Software Lifecycle & Process Models

CSE 403 - Spring 1997

Outline

- Software Lifecycle
- Ad Hoc vs. Systematic Processes
- Waterfall Model
  - Phases in the waterfall model.
  - Benefits and shortfalls of the waterfall model.
- Evolutionary Model
- Spiral Model
  - Benefits and shortfalls of the spiral model.

Software Lifecycle

- The "software lifecycle" is a generic term used to denote a software process.
  - The activities used to produce the product are called a software process, including the order in which these activities are applied.
- The basic goal of a software lifecycle is to define how a software system proceeds from the original concept through development, release, use, and maintenance of a software product that meets that concept.

There is Always a Process

- Every software product goes through stages from womb to tomb.
- Thus, every product has had some software process applied.
  - There have been a collection of activities, in some order, that have been applied.
  - It may not have been planned and it may not have been systematic, but it exists.

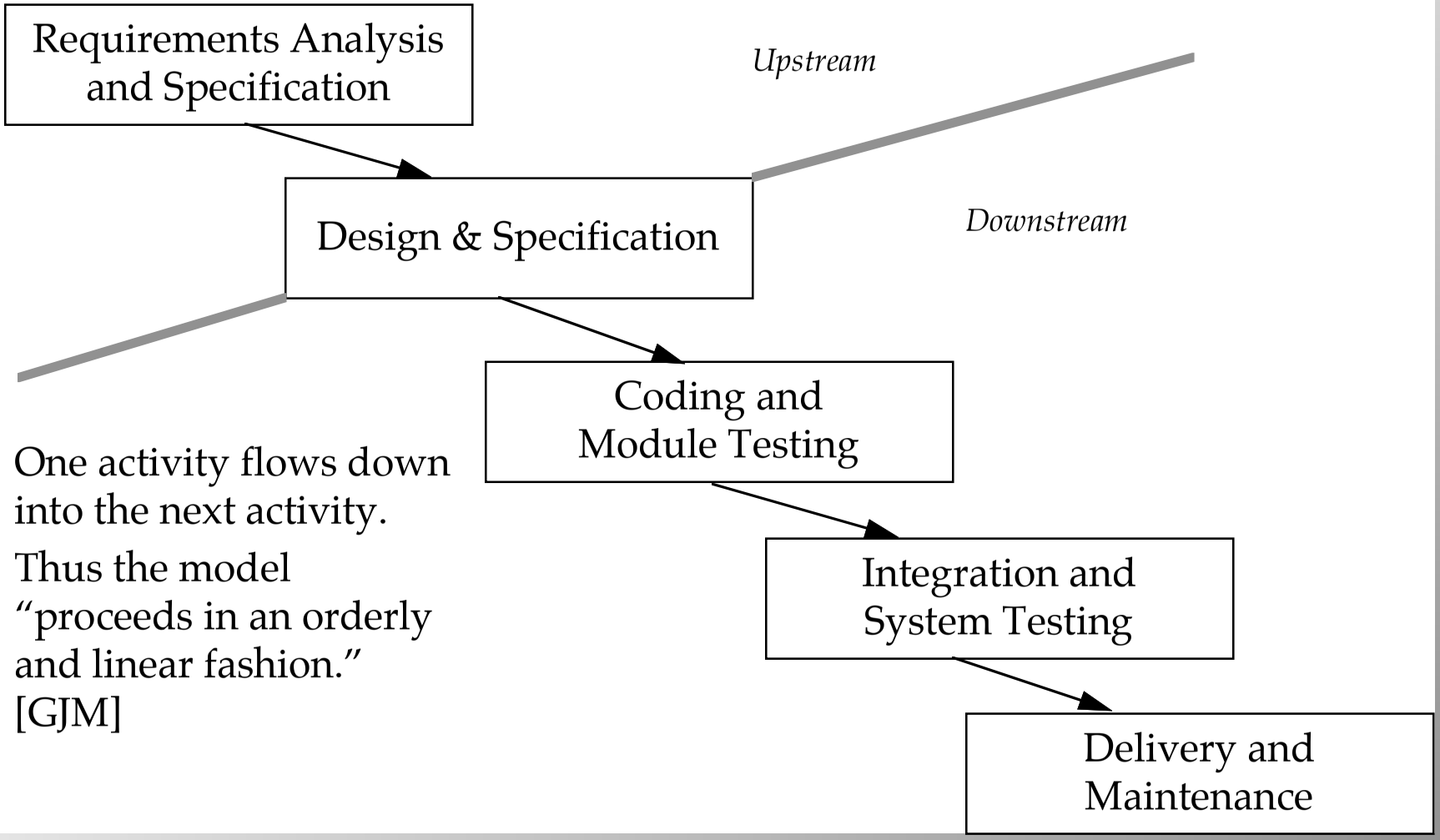
Ad Hoc vs. Systematic Processes

- An *ad hoc* process---that is, the absence of a well-defined process---is chaotic and in general leads to poorer software products.
  - Why?
- A *systematic* process does not necessarily yield a high-quality product.
  - Not all systematically planned processes play out systematically?
    - Examples?
  - A systematic process won't overcome a lack of understanding of domain, technical issues, etc.

Process Models

- A process model is "[a] purely descriptive representation of the software process. A software process model should represent attributes of a range of particular software processes and be sufficiently specific to allow reasoning about them [Dowson]."
- That is, a process model characterizes a collection of software processes.
  - E.g., waterfall model, evolutionary model, spiral model.
- Artifacts and commitments between the phases.

Waterfall Model



- One activity flows down into the next activity.
- Thus the model "proceeds in an orderly and linear fashion." [GJM]

Phases in the Waterfall Model

- Requirements analysis & specification
  - Feasibility study: technical evaluation, economics, etc.
  - Understand, clarify and precisely state the actual requirements for the system.
- Design.
  - Take from "what" to "how"
    - Narrow down from the space of possible solutions.
    - Selection criteria?
- Coding and module testing.
  - Create individual modules and test them for correctness.
- Integration and system testing.
  - Test modules collectively as sybsystems and as a complete system.

Phases in the Waterfall Model

- Delivery and maintenance.
  - Deliver the product to the customer, installed, and used.
    - Other activities?
  - Maintenance: corrective, adaptive, and perfective.
    - Maintenance accounts for over 60% of the total costs of software development.
    - Then why just one box?

Benefits of the Waterfall Model

- The model itself is clearly stated.
- The model maps quite closely to a set of reasonable deliverables.
- Methods can be derived for each phase, giving guidance to the software engineers about how to proceed.

Weaknesses of the Waterfall Model

- For poorly understood and/or complex applications, the prohibition against backtracking is a heavy one.
- Evolution is neither anticipated or planned.
- It requires monolithic planning a priori, which is not generally feasible, because it requires estimation in the face of limited information.
- The stress on documents can lead to a focus on syntax without semantics.

Evolutionary Model

- Deliver-Measure-Adjust.
- Plan to throw one away.
  - An alternative to the waterfall model is characterized by Brooks': "Plan to throw one away; you will anyway."
  - The first version is a kind of prototype.
  - In "theory," the second system can be developed with a waterfall model, since the dangers and difficulties have been identified.
    - What's wrong with this picture?

Second System Syndrome

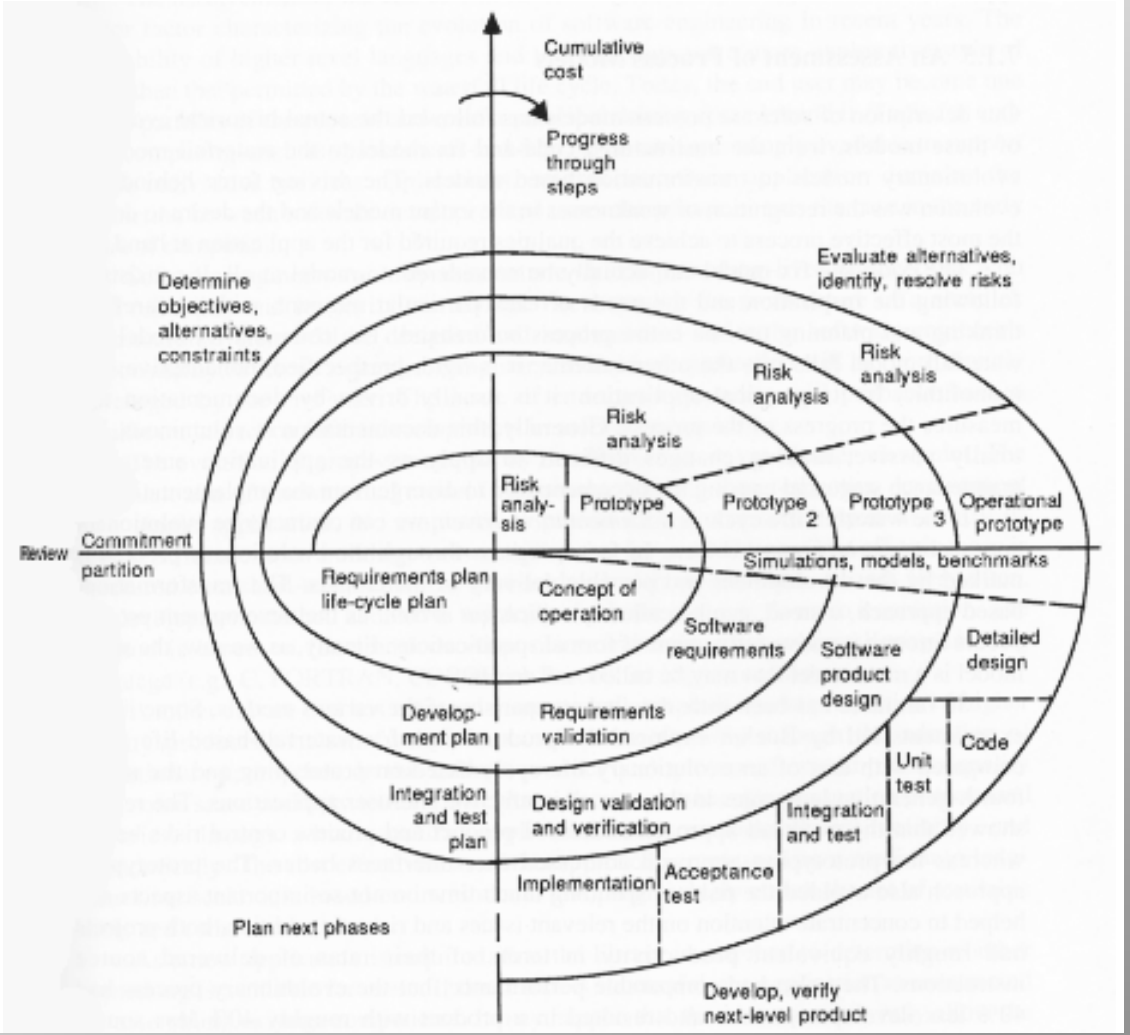
- Brooks also warns against the second system syndrome, where the danger of adding in bells and whistles is enormous.
- Balancing “plan to throw one away” with the “second system syndrome” is needed.
  - Any suggestions?

Spiral Model (Boehm)

- Boehm's spiral process model is based on the notion of risk management.
  - A "discipline whose objectives are to identify, address and eliminate software risk items before they become either threats to successful software operation or a major source of expensive software rework."
- The spiral model uses risk to treat minor and severe problems differently, which the waterfall model can't do.

Four Activities in the Spiral Model

- The radius represents the cost so far.
- The angle represents the current activity.
- The number of turns in the spiral represents the degree to which the application domain is well-understood.



The Spiral

- "After one cycle of the spiral, unstated requirements are checked as part of the robustness of the application. As we realize that they should have been specified as requirements, they become part of the specification of the next cycle, if any. Consequently, after each iteration through the cycles, robustness approximates correctness more closely [GJM]."

How Do You Finish a Spiral?

- Successful completion is one way.
- Another way is to view the spiral as testing a hypothesis about a product.
- If the hypothesis fails, the spiral should be terminated without completing the project.
  - Delay causes the product to miss a marketing window.
  - A superior product becomes commercially available.

Subsumes Other Process Models

- With low-risk in user interfaces and performance, but with high-risk in scheduling, budget, etc., the spiral model reduces to the water fall model.
- With stable requirements, but risk in error-prone implementations, spiral reduces to precise specification and formal deductive program development.

Advantages of Spiral Model

- Accommodates good features of existing process models, while risk-approach reduces their problems.
- Focuses early attention on options involving reuse.
- Helps prepare for evolution.
- Relates desired objectives (ease of use, reliability, etc.) to product development.
- Focuses on early elimination of errors and weak approaches.

Advantages of Spiral Model

- Helps to answer "How much is enough?" question in resource allocation.
- Permits combination of development and evolution approaches.
- Applies to system design, not only software design.

Weaknesses of Spiral Model

- Applies better to internal development than to contract software efforts (since there are generally more constrains in those situations.)
  - Like?
- Relies on expertise in risk-assessment.
- Needs more elaboration of the different steps in a cycle.