Software development processes: from the waterfall to the Unified Process

Perdita Stevens

School of Informatics
University of Edinburgh

The Waterfall Model

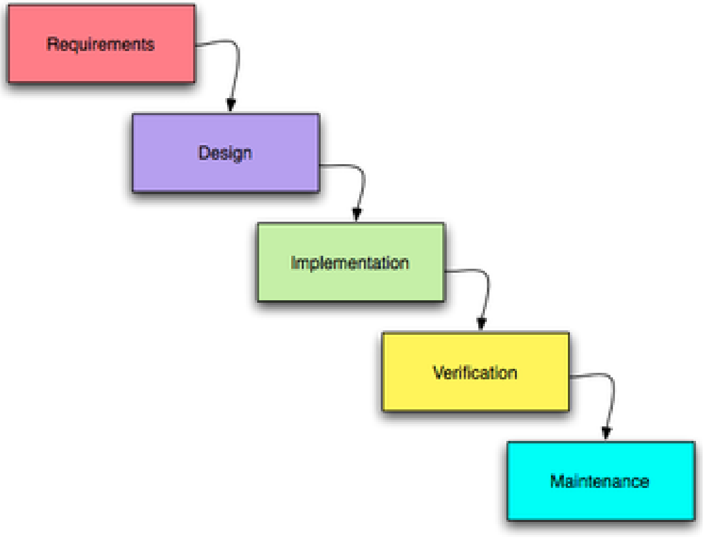


Image from Wikipedia

Pros, cons and history of the waterfall

+ better than no process at all – makes clear that requirements must be analysed, software must be tested, etc.

– inflexible and unrealistic – in practice, you cannot follow it: e.g., verification will show up problems with requirements capture.
  slow and expensive – in an attempt to avoid problems later, end up "gold plating" early phases, e.g., designing something elaborate enough to support the requirements you suspect you've missed, so that functionality for them can be added in coding without revisiting Requirements.

Introduced by Winston W. Royce in a 1970 paper

Pros, cons and history of the waterfall

+ better than no process at all – makes clear that requirements must be analysed, software must be tested, etc.

– inflexible and unrealistic – in practice, you cannot follow it: e.g., verification will show up problems with requirements capture.
  slow and expensive – in an attempt to avoid problems later, end up "gold plating" early phases, e.g., designing something elaborate enough to support the requirements you suspect you've missed, so that functionality for them can be added in coding without revisiting Requirements.

Introduced by Winston W. Royce in a 1970 paper

as an obviously flawed idea!

Spiral models

Split project into controlled iteration: each iteration is a mini-waterfall.

- + Mitigate risk. E.g. check user requirements, try out technology, practice new techniques in an early iteration to catch errors before main cost of project starts.
- – Cost: e.g., repeated testing and documentation. A few projects are so low risk that iteration isn't cost-effective.

  In practice, need for rework: essential to allow time for refactoring.

  Big projects need different approaches to different iterations.

Steps towards the Unified Process

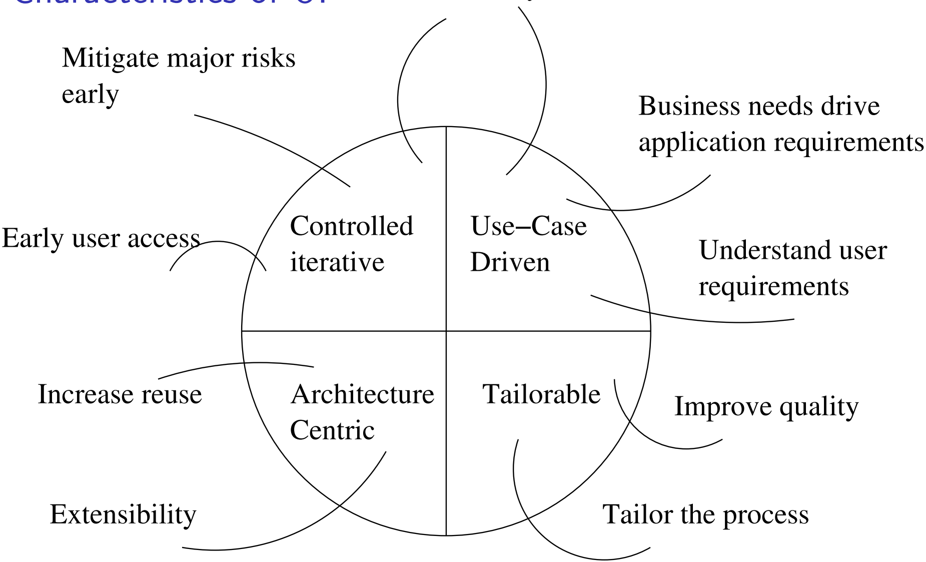
- 1960s - 1987: Ivar Jacobson at Ericsson: early component-based development, architectural block diagrams.
- 1987-1995: Jacobson founded Objectory (contraction of "Object factory"), added use cases
- 1995: Grady Booch, Jim Rumbaugh and Ivar Jacobson together at Rational, which bought Objectory. "The methods war is over – we won." First version of Unified Method produced. Controversial: quickly overshadowed by UML.
- 1995-1997: Rational Objectory: added controlled iteration
- 1998: (Rational) Unified Process

Unified process: the public domain, generic ideas

Rational unified process: more detailed, commercial. Now IBM.

Lots of variants, e.g. OpenUP, EnterpriseUP...

Characteristics of UP



Get early feedback

Mitigate major risks early

Business needs drive application requirements

Controlled iterative

Use–Case Driven

Early user access

Understand user requirements

Increase reuse

Architecture Centric

Tailorable

Improve quality

Extensibility

Tailor the process

(adapted from Rational slide)

The four Ps

- People
- Project
- Product
- Process

Workflows (one variant)

Engineering workflows:

- Business modelling
- Requirements
- Analysis and design
- Implementation
- Test
- Deployment

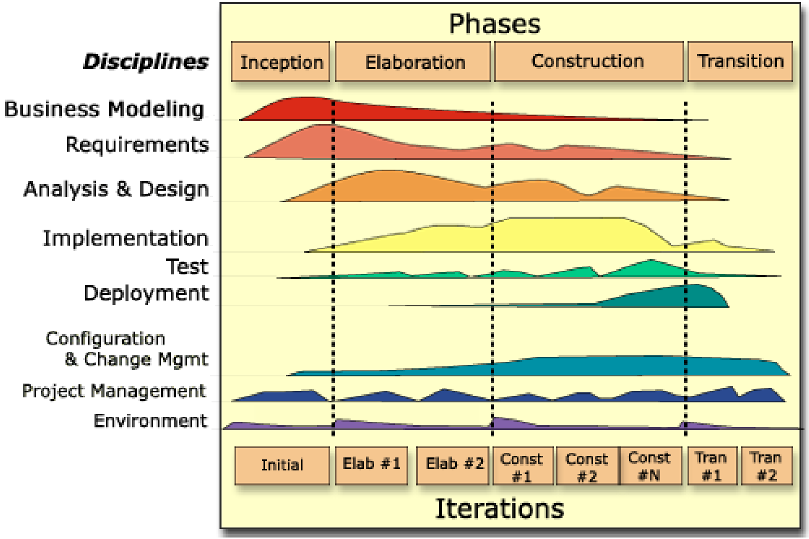
Supporting workflows:

- Configuration and change management
- Project management
- Environment (e.g. process and tools)

UP phases (iterative: end with review)

- Inception ends with commitment from the project sponsor to go ahead: business case for the project and its basic feasibility and scope known.
- Elaboration ends with
  - basic architecture of the system in place,
  - a plan for construction agreed,
  - all significant risks identified,
  - major risks understood enough not to be too worried.
- Construction (definitely iterative) ends with a beta-release system .
- Transition is the process of introducing the system to its users.

Workflows against phases



(adapted from Rational slide)