Introduction to Software Engineering & Recent Trends in Industries

Anirban Mitra

When do we consider a Software Application successful?

- Development Completed
- It is Useful
- It is Usable
- It is Used

But it is true that Software development project have not always been successful

Reason of Failure ?

- Schedule slippage & Deliverables to user not identified
- Poor understanding of user requirement
- Poor understanding of cost & effort by both developers & user.
- Cost overruns
- Poor quality of Software
- Ad-hoc software development results in such problems
- No planning of development work

Why Software Engineering?

- All of the above reasons are the attributes of "**Software Crisis**"
- Late 1960 H/W prices were falling but S/W prices rising. Many Software Project failed. They did not satisfy the requirements.

  The term "Software Engineering" First introduced at a conference in late 1960 to discuss the "Software Crisis"

"You can't always predict ,but you can always prepare"

Difference from other Engineering :

- Large projects common & successfully done
  - Building bridges, dams
  - Power Plants
  - Aircraft, missiles

Software Project Vs Hardware Project

- Software is different from the other product:
  - Cost of production concentrated in development
  - Maintenance consists of making corrections && enhancing or adding functions
  - Progress in development is difficult to measure:

    80% or 90% complete.

What is Software?

- Software is a collection of programs, procedures, rules & associative documentation to provide functions and performance of desire tasks.

What is Engineering

- Engineers try to discover problems and find / develop and apply appropriate theory, methods, tools to the solution of the problems. It is a discipline that integrates method ,tools & procedure for the development of computer software
- The systematic approach to the development , operation , maintenance,& retirement of software.

What is Software Engineering?

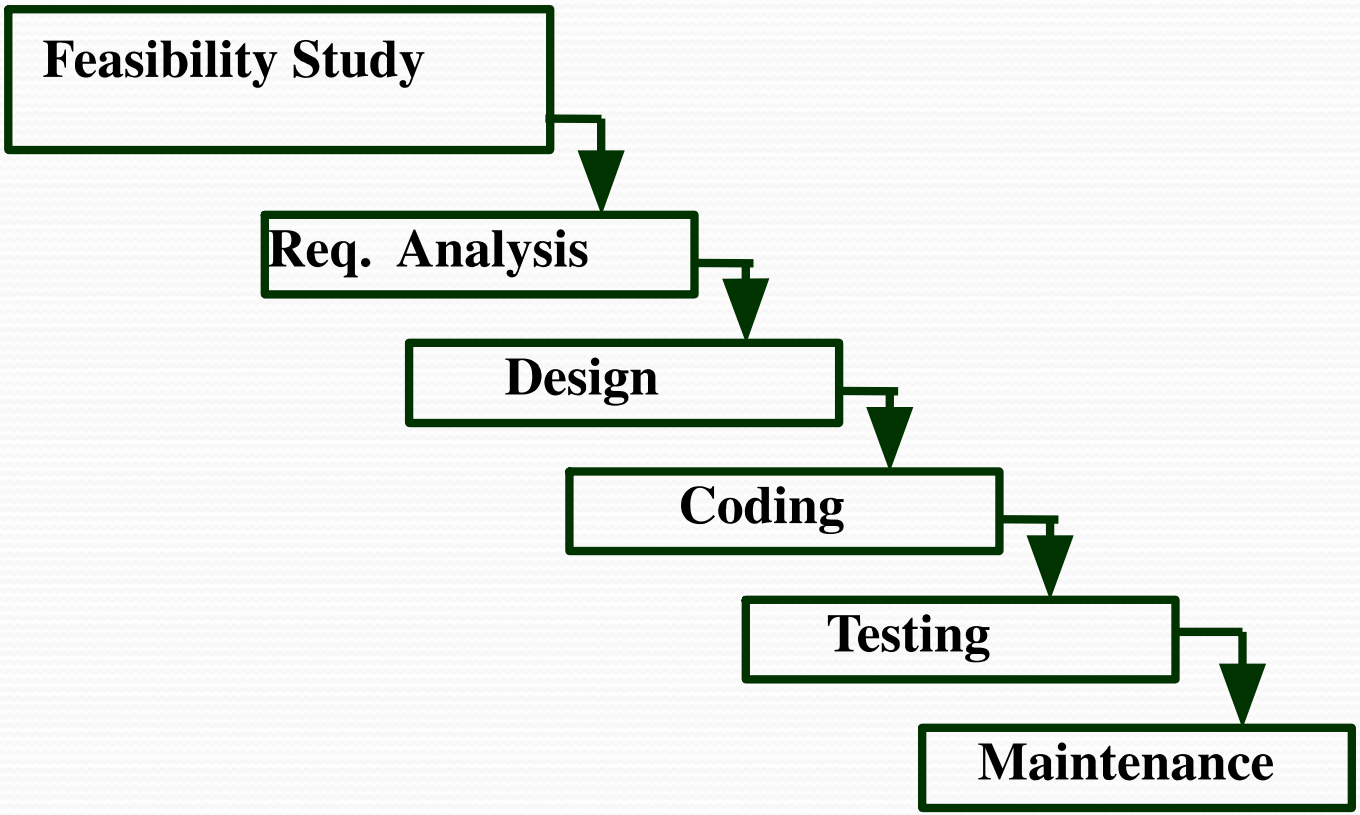
- "SE is concerned with software system developed by teams rather than individual programmers, uses engineering principles in the development of these system and is made up of both technical & non-technical aspects."

Software Process Model

Classical Waterfall Model

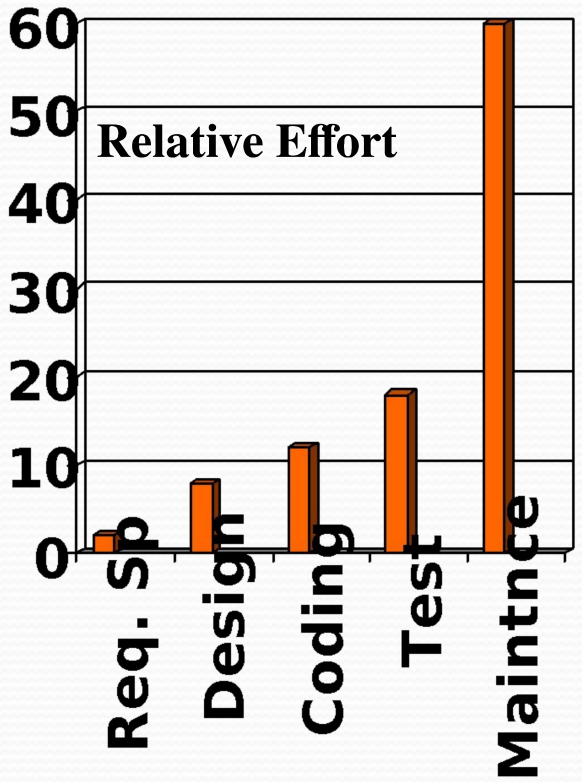
- Classical waterfall model divides life cycle into phases:
  - feasibility study,
  - requirements analysis and specification,
  - design,
  - coding and unit testing,
  - integration and system testing,
  - maintenance.

Classical Waterfall Model



Relative Effort for Phases

- Phases from requirement specification to testing
  - known as development phases.
- Among all life cycle phases
  - maintenance phase consumes maximum effort.
- Among development phases,
  - testing phase consumes the maximum effort.



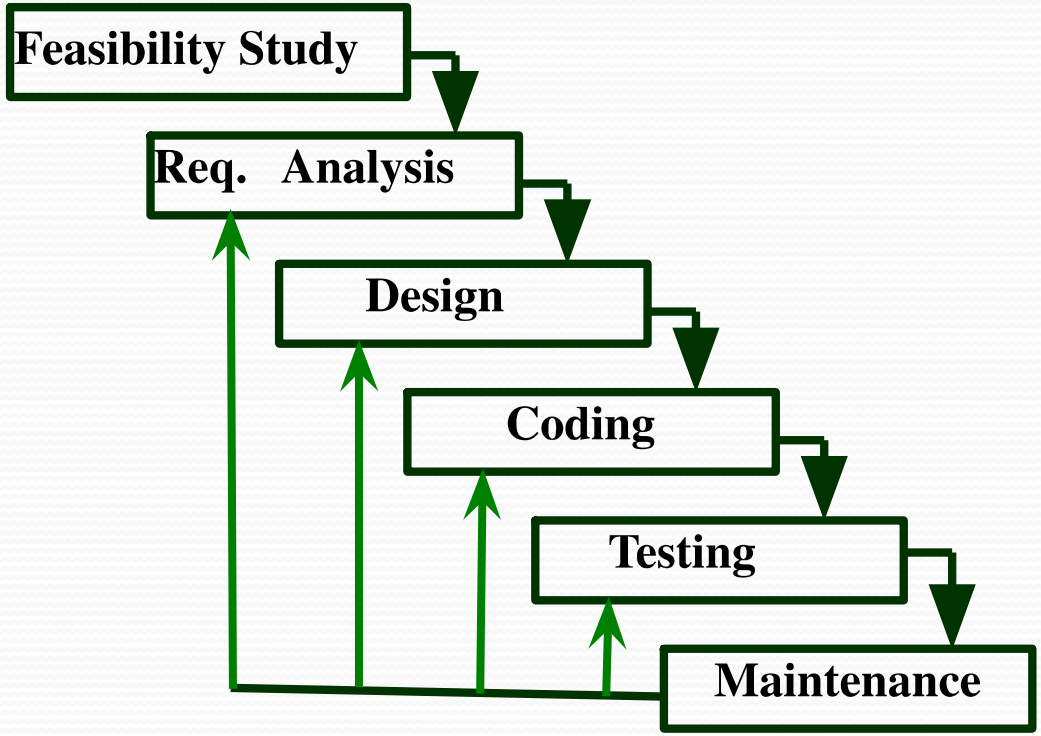
Shortcoming Of WM

- Requirements may not be clearly known, Especially for applications not having existing counterpart
  - Railway reservation: Manual system existed, so SRS can Be defined
  - On-line container management for railways
  - Knowledge management for central bank
- Requirement change with time during project life cycle itself…

Shortcoming Of WM…

- Requirement change with time during project life cycle itself…
  - User may find solution of little use
  - Better to develop in parts in smaller increments
  - This is common for packages, System software
- Consider documentation heavy: so much documentation may not be required for all types of project.

Iterative Waterfall Model (CONT.)



Recent trends in Industry.....

Rapid Adaptable

Cooperative AGILE Iterative

Quality-driven

Not a process, it's a **philosophy** or **set of values**

Agile Manifesto

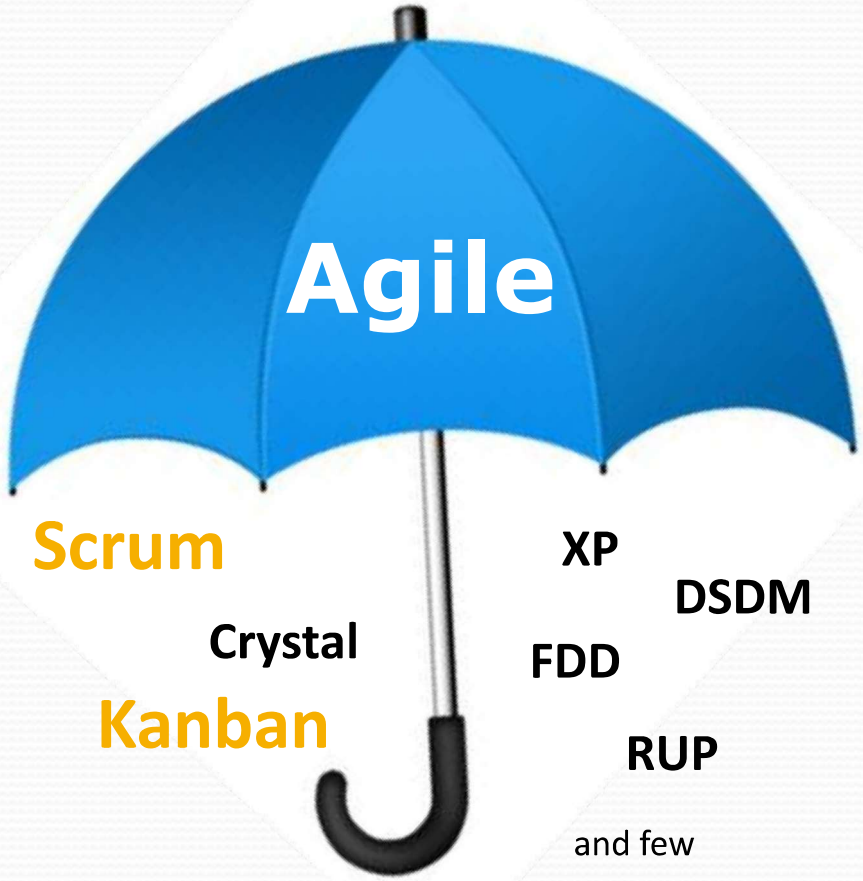
Individuals and interactions over **processes and tools**

Working software over **comprehensive documentation**

Customer collaboration over **contract negotiation**

Responding to change over **following a plan**

Agile Umbrella



More Prescriptive

more rules to follow

- RUP
  RUP has over 30 roles, over 20 activities, and over 70 artifacts
- XP
- Scrum
- Kanban
- Do Whatever!!

More Adaptive

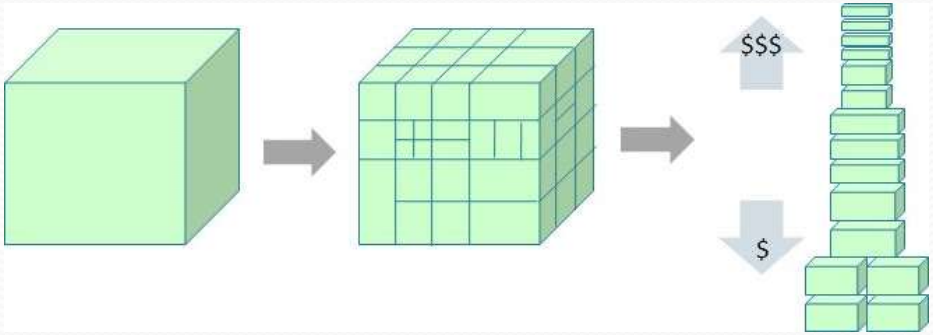
fewer rules to follow

Scrum

A **light-weight agile process tool**

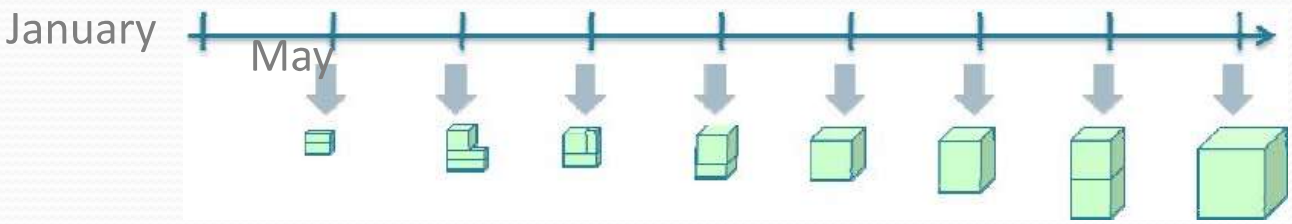
Split your organization into small, cross-functional, self-organizing teams.

Split your work into a list of small, concrete deliverables. Sort the list by priority and estimate the relative effort of each item.



Scrum (contd..)

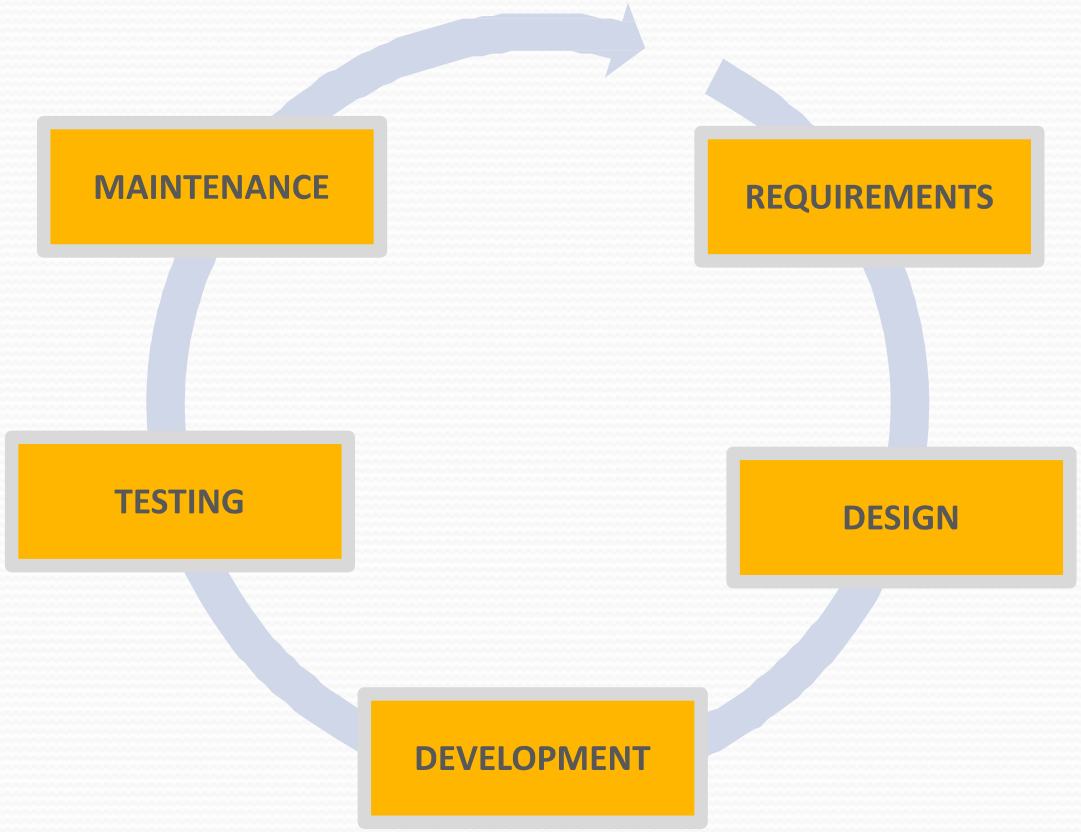
Split time into short fixed-length iterations/ sprints (usually 2 – 4 weeks), with potentially shippable code demonstrated after each iteration.



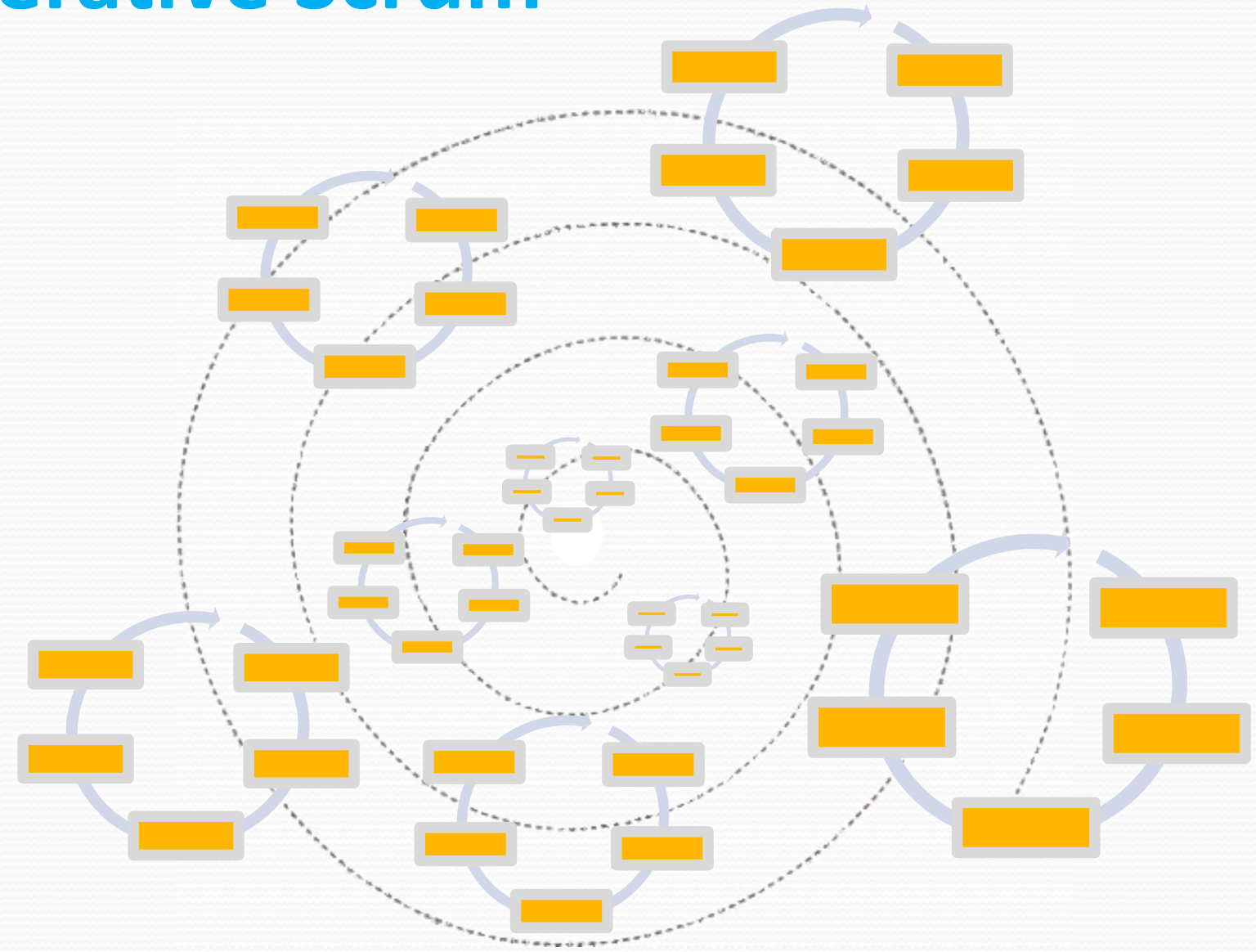
Optimize the release plan and update priorities in collaboration with the customer, based on insights gained by inspecting the release after each iteration.

Optimize the process by having a retrospective after each iteration.

Scrum vs. Waterfall



Iterative Scrum



Things we do in Scrum

a.k.a Scrum terminologies

The project/ product is described as a list of features: the **backlog**.

The features are described in terms of **user stories**.

The scrum team **estimates** the **work** associated with each story.

Features in the backlog are **ranked** in order of importance.

Result: a **ranked** and **weighted** list of product features, a **roadmap**.

Daily scrum meeting to discuss **What did you do y'day**? **What will you do today**? **Any obstacles**?

Scrum Artifacts

Sample Userstory

USERS SHOULD BE ABLE TO UPLOAD MULTIPLE PHOTOS AT ONCE

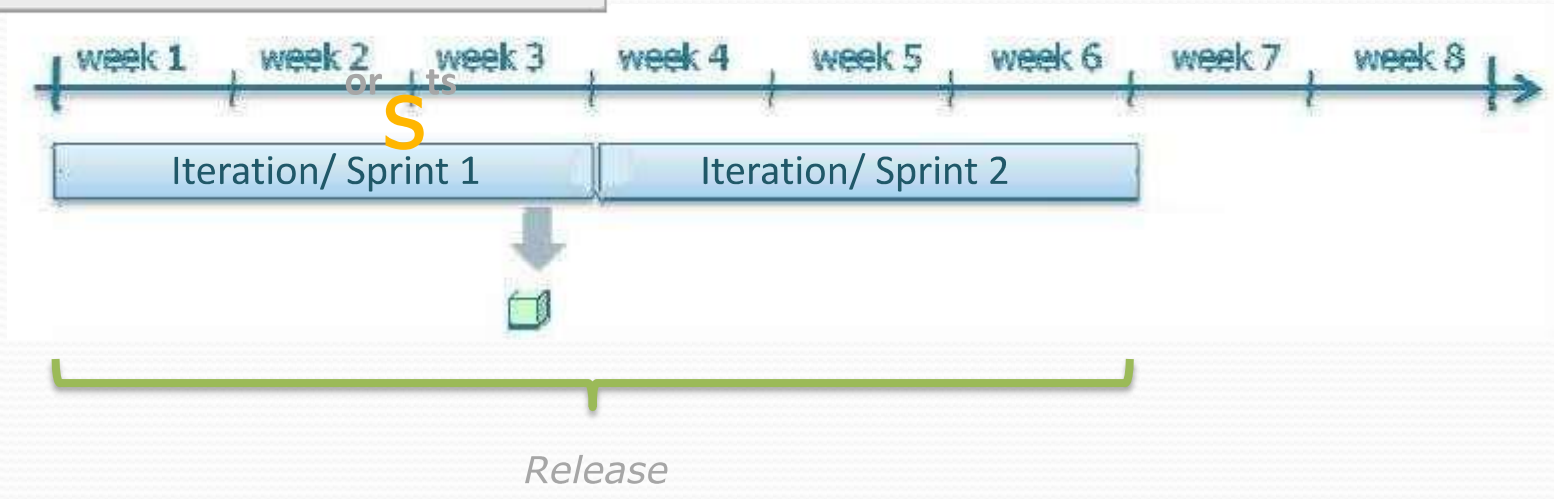
- Test with JPEG, PNG, GIF (supported)
- Test with an unsupported image format
- Test with Flash not present
- Test with more than 20 MB of total POST data

10h Effort

Efforts: 2hrs IA, 6hrs Development, 2hrs Testing

The total effort each iteration can accommodate **leads to** number of user story per iteration

Iterations View



One **release** may contains **number of iterations**

Scrum planning example

Iteration cycle of **3 weeks**

Working hours per day is **8**

Total hours of **work iteration can accommodate**

8hrs x 5days x 3weeks = **120hrs**

Product backlog of **20 stories**

Each story effort is **10 hrs**

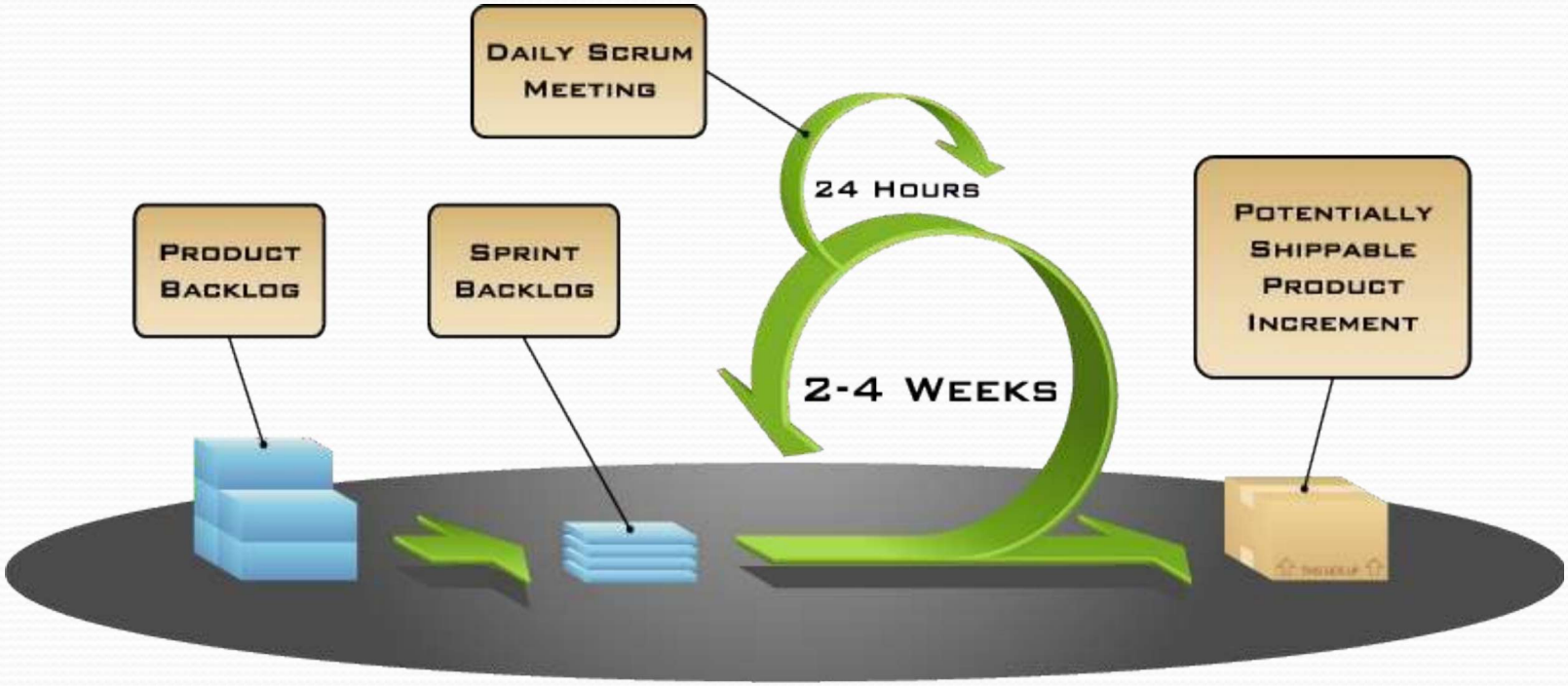
Iteration backlog or **number of stories per iteration**

12 user story

Scrum in a nutshell

So instead of a **large group** spending **a long time** building a **big thing**, we have a **small team** spending a **short time** building a **small thing**.

But integrating regularly to see the whole.



Thank You