Software Development Life-Cycle Models

Essential Processes in Software Development

- **Requirements**
- **Design**
- **Implementation**
- **Test**

Models

A "Software Life-Cycle Model" specifies when the processes are conducted and how they feed into each other.

"Life-Cycle" Models

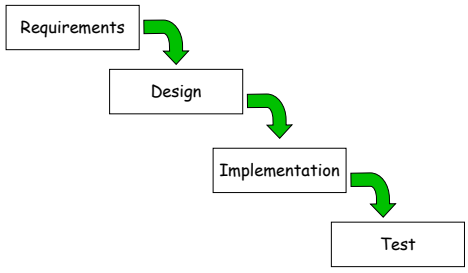
- Single-Version Models
- Incremental/Iterative Models

"Life-Cycle" Models

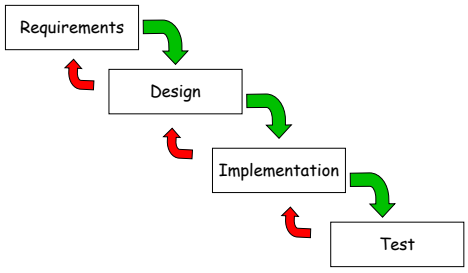
- Single-Version Models
  - Big Bang Model
  - Waterfall Model
  - Waterfall Model with "back flow"
  - "V" model: Integrating testing

Big Bang Model

Waterfall Model

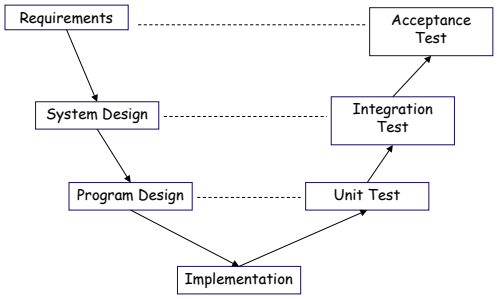


Waterfall Model



"V" Model

Each phase has corresponding test



When to use single version?

- For simple projects!

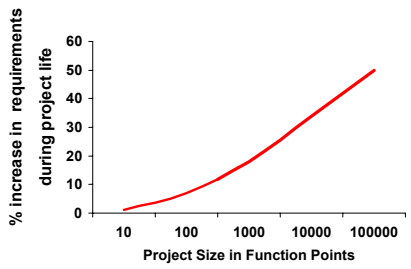
When not to use single version?

- For everything else!

Why not single-version?

- Initial requirements are *speculative*

Growth in requirements



Source: Applied Software Measurement, Capers Jones, 1997. Based on 6,700 systems.

Why not single-version?

- Initial requirements are *speculative*
- Initial designs are *speculative*

Software is "hard"

- Software is very "hard".
  - Discover Magazine, 1999: Software characterized as the most complex "machine" humankind builds.

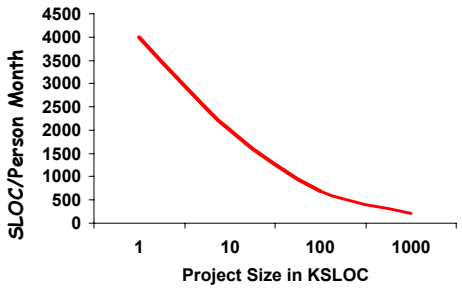
Why not single-version?

- Initial requirements are *speculative*
- Initial designs are *speculative*
- Speculative decisions compound in absence of feedback

Why not single-version?

- Initial requirements are *speculative*
- Initial designs are *speculative*
- Speculative decisions compound
- High complexity/low adaptability

Decrease in Productivity



Source: Measures For Excellence, Putnam, 1992. Based on 1,600 systems.

Why not single-version?

- Initial requirements are *speculative*
- Initial designs are *speculative*
- Speculative decisions compound
- High complexity/low adaptability
- High risk issues identified/addressed late in the life cycle

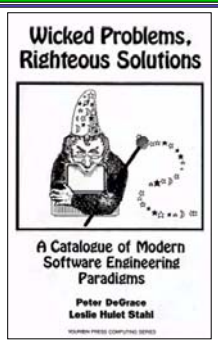
Risk

not the game…

"Wicked Problems"

"problems that are fully understood only after they are solved the first time" (however poorly)

Source of some of this



Prentice-Hall, 1990

basically a criticism of the waterfall model

"wicked" term first used in H. Rittel and M. Webber, *Dilemmas in a general theory of planning*, Policy Sciences, **4**, pp. 155-169, Elsevier, 1973.

Some Roots of Wickedness

- **Risk**: A *customer* not knowing exactly what he/she wants; changing expectations as project progresses.
- **Risk**: *Staff* who are inexperienced in the problem domain, or with the appropriate implementation techniques.

US Air Force Risk Classification

- **Performance risk**: The project might not meet requirements or otherwise be fit for use.
- **Cost risk**: The budget might get overrun.
- **Support risk**: The software might not be adaptable, maintainable, extendable
- **Schedule risk**: The project might be delivered too late.

The Waffle Principle

- "Plan to throw the first one away; you will anyhow."

Fred Brooks, "The Mythical Man-Month: Essays on Software Engineering", Addison Wesley, 1975.
Revised in 1995.

"Life-Cycle" Models

- Single-Version Models
- Incremental/Iterative Models

Iterative vs. Incremental

- To some people these are different
- To some they are the same ⬅ You are here

Front-Loading/Risk Driven

- Tackle the unknown and harder parts earlier rather than later.
- Better to find out about infeasible, intractable, or very hard problems early.
- The easy parts will be worthless if the hard parts are impossible.
- Find out about design flaws early rather than upon completion of a major phase.

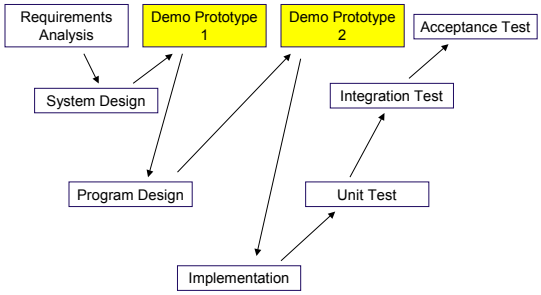
Iterative Development

- In each iteration:
  - Identify the largest risks of the project
  - Brainstorm on ways to reduce or eliminate these risks
  - Form a concrete plan with specific artifacts
  - Carry out the plan

Iterative/incremental "Life-Cycle" Models

- Sawtooth Model
- Spiral Model & Variants
  - ROPES Model
  - Controlled Iteration Model: Unified Process
  - Time Box Model
- Scrum Model
  - Fountain Model

Sawtooth Model (Brugge)



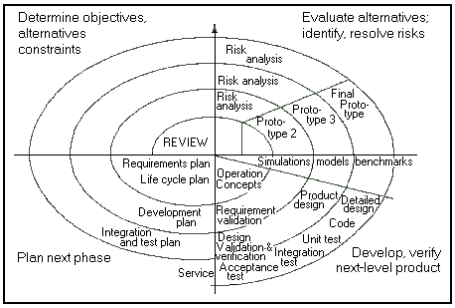
Boehm Spiral Model

Iterates cycles of these project phases:

1. Requirements definition
2. Risk analysis
3. Prototyping
4. Simulate, benchmark
5. Design, implement, test
6. Plan next cycle (if any)

Prof. Barry Boehm

Boehm Spiral Model



ROPES Model - Similar to Spiral

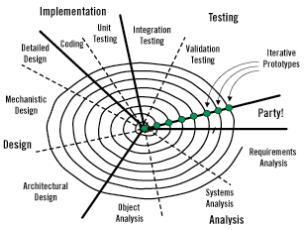
Rapid Object-Oriented Process for Embedded Systems
Bruce Douglass

- Iterates the following sequence of phases repeatedly:
  - Requirements analysis
  - System analysis
  - Object analysis
  - Architectural design
  - Design
  - Mechanistic design
  - Detailed design
  - Coding
  - Unit testing
  - Integration testing
  - Validation testing
  - Iterative prototypes

http://www.sdmagazine.com/breakrm/features/s999f1.shtml

ROPES Model

Rapid Object-Oriented Process for Embedded Systems
Bruce Douglass

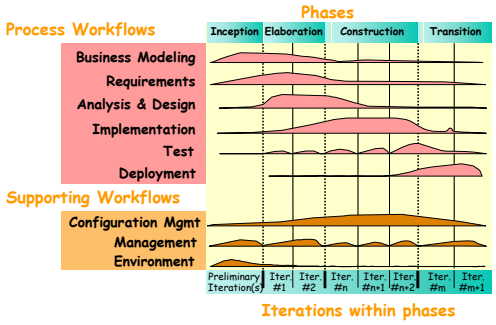


Controlled-Iteration Model

- Four phases per iteration
  - **Inception**: Negotiate and define product for this iteration
  - **Elaboration**: Design
  - **Construction**: Create fully functional product
  - **Transition**: Deliver product of phase as specified
- The next phase is started **before** the end of the previous phase (say at 80% point).

Rational Unified Process

(a form of controlled iteration)



Time-Box Requirement

(can be used in iterative or incremental)

- Requirements analysis
- Initial design
- while(not done)

  {
  Develop a *version* **within a bounded time**
  Deliver to customer
  Get feedback
  Plan next version
  }

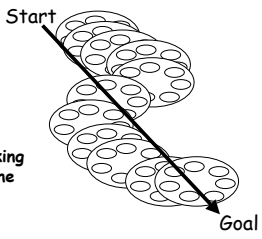
Scrum, A cure for Wicked?

Scrum first mentioned in
"The New New Product Development Game" (Harvard Business Review 86116:137-146, 1986)

Scrum Model

(incremental model,
includes some aspects of team structure, as well as process)

A **small group** is responsible for **picking up the ball and moving it toward the goal**.



See http://www.cetus-links.org/oo_ooa_ood_methods.html

Argument for the Scrum Model over other iterative models

- A software development project might not be compartmentalizable into nice clean phases as the Spiral models suggest.
- Scrum may be "just the thing" for wicked problems, because the team can quickly react to new information.

Some Principles of Scrum Model

- **Always have a product** that you can theoretically ship: "done" can be declared at any time.
- **Build early, build often**.
- **Continuously test** the product as you build it.
- **Assume requirements may change**; Have ablility to adapt to marketplace changes during development.
- **Small teams** work in parallel to maximize communication and minimize overhead.

Use of Iteration in Scrum

http://www.controlchaos.com/scrumwp.htm

- Each **iteration** consists of all of the standard Waterfall **phases**,
- *but* each iteration only addresses **one set of functionality**.
- Overall project deliverable has been **partitioned** into prioritized subsystems, each with clean interfaces.
- **Test the feasibility** of subsystems and technology in the initial iterations.
- Further iterations can **add resources** to the project while ramping up the speed of delivery.
- U**nderlying development processes are still defined** and linear.

Fountain Model

(Ian Graham, et al., The OPEN Process Specification
OPEN = Object-oriented Process Environment and Notation)