INSTITUTO SUPERIOR DE TECNOLOGIAS AVANÇADAS DE LISBOA

Software Engineering

João Caldeira

Invited Professor

Email. joaocarlos.caldeira@my.istec.pt

Mob. +351 917769544

November 2, 2022

INSTITUTO SUPERIOR
DE TECNOLOGIAS
AVANÇADAS DE LISBOA

Software Engineering Process

November 2, 2022

Software Engineering

Set of activities

- **Requirements**
  - End-User / customer business needs
- **Specification**
  - The functionality of the software and constraints on its operation must be defined
- **Design & Implementation**
  - The functionality of the software and constraints on its operation must be defined
- **Verification & Validation**
  - The software must be validated to ensure that it does what the end-user / customer wants
- **Evolution**
  - The software must evolve to meet changing customer needs

Software Engineering Process

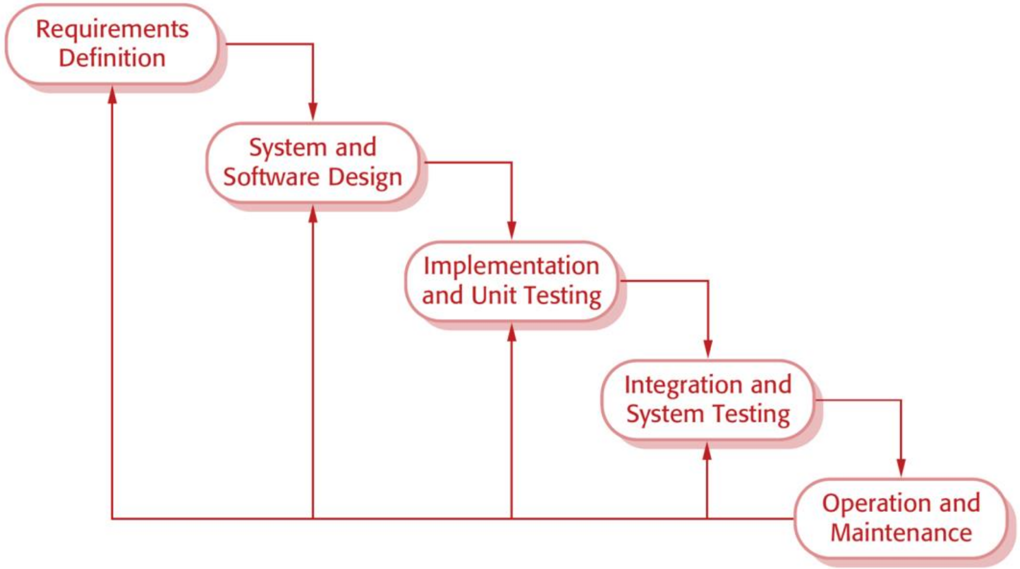
Generic Software Process Models

- **The waterfall model**
  - The fundamental process activities of specification, development, validation, and evolution are represented as separate process phases such as requirements specification, software design, implementation, testing, etc…
- **Incremental development**
  - This approach interleaves the activities of specification, development, and validation
  - The system is developed as a series of versions (increments), with each version adding functionality to the previous version
- **Integration and Configuration (Reuse-oriented software engineering)**
  - This approach is based on the existence of a significant number of reusable components
  - Focuses on integrating components into a system rather than developing from scratch

Software Engineering Process

Waterfall Model Phases

1. **Requirements analysis and definition**
   - The system's services, constraints, and goals are established by consultation with system users
2. **System and software design**
   - The systems design process allocates the requirements to either hardware or software systems by establishing an overall system architecture. Software design involves identifying and describing the fundamental software system abstractions and their relationships
3. **Implementation and unit testing**
   - Software design is realized as a set of programs or program units. Unit testing involves verifying that each unit meets its specification
4. **Integration and system testing**
   - The individual program units or programs are integrated and tested as a complete system to ensure that the software requirements have been met
5. **Operation and maintenance**
   - Normally (although not necessarily), this is the longest life cycle phase. The system is installed and put into practical use. Maintenance involves correcting errors which were not discovered in earlier stages of the life cycle


Requirements Definition
System and Software Design
Implementation and Unit Testing
Integration and System Testing
Operation and Maintenance

Software Engineering Process

Waterfall Model Problems

1. Inflexible partitioning of the project into distinct stages makes it difficult to respond to changing customer requirements.
   - One phase has to be complete before moving onto the next

2. This model is only appropriate when the requirements are well-understood and changes will be fairly limited
   - Few business systems have stable requirements

3. The waterfall model is mostly used for large systems/projects where a system is developed at several sites

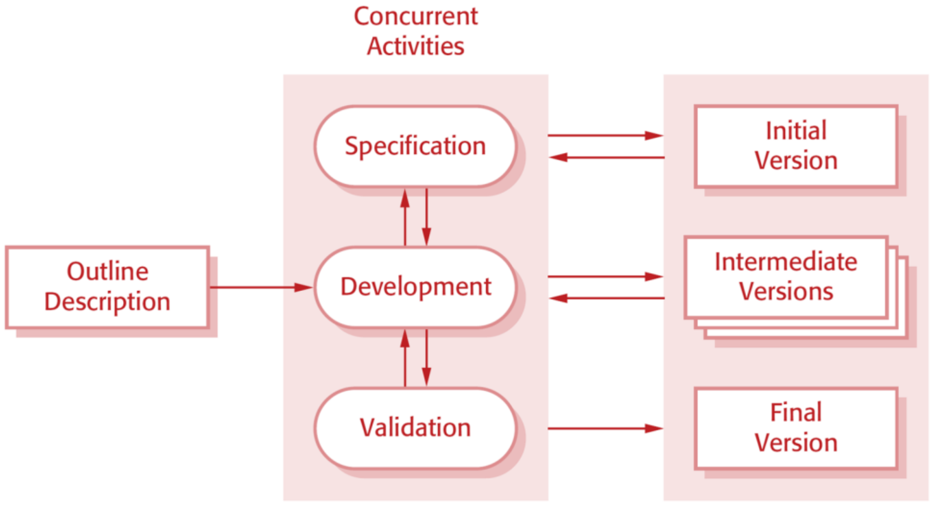
Software Engineering Process

Incremental Development Model

- Incremental development is based on the idea of developing an initial implementation, exposing this to user comment and evolving it through several versions until an adequate system has been developed

- Specification, development, and validation activities are interleaved rather than separate, with rapid feedback across activities

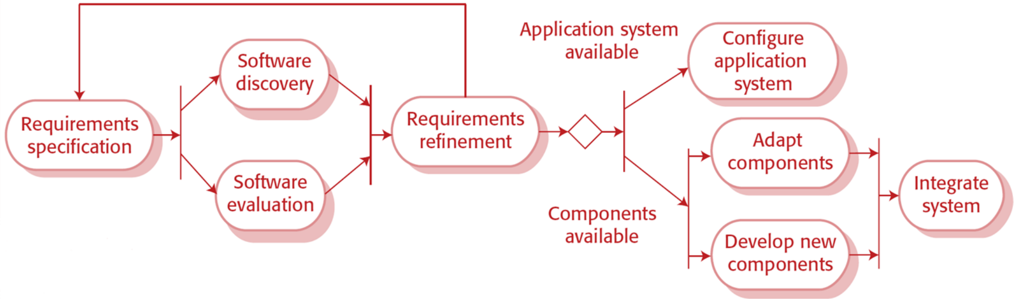
- **Benefits over Waterfall**
  - The cost of accommodating **changing customer requirements is reduced**. The amount of analysis and documentation that has to be redone is **much less than is required with the waterfall model**.
  - **It is easier to get customer feedback** on the development work that has been done. Customers can comment on demonstrations of the software and see how much has been implemented. Customers find it difficult to judge progress from software design documents
  - More rapid delivery and deployment of useful software to the customer is possible, even if all of the functionality has not been included. **Customers are able to use and gain value from the software earlier** than is possible with a waterfall process



Software Engineering Process

Integration and Configuration

- In software projects, there is some software reuse
- Developers look for these, modify them as needed, and integrate them with the new code that they have developed
- Since 2000, software development processes that focus on the reuse of existing software have become widely used
- Reuse-oriented approaches rely on a base of reusable software components and an integrating framework for the composition of these components



- **Software components frequently reused**
  - **Stand-alone application** systems that are configured for use in a particular environment. These systems are general-purpose systems that have many features, but they have to be adapted for use in a specific application.
  - **Collections of objects** that are developed as a component or as a package to be integrated with a component framework such as the **Java Spring framework**
  - **Web services** that are developed according to service standards and that are available for remote invocation over the Internet

Software Engineering

Set of activities

- **Requirements**
    - End-User / customer business needs
- **Specification**
    - The functionality of the software and constraints on its operation must be defined
- **Design & Implementation**
    - The functionality of the software and constraints on its operation must be defined
- **Verification & Validation**
    - The software must be validated to ensure that it does what the end-user / customer wants
- **Evolution**
    - The software must evolve to meet changing customer needs

Software Engineering Process

Agile Methods

- **The philosophy behind agile methods is reflected in the agile manifesto (http://agilemanifesto.org)**
- **This manifesto states:**
    - Individuals and interactions over processes and tools
    - Working software over comprehensive documentation
    - Customer collaboration over contract negotiation
    - Responding to change over following a plan

Software Engineering Process

Agile Methods

- All agile methods suggest that software **should be developed and delivered incrementally**
- These methods are based on different agile processes but they share a set of principles, based on the agile manifesto, and so they have much in common.

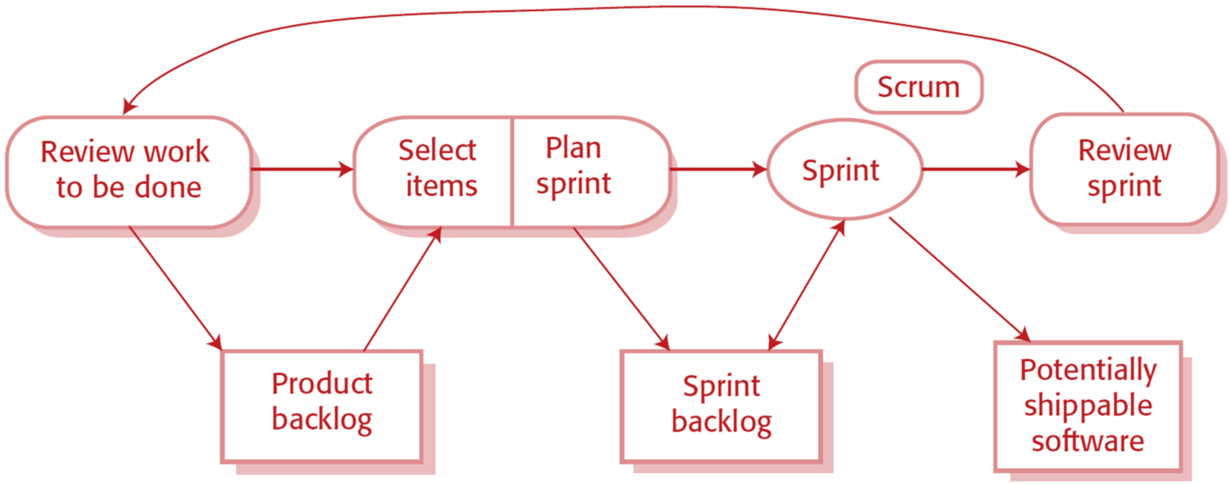
- Agile methods have been particularly successful for two kinds of system development:

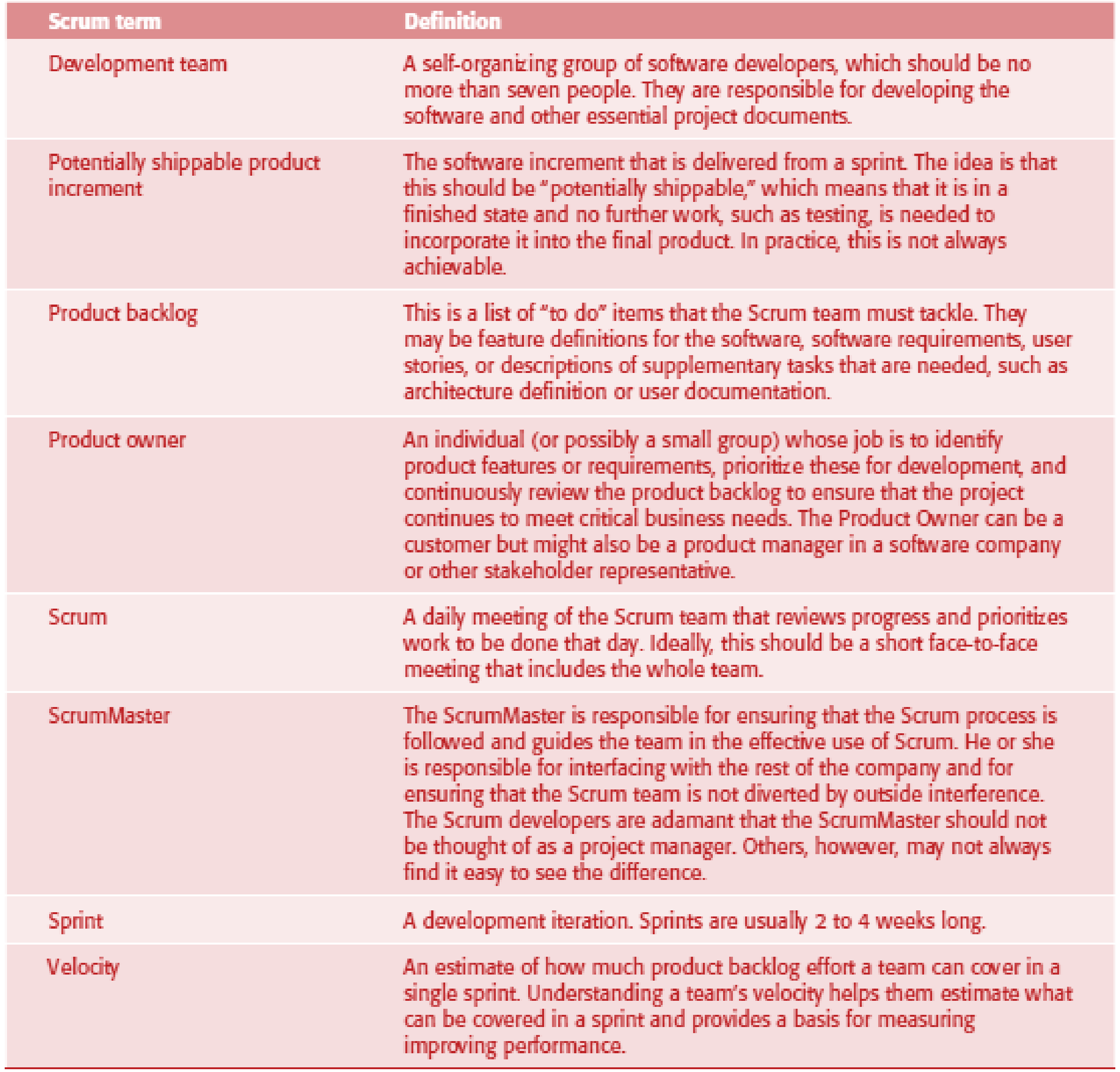
  1. Product development where a software company is developing a small or medium-sized product for sale. Virtually all software products and apps are now developed using an agile approach

  2. Custom system development within an organization, where there is a clear commitment from the customer to become involved in the development process and where there are few external stakeholders and regulations that affect the software

Software Engineering Process

Agile Project Management





Scrum term	Definition
Development team	A self-organizing group of software developers, which should be no more than seven people. They are responsible for developing the software and other essential project documents.
Potentially shippable product increment	The software increment that is delivered from a sprint. The idea is that this should be "potentially shippable," which means that it is in a finished state and no further work, such as testing, is needed to incorporate it into the final product. In practice, this is not always achievable.
Product backlog	This is a list of "to do" items that the Scrum team must tackle. They may be feature definitions for the software, software requirements, user stories, or descriptions of supplementary tasks that are needed, such as architecture definition or user documentation.
Product owner	An individual (or possibly a small group) whose job is to identify product features or requirements, prioritize these for development, and continuously review the product backlog to ensure that the project continues to meet critical business needs. The Product Owner can be a customer but might also be a product manager in a software company or other stakeholder representative.
Scrum	A daily meeting of the Scrum team that reviews progress and prioritizes work to be done that day. Ideally, this should be a short face-to-face meeting that includes the whole team.
ScrumMaster	The ScrumMaster is responsible for ensuring that the Scrum process is followed and guides the team in the effective use of Scrum. He or she is responsible for interfacing with the rest of the company and for ensuring that the Scrum team is not diverted by outside interference. The Scrum developers are adamant that the ScrumMaster should not be thought of as a project manager. Others, however, may not always find it easy to see the difference.
Sprint	A development iteration. Sprints are usually 2 to 4 weeks long.
Velocity	An estimate of how much product backlog effort a team can cover in a single sprint. Understanding a team's velocity helps them estimate what can be covered in a sprint and provides a basis for measuring improving performance.

Software Engineering Process

Problems of Agile Methods

- **For large, long-lifetime systems that are developed by a software company for an external client, using an agile approach presents a number of problems :**
  - The informality of agile development **is incompatible with the legal approach to contract** definition that is commonly used in large companies
  - Agile methods **are most appropriate for new software development rather than for software maintenance**. Yet the majority of **software costs in large companies come from maintaining** their existing software systems
  - Agile methods **are designed for small co-located teams**, yet much software development **now involves worldwide distributed teams**

INSTITUTO SUPERIOR DE TECNOLOGIAS AVANÇADAS DE LISBOA

Software Engineering

João Caldeira

Invited Professor

Email. joaocarlos.caldeira@my.istec.pt

Mob. +351 917769544

November 2, 2022