Universitas Esa Unggul

Smart, Creative and Entrepreneurial

CPL230-PENGEMBANGAN PERANGKAT LUNAK (PERTEMUAN-3)

Dosen Pengampu :

5165-Kundang K Juman

Prodi Teknik Informatika Fakultas Ilmu Komputer

www.esaunggul.ac.id

SOFTWARE DEVELOPMENT METHODOLOGIES

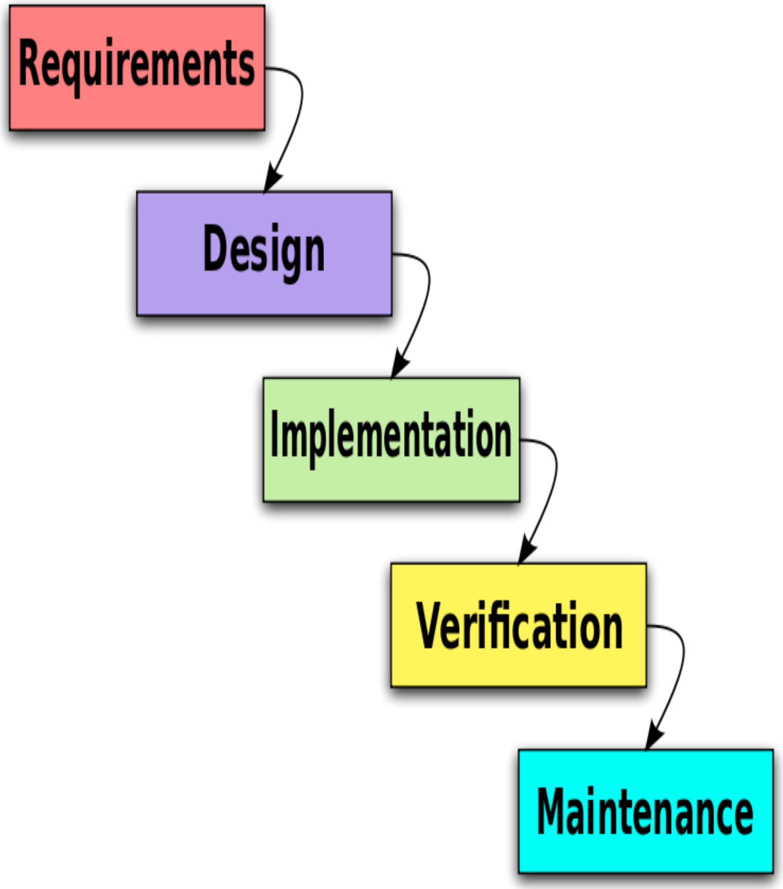
2013.04.30

Methodologies

- Waterfall
- Prototype model
- Incremental
- Iterative
- V-Model
- Spiral
- Scrum
- Cleanroom
- RAD
- DSDM
- RUP
- XP
- Agile
- Lean
- Dual Vee Model
- TDD
- FDD

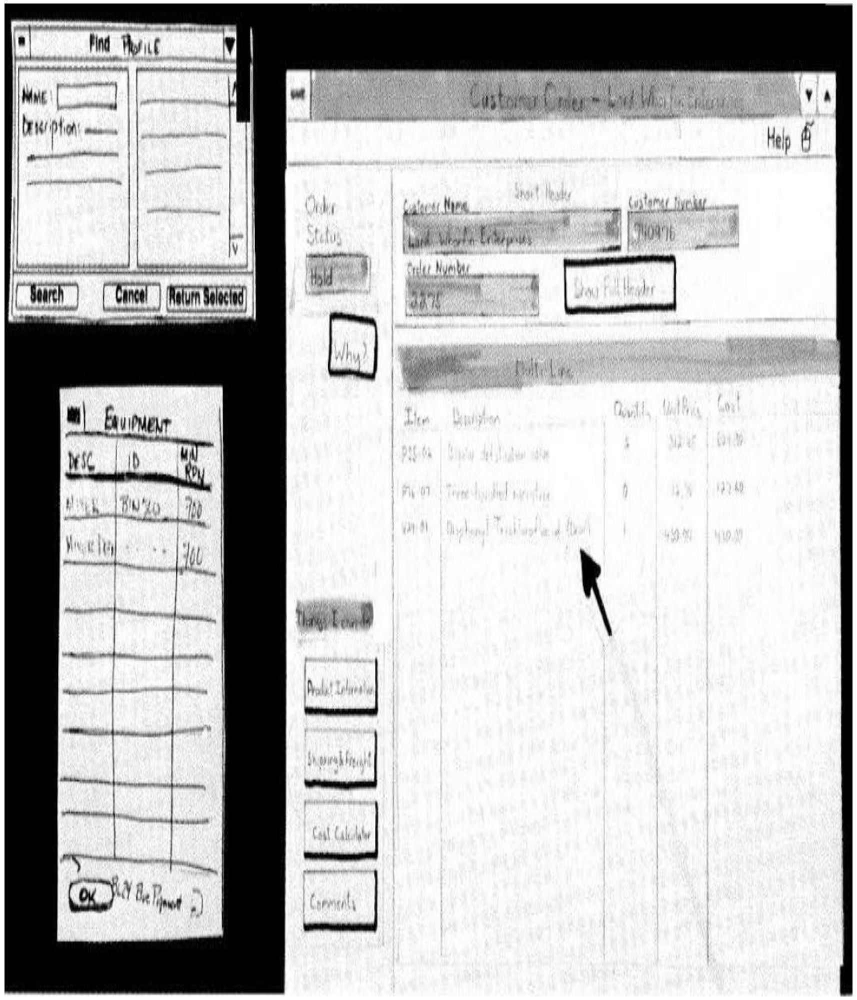
Waterfall

- Sequential design process
- Progress is seen as flowing steadily downwards (like a waterfall) through SDLC



Prototyping

- Creating prototypes of software applications i.e. incomplete versions of the software program being developed
- A prototype typically simulates only a few aspects of, and may be completely different from, the final product.

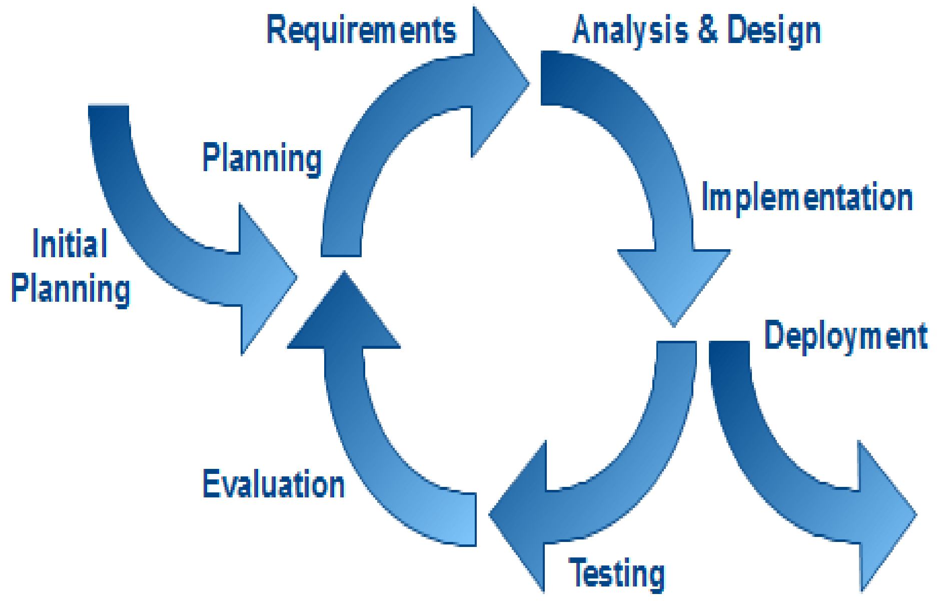


Incremental Build Model

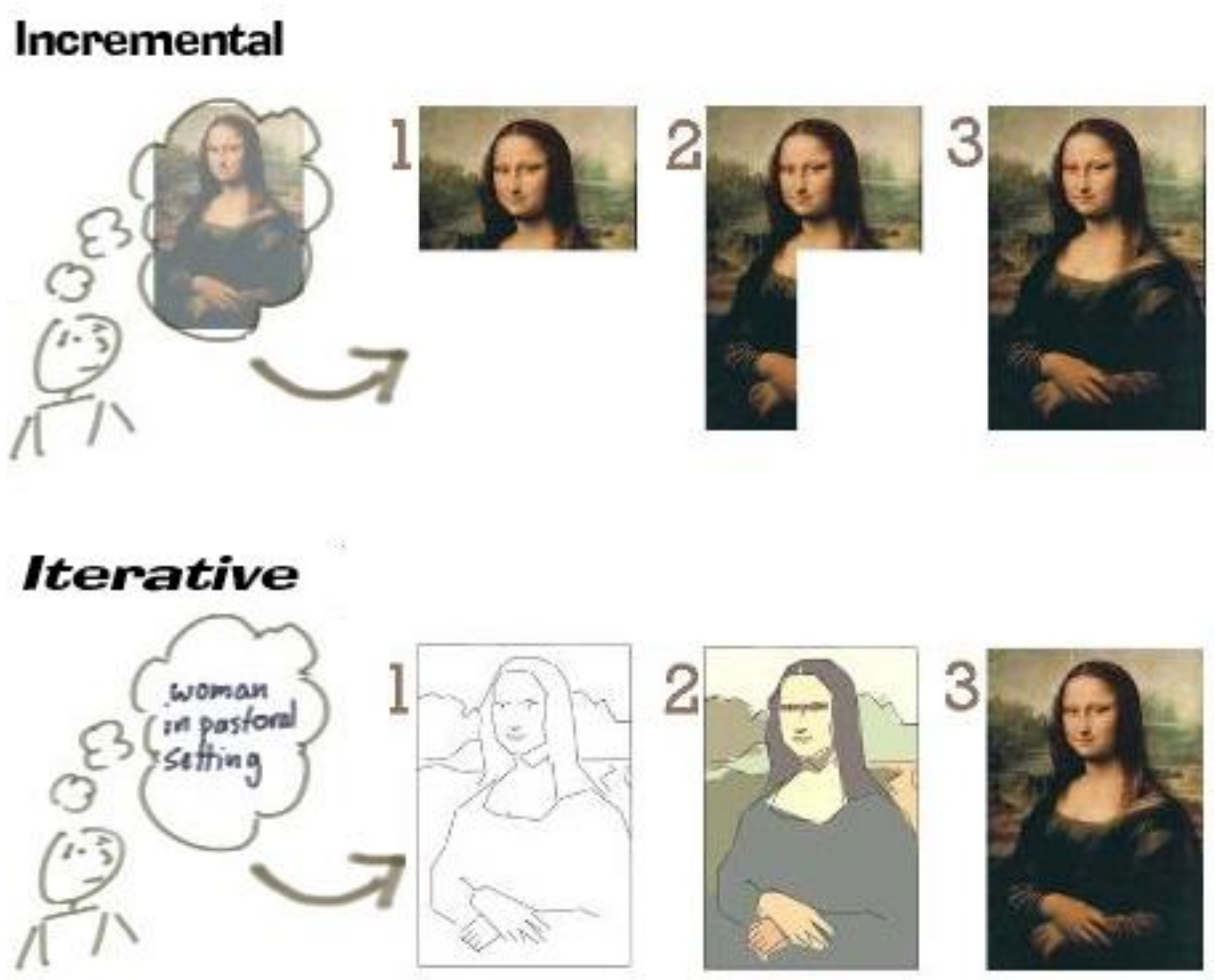
- The model is designed, implemented and tested incrementally (a little more is added each time).
- Finished when satisfies all the requirements.
- Combines the elements of the waterfall model with the iterative philosophy of prototyping.

Iterative and Incremental Development

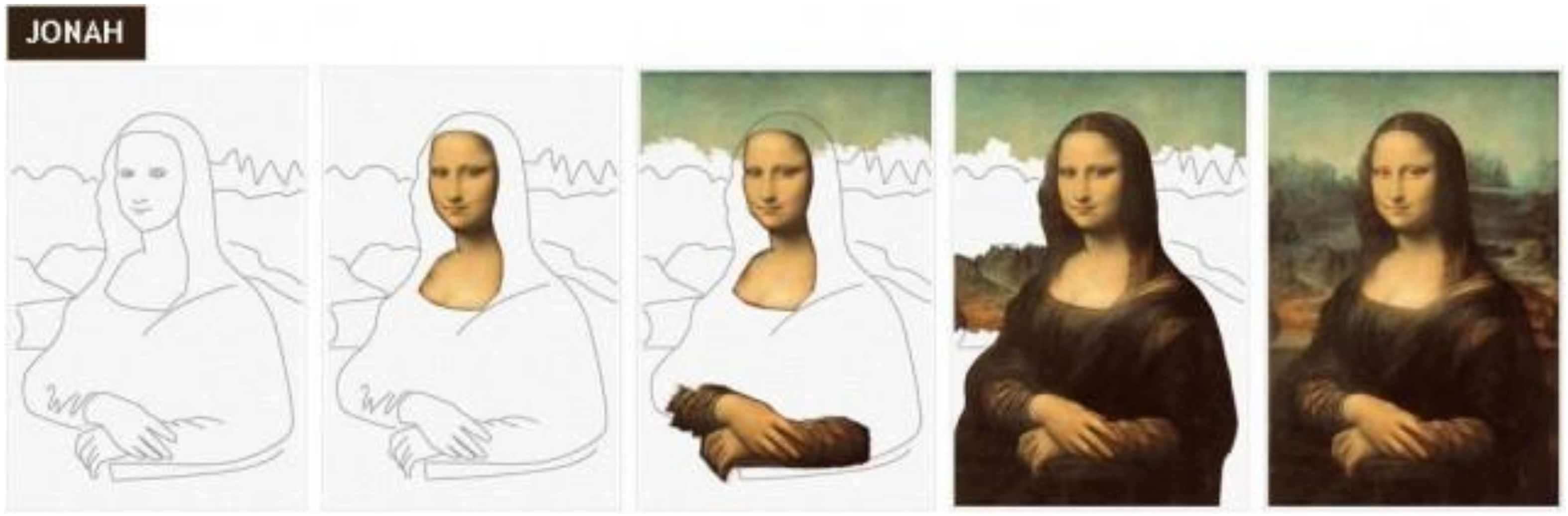
- Iterative and incremental development is any combination of both iterative design or iterative method and incremental build model for development.



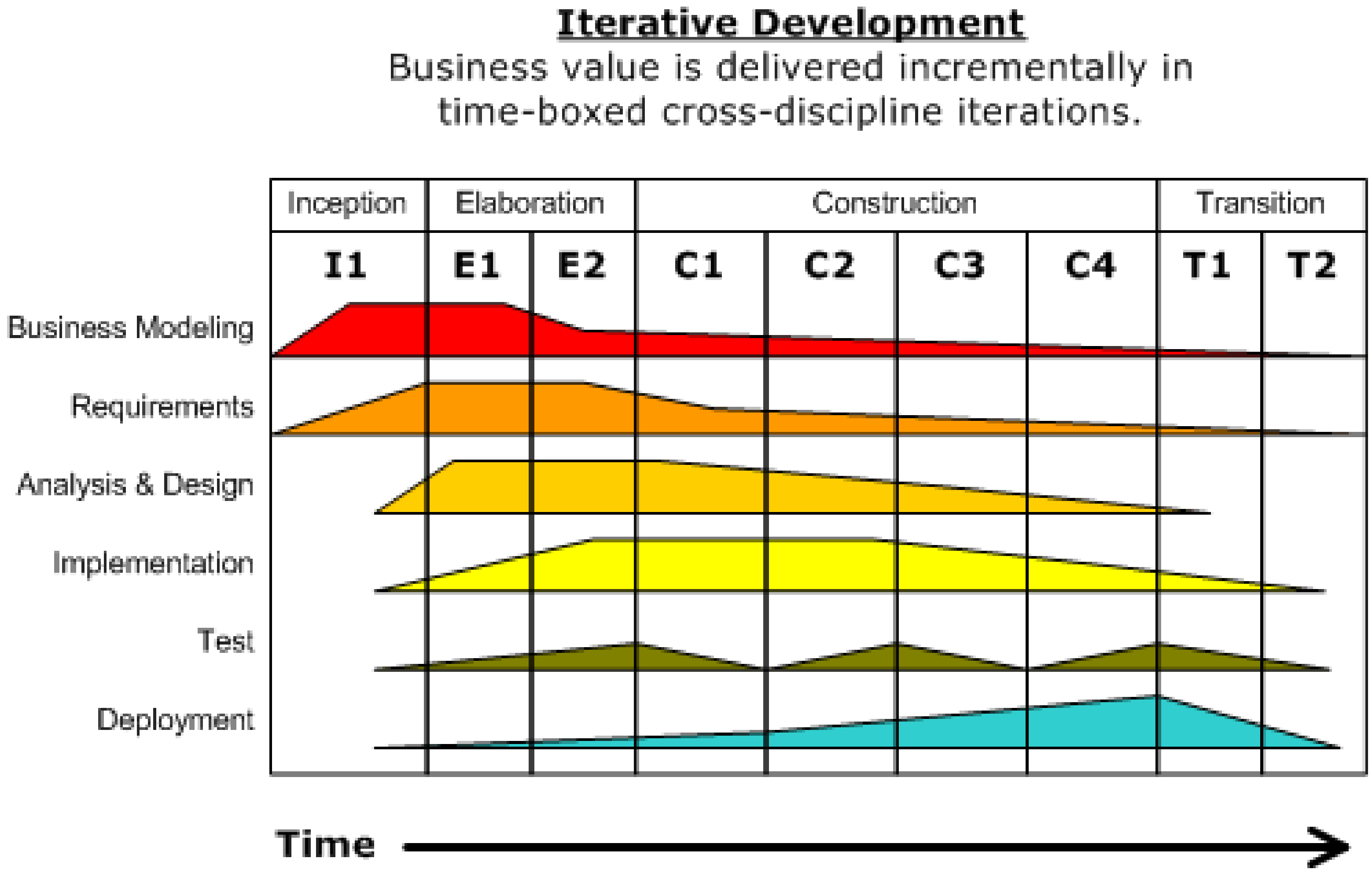
Incremental vs. Iterative



A Bit Different Understanding

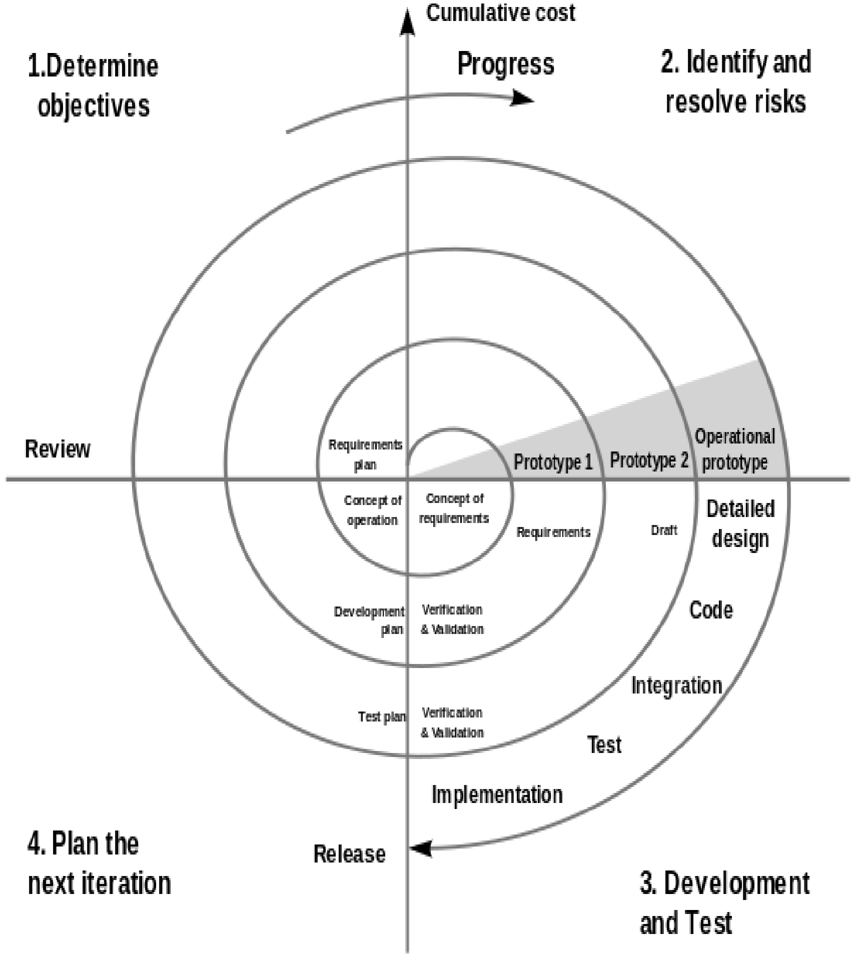


Effort in Iterative Development



Spiral Model

- Combining elements of design and prototyping-in-stages
- Combines the features of the prototyping and the waterfall model
- The spiral model is intended for large, expensive and complicated projects
- Advantages of top-down and bottom-up concepts



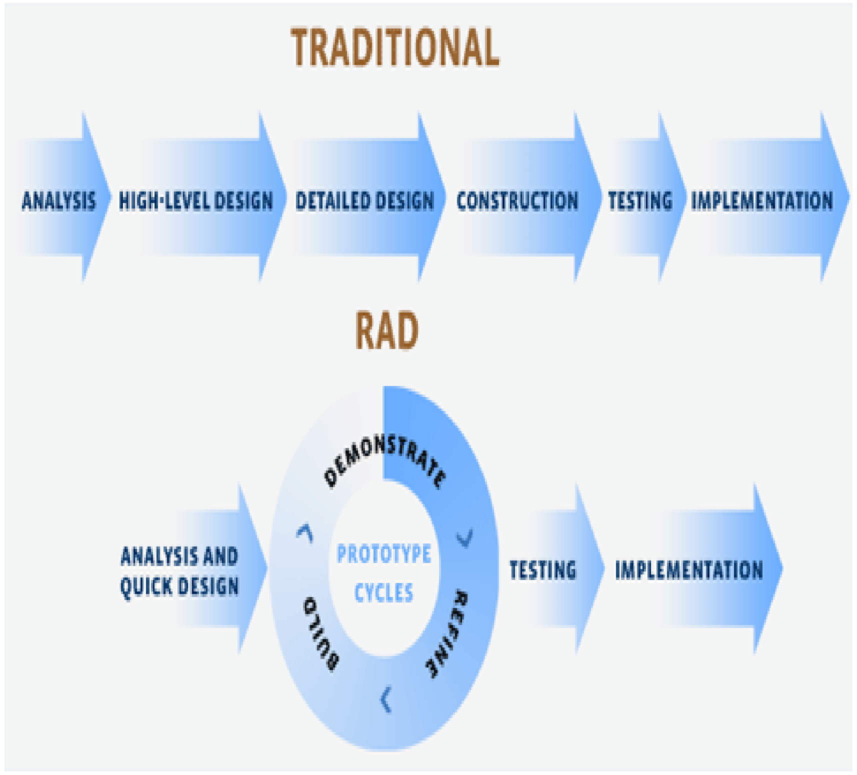
Background

- Top-down
  - deductive reasoning
  - analysis or decomposition
  - Descartes
  - G => 1

- Bottom-up
  - inductive reasoning
  - synthesis
  - Bacon
  - 1 => G

RAD

- Minimal planning and fast prototyping.
- Developing instead of planning
- The lack of pre-planning generally allows software to be written much faster, and makes it easier to change requirements.

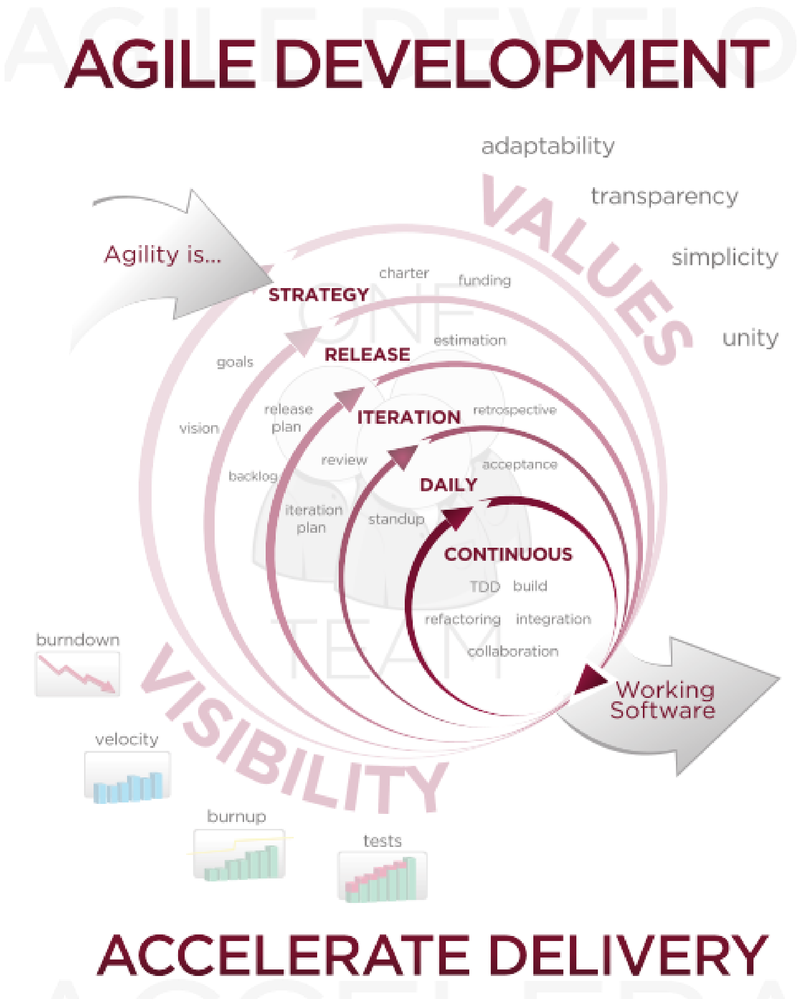


Cleanroom

- The Cleanroom process embeds software development and testing within a statistical quality control framework.
- Mathematically-based software development processes are employed to create software that is correct by design, and statistical usage testing processes are employed to provide inferences about software reliability.
- This systematic process of assessing and controlling software quality during development permits certification of software fitness for use at delivery.

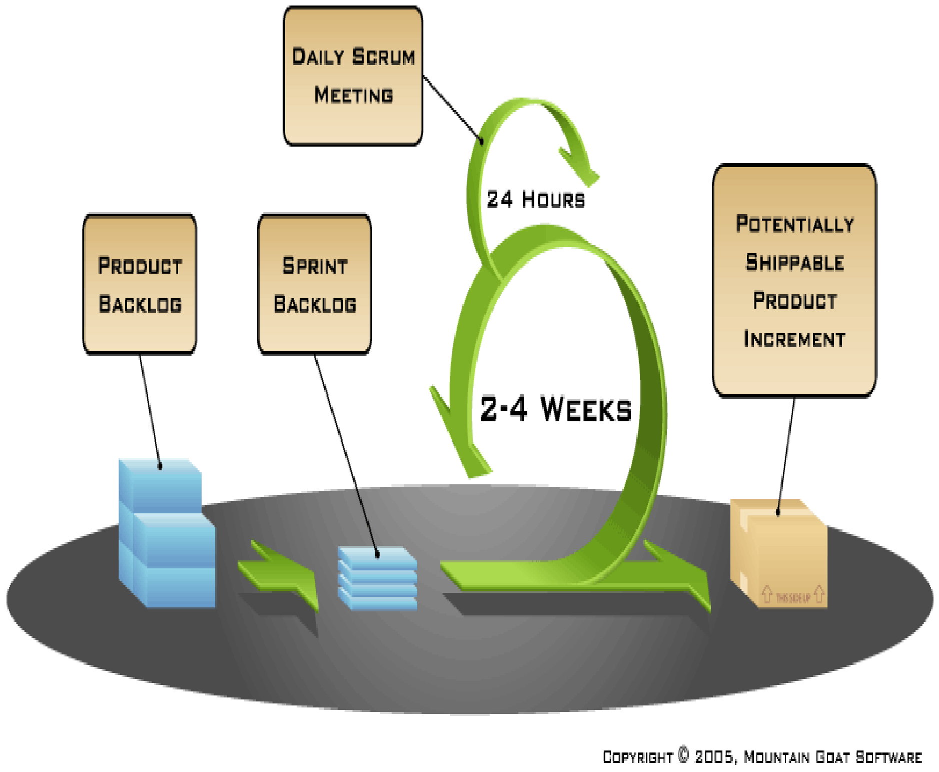
Agile

- Group of software development methods
- Based on iterative and incremental development
- Most important phrases
  - self-organizing, cross-functional teams
  - adaptive planning,
  - evolutionary development and delivery,
  - a time-boxed iterative approach,
  - rapid and flexible response to change.
- A conceptual framework
- The Agile Manifesto in 2001.



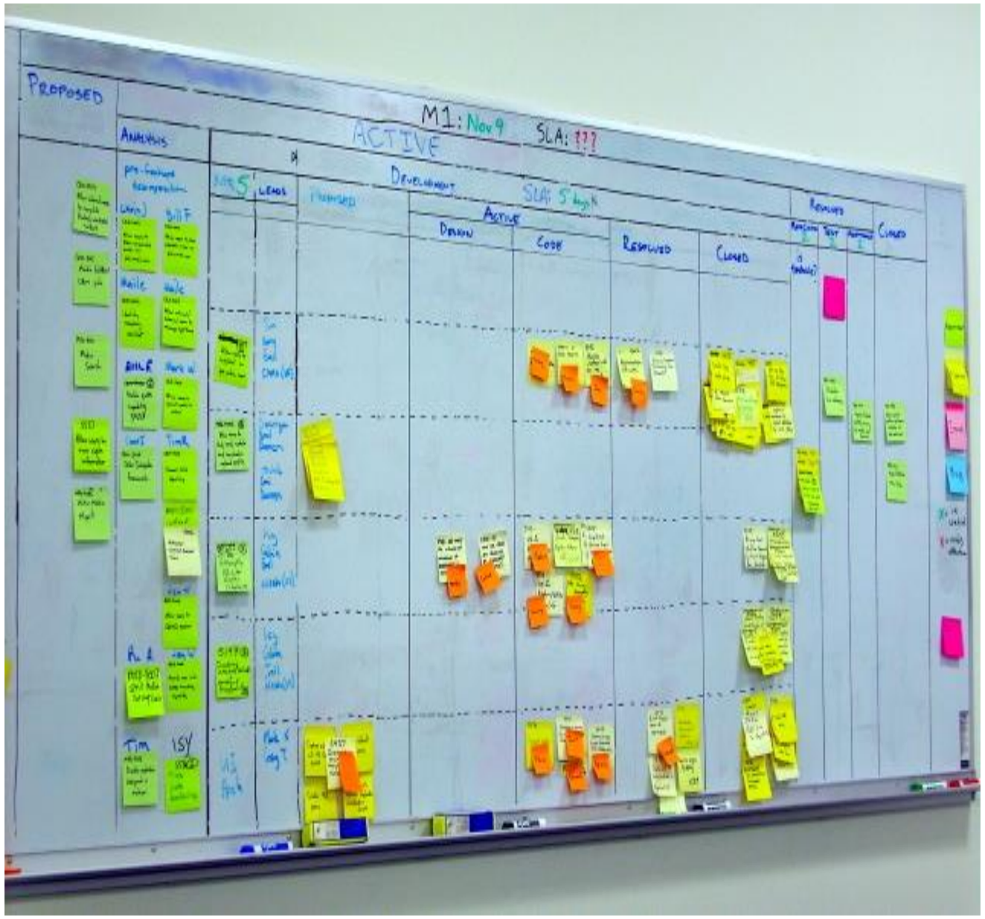
Scrum

- Scrum is an iterative and incremental agile software development framework
- A flexible, holistic product development strategy
- Development team works as an atomic unit
- Opposing to sequential approach


Daily Scrum Meeting
24 Hours
Product Backlog
Sprint Backlog
2-4 Weeks
Potentially Shippable Product Increment
Copyright © 2005, Mountain Goat Software

Lean (Kanban)

- A translation of lean manufacturing principles and practices
- Toyota Production System,
- Today part of Agile community.

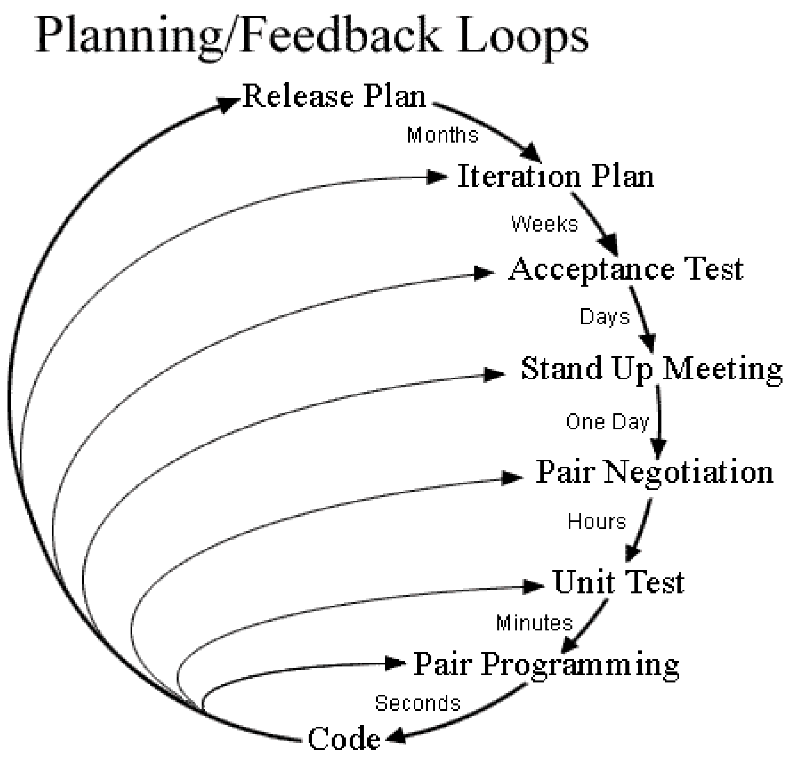


Lean Principles

1. Eliminate waste
2. Amplify learning
3. Decide as late as possible
4. Deliver as fast as possible
5. Empower the team
6. Build integrity in
7. See the whole

Extreme Programming (XP)

- Improve software quality and responsiveness to changing customer requirements
- A type of agile software development
- Frequent "releases" in short development cycles
- Introduce checkpoints where new customer requirements can be adopted.



XP Concepts (examples only)

- Pair programming
- Planning game
- Test-driven development
- Continuous integration

DSDM

- An agile project delivery framework, primarily
- DSDM fixes cost, quality and time at the outset and uses the MoSCoW prioritization of scope
- Pareto principle

- **M - MUST:** Describes a requirement that must be satisfied in the final solution for the solution to be considered a success.
- **S - SHOULD:** Represents a high-priority item that should be included in the solution if it is possible. This is often a critical requirement but one which can be satisfied in other ways if strictly necessary.
- **C - COULD:** Describes a requirement which is considered desirable but not necessary. This will be included if time and resources permit.
- **W - WOULD:** Represents a requirement that stakeholders have agreed will not be implemented in a given release, but may be considered for the future.

Test-driven development (TDD)

- Relies on the repetition of a very short development cycle: first the developer writes an (initially failing) automated test case that defines a desired improvement or new function, then produces the minimum amount of code to pass that test, and finally refactors the new code to acceptable standards.

- Test-first programming concept of extreme programming in the beginning
- Today standalone methodology

Feature-driven development (FDD)

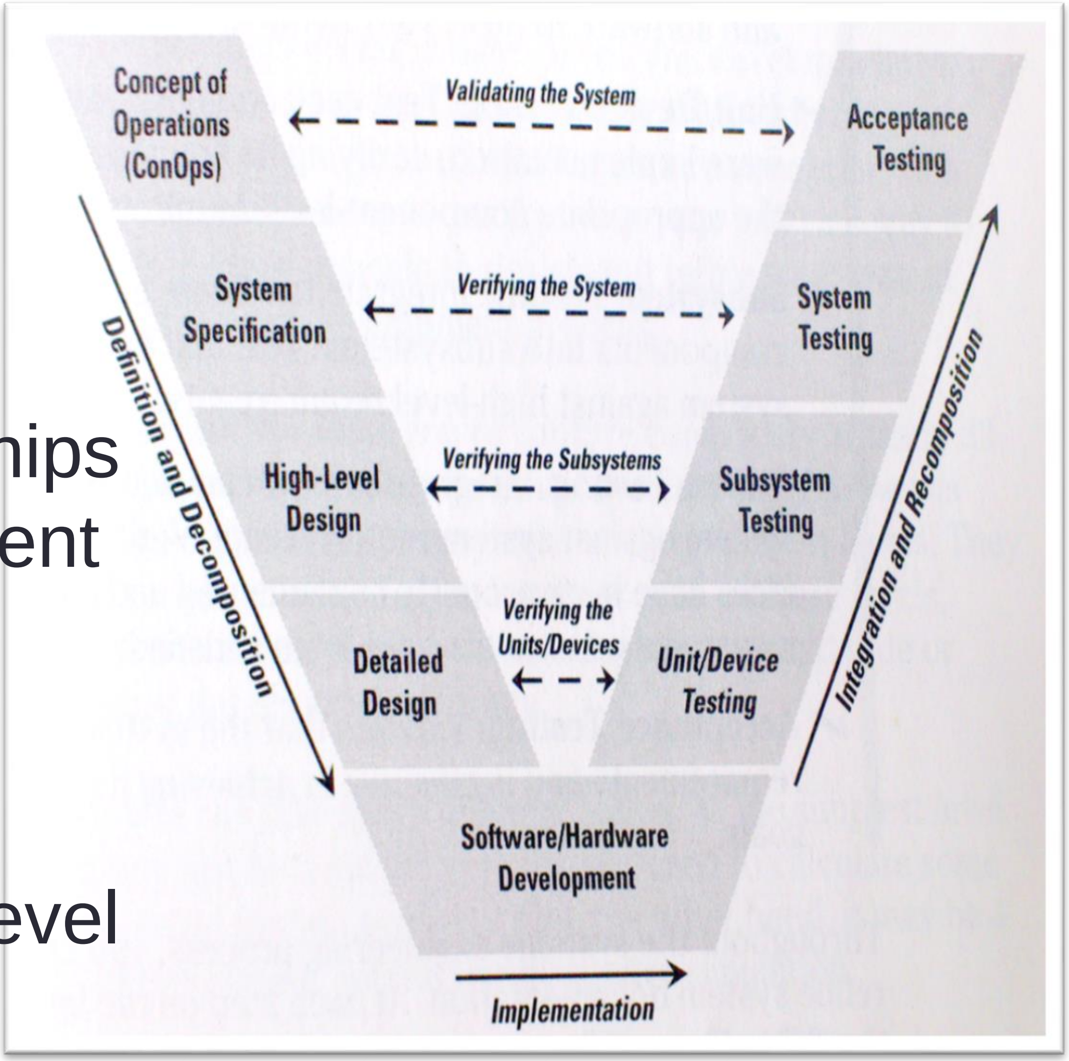
- Iterative and incremental development process.
- An Agile method
- Driven from a client-valued functionality (feature) perspective
- Mostly part of other methodologies

Rational Unified Process (RUP)

- An iterative software development process framework created by the Rational Software Corporation (IBM)
- Not a concrete prescriptive process, but an adaptable framework, intended to be tailored by the development organizations
- Expected to select elements of the process that are appropriate

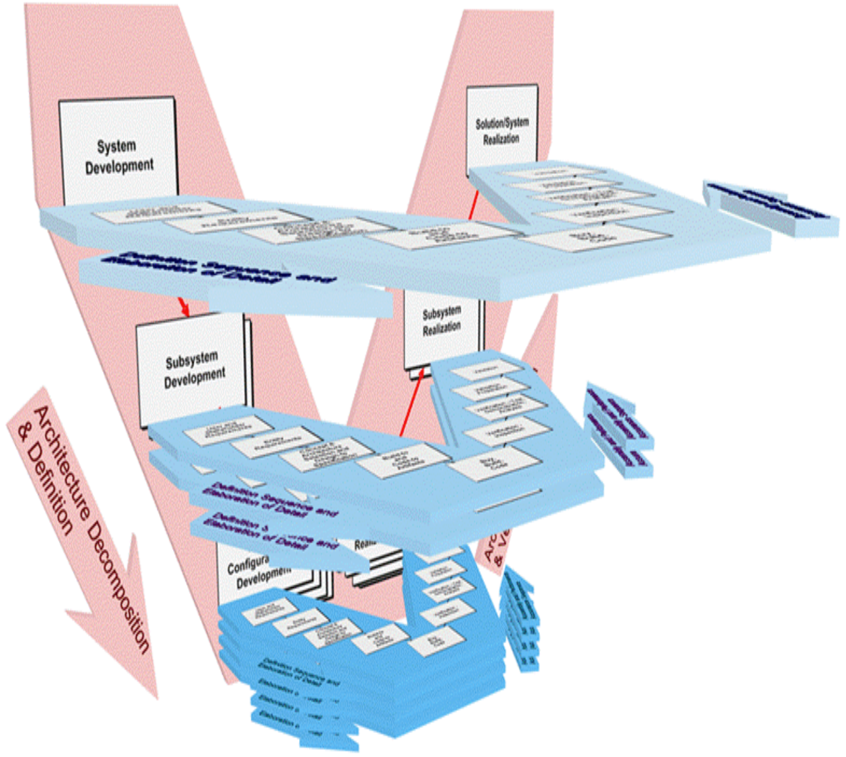
V-model

- The V-model is an extension of the waterfall model.
- Show the relationships between development phases and test phases
- Time and project completeness vs. level of abstraction

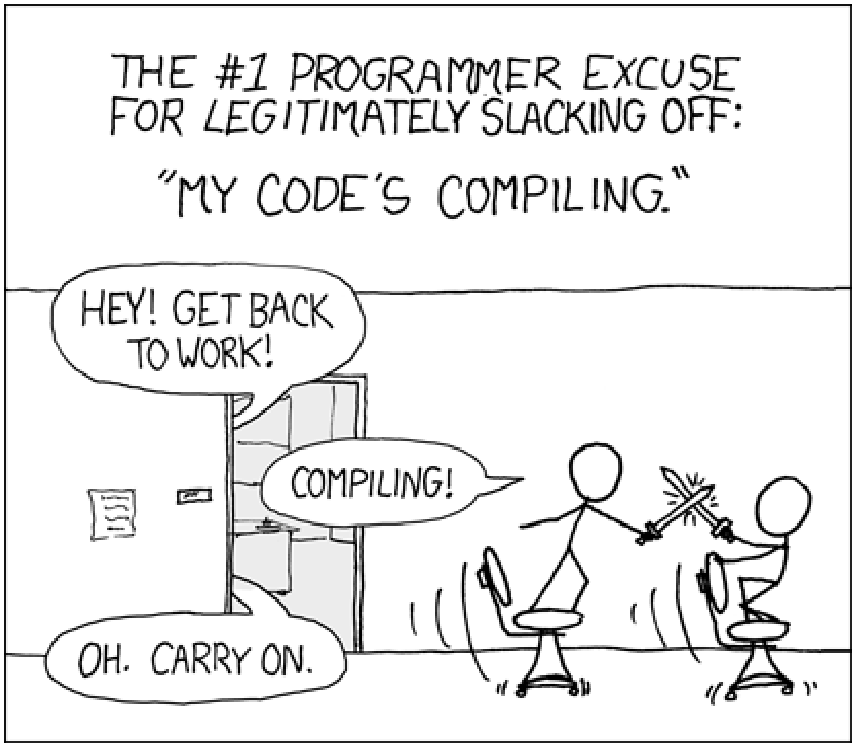


Dual Vee Model

- Describes a model of complex development
- For example:
  - Hardware
  - Platform
  - Application software
- Development of a system's architecture is the “big V”
  - Components’/entities’ developments are the “small V”-s
- It shows interactions and sequences of developing a complex system and a system of systems.


System Development
Solution/System Realization
Subsystem Development
Subsystem Realization
Configura Developme
Architecture Decomposition & Definition
Definition Sequence and Elaboration of Detail

Shouldn't forget



WATERFALL

Details

Waterfall #1

- Jump to next phase only if the prior one is completed

- PROs
  - Detailed early analysis cause huge advantages at later phases
  - If a bug found earlier, it is much cheaper (and more effective) to fix than bugs found in a later phase
  - Requirement should be set before design starts
  - Points to importance of documentation (minimized “broken leg” issue)
  - Disciplined and well-structured approach
  - Effective for stable software projects
  - Easy to plan from project management point of view

Waterfall #2

- CONs
  - Changes are expensive
  - Client does not explicitly know what he or she wants
  - Client does not explicitly know what is possible to have
  - Need to finish every phase fully
  - Long projects, difficult to keep the plan
  - Designers may not know in advance how complex a feature's implementation
  - "Measure twice, cut once"