Software Process Models

Table of Contents

- Waterfall Model
- Waterfall Model Phases
- Waterfall Model Phases (cont)
- Waterfall Model Phases (cont)
- Waterfall Model Phases (cont)
- Spiral Model
- Spiral Model (cont)
Waterfall Model

Phases

- Requirements
  - System analysis
  - Problem Description
- Specification
  - Problem Definition
- High Level Design
  - Logical Design
  - Modular Composition
- Low Level Design
  - Physical Design
  - Step-Wise Refinement
  - Top-Down Design
- Coding
- Integration
- Testing
- Deployment
- Maintenance

Advantages / Disadvantages

- Most-widely used process model
- Controls schedules, budgets & documentation
- Tends to favor well-understood system aspects over poorly understood system components, (no risk analysis)
- Does not detect development areas behind schedule early in the lifecycle stages.

Document-driven process

- Deliverables: documents produced at the end of each phase, usually in accordance to contract deadlines
Waterfall Model: Phases

Requirements
- A statement of the functions and behavior of the system required by its users & operators
- General Requirements
  † Defines broad & detailed objectives of the system
  † e.g., reliable, correct, efficient, user-friendly, expandable
- Gives relationship of Qualitative & Quantitative System Goals

Specification
- Listing of specific, measurable behavioral system constraints that satisfy system requirements
- Clearly communicates system operations with end user(s)
  † complete, unambiguous, minimal, understandable, testable
- Cross-reference indexed to requirement items
- Defines the design validation & final system testing criteria
- Provides chief mechanism for estimating the project’s progress

Design: Representation or model of a system
Coding and Debugging (implementation)
- Translation of design into a programming language
- Indispensable Programmer Phenomena
- Program Unit Notebooks
  1. Documents programmer’s work activities
  2. Maintains current unit (module) documentation
  3. Passed from programmer to programmer during development

<table>
<thead>
<tr>
<th>SECTION</th>
<th>CONTENTS</th>
<th>DUE DATE</th>
<th>COMPLETED DATE</th>
<th>REVIEWER/DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>RQMTS.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td>ARCH. DESIGN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>DFTAIL DESIGN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>TEST PLAN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>TEST RESULTS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.</td>
<td>CHANGE REQUESTS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.</td>
<td>SOURCE CODE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.</td>
<td>NOTES</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

RELEASE APPROVAL: ___________________________ DATE: ___________
Waterfall Model: Phases (cont)

Integration and Testing
- Unit testing: individual modules (functions) are tested separate from other modules
- Integration testing: system modules are tested together

Deployment & Maintenance
- Requires previous phases to be repeated
- Makes up 70%-90% of total system cost
- Majority of maintenance time (50%) spent on system understanding -> system documentation
- Maintenance Tasks
  † collection, analysis and prioritization of user trouble reports
  † new system release installations
  † documentation (user’s manuals) changes
  † configuration control issues
The Spiral Software Process Model Diagram

- Development phases reiterates through four cycles:
  † Set goals and determine constraints for the phase
  † Evaluate and resolve risks for the phase
  † Develop the prototype for the phase
  † Plan the next stage activities

- Step 2 involves a Risk Analysis that identifies:
  - less understood system areas
  - systems areas that pose the greatest jeopardy to development

Prototype Based

- Prototype: a limited, semi-functional, task restricted, partially operational system
  † Analogous to a model or mockup that allows evaluation of development alternatives before commitment

- Rapid Prototyping Systems
  † Authoring/scripting (multimedia) systems used to quickly develop multiple interfaces for user evaluation, cannot serve as a kernel for future iterative system prototype development
  † Users tend to view prototypes as final versions of the system

Mimic

- Risk analysis produces a risk-resolution strategy
  † Feasibility Study: determination of a strategy achieving set goals and requirements within stated constraints.
  ‡ Address development factors of expertise, experience, resources and motivation
  † Extension of cost/benefit analysis
  ‡ Cost & benefits are estimated for best & worst case outcomes which are multiplied by their probability of occurrence giving an expected value.
  ‡ Decisions on strategies are made to minimize cost and maximize benefits

- Cycles are modified to concentrate on different areas of system development driven by the risk-resolution plan
- Spiral model tends to behave like other process models due to differing cycles