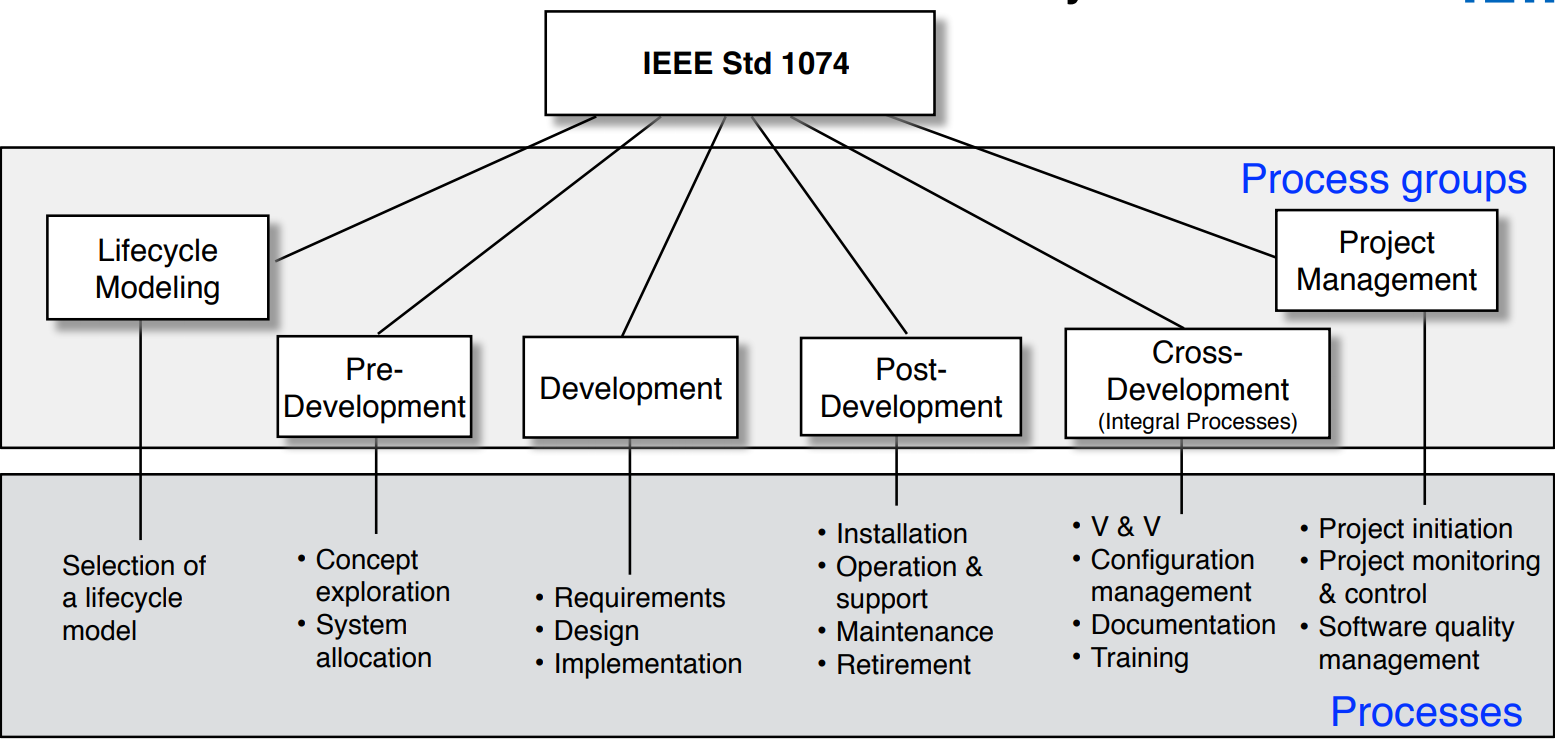
**POM 04**

* The IEEE 1058 standard describes the structure of a software project management plan, applicable to any type or size



* What it does:
  + Specifies the format and contents of software project management plans
  + It provides a standard set of abstractions for a project manager or a whole organization to build its set of practices and procedures for developing software project management plans

**Two styles of navigation [Gladwin 1964]**

1) European navigation (Traditional Planning)

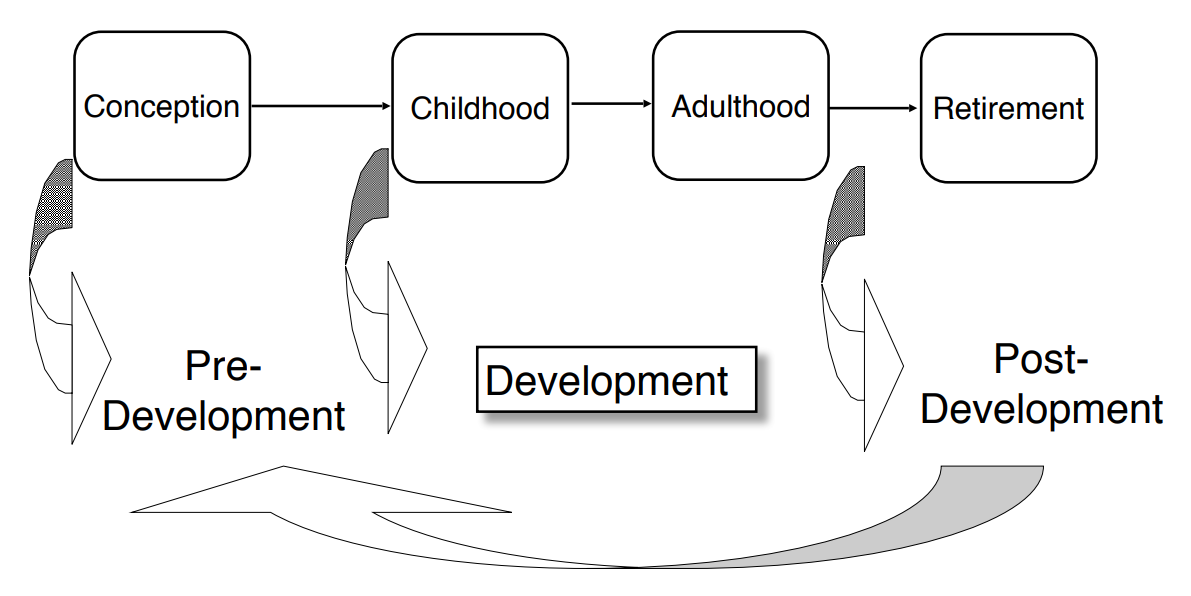
2) Polynesian navigation (Agile Planning)

➡Main difference: reaction to unexpected events (“change”)

**Properties of software project plans**

* Useful at the beginning of a project
* Useful also for projects if the outcome is predictable or when no major change occurs
* Problematic when the outcome is unpredictable or when unexpected (“unusual”) events occur that change the project context

**Software lifecycle**



* **Software lifecycle** (also software process): Set of activities and their relationships to each other to support the development of a software system
* **Software lifecycle model** (also software process model): An abstraction that represents a software lifecycle for the purpose of understanding, monitoring, or controlling the development of a software system.

**Where do we need models?**

* Communication: The model provides a common vocabulary. An informal model is often used to communicate an idea
* Analysis/Design: Models enable developers to reason about the future system
* Archival: Compact representation for storing the design and rationale of an existing system (Documentation)

**Support communication**

* Most often used in the early phases of a project and during informal communication
* The model is used only to communicate an idea to a person
* Communication media: whiteboard, a mockup, or even a napkin design

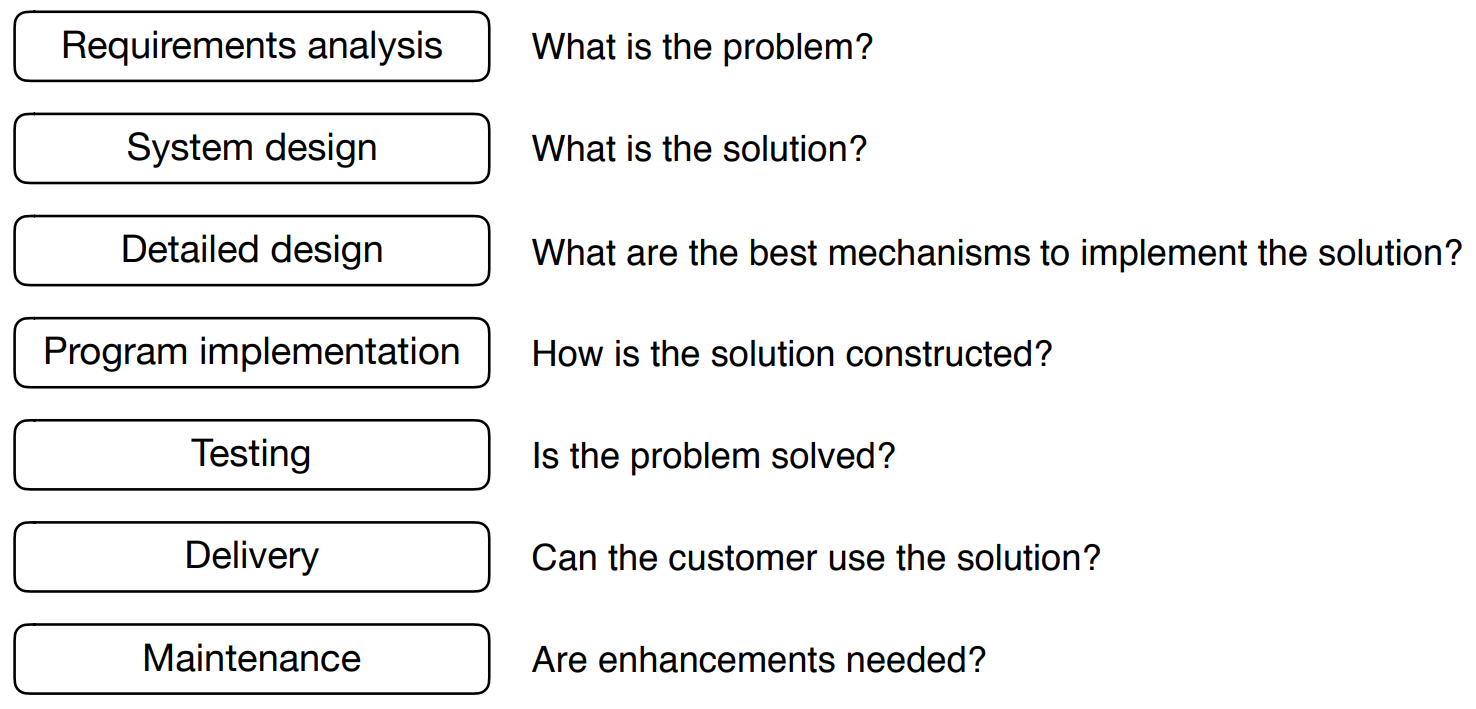
**Support analysis and design**

* The model provides a representation that enables developers to reason about the system
* The model is used to communicate an idea to a tool
* UML is our preferred notation for models that support analysis and design

**Managerial challenges of modeling**

* Formalizing knowledge is expensive
  + Takes time and effort from developers
  + Requires validation and consensus
* Models introduce redundancy
  + If the system is changed, the models must be changed
  + If several models depict (abbilden) the same aspects of the system, all of them must be updated
  + If one model becomes out of sync, it loses its value
* Models can become too complex
  + As the model complexity becomes similar to the complexity of the system, the benefit of having a model is reduced significantly

**Software development activities (Example)**



**Tailoring**

* There is no “one size fits all” software lifecycle model that works for all possible software engineering projects
* Tailoring: adjusting a lifecycle model to fit a project
* Naming: adjusting the naming of activities
* Cutting: removing activities not needed in the project
* Ordering: defining the order the activities take place in

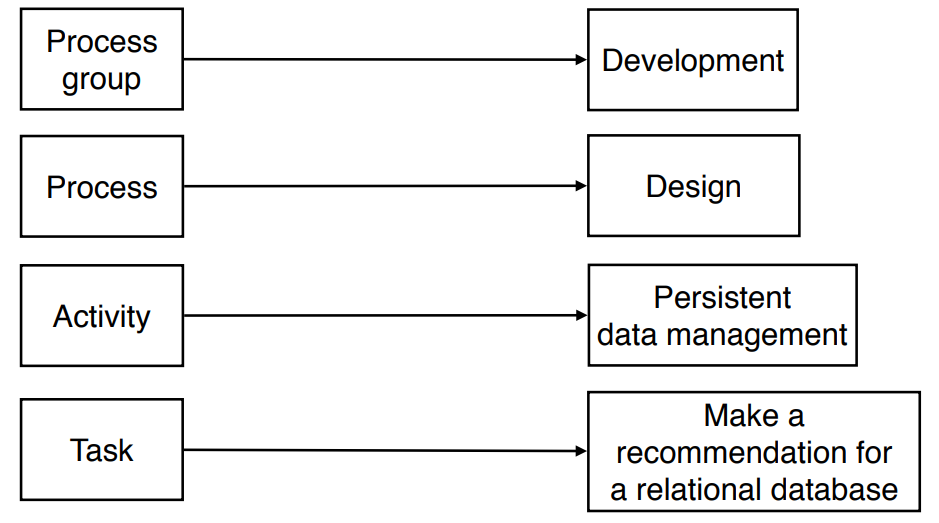
**Modeling a software lifecycle (POM 04 ab Folie 37)**

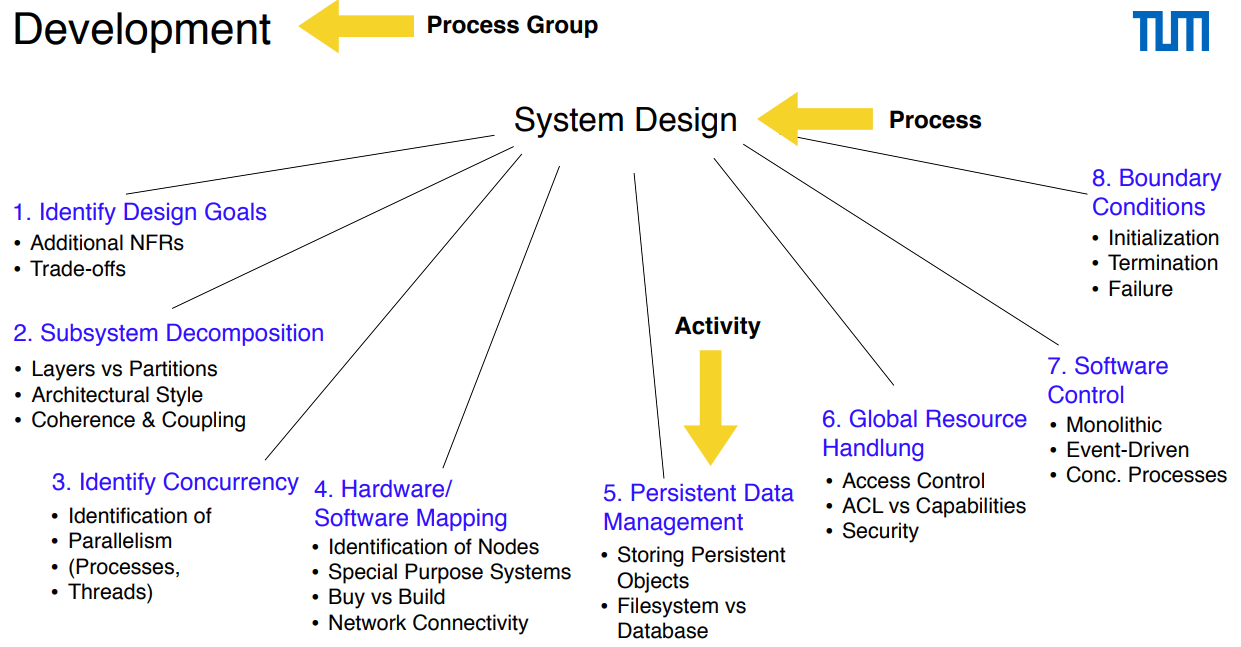
* Functional model of a software lifecycle
  + Scenarios, user stories
  + Use case model
* Structural model of a software lifecycle
  + Object identification
  + Class diagrams
* Dynamic model of a software lifecycle
  + Sequence diagrams, state chart and activity diagrams

**ZWISCHEN 37 und 44 WOAS NET OB EINI TIAN**

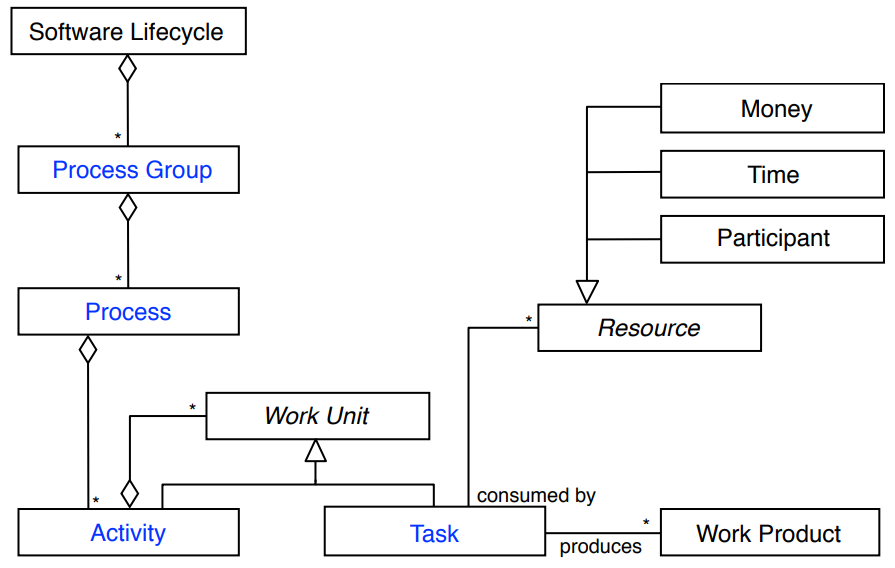
Process groups, processes, activities and tasks

* Process group: consists of a set of processes
* Process: consists of activities
* Activity: consists of sub activities (phases, steps,…) and tasks



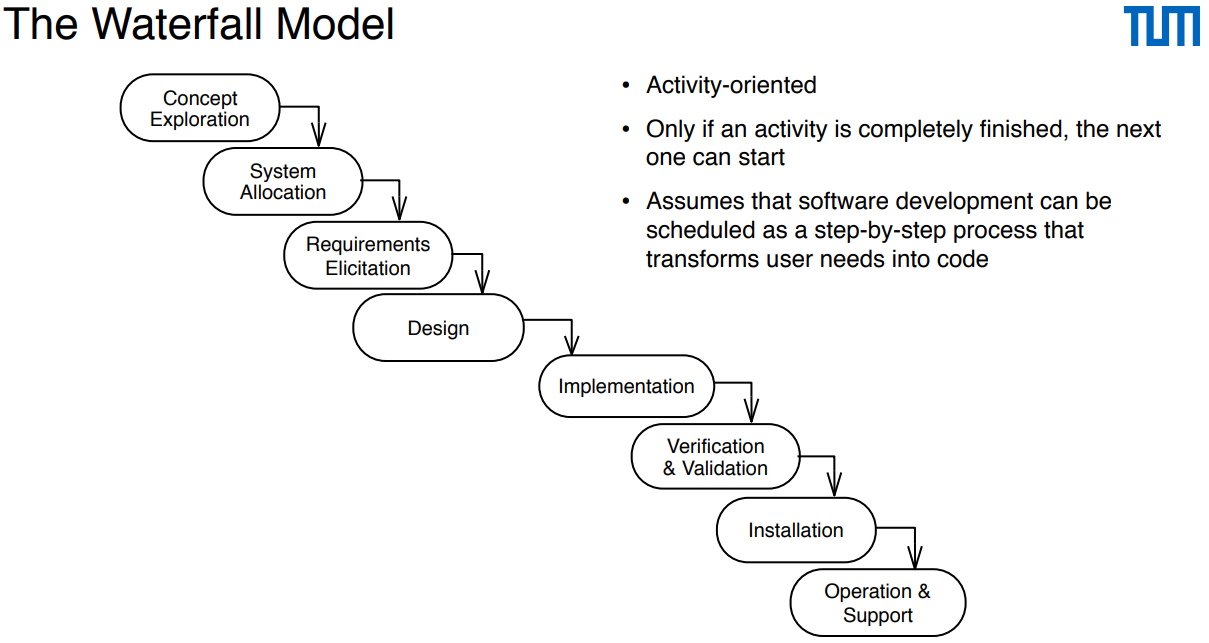


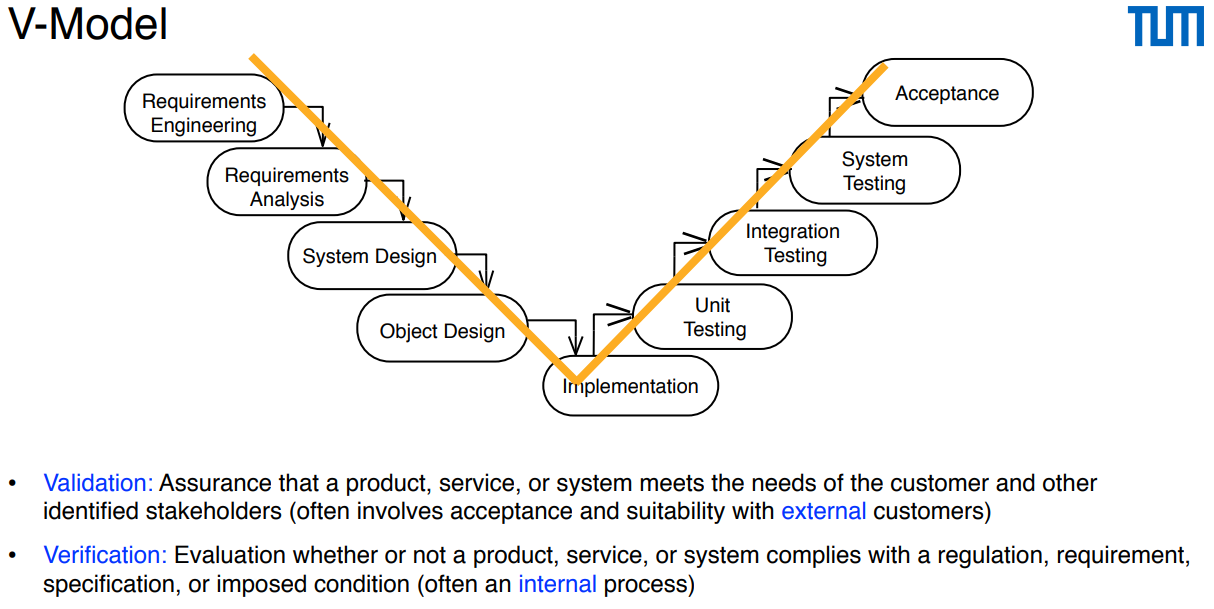
**Object model of the IEEE 1074 standard**

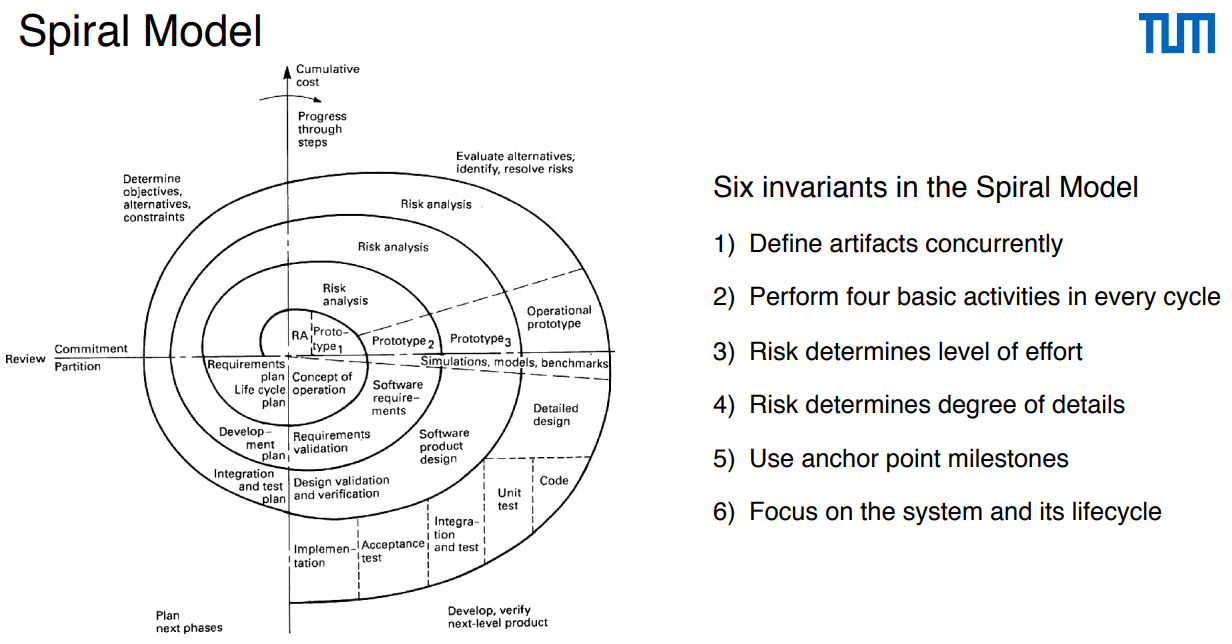


**Overview of software lifecycle models**

* Sequential
  + Waterfall Model
  + V-Model
* Iterative
  + Spiral Model
  + V-Model XT
  + Unified Process
* Agile
  + Extreme Programming (XP)
  + Kanban
  + Scrum







**V-Model XT**

• The V-Model XT is the successor of the V-Model

• Standard for German government contracts

• Allows for tailoring and iterations

• Also incorporates duties of the client

• Goals:

• Minimization of project risk

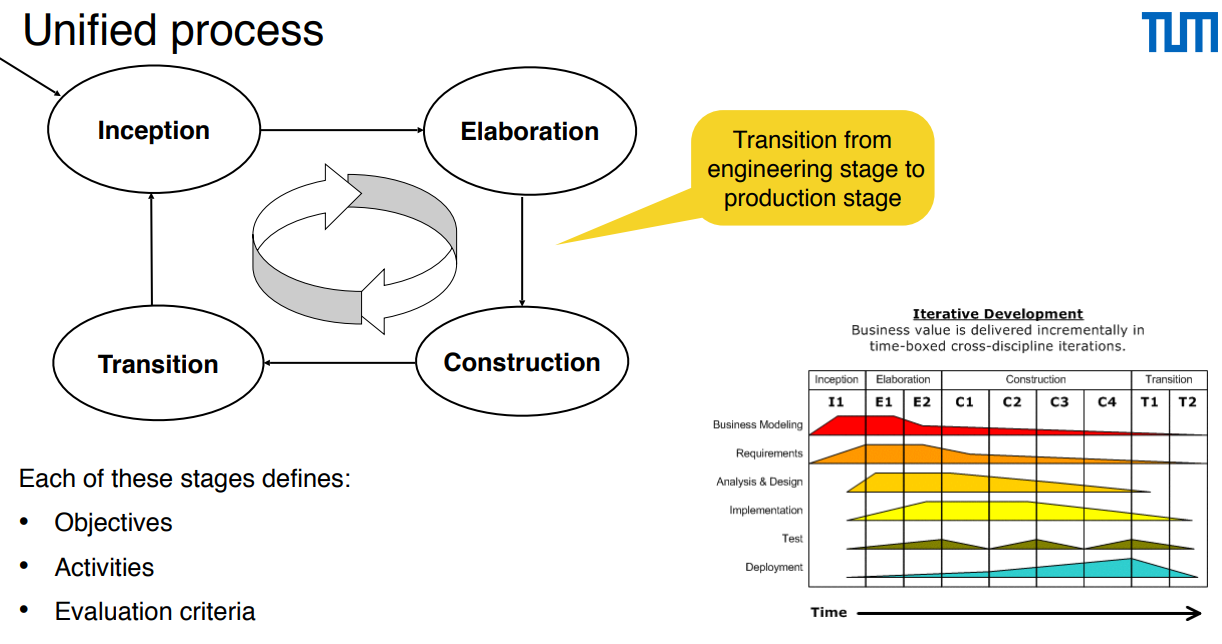
• Improvement and guarantee of quality

• Reduction of cost

• Improvement of communication.

• Focus on products instead of activities

• No defined chronology of activities



**Extreme programming (XP) (??? Nur geschichtliche Infos)**

• First use of XP in the Chrysler Comprehensive Compensation project (C3 Project) in 1995

• Product release in 1997

• Lots of initial excitement followed by a lot of problems

• Mostly performance problems: 1000 hours to run the monthly payroll

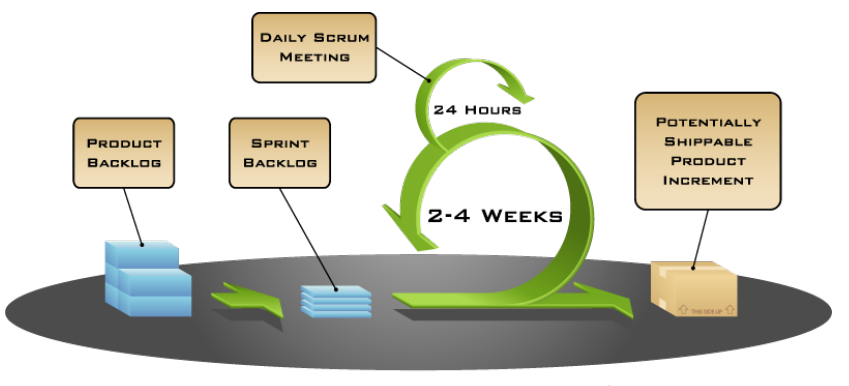
• Daimler shut down the C3 Project in 2000 and even banned XP for some time [Hendrickson 2001].

**Kanban**

* Start with existing process
* Agree to pursue incremental, evolutionary change
* Respect the current process, roles, responsibilities and titles
* Leadership at all levels

**Eigene Defintion:** Kanban ist eine agile Methode für evolutionäres Change Management. Das bedeutet, dass der bestehende Prozess in kleinen Schritten (evolutionär) verbessert wird. Indem viele kleine Änderungen durchgeführt werden (anstatt einer großen), wird das Risiko für jede einzelne Maßnahme reduziert.

Scrum



• Management and control process

• Focus on early risk management

• Deliver working software, incrementally and empirically (auf Erfahrung)

• Simple framework (rahmen) for effective team collaboration

**Summary**

* A software lifecycle model (also software process model) represents a software lifecycle (concrete instance of a project)
* A software lifecycle model has the purpose to understand, monitor, or control the development of a software system
* A software lifecycle model consists of activities and their relationship
* We use UML activity diagrams to model software lifecycles