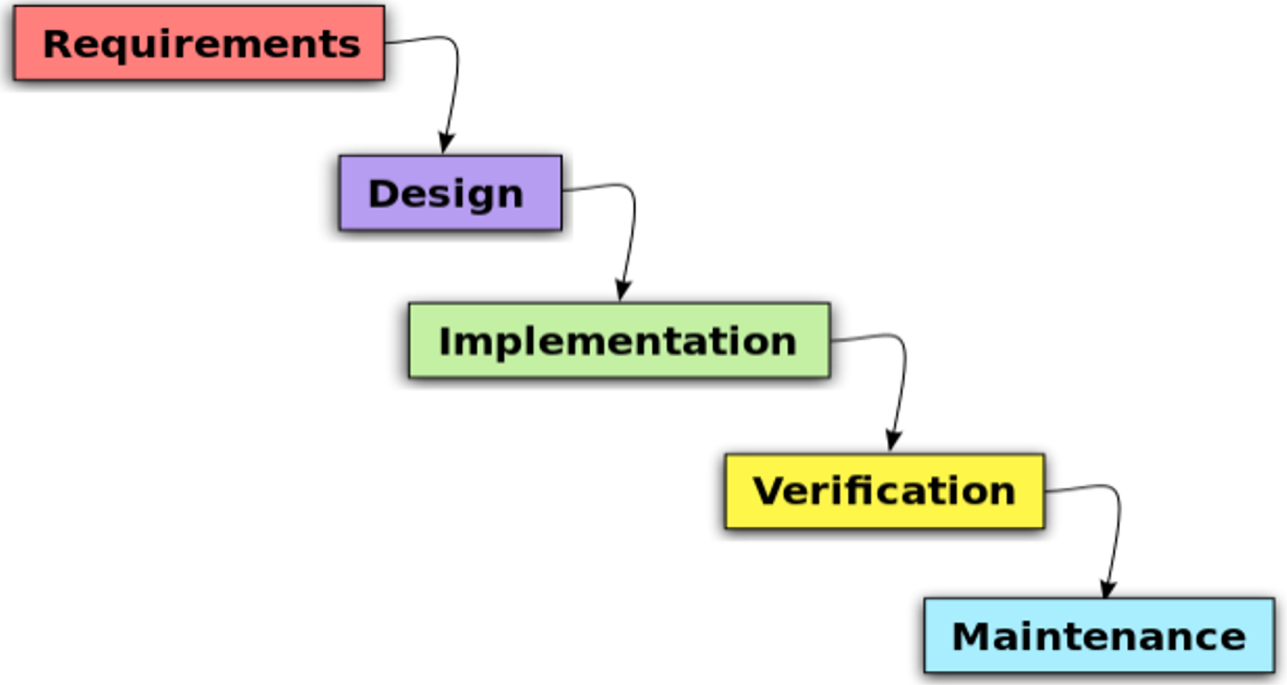
ENSF 619
Software Requirement Analysis and Process Management

What is Software Development Processes?

Software Development Process

- Software Development Process or also called Software Development Life Cycle is a splitting of software development activities in different phases.
- Waterfall Development Process:

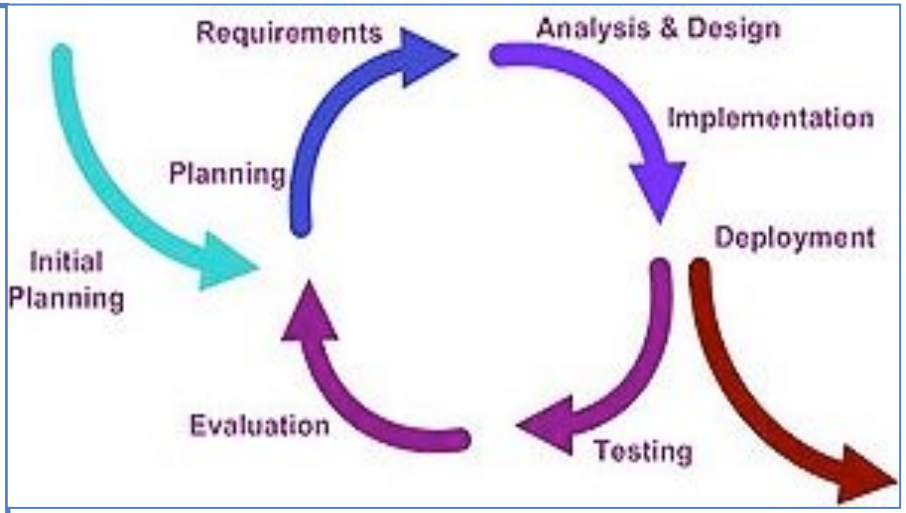


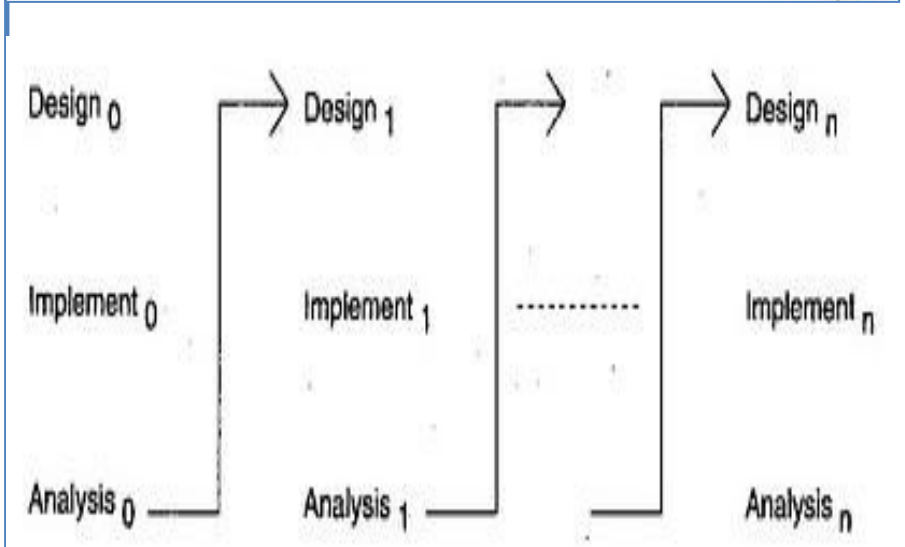
Waterfall Process – Pros & Cons

- 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
  - Suitable for projects that are well defined, scope is fixed and stable and technology is clearly understood
  - 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
  - High degree of uncertainty and risk
  - Not suitable for complex projects

Software Development Process

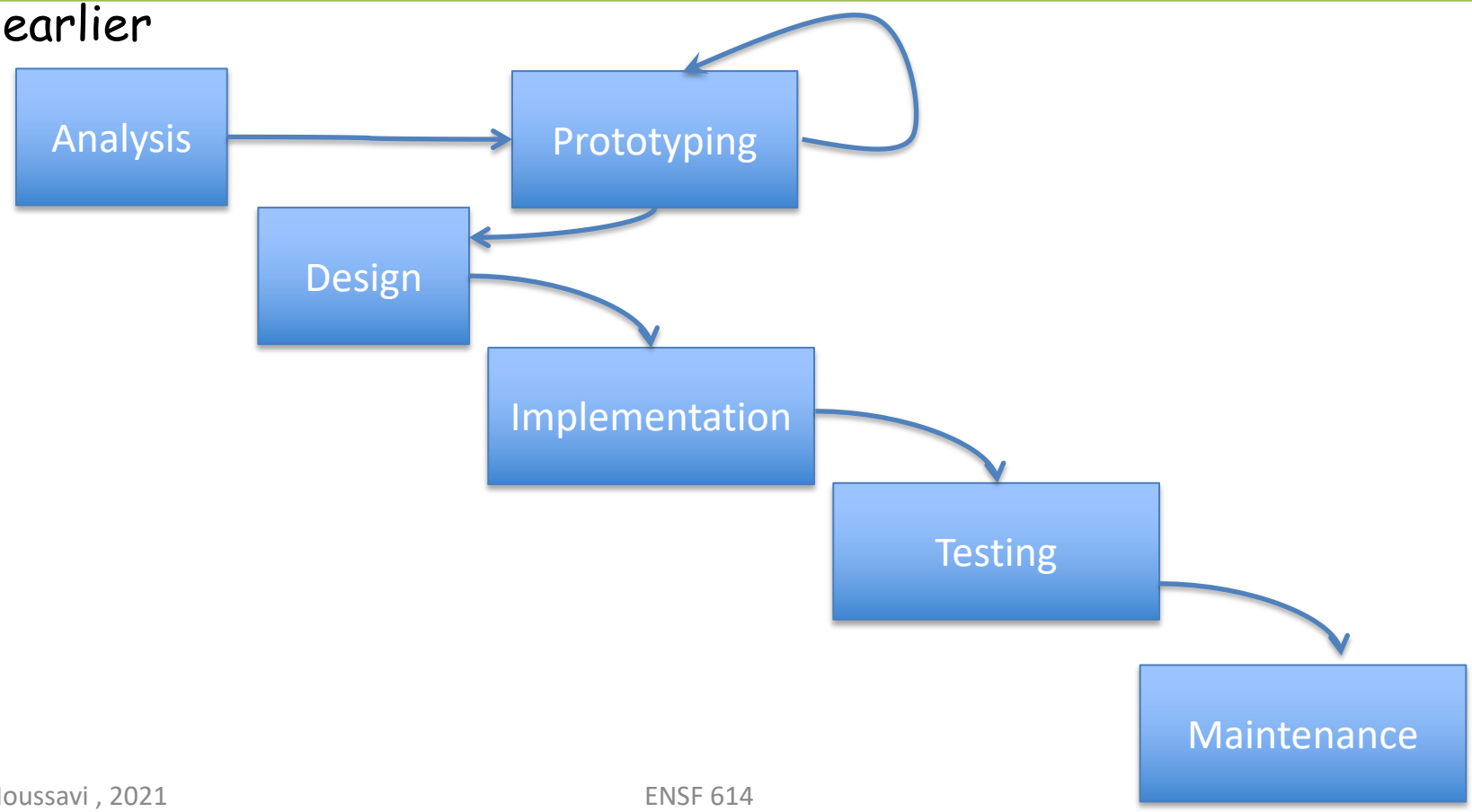
- Incremental and Iterative
  - Pros
    - Early working product
    - Easier to mange the requirement changes
    - Easier for testing and debugging
    - Lower risk factor





Prototyping

- Early approximation of a final product
- Prototyping can improve the quality of requirements and specifications provided to developers.
- Can help to reduce cost of requirement errors will be detected earlier

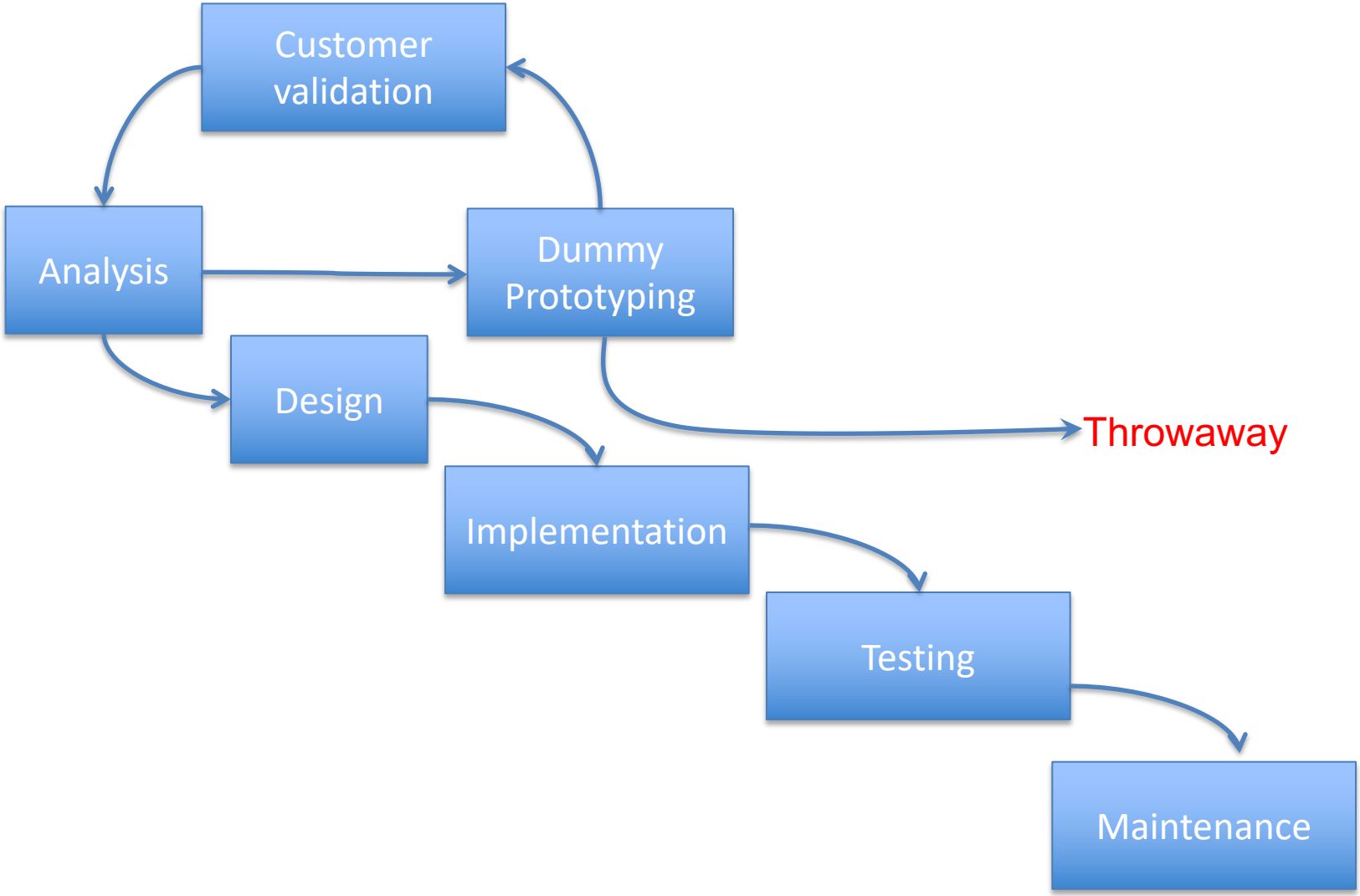


Prototyping Disadvantages

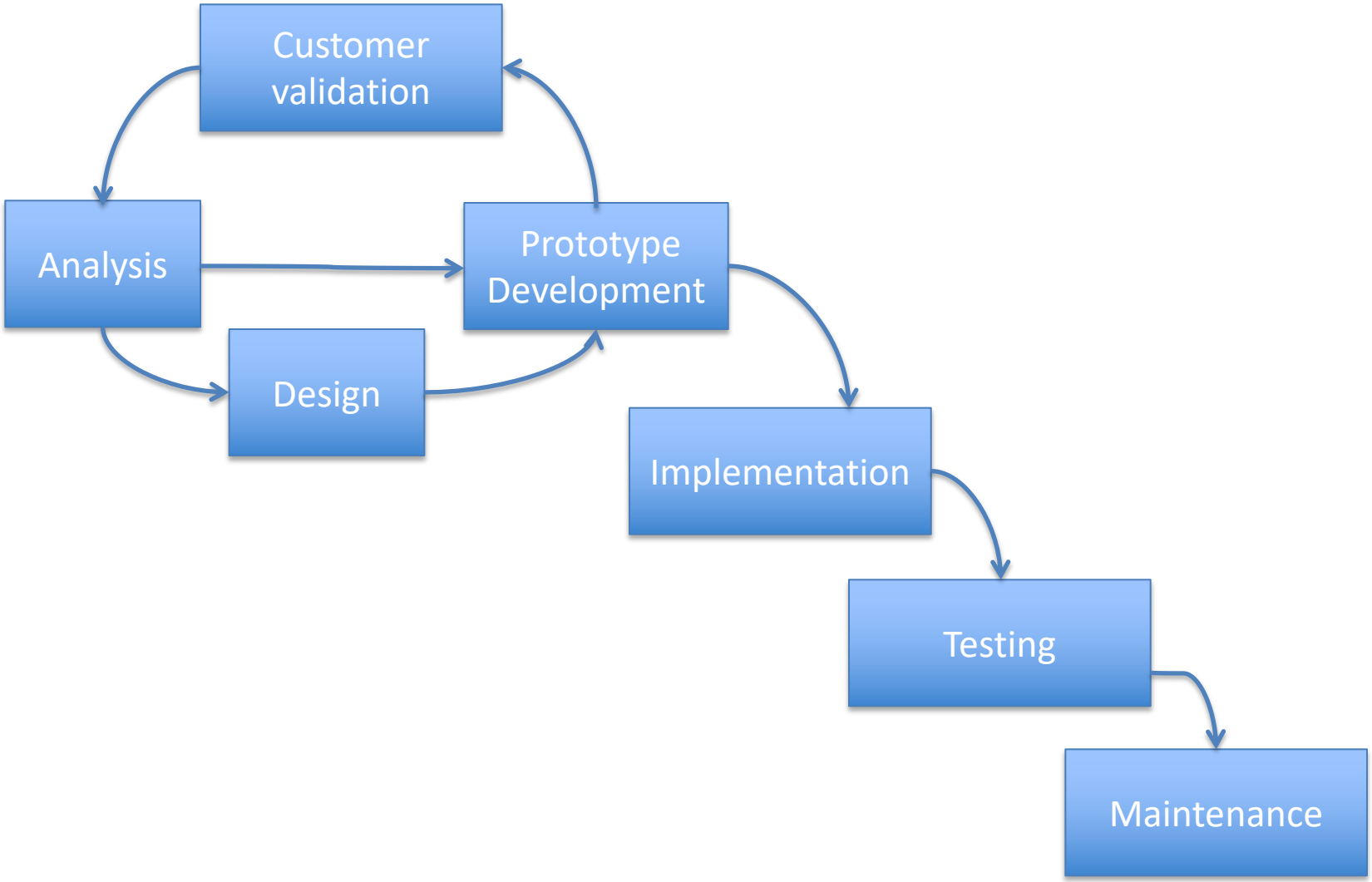
- Due to insufficient formal analysis and design may increases the complexity of the overall system
- Involves exploratory (trial and error) methodology and therefore involves higher risk.
- Involves implementing and then repairing the way a system is built, so errors are an inherent part of the development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

Different Prototyping Approaches

Throwaway Prototyping Approach



Evolutionary Prototyping Approach



Agile Development Approach

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 Project Management
Scrum

SCRUM

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

Review of Requirements "Backlog"

Working Software Release # n Delivered

FINISH

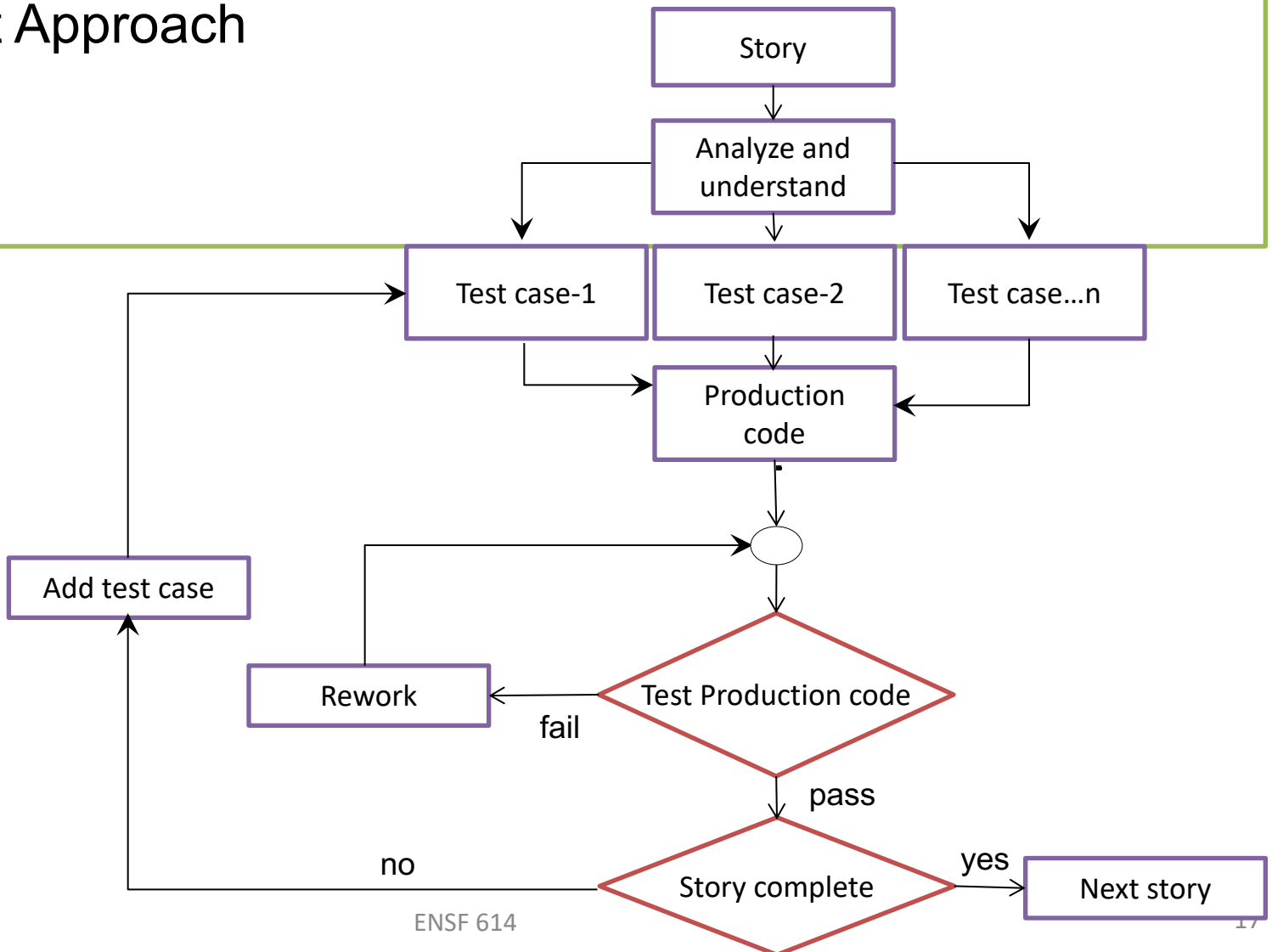
Agile Misconceptions?

- 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…”

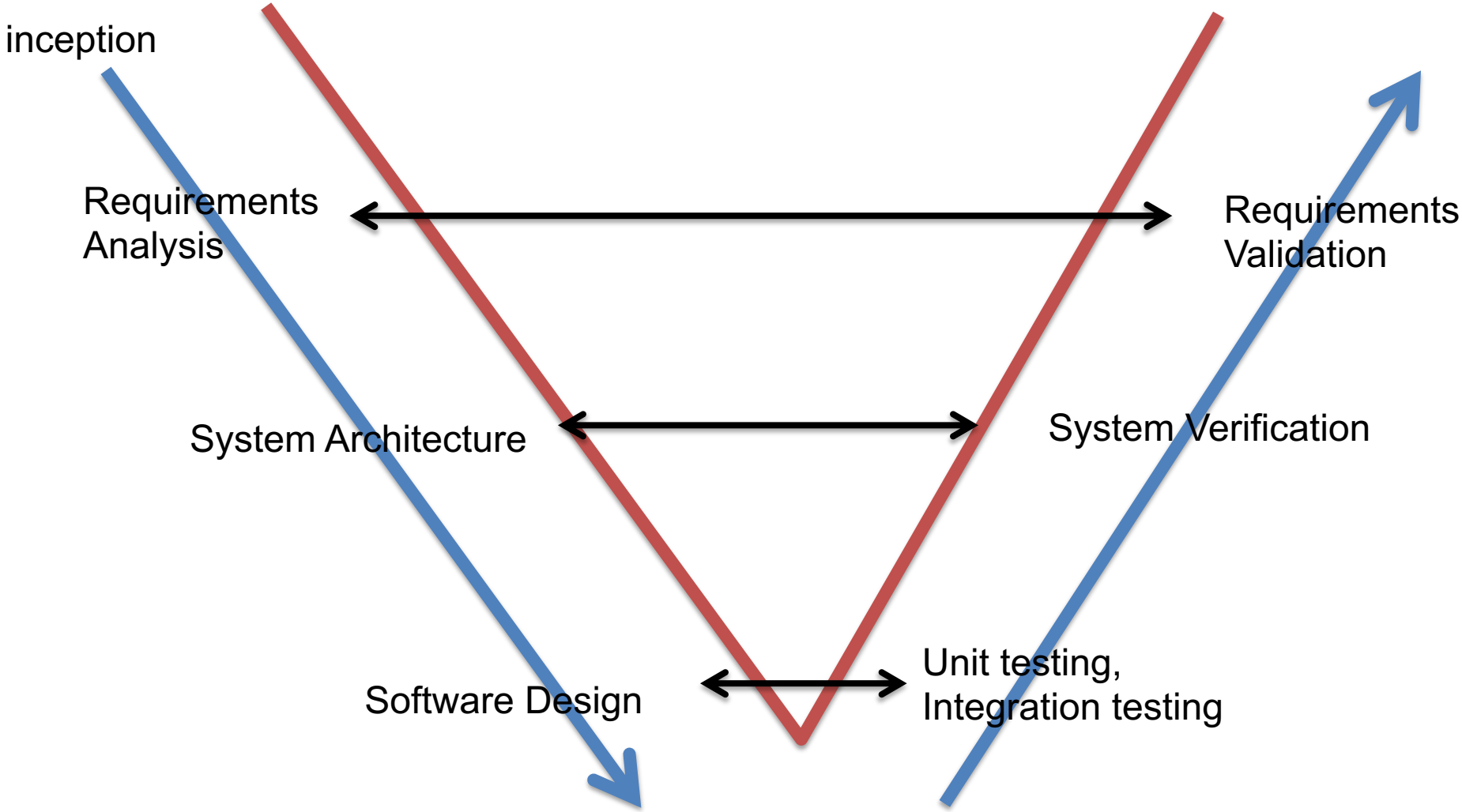
Test-Focused Approaches

Test-Focused Approaches (TDD)

- TDD is a software development process that relies on the repetition of a very short development cycle:
- Test First Approach

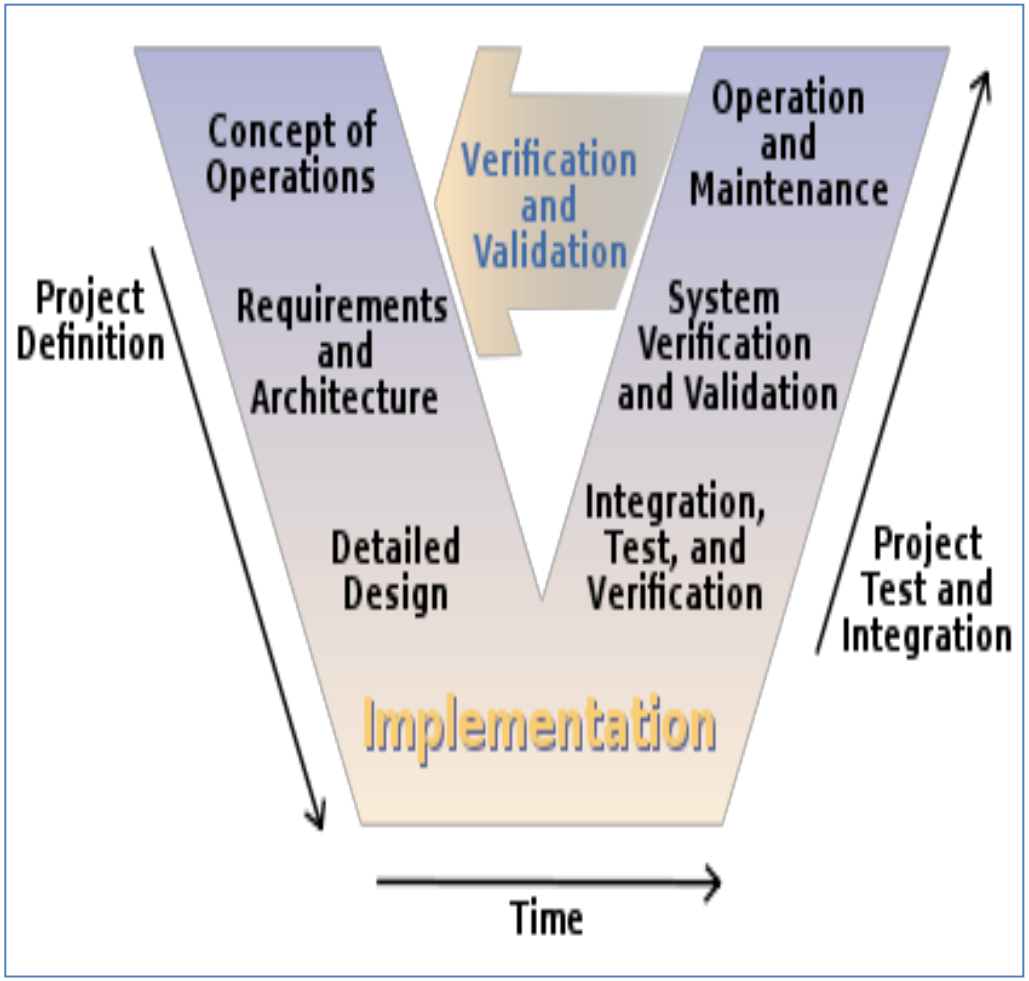


Test Focused Models: The V Model



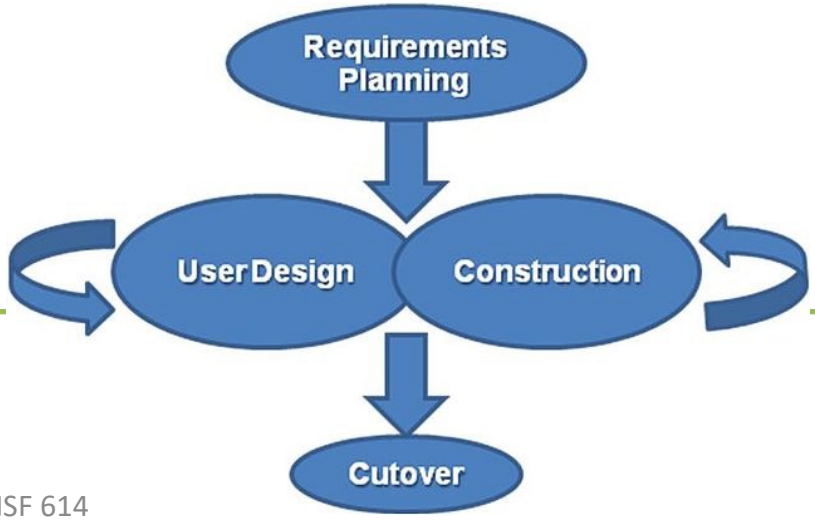
Test-Focused Approaches

- V-Model
  - A derivation of Waterfall Approach



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.



Spiral Development Process

- An iterative and incremental with the waterfall approach, and particular focus on risk analysis

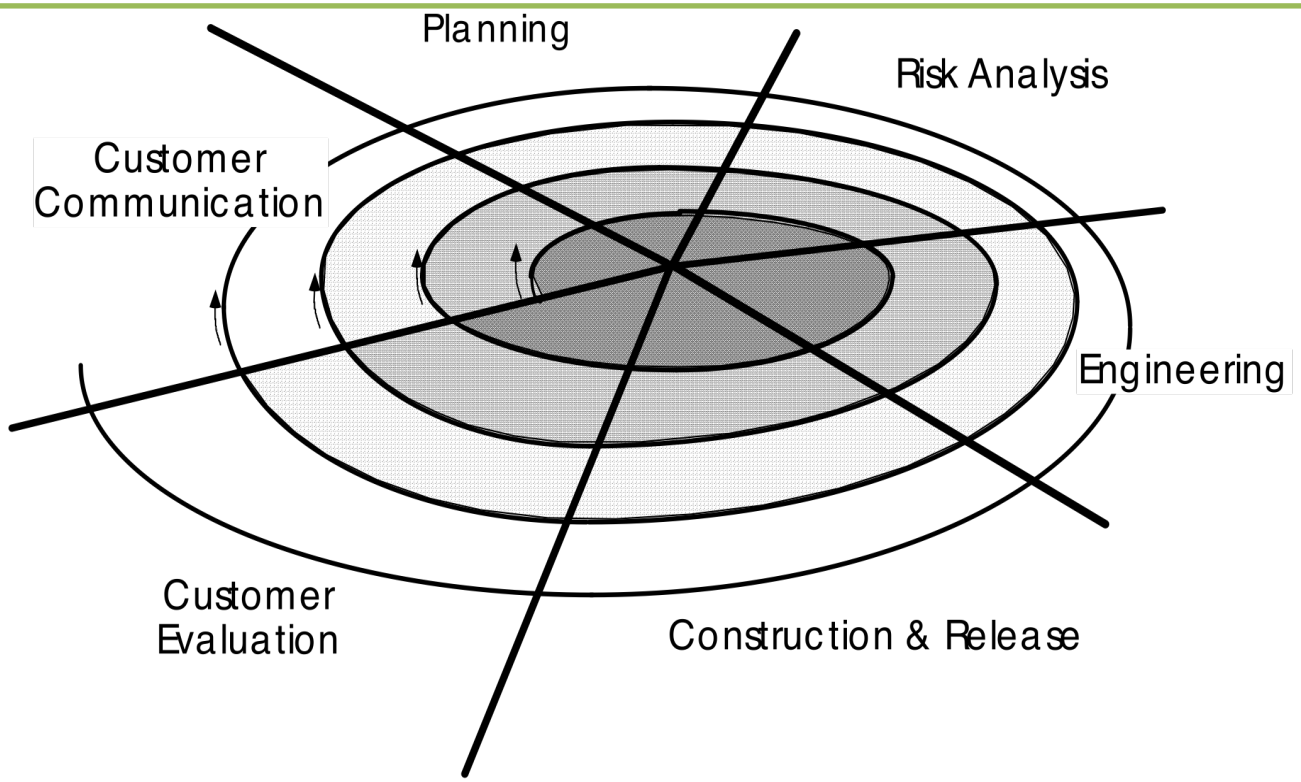


Figure from Software Engineering. A Practitioner's Approach, McGraw-Hill

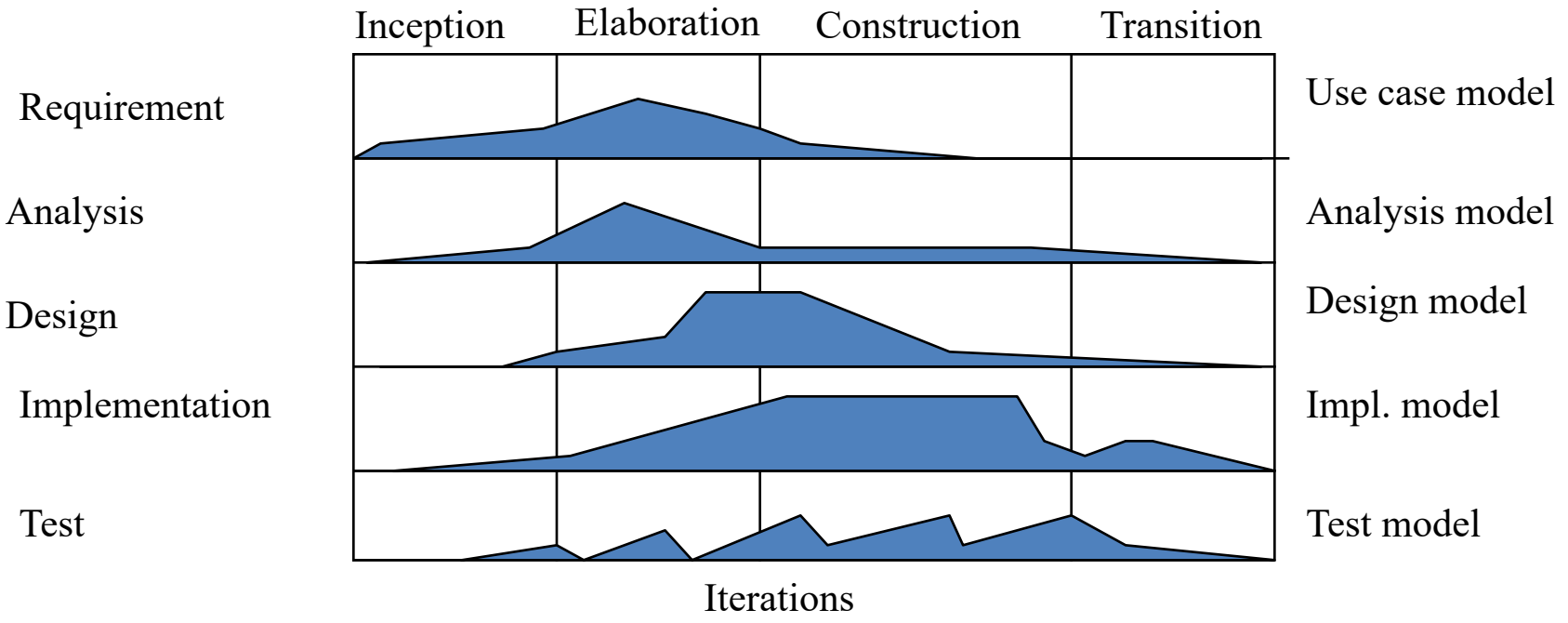
Rational Unified Development Process (RUP)

The Unified Process

- The Rational Unified Process is structured along two dimensions:
  - Process components: Production of a specific set of artifacts with well-defined activities.
  - Time: Division of the life cycle into phases and iterations
- Both dimensions must be taken into account for a project to succeed.

The Rational Unified Process

- Process component dimension includes the following activities: Requirement, Analysis, Design, Implementation, Testing
- Time dimension involves the adoption the following time based phases: Inception, Elaboration, Construction, Transition



During 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.
  - One or more prototypes.

During 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.

During 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?