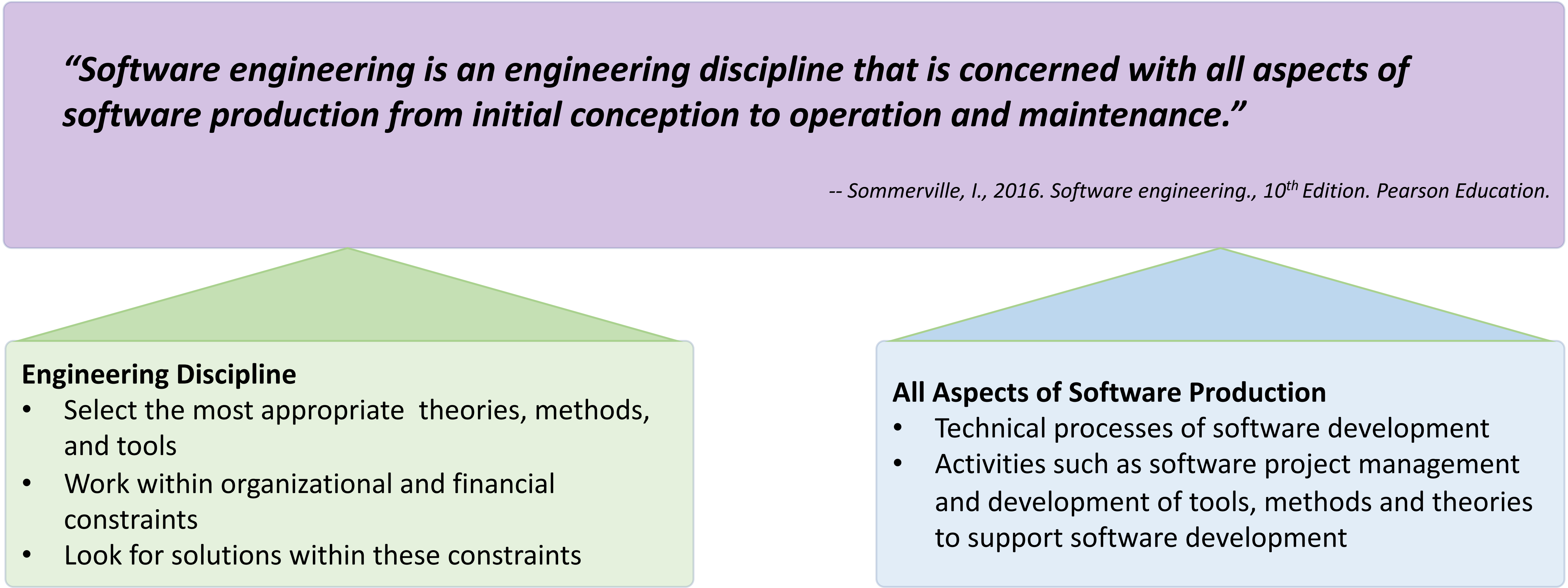
CS335, 2021-2022

Introduction to Software Engineering

Dapeng Dong

What is Software Engineering?



“Software engineering is an engineering discipline that is concerned with all aspects of software production from initial conception to operation and maintenance.”

-- Sommerville, I., 2016. Software engineering., 10th Edition. Pearson Education.

Engineering Discipline

- Select the most appropriate theories, methods, and tools
- Work within organizational and financial constraints
- Look for solutions within these constraints

All Aspects of Software Production

- Technical processes of software development
- Activities such as software project management and development of tools, methods and theories to support software development

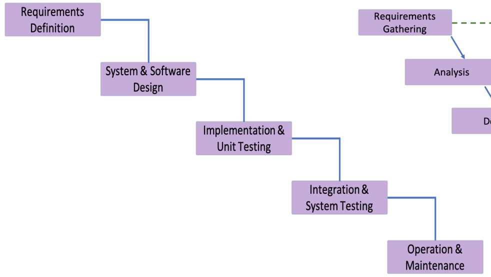
Reading (Software Engineering): *Naur, P., 1968. Software engineering-report on a conference sponsored by the NATO Science Committee Garimisch, Germany. NATO,1968.*

What is Software Process?

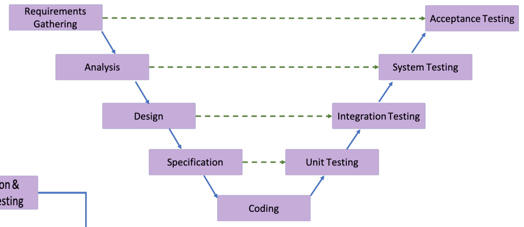
> *"The systematic approach that is used in software engineering is sometimes called a software process. A software process is a sequence of activities that leads to the production of a software product."*
>
> *-- Sommerville, I., 2016. Software engineering., 10th Edition. Pearson Education.*

A simplified and abstracted representation of a software process is called a software process model.

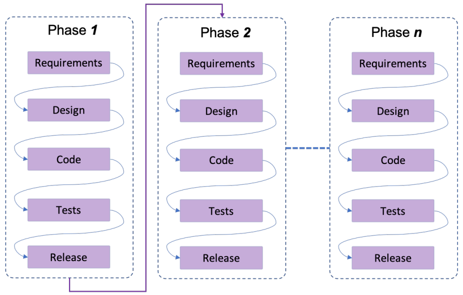
- A software process model is often called a Software Development Life Cycle (SDLC) model
- Depending on the type of software application to be developed, different software process models may be used
- E.g., V-model, Spiral model, RAD (Rapid Application Development) model, Agile model, Waterfall model, Increment model, etc.



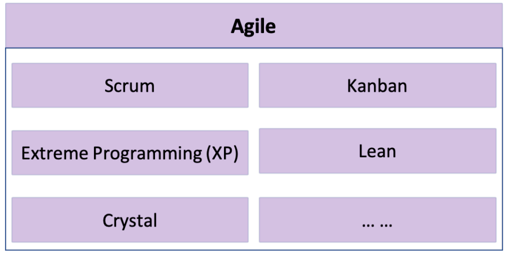
Waterfall Model



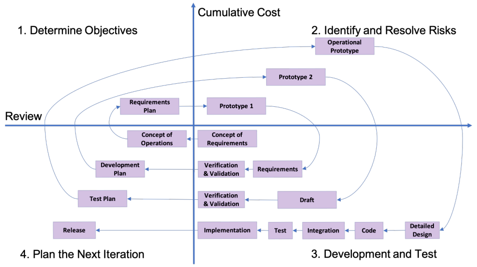
V-Model



Incremental Model



Agile Model

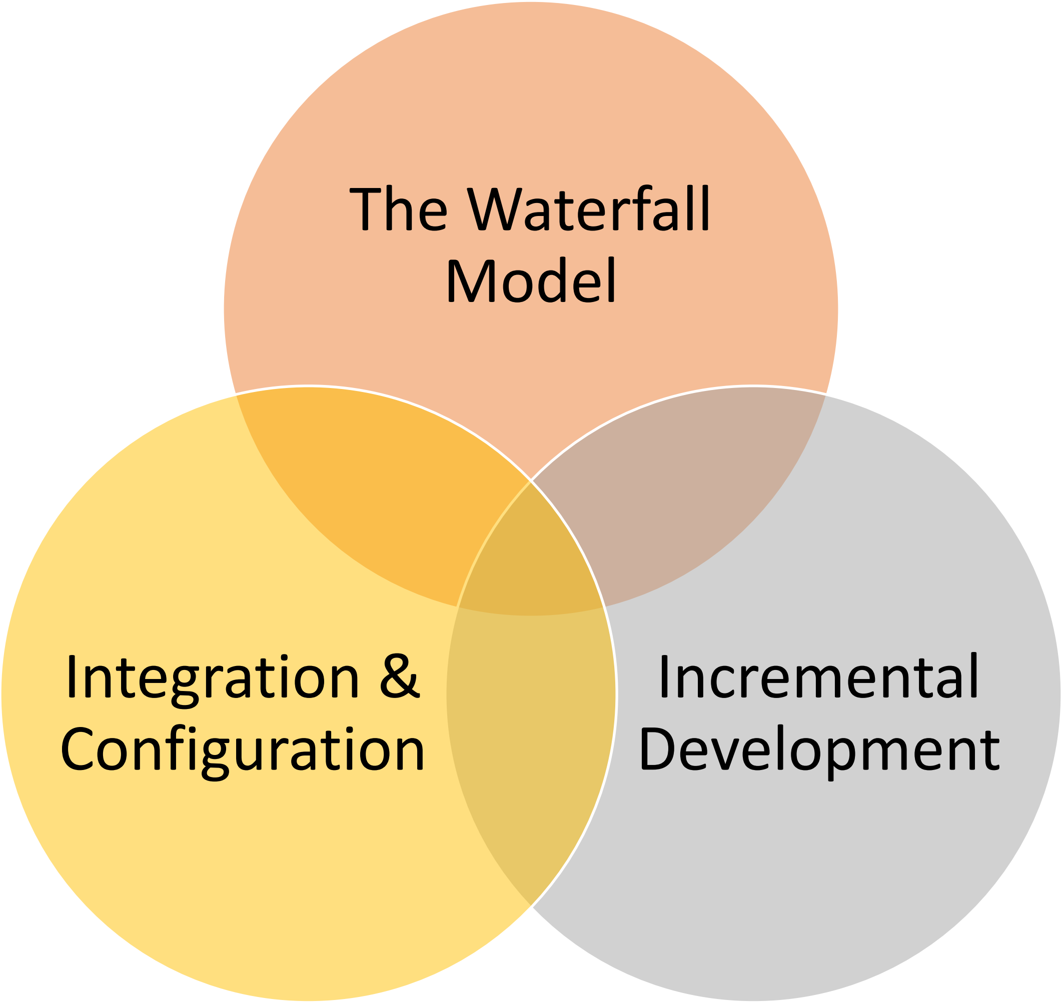


Spiral Model

Software Paradigms

“These generic models are high-level, abstract descriptions of software processes that can be used to explain different approaches to software development.”

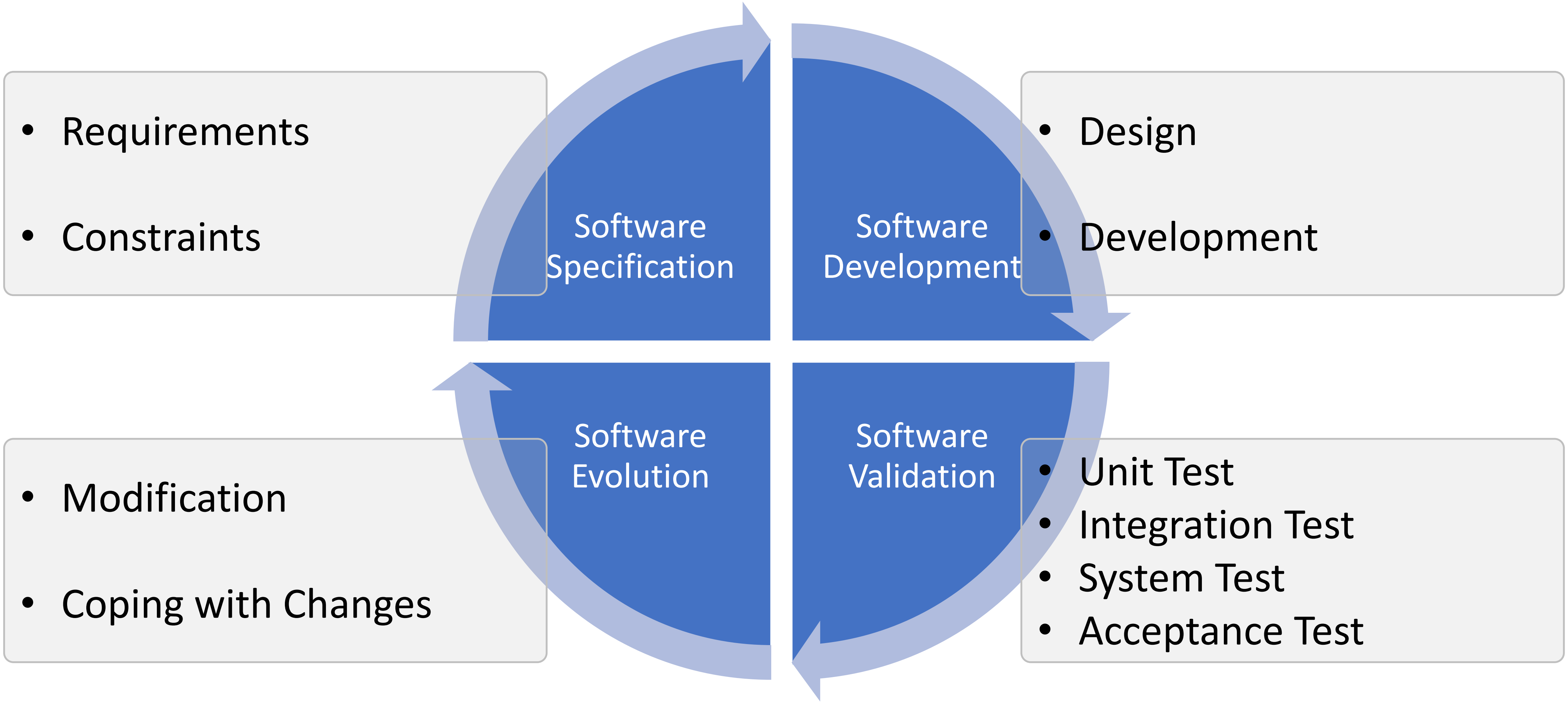
-- Sommerville, I., 2016. Software engineering., 10th Edition. Pearson Education.



Process Paradigms

Each of them can be extended and adapted to create more specific software engineering processes. They can also be seen as process frameworks.

Fundamental Activities for Software Processes



- Requirements
- Constraints

Software Specification

Software Development

- Design
- Development

Software Evolution

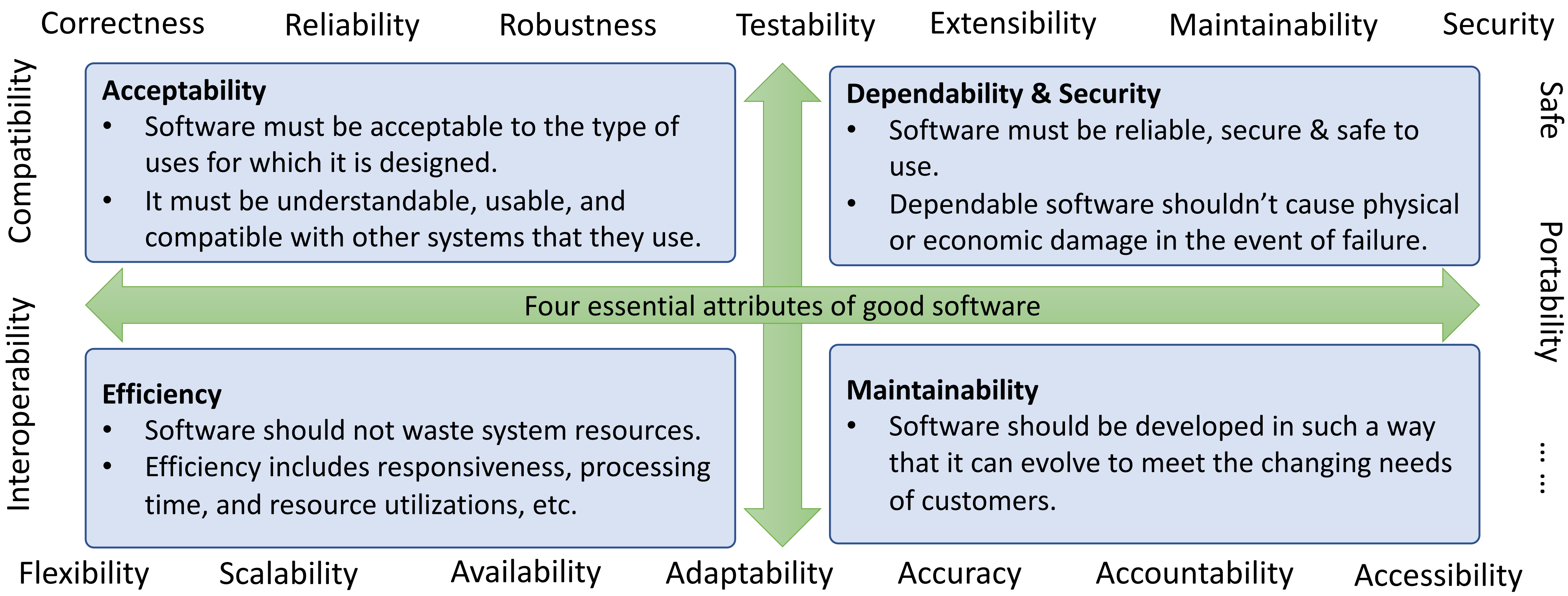
- Modification
- Coping with Changes

Software Validation

- Unit Test
- Integration Test
- System Test
- Acceptance Test

The four fundamental activities that are common to all software processes

Quality of Software Product



Correctness Reliability Robustness Testability Extensibility Maintainability Security

Compatibility

Interoperability

Acceptability
- Software must be acceptable to the type of uses for which it is designed.
- It must be understandable, usable, and compatible with other systems that they use.

Dependability & Security
- Software must be reliable, secure & safe to use.
- Dependable software shouldn’t cause physical or economic damage in the event of failure.

Four essential attributes of good software

Efficiency
- Software should not waste system resources.
- Efficiency includes responsiveness, processing time, and resource utilizations, etc.

Maintainability
- Software should be developed in such a way that it can evolve to meet the changing needs of customers.

Safe

Portability

…

Flexibility Scalability Availability Adaptability Accuracy Accountability Accessibility

Software quality is not just concerned with what the software does, it also includes the software’s behavior while it is running and the structure and organization of the system programs and associated documentation.

Types of Software (1)

- Stand-alone applications
  - Applications that run on personal computers, mobile devices or mainframes
  - For example, Adobe Photoshop, Windows Calculator, etc.
- Interactive transaction-based applications
  - Applications that run on a remote computer, but accessed by users from their own computers
  - For example, Web applications, eBay Apps, cloud-based services, etc.
- Embedded control systems
  - Software that control and manage hardware devices such as refrigerator temperature control, microware cooking functions, oil pump control, etc.

Types of Software(2)

- Batch processing systems
  - Application systems that process data in large batches
  - For example, cell phone billing systems, staff salary payment systems, bank transaction processing systems, etc.
- Entertainment systems
  - Most of these systems are games that can run on personal computers or special console hardware, e.g., Xbox, PS4, Nintendo Switch, etc.
- System for modeling and simulation
  - Application systems developed for scientists and engineers to model and simulate physical processes, chemical reactions, protein folding, visualization, etc.

sage

Payroll

HALO

INFINITE

OpenSim

Types of Software(3)

- Data collection and analysis systems
  - Software systems that collect data from their environment and send that data to other systems for processing
  - For example, data warehouse, data lake, big data analytics systems, sensor data processing systems, etc.
- Systems of systems
  - Systems or subsystems used in the large-scale enterprise or organizations.
  - For example, Enterprise Resource Planning (ERP) system, or systems that are composed from other discrete systems.

NOTE: there are on clear boundaries between these types of software. As the world is becoming more and more software controlled, other types of software may emerge, thus new software processes are likely to be developed.

The Importance of Software Engineering

To be able to produce reliable and trustworthy systems economically and quickly.

It is usually cheaper, in the long run, to use software engineering methods and techniques.

- How to plan software development activities across a long term?
  - E.g., The electronic trading platform project (TAURUS) for UK stock exchange lasted for more than 14 years
- How to allocate budget?
  - E.g., The air traffic control project (FAA Advanced Automation System) for US federal aviation administration received 3 – 6 Billion US dollars
- How to deal with hundreds and thousands of the requirements gathered from users? And how to prioritize them?
- How to deal with conflict requirements from different stakeholders?
- How to design an architecture that can scale to thousands of servers concurrently?
- How to manage hundreds and thousands of source code files and their dependencies?
- How to test, deploy, configure, manage software products?
- How to …

Software Development Failures

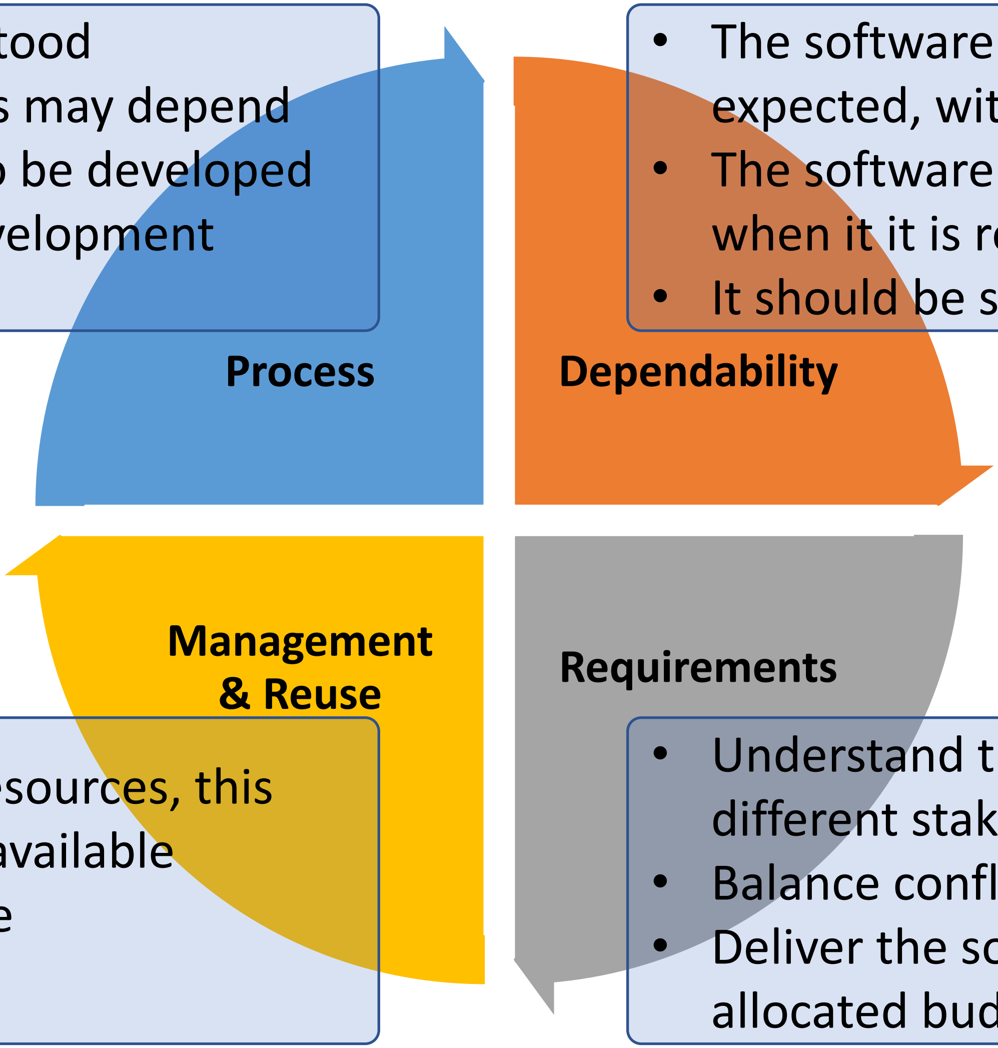
Date	Project Name	Type of System	Country	Problem	Expected Cost	Result
1984 – 1994	FAA Advanced Automation System	Air Traffic Control	USA	Cost overrun, Too complex to implement	$3 – 6 Billion	Scrapped
2005 – 2012	Expeditionary Combat Support System	Air Force	USA	Time overrun, cost overrun, too complex to implement	$1.1 Billion	Cancelled
2008 – 2013	Digital Media Initiative	State Broadcaster	UK	Time overrun,	£98 Million	Cancelled
2009 – 2013	SIREN	Police Force	UK	Not fit for purpose	£14.8 Million	Scrapped

NOTE: Inappropriate use of software engineering methods can lead to higher costs for testing, quality assurance, and long-term maintenance.

Software Engineering Fundamentals

- Use managed and understood development process, this may depend on the type of software to be developed
- Organize and plan the development process

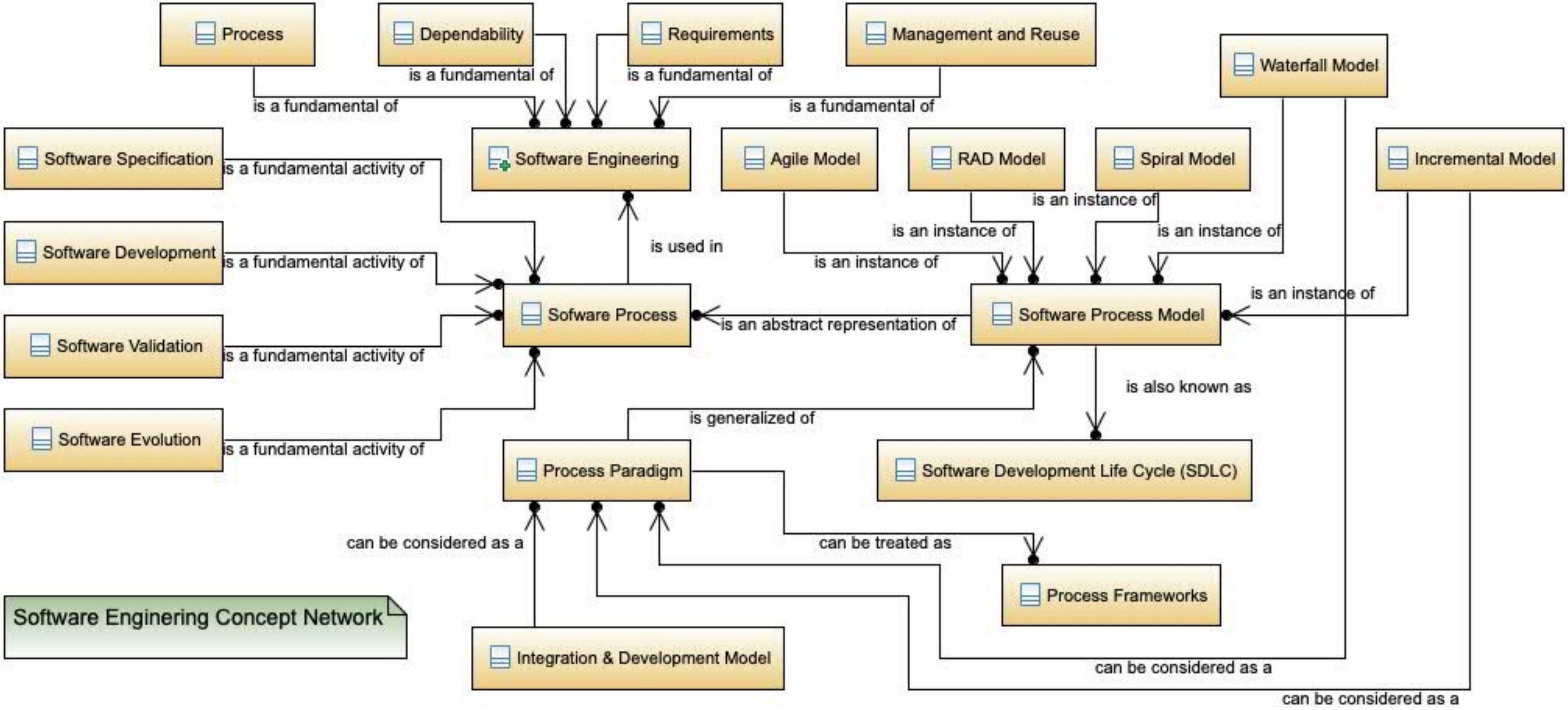
- The software system should behave as expected, without failures
- The software system should be available when it it is required
- It should be safe and secure



- Effective use of existing resources, this may include staffs, skills, available technologies and software

- Understand the expectations from different stakeholders
- Balance conflict requirements
- Deliver the software system within the allocated budget and time

Concept Map


Process
Dependability
Requirements
Management and Reuse
Waterfall Model
is a fundamental of
is a fundamental of
is a fundamental of
is a fundamental of
Software Specification
Software Engineering
Agile Model
RAD Model
Spiral Model
Incremental Model
is a fundamental activity of
is an instance of
Software Development
is a fundamental activity of
is used in
is an instance of
is an instance of
is an instance of
is an instance of
Sofware Process
is an abstract representation of
Software Process Model
Software Validation
is a fundamental activity of
is also known as
Software Evolution
is a fundamental activity of
is generalized of
Process Paradigm
Software Development Life Cycle (SDLC)
can be considered as a
can be treated as
Process Frameworks
Software Enginering Concept Network
Integration & Development Model
can be considered as a
can be considered as a