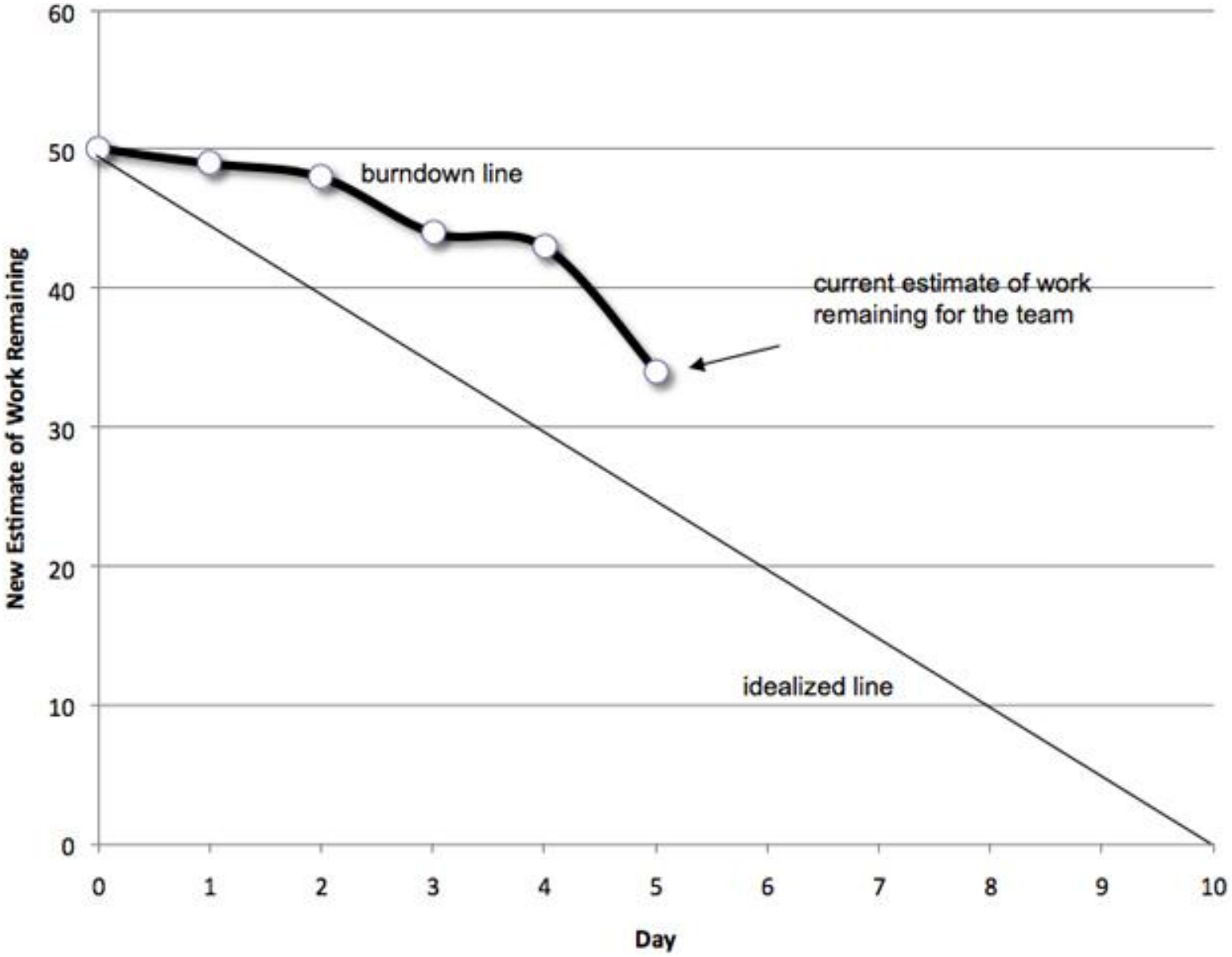
Working with the Sprint Backlog

- Any Team member can update
- Member volunteer for tasks
- All members work on a single item
- Try to keep low the WIP
- Try to take decision as late as possible
- Update daily the estimation of work remaining
- Room for stop & fix
  - Bugs
  - Improvement experiment
- New tasks can be discovered during the sprint

Daily Scrum

- Very quick daily report
  - Inspect and adapt
  - 15 minutes timebox
  - Everyone remain standing
- Everyone reports in brief
  - What was done
  - Plan for finishing tasks for next meeting
  - Any blocks or impediments
- Revise the tasks
  - Move tasks between columns
  - Add new tasks (if necessary)
  - Update the effort estimation
- Longer discussions postponed to follow-up meetings
- Only for the team!

Sprint Burndown Chart



Product Backlog Refinement

- Dedicated meeting near the Sprint end
  - Team
  - Product Owner
  - Scrum Master
- The Team refines the product backlog
  - Detailed requirement analysis
  - Splitting larger items into smaller ones
- Eases the Planning of the next Sprint

Questions?

Thank you!