ENSF 613

Software Requirement Analysis and Process Management

What is Software Development Processes?

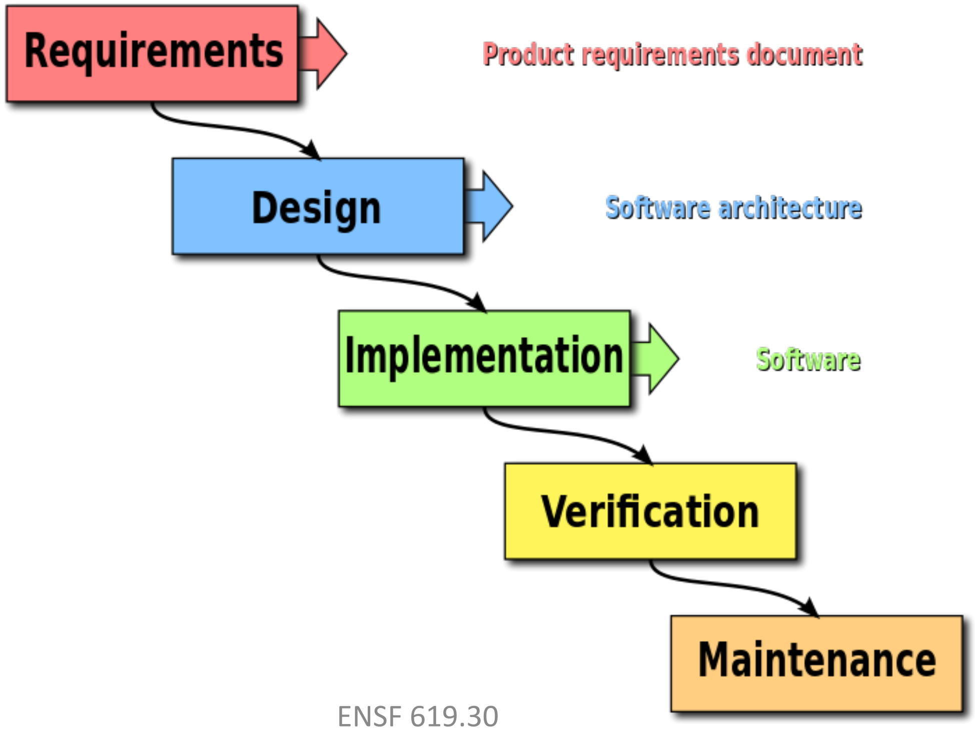
Software Development Process

- Different process models have been around since software engineering become a profession.
- Software Development Process or also called Software. Development Life Cycle (DLC) is a splitting of software. development activities in different phases.
- Some process models have sequential flow and some incremental and/or iterative.
- DLC models also provide a timeline for each phases of development to assist with project management.
  - One of the main concerns is: "what to do and how long it should take to complete the product release.
- Each model has a different focus, and each one might be useful for a different situation. Also a project select a combination of two or more models at different stages.

Waterfall Process Model

Waterfall Model

- Originated in the Manufacturing and Construction Industry
- In 1985, the United States Department of Defense captured this approach as their standards for working with software development contractors, which stated that "the contractor shall implement a software development cycle that includes the following six phases: Preliminary Design, Detailed Design, Coding and Unit Testing, and Integration, Testing" [Wikipedia]:



Application of Waterfall Process Model

- Questions to be asked when thinking of waterfall process model:
  - Are software requirements known prior to of its implementation and not subject to change in future?
    - Are requirements compatible with the stakeholders (key users/customers, developers, etc.) expectations?
  - Is systems architecture well defined and well-understood
  - Aren't requirements subject to high-risk, such as: cost, schedule, user interface, safety, user interface, internal and external organizational, legal or political impacts?
  - Is the full documentation of the system requirements, and the following phases one of the customers main concern?
  - Does the project have fixed budget, and customer needs a clear estimation of cost for each phase of the development?

Waterfall Model Pros and Cons

- Pros
  - Provides a structured approach, therefore is easier to understand.
  - Time spend early in the production cycle can reduce the cost.
  - Emphasize on documentation helps future developments.
  - Independency of phases.
  - Can be useful for cases that project has a limited budget and must be carried out by a bidding process.
  - Can be useful for cases that customer and designer mixers is not relevant.
- Cons
  - No output until end.
  - Designers may not be aware of future difficulties.
  - Hard to manage requirement changes.
  - Not suitable for complex projects.

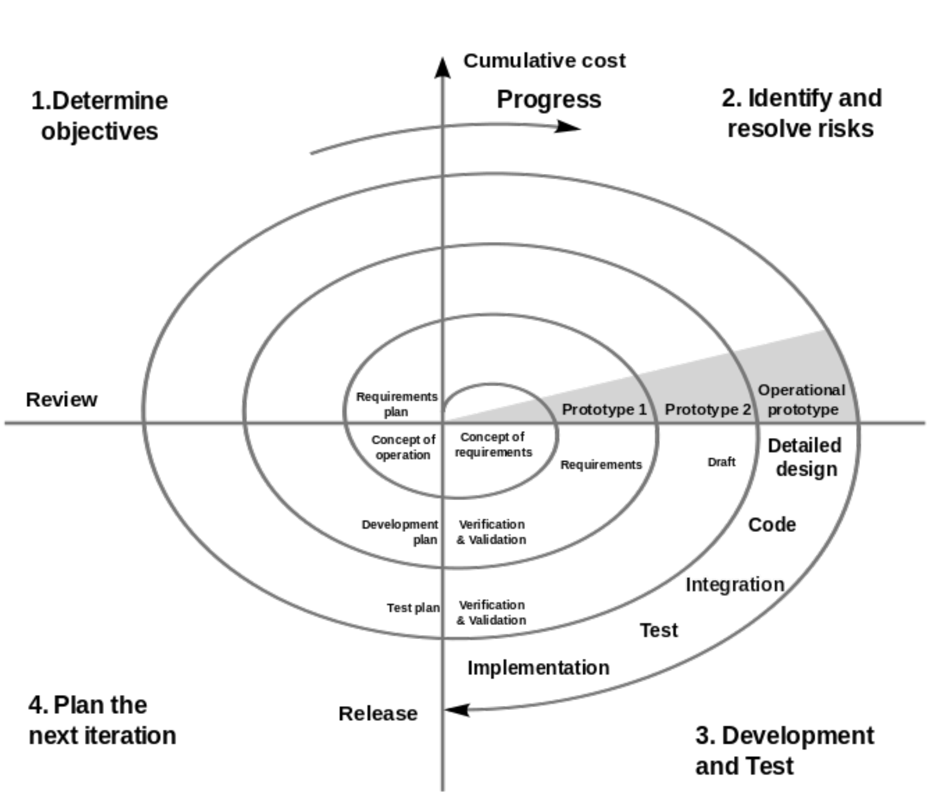
Spiral Process Model

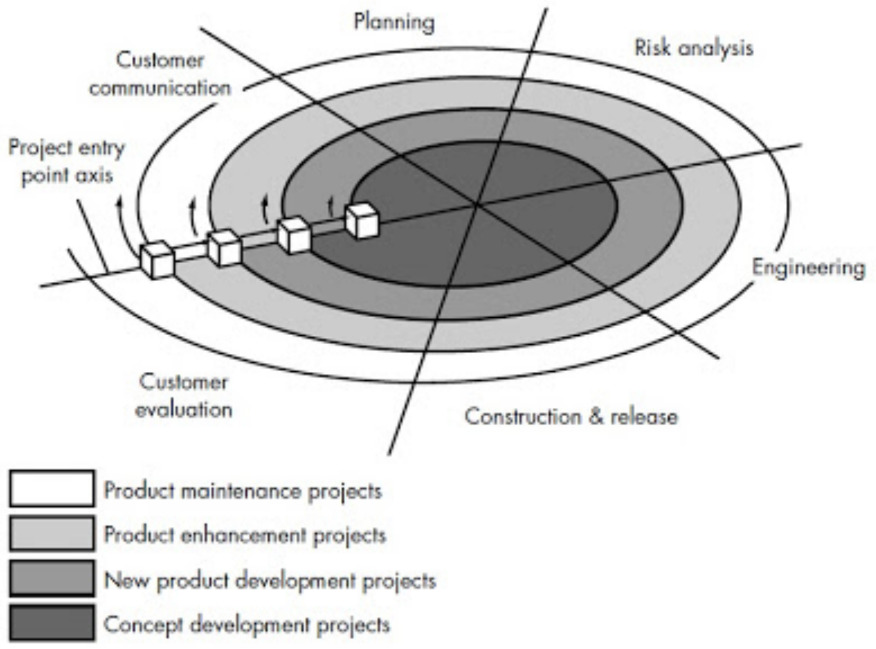
Spiral Development Process

- This model was first described by Barry Boehm in 1986.
- It is an iterative version of the waterfall approach, with particular focus on risk analysis.
  - One of the phases of the development process focuses on identifying risks and mitigating them.
- There are 4 distinct phases in the spiral model:
  - Identify objectives
  - Identify and resolve risks
  - Build and test
  - Review and plan for next iteration
- For each phase, development team must decide how much time and effort is enough. Spending additional time an effort can be costly and should be avoided.

Spiral Models

- There is a variety of presentations of spiral models:
  - Figure on the left shows Boehm's Spiral Model (1988)
  - Figure on the right is slightly different presentation by Pressman (2010).





⬜	Product maintenance projects
▨	Product enhancement projects
▨	New product development projects
⬛	Concept development projects

From: Software Engineering a Practitioner's Approach, Pressman (2010)

Typical uses of a Spiral Model −

- For medium to high-risk projects.
- When there is a budget constraint and risk evaluation is important.
- When customers are not sure of their requirements.
- When customers’ requirements are complex and need evaluation to get clarity.
- New product line which should be released in phases to get enough customer feedback.
- When significant changes are expected in the product during the development cycle.

Spiral Model - Pros

- Spiral is a better sequential model, because
  - Changing requirements can be accommodated.
  - Allows extensive use of prototypes.
  - Requirements can be captured more accurately.
  - Users see the system early.
  - Development can be divided into smaller parts and the risky parts can be developed earlier which helps in better risk management.

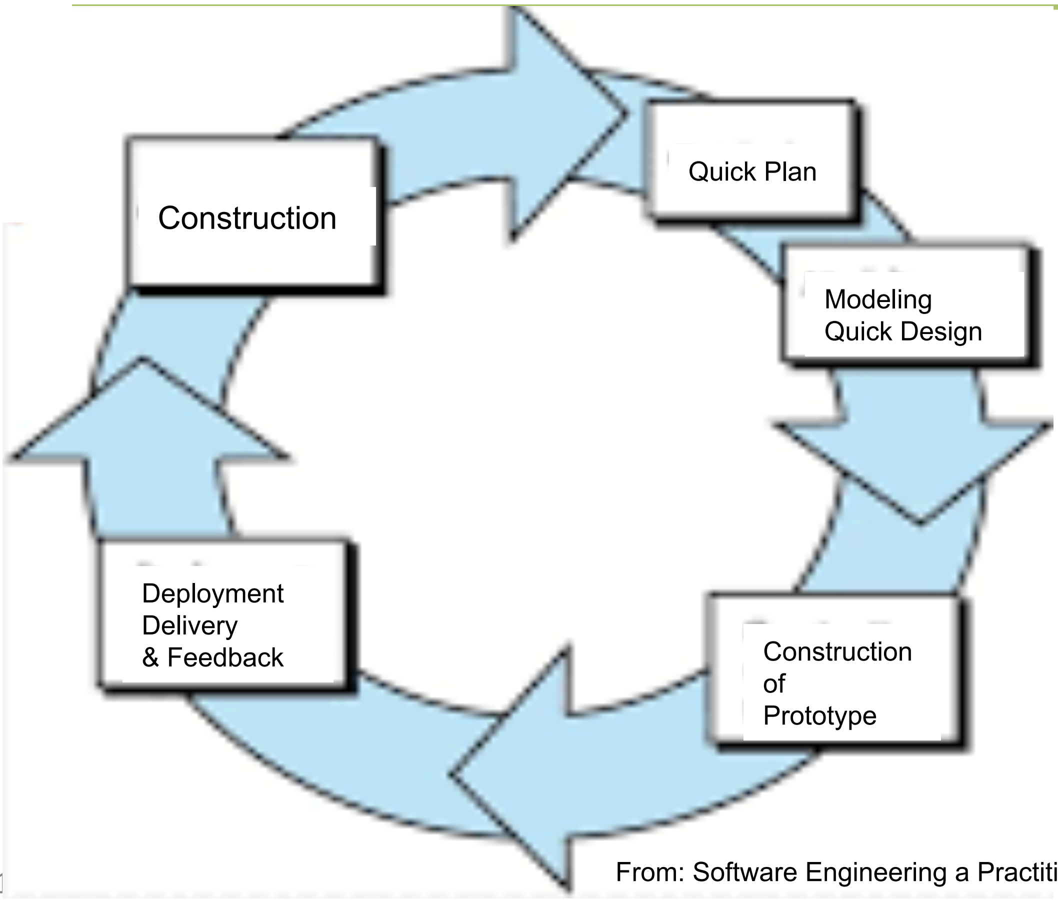
Spiral Model - Cons

- It takes a very strict management to complete such products and there is a risk of running the spiral in an indefinite loop. So, the discipline of change and the extent of taking change requests is very important to develop and deploy the product successfully.
- Summary of Cons
  - Process and its management is more complex.
  - End of the project may not be known early.
  - Not suitable for small or low risk projects and could be expensive for small projects.
  - Large number of intermediate stages requires excessive documentation.

Prototyping

Prototyping

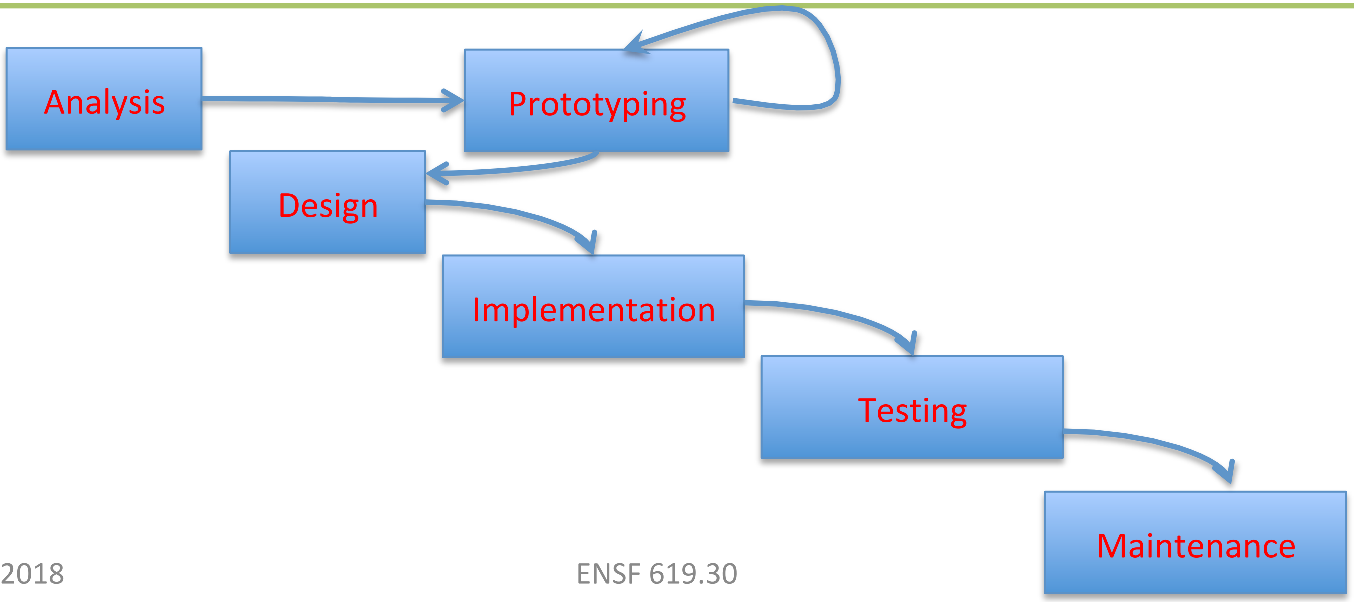
- The prototyping model is a development method in which a prototype is built, tested and then reworked as needed until an acceptable outcome is achieved from which the complete system or product can be developed.
- Starts with a quick plan, followed by a quick design, construction of the prototype, delivery and feedback, and finally construction..



From: Software Engineering a Practitioner's Approach, Pressman (2010)

Advantages

- Prototyping can improve the quality of requirements.
  - Ideally, prototypes serves as a mechanism for identifying software requirements
- Can help to reduce cost of requirement errors that will be detected earlier.
- It can help better understanding and better communication among developers and other stakeholder.
  - Although prototyping can be used as a stand-alone process model, any process model may adopt it for situations that requirements are fuzzy.

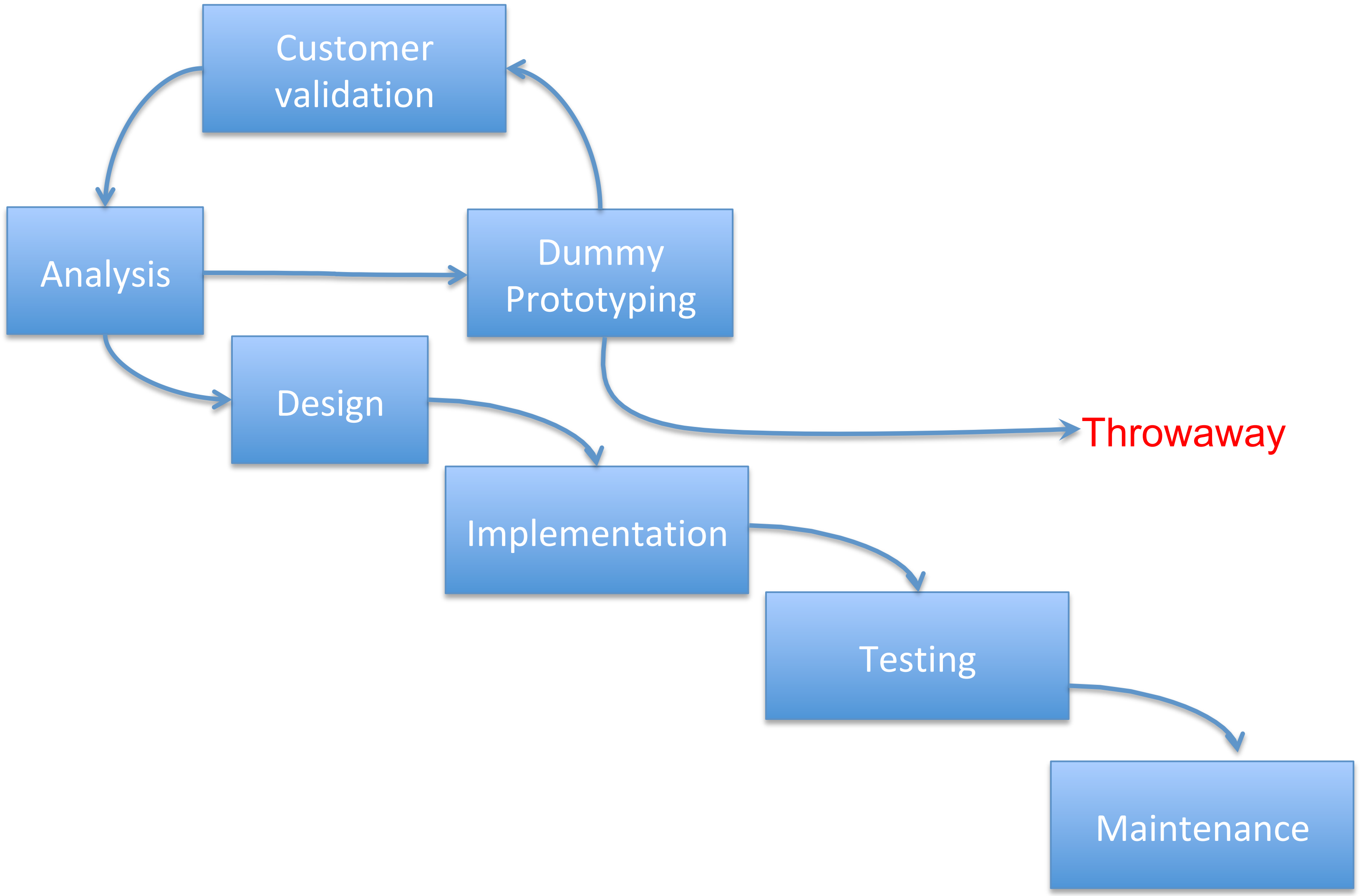


Prototyping Disadvantages

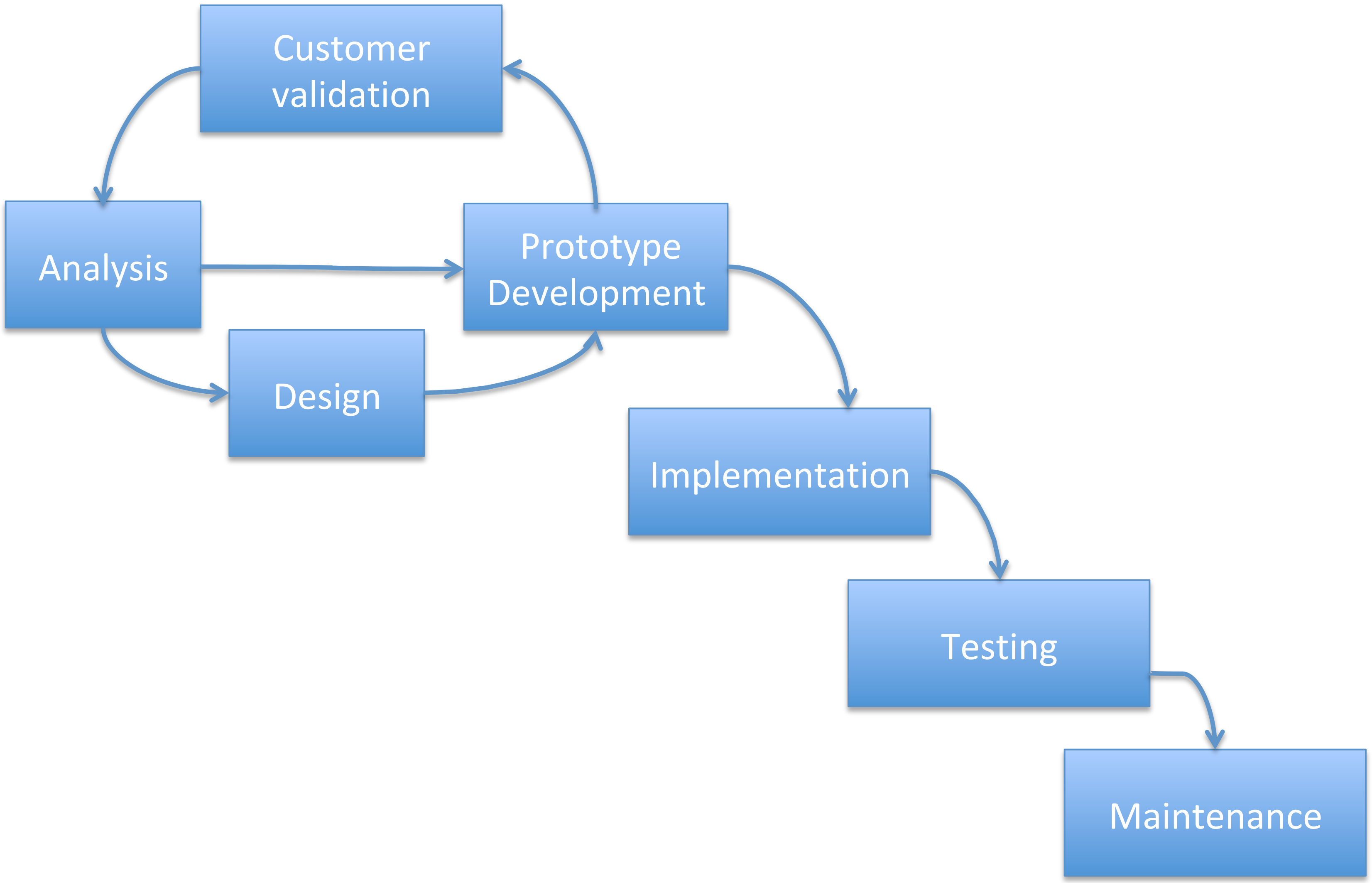
- Due to less emphasize on formal analysis and design, it may increases the complexity of the overall system
- Sometime involves exploratory (trial and error) methodology and therefore involves higher risk.
- Involves implementing and then repairing the way a system is built, so errors are an inherent part of the development process
- Costs of producing the prototypes.

Different Prototyping Approaches

Throwaway Prototyping Approach



Evolutionary Prototyping Approach



Test-Focused Approaches

Test-Driven Approaches

- It is important to note that Test driven development approaches are not only a testing technique. In fact they are the process of designing, developing and testing. But they put a lot of emphasize on testing.
- There are two test-driven process that we will discuss here:
  - V Model
  - TDD Model – Test Driven Development Model

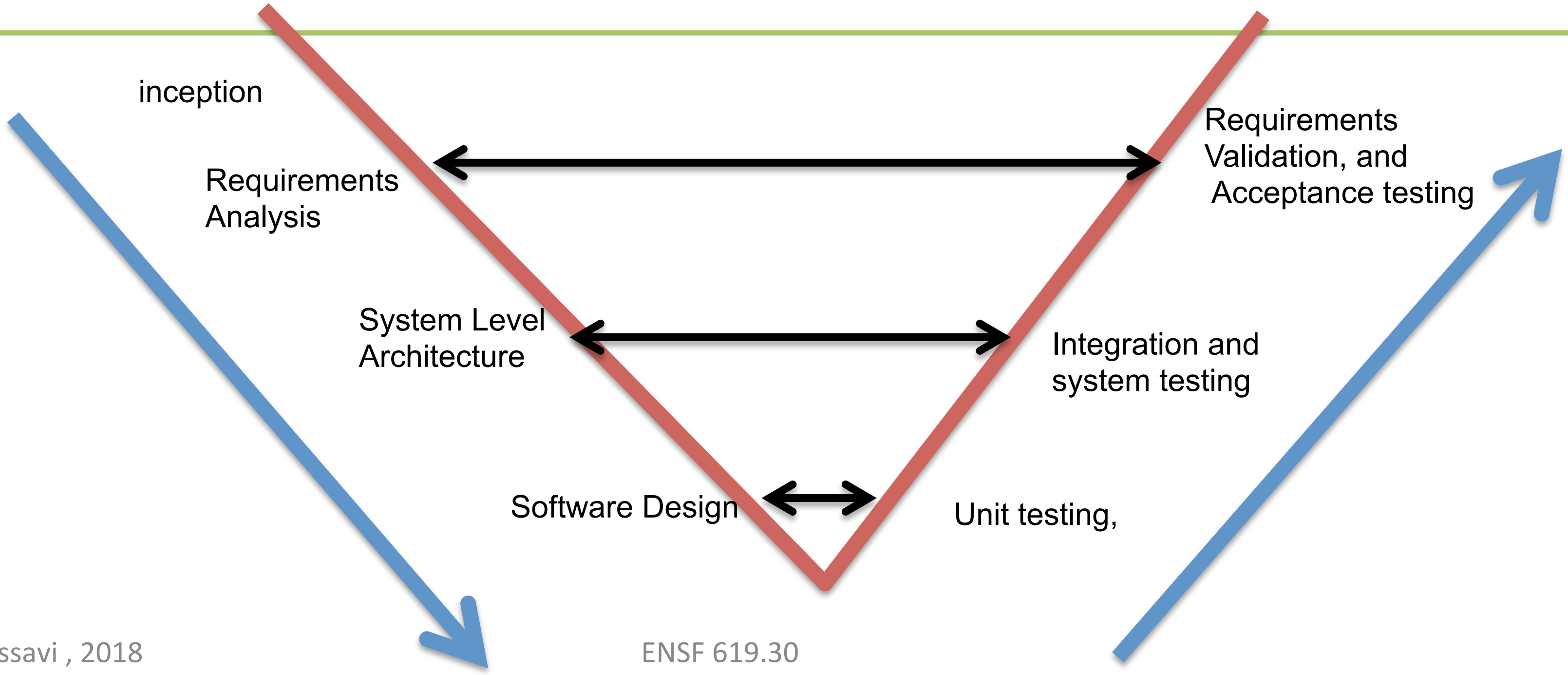
V Model

V Model

- The V-Model was first proposed by Paul Rook.
- It is named V-model because the process takes the shape of letter V, and focuses on two Vs: Validation and Verification.
- Each phase must be completed before the next phase commence. This model is also an extension to waterfall model.
- The overall step of V shape process modeling is similar to waterfall model but the key difference is that each stage is associated with some verification/validation and testing activities.

Test Focused Models: The V Model

- Major testing phases are:
  - Unit Testing: happens during the design phase.
    - Inclusion of unit test provides means for verification of smallest programming units
  - Integration Testing: happens during architecture design or high level system design.
    - This test is to confirm that smaller units of a product can coexist and communicate properly with each other.
  - Requirements Validation & Acceptance Testing : happens during the requirements analysis, and must be developed in collaboration with customers business team. Its target is to ensure customer requirements are met.


inception
Requirements Analysis
System Level Architecture
Software Design
Requirements Validation, and Acceptance testing
Integration and system testing
Unit testing,

Pros and Cons

- Pros:
  - V-model is easy and simple to use.
  - It improves the quality and reliability.
  - Testing activities are done well before coding which saves lot of time, so it has higher success rate as compared to waterfall model.
  - Defects are found in initial stages,
  - The downward flow of defects are prevented.
- Cons:
  - V-mode is very rigid and not flexible.
  - Lot of money and resources are required in this process.
  - Not suitable for large projects.
  - Early prototypes of the software are not produced as software is developed during implementation phase.
  - The requirement documents and test documents has to be updated if there is any change in the middle of the project.

TDD

TDD

- TDD is a software development process that relies on the repetition of a very short development cycle:
  - It is an approach to develop a software with combination of **test-first** development and refactoring the code until all test are passed.
- TDD can be consider as a way of defining software specification, or as method of writing clean code.
- Some developers adopt TDD as a methodology for agile requirements analysis and an agile design approach.

TDD

- Here are the major steps of TDD:
  1. Developer begins the process by analyzing the system requirements and extracting a feature or specific function from the requirements.
  2. Developer writes an automated test that will test the function.
  3. Developer writes a minimum code to run the test.
  4. Initially the test fails since the function is not developed.
  5. Then developer writes the function and runs automated test. If failed, he/she makes attempt until the test passes.
  6. Next, it is time for refactoring which is the key step in TDD. Refactoring means: Developer looks all possible changes to make the code more: reliable, efficient, precise, maintainable, etc.
  7. In the next step, developer repeats the same process for another feature/function.

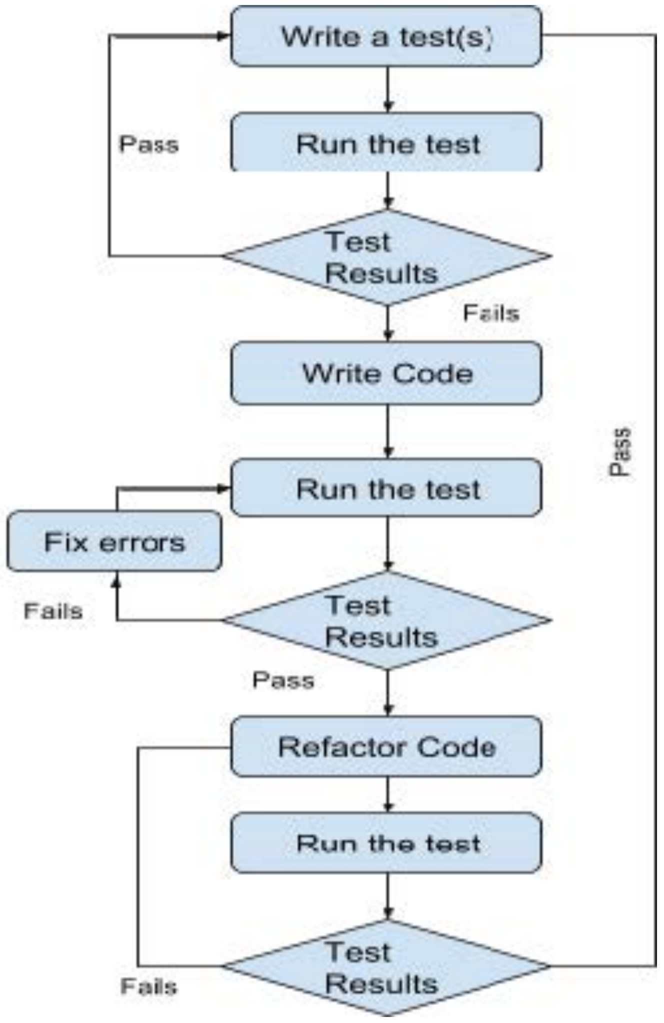


Figure from: Sumanth Yenduri, et al , 2005

Tools

- Some of the available tools for TDD include:
  - JUnit
  - HttpUnit
  - PHPUnit
  - PyUnit (Python)
  - Testoob (Python)
  - Cpputest
  - csUnit (.Net)

Pros and Cons

Pros:

- Forces developers to take small steps in building software.
- Although it requires some additional code writing for testing, it can reduce the overall construction time of the software.
- Results in cleaner code.
- Due to extensive unit testing most likely defects are caught in early stages of the development which results in saving money.

• Cons:

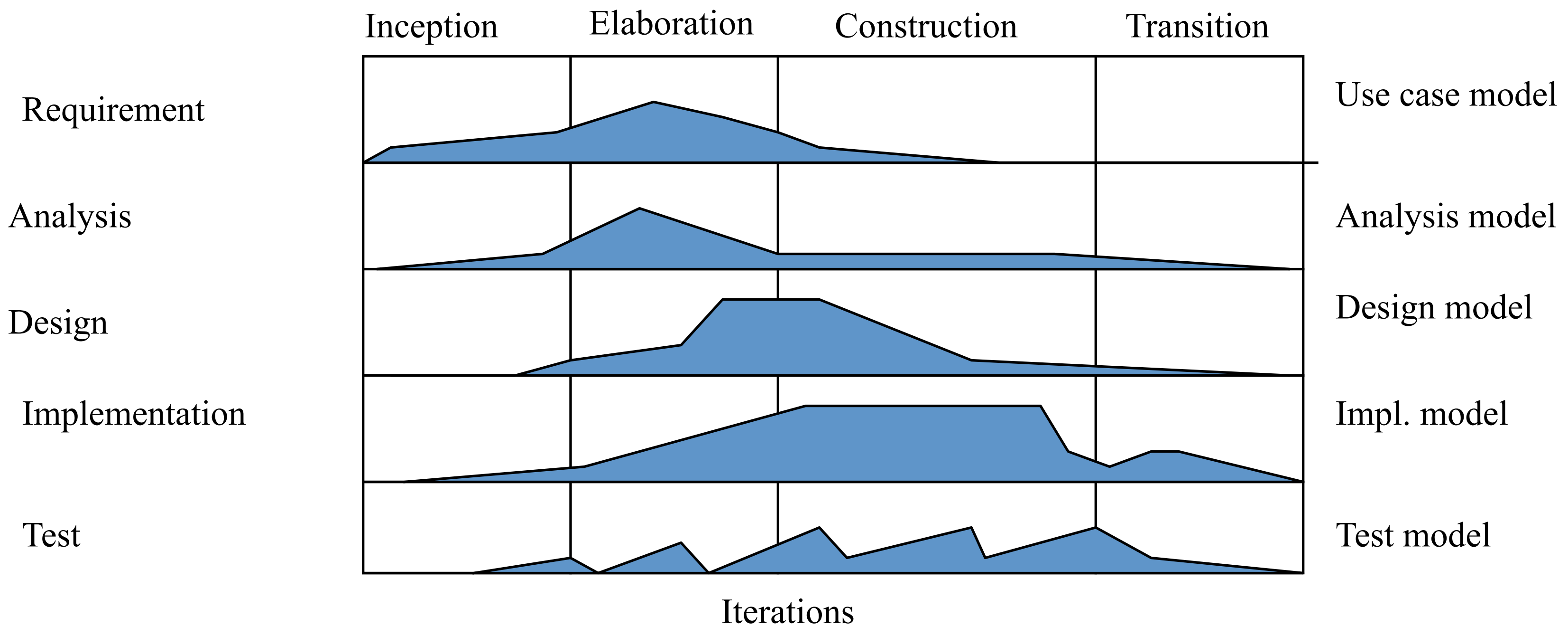
- Requires some time and effort to design test cases and write the unit tests.
- It may slow down the process at the beginning of the development.
- Requires all team member to understand and be able to apply TDD

Incremental and Iterative Approaches

Rational Unified Development Process (RUP)

The Rational Unified Process

- The Rational Unified Process is structured along two dimensions:
  - Process Dimension: Production of a specific set of artifacts with well-defined activities.
    - Includes the following activities: Requirement, Analysis, Design, Implementation, Testing
  - Time: Division of the life cycle into phases and iterations
    - involves the adoption the following time based phases: Inception, Elaboration, Construction, Transition



Inception Phase:

- Establish the business rationale for the project and decide on the scope of the project.
- Identifies the system’s feasibility and constraints
- Outcome:
  - A vision document, expressing the key requirements and key features
  - An initial project glossary.
  - An initial business case, which includes
    - business context
    - Success criteria (revenue projection, market recognition, and so on).
    - Financial forecast
    - An initial risk assessment
    - A project plan, which shows the phases and iterations.
  - A domain model, which is more sophisticated than a glossary.

Elaboration Phase

- Collecting more detailed requirements
- Do high-level analysis and design to establish baseline architecture, and create the plan for construction.
- Outcomes:
  - A use case model
  - A software architecture description
  - A revised risk list
  - A development plan for the overall project, including the coarse-grained project plan, showing iterations and evaluation criteria for each iteration.
  - Documenting the nonfunctional requirements and any requirements that are not associated with a specific use case.

Construction Phase:

- Build the system in a series of releases.
  - Each release may be an executable version of one or more module (subsystem), or can be the executable version of the entire system.
- Outcomes:
  - The software product, possibly integrated on the adequate platforms
  - The user manuals
  - A description of the current release

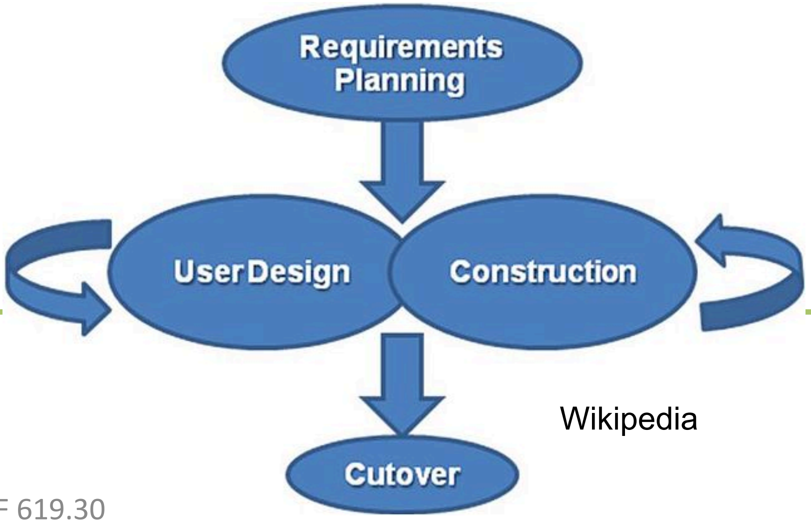
During Transition Phase

- This phase is mainly consisted of:
  - Beta testing
    - a test by the end users
  - Performance tuning
  - User training.
- Outcomes:
  - Product package, ready to be deployed.
  - The evaluation criteria for this phase are as follows:
    - Is the user satisfied?
    - Are the actual resources expenditures versus planned expenditures still in balance?

Agile/Rapid Approaches

Rapid Application Development (RAD) Approach

- Aims to:
  - produce high quality systems quickly, primarily via incremental and iterative evolutionary prototyping.
  - Use computerized development tools such as: GUI builders, Computer Aided Software Engineering (CASE) tools, Database Management System (DBMS).
  - Intensively involve user.
  - Control the project control via "timeboxes". If the project starts to slip, emphasis is on reducing requirements to fit the timebox, not in increasing the deadline.
  - Use Joint Application Design (JAD) approach. Users are intensively involved.



Wikipedia

Agile Development

Agile Software Development

- Agile software development methodology is growing rapidly and gaining popularity among software developers.
  - Starts with customers vision of the product.
  - Doesn't' need all the requirements upfront.
  - Customers are actively involved and they define and add requirements, which produces a backlog of requirements.
  - With the help of development team and continuous involvement of customers the requirements are prioritized. Requirement are prioritized based on customer's requirements, time pressure, time to market and competition concerns, quality, resources, etc.
  - The development team works on the prioritize requirements for a fixed time using Extreme Programming (XP) approach.

Agile Misconceptions?

- Agile means: "letting the programming team do whatever they need to with no project management, and no architecture, allowing a solution to emerge, the programmers will do all the testing necessary with Unit Tests…"

Extreme Programming

- Extreme Programming is a disciplined software development methodology with the focus on simplicity, communication, feedback, respect, etc.
- Core Practices include:
  - Simple Design,
  - Pair Programming,
  - Test-Driven Development,
  - Design Improvement

Agile Manifesto

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 timescale.
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 trust them to get the job done.
6. The most efficient and 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.
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 teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

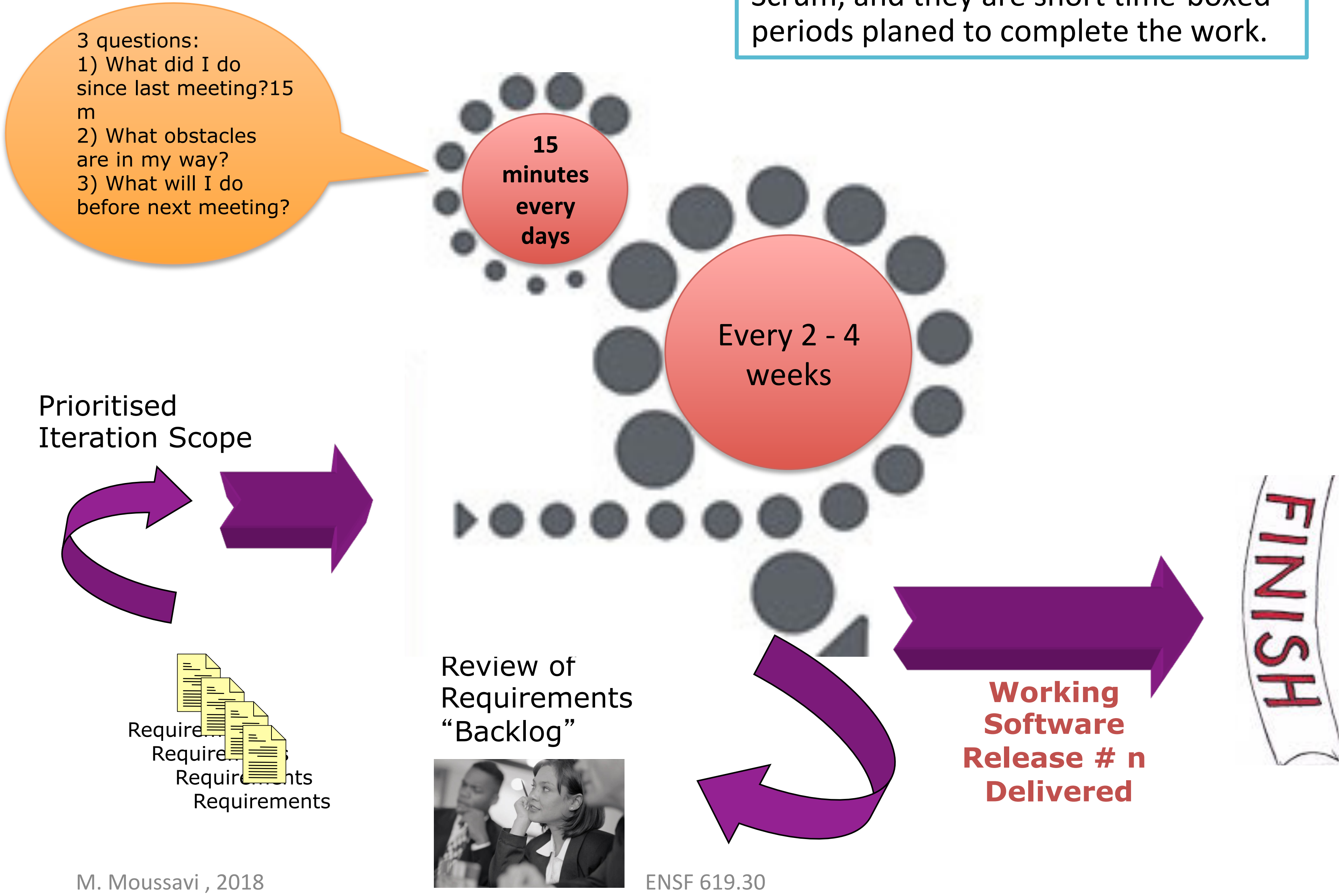
Agile Process Management
Scrum

Scrum

- A management process model used by agile developer is Scrum.
- Scrum is managed by a scrum master and needs to have a combination of technical and soft skills.
  - Note that, a highly skilled software engineer, do not necessarily have the required soft skills for effectively managing people.
- Scrum master is responsible for success of scrum and his main role is to:
  - provide a suitable environment for user community.
    - Not only involvement of the users but also commitment to the project.
  - act as Liaison between scrum team and management.
  - build the scrum team
  - remove impediments
  - plan iterations
  - assign work-assignments
  - estimate efforts
  - manage daily standup meetings and monitor the progress
  - coordinate and conduct the sprint review meetings

SCRUM

Sprints which are at the very heart of Scrum, and they are short time-boxed periods planed to complete the work.


3 questions:
1) What did I do since last meeting?15 m
2) What obstacles are in my way?
3) What will I do before next meeting?
15 minutes every days
Every 2 - 4 weeks
Prioritised Iteration Scope
Requirements
Requirements
Requirements
Requirements
Review of Requirements "Backlog"
Working Software Release # n Delivered
FINISH