Service Science
UniMiB
F9101Q022

The Sprint Approach

How to solve Problems in one Week

mirko.cesarini@unimib.it

Pivot or Persevere (Recap)

- Create an **MVP** and select **metrics**
- Repeat several times
  - **Tune** the **engine** from the baseline toward the ideal
  - **Pivot** or **Persevere** i.e.,
- Problems
  - Endless pivot loops
  - Exacerbated if the start-up runway is long

Introducing the Sprint Approach

- Organizations should perform **decision making** about critical activities in **short time-frames** e.g.,
  - Designing the next MVP
  - Design a split test experiment
  - Collect further information about whether to pivot or not
  - …
- Let's introduce the "**Sprint approach**"
  - Motto: "**Solve big Problems** and **test Ideas** in **5 days** (i.e. a working week)"
  - Developed at Google
    - Used in Google … for everything
    - And in a lot of other (innovative) companies

Sprint Introduction

- **Frame activities** in a very **tight time-frame** (e.g., problem analysis, solution proposal, decision making, implementation and tests)
  - 5 working days
  - 6 hours a day
- Gather relevant people and promote **idea sharing** and collaboration
  - Decision Makers
  - Business experts
  - Marketing experts
  - Technicians
- Rationale
  - To perform **decision making** in a **limited time frame**
  - Time limits are helpful
    - To **focus** on the relevant aspects and …
    - … to **discard** non relevant ones
    - People are forced to do as best as they can within the time limit
  - Experts **people time is expensive**. Time limits help staying **focused**

Sprint Week (Goals & Output)

- Day 1 (Monday)
  - Goal: **identify** the **problems** and the sprint **goal**
  - Output
    - A **Conceptual Map** summarizing
      - Key Actors/Customers
      - How customers interact with the proposed product/service/…
      - Project goal(s)
    - List of **assumptions** and **challenges**
- Day 2 (Tuesday)
  - Goal
    - **Propose solution(s)**
    - Start **customer recruiting** process (for Friday test)
  - Output: solution **sketches**
- Day 3 (Wednesday)
  - Goal
    - Explore solutions and
    - **Decide** which **solution(s) to implement**
  - Output: **solution Storyboard** (helps clarifying implementation details)
- Day 4 (Thursday)
  - Goal: **create the prototype**
  - Output
    - Prototype
    - Customer **Interview Script** (for Friday)
- Day 5 (Friday)
  - Goal: **evaluate** the prototype with real customers
  - Output: test **results and evaluation**

Sprint Key Concepts

- Gather **all** the relevant **experts together**
  - People with different backgrounds can contribute in “unveiling” the different solution facets
  - If the expert aren’t available for the whole week, they should be present the **first day** or the **first two days**
- **Time limit for each activity** (e.g., ideas proposal in half day, ideas selection in half day, prototyping in one day, …)
  - People will focus on the most important steps/parts/elements
  - Time constrains will force the team to discard the activities less important elements

Sprint Key Concepts (2)

- **Establishing** a **common vision** about the problem among the participants
  - **Conceptual map** to let people share a common view
  - **Identify** the **border** i.e., which problem(s) to solve and which not
- Promote **idea sharing**, avoid **frictions**/conflicts
  - Proposal anonymously written on **post-its** (to avoid personal frictions)
  - Post-its pinpointed to a **blackboard**
  - People **votes** and the winner solution is selected, unless the CEO (or equivalent) go for a different direction

Sprint Key Concepts (3)

- **Focused work**
  - 6 hours a day (max)
  - Suggested 10:00-13:00 14:00-17:00
    - Start at 10:00 so people can manage in advance daily urgent issues (and the later meeting won't be interrupted)
- Keep people focused and energized (even trivial stuffs matter)
  - Breaks every 90 minutes
  - Don't skip lunch
  - Provide food and drinks

Questions?

- Why
  - spending one day to understand the problem? And
  - another day for solution proposal?
- Why building the prototype in one day only?
- Answers
  - Previous-Day-Work speeds-up the development
    - The team has a shared overview of the prototype
    - Requirement specification is paramount (i.e., what to implement and what not to)
  - Similar rationale behind MVPs: resource (time) limitations will prevent loosing time on non-important parts

5th Day Evaluation

- **Real users** are invited for prototype **evaluation**
  - User-selection process started the 2nd day
    - User selection process through **questionnaire**.
      - Sent to a large set of people
        - Newspaper **ads**
        - Craiglist, …
      - **5 people** are selected. Empirical results show that more than 5 customers don't bring significative improvements
        - Selected those whose answers best **match** the **desired profile**
        - They are invited to **participate** the 5th day (Friday) **test**
        - Selected people are **rewarded** (e.g., 100$ gift card)
- The team observes the users interacting with the prototype
- **Results** are **collected** and then **discussed** by the team

Open Discussion

- Similarities, differences, shared concepts between
  - Innovation Accounting (Lean start-up)
  - Sprint approach (How to solve … in 5 days)

Other Development Methodologies

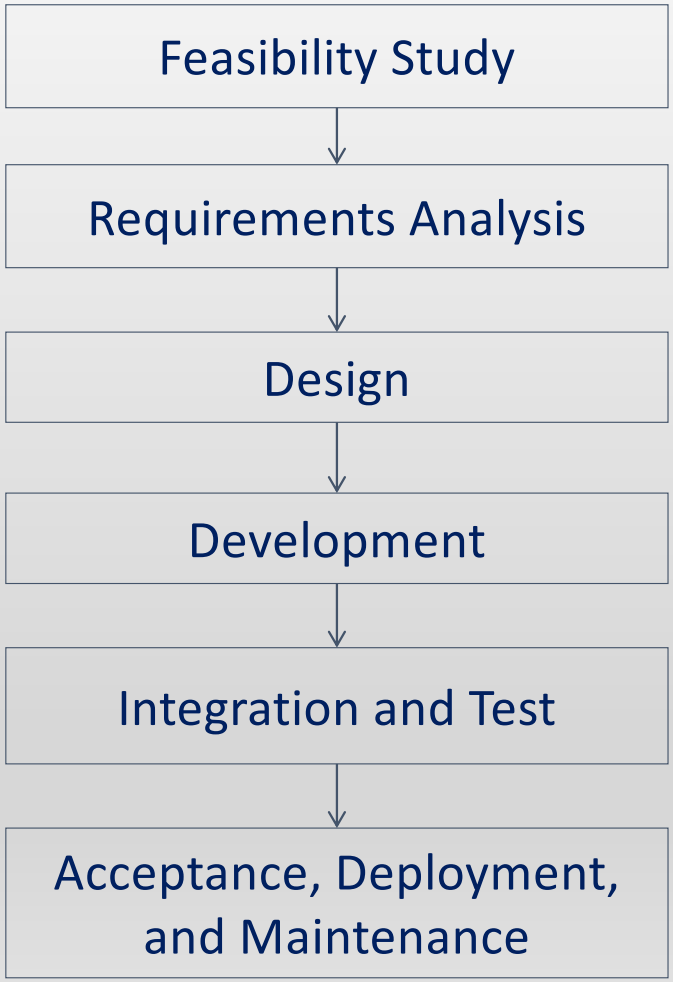
Mostly Borrowed by the Software Development Domain

Roadmap

- Focus on Software Development
  - Requirement specification is a problem of knowledge acquisition
  - Requirement specification has been investigated for long time
- Waterfall Model (one of the first theoretical model for software development)
- Agile Methodologies (created to address the problems of the waterfall model)

Waterfall Model

- Scenario: Software Development
- **Waterfall model**
  - The secret desire of every Project Manager
  - After a phase **completion**, the **next one** is executed
  - **Backtracking** (going backward) is considered a **failure**
- Prerequisite: the required **knowledge** is (very well) known
- Pros: very **efficient** execution of activities
- Cons:
  - User **feed-backs** only **at the end** of the project (months or years)
  - If the user requirements were wrongly collected, the whole project is a failure (**months/years of work wasted**)



Agile Software Development

- Agile is a set of **practice**s, values, and **principles** for **software development** (they applies also for product and service development)
- **Requirements** and solutions **evolve** through the **collaborative effort** of self-organizing and cross-functional **teams** and their **customer**(s)/end **user**(s)
- Core ideas in Agile Development
  - **Adaptive**. The teams and the process should be flexible in the presence of "rapid-fire change"
  - **Iterative** and **incremental**. Agile Development produces working products in **stages** – a growing set of "completed and working software"
  - **People-oriented**. The team organization and processes will support people, who are the most important ingredient to project success
- Agile is an umbrella term for several approaches
- For the interested reader: Manifesto for Agile Software Development
  http://agilemanifesto.org/

Example of Agile Methodologies

- Scrum

We will quickly introduce only this

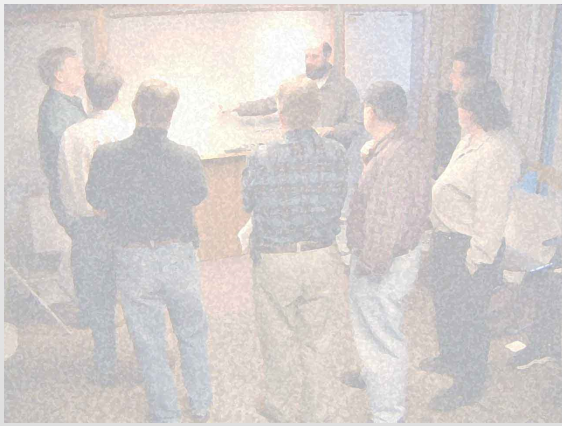
- Extreme Programming
- Lean Programming
- …

The Scrum Framework

- **Scrum** is an **iterative** and **incremental** framework for managing product development
- Key Principle
  - **Customers** will **change their minds** about what they want or need (requirements volatility)
  - **Unpredictable challenges** (no planned approach is suited)
- Solution:
  - **Time limited** development **activities** (called **Sprint** or Iteration)
  - Each iteration **ends** with a tangible result (a **demo** has to be given)
  - **Feed-backs** are collected
  - The iterations will be repeated until the end of the project

Name inspiration for several methodologies

A Software Scrum



A Rugby Scrum

The Sprint

- A **sprint** (or iteration) is the basic unit of development in Scrum
  - Contains a list of **tasks** and the **expected output**
  - **Time limited** effort (between 1 week and 1 month, frequently **2 weeks**)
  - The sprint result should be a
    - **working product**
    - potentially ready for selling
- Each sprint starts with a **Sprint planning event**
  - to define the sprint **backlog** (activities/work to be done)
  - Let each team member have a **shared idea** of what they will be working on
- Each Sprint ends with
  - A **sprint review** (product **Demo** to show results to stakeholders)
  - A **Sprint retrospective** (identify lessons and improvements for the next sprints)
- Deadlines matter
  - "Don't say – we can finish everything in 2 more days. Just deliver and run the next iteration planning meeting."
  - The team learns to make good short-term estimates – so **over time**, most of the iterations **will deliver** as expected

Product and Sprint Backlog

- **Product Backlog**
  - A list of all the features/products to be done (over several sprints)
  - Feature/products are described using **narrative stories**
  - It is OK to add things to the Product Backlog any time
  - At the **sprint planning event**, decision making about which elements to **load** from the Product Backlog to the Sprint Backlog (i.e., what to do in the sprint)
- **Sprint Backlog**
  - The **list of work** the Team is addressing during the **current Sprint**
  - It is a subset of the Product Backlog
  - Each activity/product/… when "in process" gets **some more details**, including
    - **Estimated effort** (in hours)
    - Primary **person responsible** for the activity

Backlog Item	Priority	(Hours)	Resp.
Subfeature 3	1	5	Mr. A
Subfeature 2	2	8	Mr. B
Subfeature 1	3	13	Mr. C
Subfeature 5	4	1	Mr. B
Subfeature 4	5	2	Mr. A

- Management should not add new requests to the Sprint
  - Any **new items** should be added to the **Product Backlog** instead
  - If new work items are **important** enough, they will get done in the **next sprint**

Scrum Actors

- **Product Owner**.
  - Own and prioritizes the **Product Backlog** (i.e., the activities to be done)
  - Has the power of stopping a Sprint
- **Scrum Master**.
  - More a **facilitator** than a traditional Project Manager
  - Facilitates the Scrum process and moderates the meetings
  - Supports the Team
  - Manages resources (e.g., renting computational power)
  - Communicates to Product Owner
- **Team**. Produces Increments of Shippable Product Functionality

Scrum Meetings

- (1) **Sprint Planning Meeting**
  - The product owner describes the **highest priority features** to the team
  - Team and Prod. Own. decide
    - which items **move to** the **sprint backlog**
    - Discussion about what the Prod. Own. Like and what can be **completed** within the sprint-end
- (2) **Daily Scrum Meeting**
  - Duration is 15 minutes (no longer)
    - **People stand-up** (to prevent never-ending meetings)
    - If someone is late, meeting starts anyway
  - Everyone in the team is supposed to speak:
    - "This is what **I did yesterday**"
    - "Here is what I am **planning to do today**"
    - "These are the **obstacles** in my way"
  - **No problem solving** in the meeting, everything is taken offline later.
- Purpose of the Daily Scrum?
  - To make sure that **problems and obstacles** are **visible to the team**
  - Obstacles are valuable input for managers

Scrum Meetings (2)

- (3) **Sprint Review Meeting**. It is composed by 2 submeetings
  - **Product Demo** (led by the Product Owner)
  - **Sprint Retrospective** (led by the Scrum Master)
    - What worked?
    - What didn’t?
    - What adjustments can be done now?

Scrum Board

- Scrum Board is a rows-and-columns depictions of work-in-progress
  - Rows: **items of work**
  - Columns: work status labels
    - **In Process** column may have **constraints** e.g., no more than X activities (to prevent people overloading)
- Work items may be split into activities/subtasks, …
- Work items migrates from left to right on the board