CS60.311- Fall 2018

University of Windsor

Introduction to Software Engineering

Lecture 02: Software Process

Agenda

Software Process

Software Process Models

Software Process Improvement

Software Process

Software Process

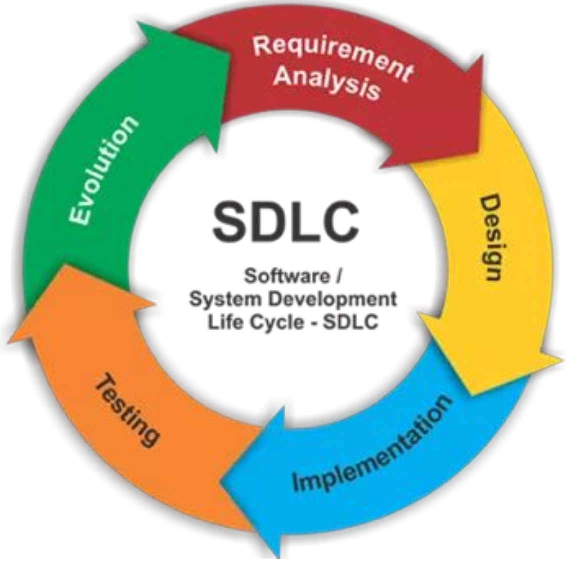
A structured set of activities required to develop a software system.

Many different software processes but all involve:

- Specification – defining what the system should do;
- Design and implementation – defining the organization of the system and implementing the system;
- Validation – checking that it does what the customer wants;
- Evolution – changing the system in response to changing customer needs.

Software Process

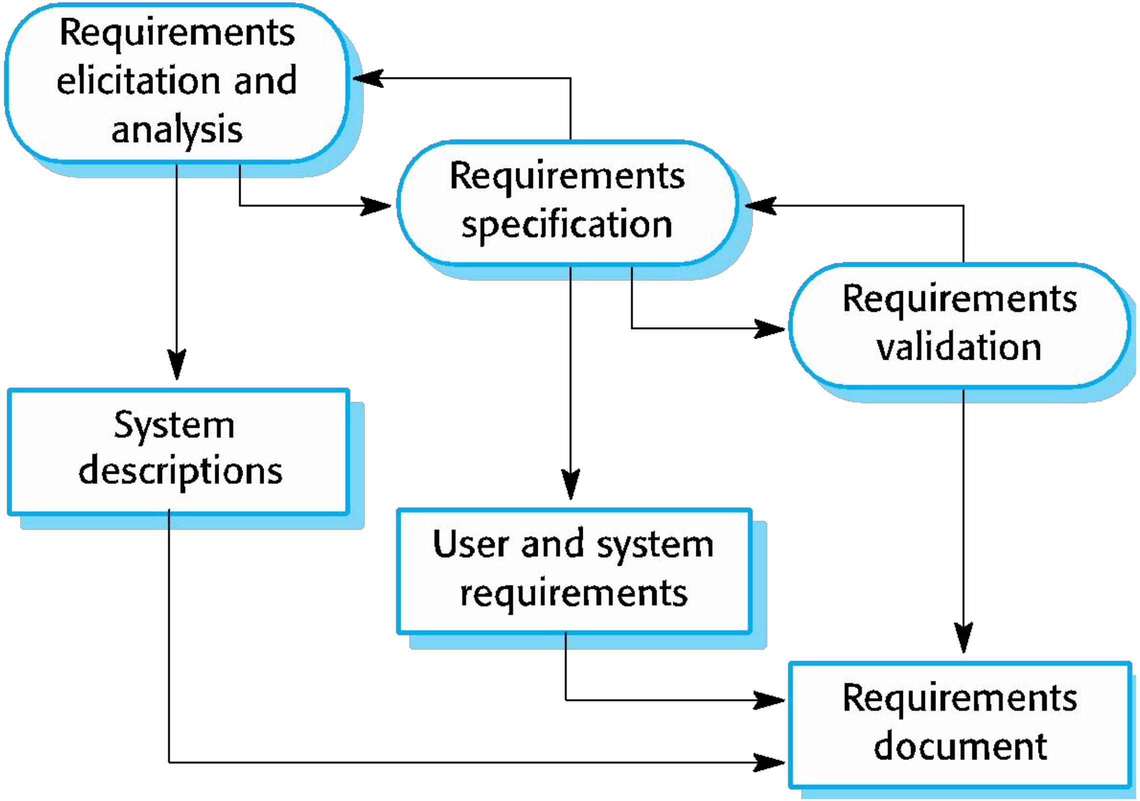
- The software process also known as software development life cycle (SDLC).
- The four basic process activities of specification, development, validation and evolution are organized differently in different development processes.
- For example, in the waterfall model, they are organized in sequence, whereas in incremental development they are interleaved.



Specification/Requirements Analysis

- The process of establishing what services are required and the constraints on the system’s operation and development.
- Requirements engineering process
  - Requirements elicitation and analysis
    - What do the system stakeholders require or expect from the system?
  - Requirements specification
    - Defining the requirements in detail
  - Requirements validation
    - Checking the validity of the requirements

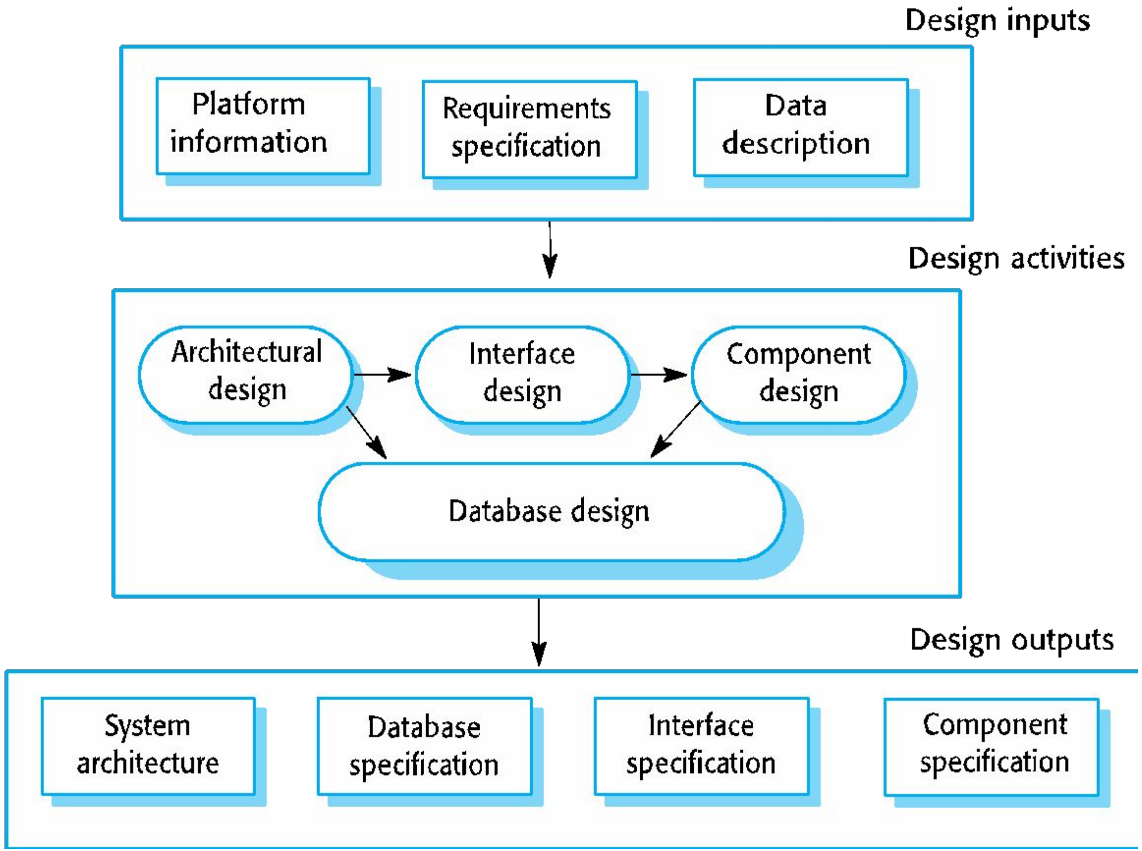
Requirements Engineering


Requirements elicitation and analysis
Requirements specification
Requirements validation
System descriptions
User and system requirements
Requirements document

Software Design and Implementation

- The process of converting the system specification into an executable system.
- Software design
  - Design a software structure that realises the specification;
- Implementation
  - Translate this structure into an executable program;
- The activities of design and implementation are closely related and may be inter-leaved.

Software Design



Design Activities

1. Architectural design, where you identify the overall structure of the system, the principal components (subsystems or modules), their relationships and how they are distributed.
2. Interface design, where you define the interfaces between system components.
3. Component selection and design, where you search for reusable components. If unavailable, you design how it will operate.
4. Database/Data Model design, where you design the system data structures and how these are to be represented in a database.

Software Validation & Testing

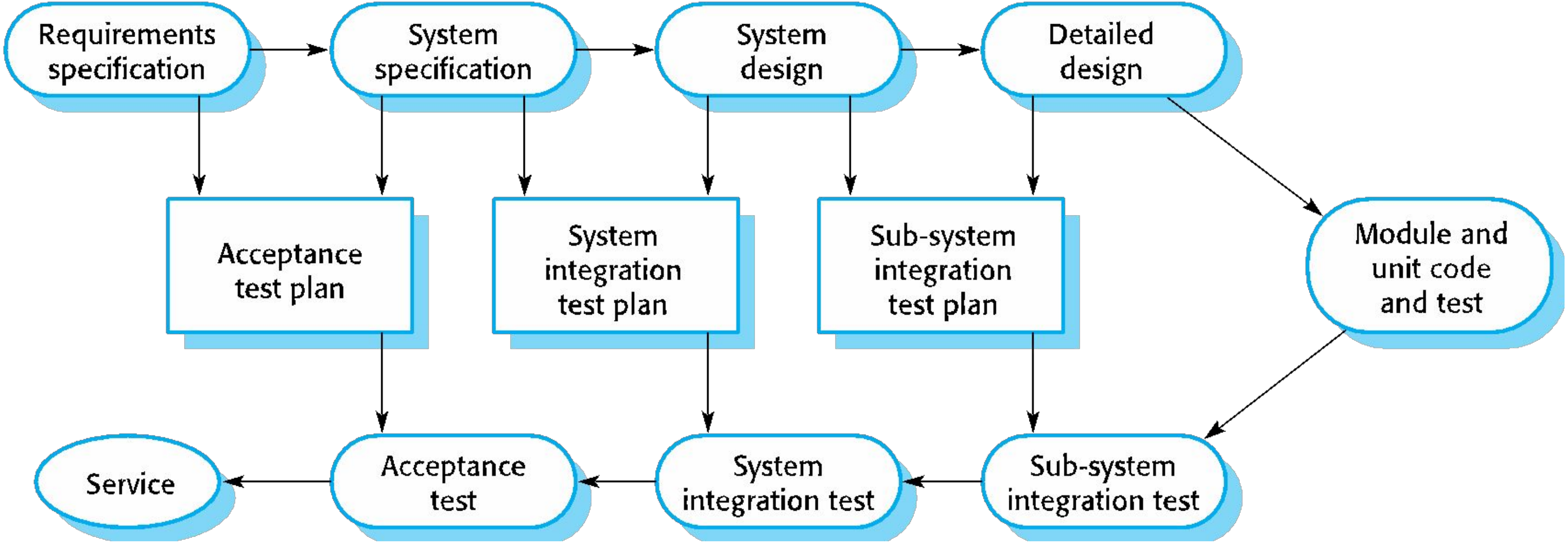
Is intended to show that a system conforms to its specification and meets the requirements of the system customer.

System testing involves executing the system with test cases that are derived from the specification of the real data to be processed by the system.

Involve the design of a test plan, test cases, and test cases executions.

There are different types of testing (e.g. unit testing, component testing, functional testing, integration testing, usability testing, load testing, etc)

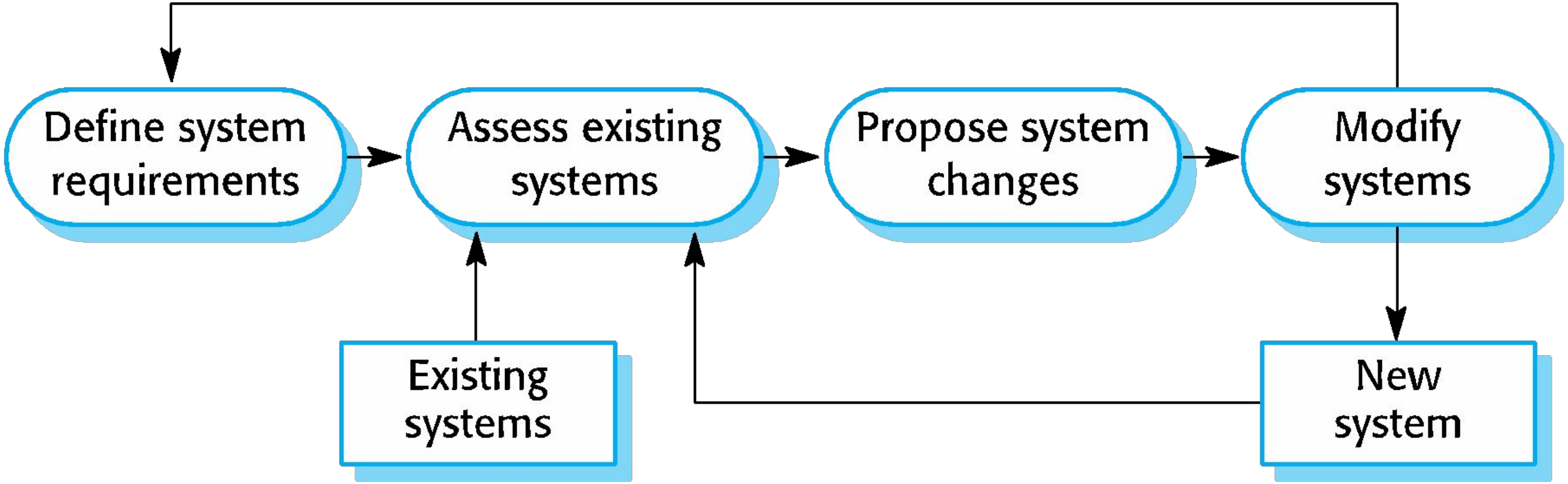
Software Validation & Testing



Requirements specification
System specification
System design
Detailed design
Acceptance test plan
System integration test plan
Sub-system integration test plan
Module and unit code and test
Service
Acceptance test
System integration test
Sub-system integration test

Software Evolution

- The software is inherently flexible and can change.
- As requirements change through changing business circumstances, the software that supports the business must also evolve and change.



Software Process Model

Software Process Model

- A software process model is an abstract representation of a process. It presents a description of a process from some particular perspective.
- The software process model identify the key phases or activities for the software process. It describes the order of these activities and the relations between them.
- The software process model does not specify the details of the different phases of activities in the software process.
- Each model has its own pros and cons. There is no one model that works for all software systems.

Software Process Model

As we mentioned, there are many software process models, but the most common ones are:

1. Waterfall Model
2. V-Shaped Model
3. Spiral Model
4. Evolutionary Prototyping Model
5. Iterative and Incremental Model
6. Integration and configuration Model
7. Agile Model

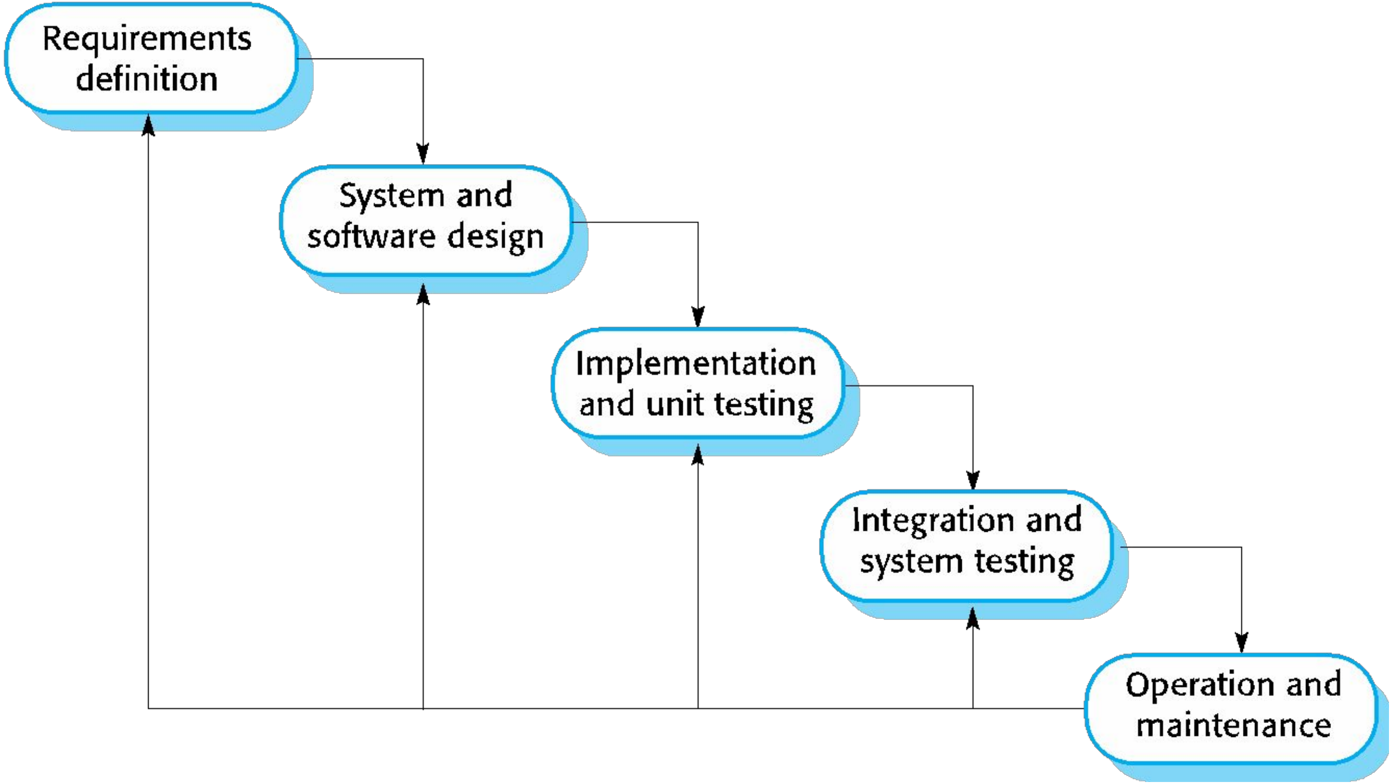
Plan Driven Processes vs Incremental Processes

- We could categorize the model into plan driven models and incremental processes model.
- Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan.
- In Incremental processes, planning is incremental and it is easier to change the process to reflect changing customer requirements.
- In practice, most practical processes include elements of both plan-driven and incremental approaches.

Selecting A Software Process Model

- Understand the pros and cons of every software process model
- Understand the capabilities and objectives of the system stakeholders
- Here are the common factors that usually affect the selection of a software process model
  - The size and skills of the software developers team
  - The technologies and tools that will be used in the project.
  - The geographical location and communication style of the development team.
  - Identify if clients require specific software process model.
  - The requirements of the software systems and the values of these requirements.

The Waterfall Model



Requirements definition
System and software design
Implementation and unit testing
Integration and system testing
Operation and maintenance

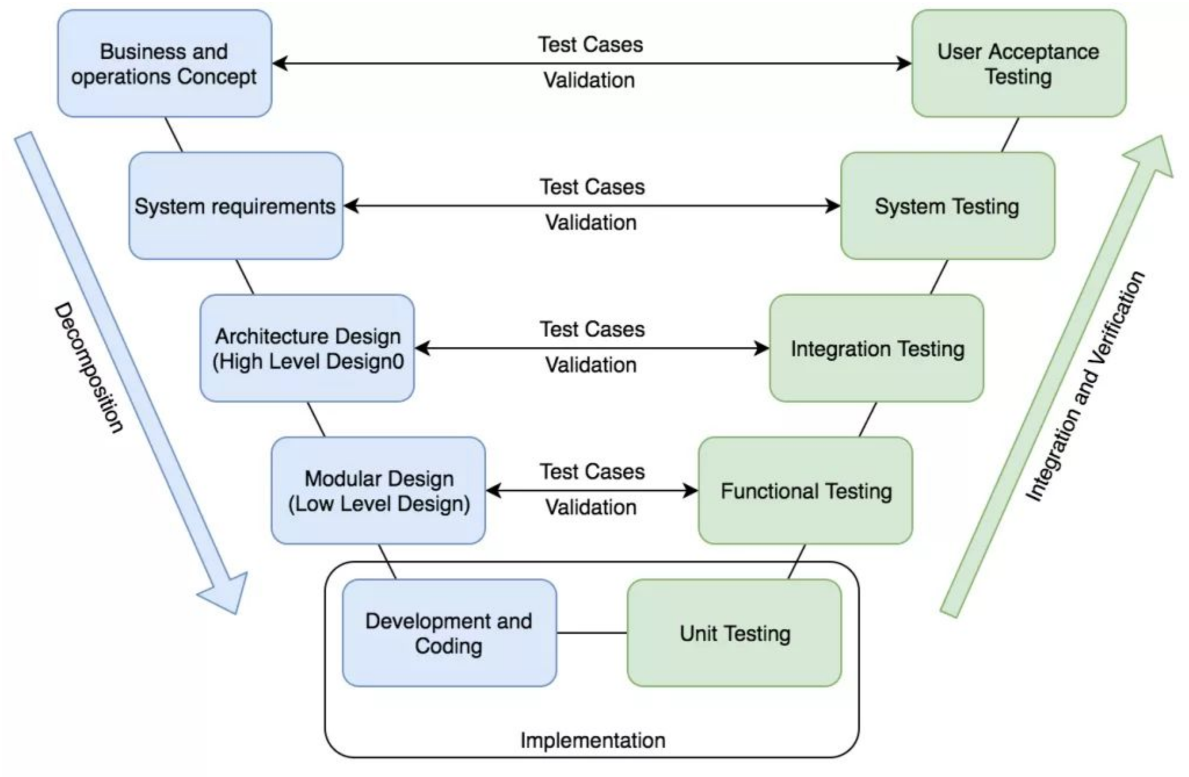
The Waterfall Model

- There are separately identified phases in the waterfall model:
  - Requirements analysis and definition
  - System and software design
  - Implementation and unit testing
  - Integration and system testing
  - Operation and maintenance
- The main drawback of the waterfall model is the difficulty of accommodating change after the process is underway. In principle, a phase has to be complete before moving onto the next phase.

The Waterfall Model: Known Issues

- Inflexible partitioning of the project into distinct stages makes it difficult to respond to changes in customer requirements.
- Therefore, this model is only appropriate when the requirements are well-understood, and changes will be fairly limited during the design process.
- Few business systems have stable requirements.
- The waterfall model is mostly used for large systems engineering projects where a system is developed at several sites.
- In those circumstances, the plan-driven nature of the waterfall model helps coordinate the work.

The V-Shaped Model



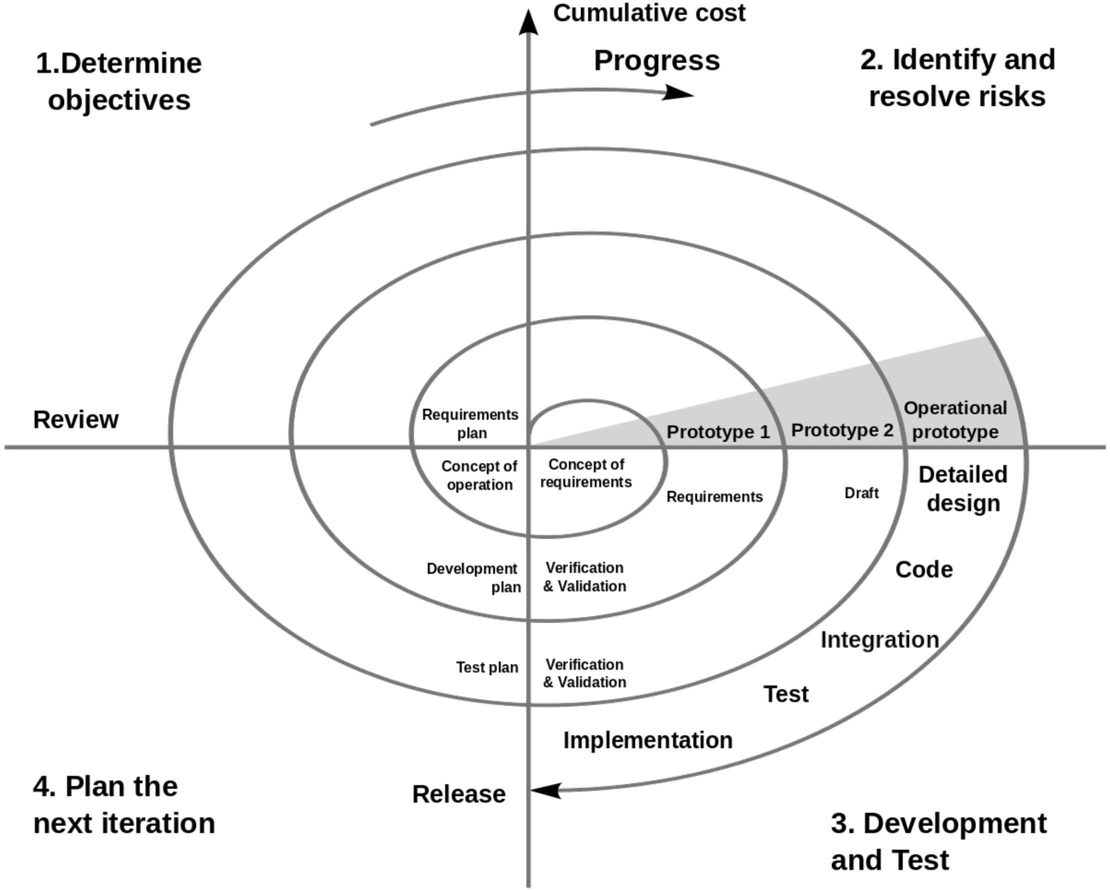
The V-Shaped Model

- The V-Shaped Model is also known as the validation and verification software process.
- It is an extension or improvement version of the waterfall model.
- It has the same issues as the waterfall model. It focuses more on the testing and validation of the system.
- The same activities of the waterfall model are executed sequentially, however, after the implementation the steps move upwards to form the typical V shape that demonstrates the relationships between each phase of the development life cycle and its associated phase of testing.

When to use the waterfall or the V-shaped Model?

- The customer has a well defined and documented requirements
- Mission Critical systems, for example, flight navigation system
- Projects with defined and clear requirements. For example, a project initiated by a request for proposals (RFPs)
- Major requirements changes are not possible or not allowed

The Spiral Model



Spiral Model Sectors

- Objective setting
  - Specific objectives for the phase are identified.
- Risk assessment and reduction
  - Risks are assessed and activities put in place to reduce the key risks.
- Development and validation
  - A development model for the system is chosen which can be any of the generic models. Development takes place.
- Planning
  - The project is reviewed and the next phase of the spiral is planned.

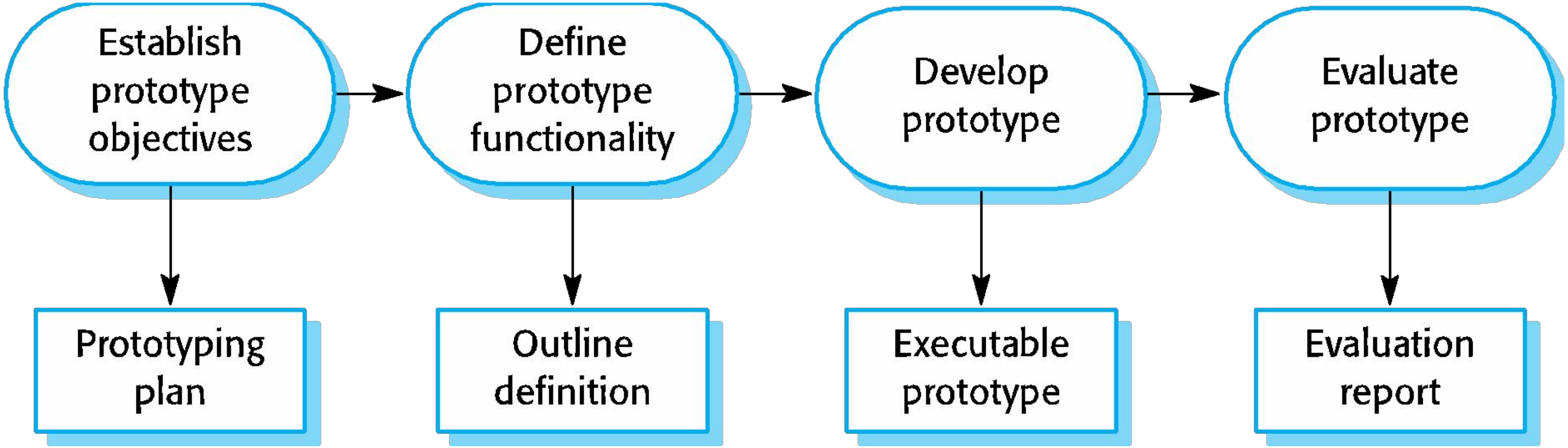
The Spiral Model

- The process is represented as a spiral rather than as a sequence of activities with backtracking.
- Each loop in the spiral represents a phase in the process.
- No fixed phases such as specification or design - loops in the spiral are chosen depending on what is required.
- Risks are explicitly assessed and resolved throughout the process. This was the motivation behind developing the Spiral Model - Risk

Spiral Model

- Spiral model has been very influential in helping people think about iteration in software processes and introducing the risk-driven approach to development.
- In practice, however, the model is rarely used as published for practical software development.

Prototyping Model



Prototyping Model

- A prototype is an initial version of a system used to demonstrate concepts and try out design options.
- A prototype can be used in:
  - The requirements engineering process to help with requirements elicitation and validation;
  - In design processes to explore options and develop a UI design;
  - In the testing process to run back-to-back tests.

Prototype Development

- May be based on rapid prototyping languages or tools
- May involve leaving out functionality
  - Prototype should focus on areas of the product that are not well-understood;
  - Error checking and recovery may not be included in the prototype;
  - Focus on functional rather than non-functional requirements such as reliability and security

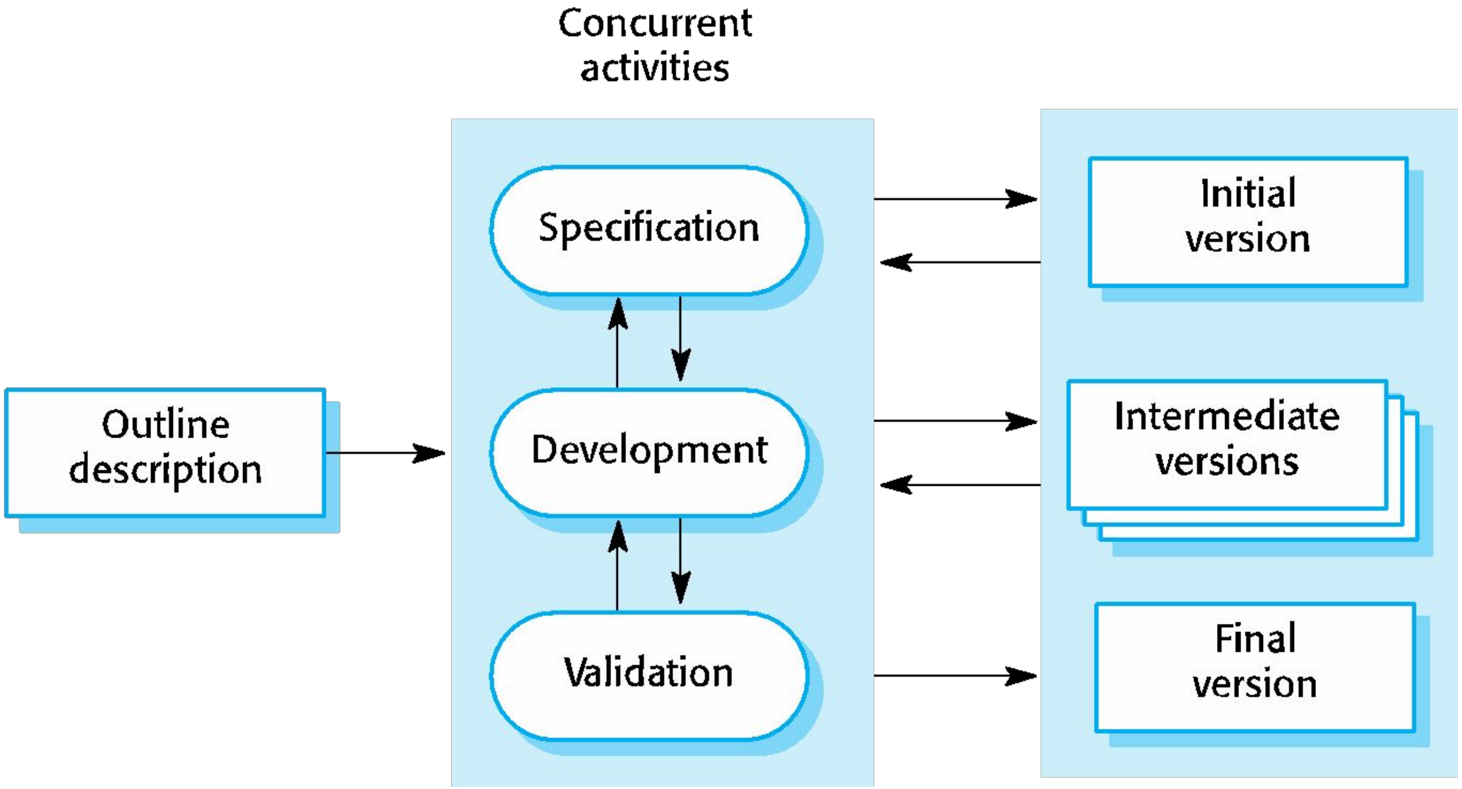
Throw-away prototypes

- Prototypes should be discarded after development as they are not a good basis for a production system:
  - It may be impossible to tune the system to meet non-functional requirements;
  - Prototypes are normally undocumented;
  - The prototype structure is usually degraded through rapid change;
  - The prototype probably will not meet normal organizational quality standards.

Evolutionary/Extreme Prototyping

- Used in web applications mainly.
- it breaks down web development into three phases, each one based on the preceding one:
    a. The first phase is a static prototype that consists mainly of HTML pages.
    b. In the second phase, the screens are programmed and fully functional using a simulated services layer.
    c. In the third phase, the services are implemented
- Recently has been used for mobile apps development

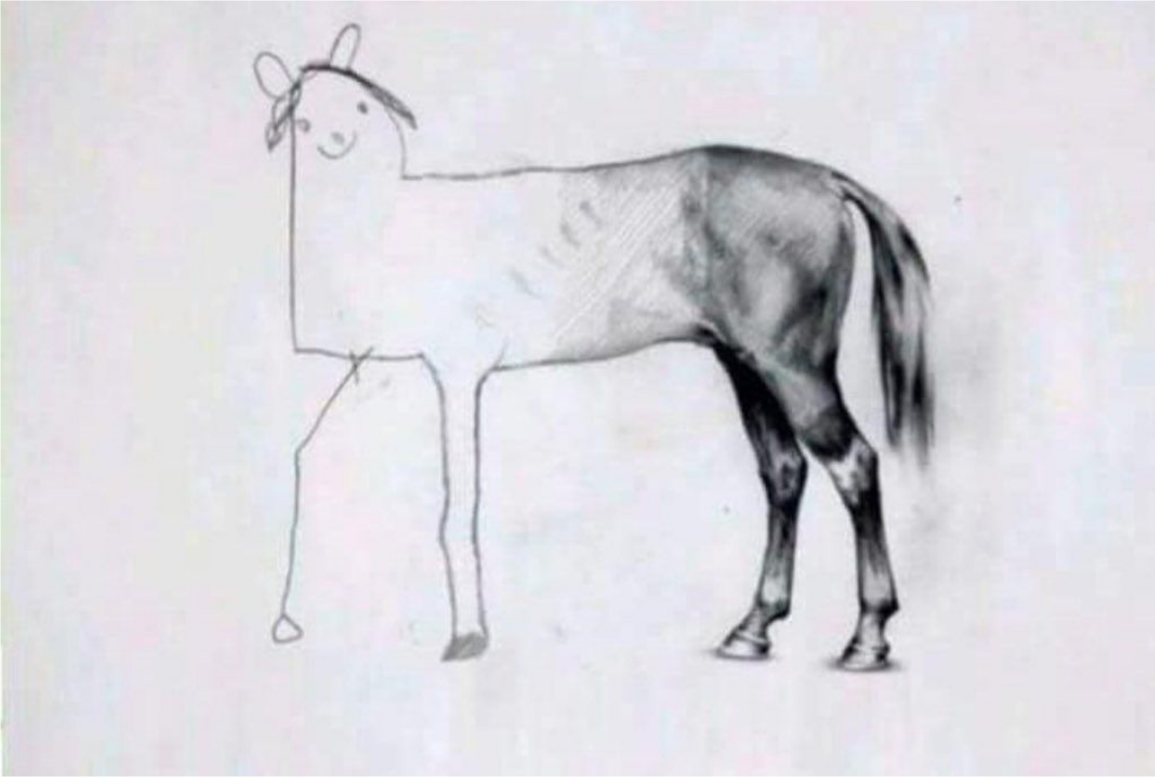
Iterative and Incremental Model



Iterative and Incremental Model

- The cost of accommodating changing customer requirements is reduced.
- The amount of analysis and documentation that has to be redone is much less than is required with the waterfall model.
- It is easier to get customer feedback on the development work that has been done.
- Customers can comment on demonstrations of the software and see how much has been implemented.
- More rapid delivery and deployment of useful software to the customer is possible.

Incremental Model: Known Issues and Usages



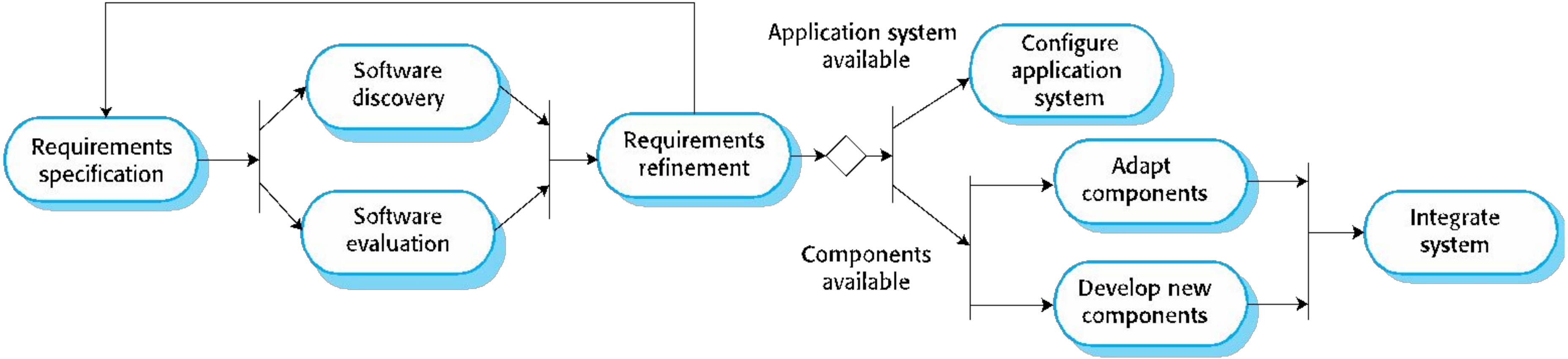
Incremental Model: Known Issues and Usages

System structure tends to degrade as new increments are added.

Unless time and money is spent on refactoring to improve the software, regular change tends to corrupt its structure. Incorporating further software changes becomes increasingly difficult and costly.

It is used in shrink-wrap application and large system which built-in small phases or segments. Also, can be used in a system has separated components, for example, ERP system. Which we can start with the budget module as a first iteration and then we can start with the inventory module and so forth.

Integration and Configuration Model



Integration and Configuration Model

Key Phases/Activities

1. Requirements specification
2. Software discovery and evaluation
3. Requirements refinement
4. Application system configuration
5. Component adaptation and integration

Integration and Configuration Model

- Based on software reuse where systems are integrated from existing components or application systems (sometimes called COTS -Commercial-off-the-shelf).
- Reused elements may be configured to adapt their behaviour and functionality to a user's requirements
- Reuse is now the standard approach for building many types of business system

Types of Reusable Software

- Stand-alone application systems (sometimes called COTS) that are configured for use in a particular environment.
- Collections of objects that are developed as a package to be integrated with a component framework such as .NET or J2EE.
- Web services that are developed according to service standards and which are available for remote invocation.

Integration and Configuration Model Pros/Cons

Pros

- Reduced costs and risks as less software is developed from scratch
- Faster delivery and deployment of system

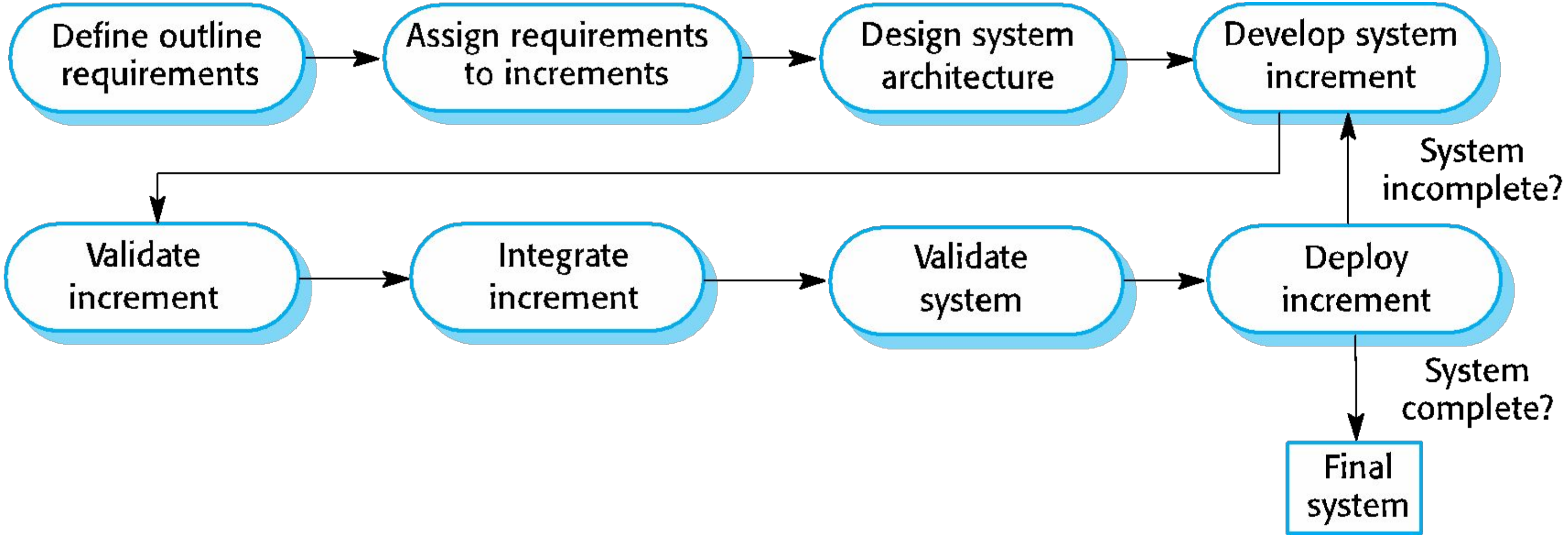
Cons

- But requirements compromises are inevitable so system may not meet real needs of users
- Loss of control over evolution of reused system elements

Agile Model

- Agile is more like a philosophy than just a software process model
- It is based on iterative and incremental development, where requirements and solutions evolve over multiple incremental deliveries.
- Planning is incremental, and it is easier to change the process to reflect changing customer requirements.
- In theory could be used with any type of the project, but it needs more engagement from the customer and to be interactive.
- There are different approaches to implement an agile model such as SCRUM, KANBAN, LEAN, SCRUMBAN

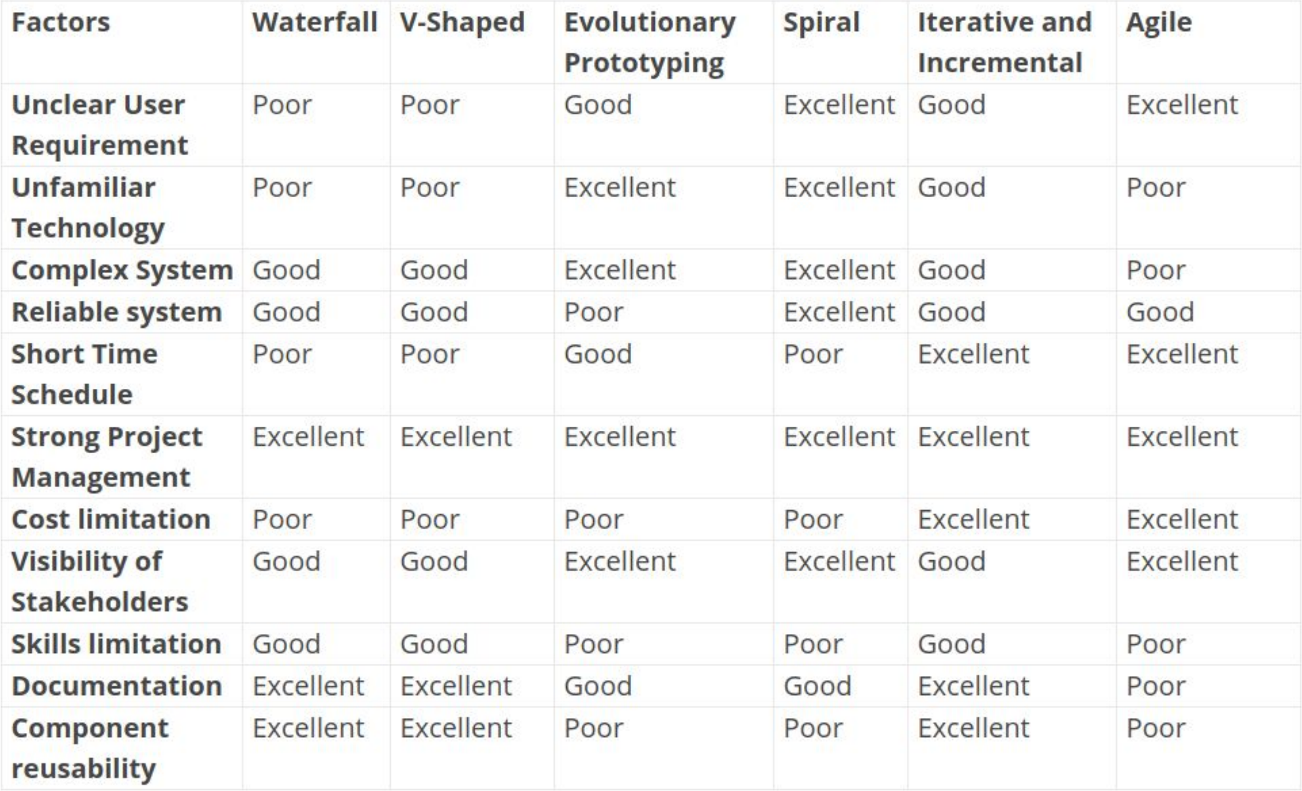
Agile Model


Define outline requirements
Assign requirements to increments
Design system architecture
Develop system increment
System incomplete?
Validate increment
Integrate increment
Validate system
Deploy increment
System complete?
Final system

Agile Model Advantages

- Customer value can be delivered with each increment so system functionality is available earlier.
- Early increments act as a prototype to help elicit requirements for later increments.
- Lower risk of overall project failure.
- The highest priority system services tend to receive the most testing.

Selecting A Software Process Model



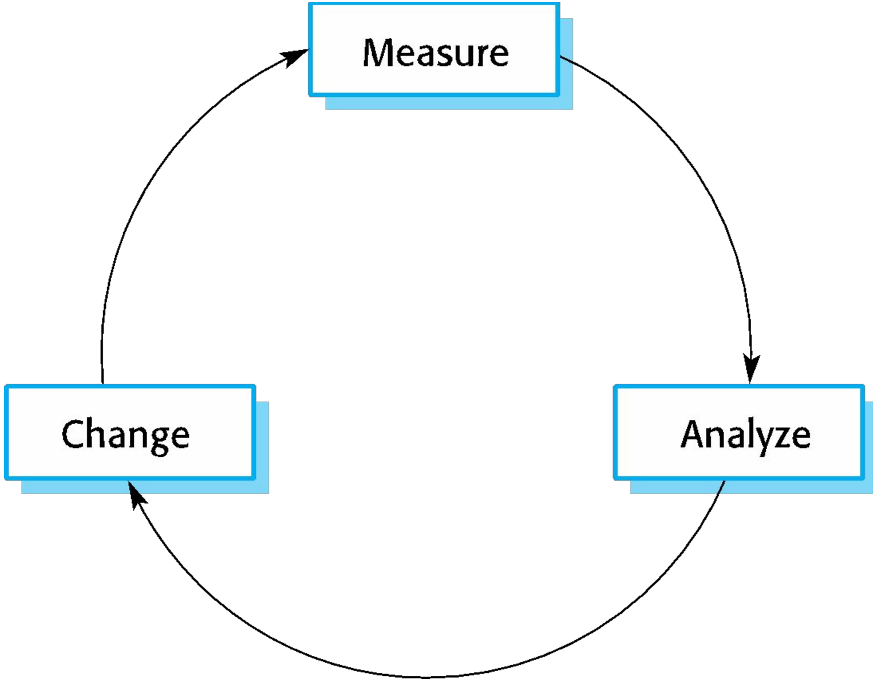
Factors	Waterfall	V-Shaped	Evolutionary Prototyping	Spiral	Iterative and Incremental	Agile
Unclear User Requirement	Poor	Poor	Good	Excellent	Good	Excellent
Unfamiliar Technology	Poor	Poor	Excellent	Excellent	Good	Poor
Complex System	Good	Good	Excellent	Excellent	Good	Poor
Reliable system	Good	Good	Poor	Excellent	Good	Good
Short Time Schedule	Poor	Poor	Good	Poor	Excellent	Excellent
Strong Project Management	Excellent	Excellent	Excellent	Excellent	Excellent	Excellent
Cost limitation	Poor	Poor	Poor	Poor	Excellent	Excellent
Visibility of Stakeholders	Good	Good	Excellent	Excellent	Good	Excellent
Skills limitation	Good	Good	Poor	Poor	Good	Poor
Documentation	Excellent	Excellent	Good	Good	Excellent	Poor
Component reusability	Excellent	Excellent	Poor	Poor	Excellent	Poor

Software Process Improvement

Process Improvement

- Many software companies have turned to software process improvement as a way of enhancing the quality of their software, reducing costs or accelerating their development processes.
- Process improvement means understanding existing processes and changing these processes to increase product quality and/or reduce costs and development time.
- We use an iterative approach for process improvement.
- There are three main activities for process improvements, namely, process measurement, process analysis, process change.
- We define a set of software metrics or software process metrics to allow us to measure the effectiveness of the process.

The Process Improvement Cycle



Process Improvements Activities

- Process measurement
  - You measure one or more attributes of the software process or product. These measurements forms a baseline that helps you decide if process improvements have been effective.
- Process analysis
  - The current process is assessed, and process weaknesses and bottlenecks are identified. Process models (sometimes called process maps) that describe the process may be developed.
- Process change
  - Process changes are proposed to address some of the identified process weaknesses. These are introduced and the cycle resumes to collect data about the effectiveness of the changes.

References

- Chapter 2, Software-Engineering-10th-Ian-Sommerville
- Mohamed Sami, "Software Engineering & Architecture Practices", 2012
  - https://melsatar.blog/2012/03/15/software-development-life-cycle-models-and-methodologies/
  - https://melsatar.blog/2012/03/21/choosing-the-right-software-development-life-cycle-model/

Questions