WaterSluice
A Software Engineering Methodology

Ron Burbank
February, 1998
# The Field of Software Engineering

<table>
<thead>
<tr>
<th>Feedback</th>
<th>plan repair, re-planning, process changes, plan optimization, chronic problem management, ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measure</td>
<td>number of faults both reported and fixed, lines of code, closeness to plan, resource utilization, performance, ...</td>
</tr>
<tr>
<td>Strategies</td>
<td>methodologies, architecture, paradigms, mission, risk analysis, scheduling, priority setting, resource utilization, decision making, life cycle management, ...</td>
</tr>
<tr>
<td>Tools</td>
<td>compilers, debuggers, environments, quality assurance, CASE, version control, databases, operating systems, networks, file systems, GUI builders, composition, ...</td>
</tr>
<tr>
<td>People</td>
<td>group interactions, skill development, group dynamics, communications, goal setting, ...</td>
</tr>
</tbody>
</table>
Methodology

The body of methods, rules, postulates, procedures, and processes that are used to manage a software engineering project through many life cycle stages.
Life Cycle Stages

Development → Deployment → Development → Deployment → Deployment → Operations → Maintenance

Deployment → Operations → Maintenance

Deployment → Operations → Maintenance

Deployment → Operations

Maintenance → Legacy
Four Fundamental Phases

- Define Goals → Analysis
- Establish Plan → Design
- Do the Work → Implementation
- Improve Quality → Testing

Every stage in the life cycle has these four phases.
Phases in the Development Stage

- Analysis:
  - requirements, domain ontology, things, actions, states, events, typical and atypical scenarios

- Design:
  - architecture, implementation plan, performance analysis, test plan

- Implementation:
  - the code

- Testing:
  - quality improvements, regression test, internal testing, unit testing, application testing, stress testing
Process Management Options

- Methodologies
  - Sequential (Waterfall)
  - Cyclical (Spiral)
  - Best-First (WaterSluice)

- Versions
  - The project may go through several versions.
  - Each version replays the methodology with the previous version used as a starting point for the next version.
  - Some features may be deferred to a later version.
Sequential Methodology
Sequential with Versions

<table>
<thead>
<tr>
<th>A</th>
<th>D</th>
<th>I</th>
<th>T</th>
<th>A</th>
<th>D</th>
<th>I</th>
<th>T</th>
<th>A</th>
<th>D</th>
<th>I</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version 1</td>
<td>Version 2</td>
<td>Version 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Waterfall Methodology

- System Requirements and Validation
  - Software Requirements and Validation
    - Preliminary Design and Validation
      - Detailed Design and Validation
        - Code, Debug, Deployment, and Test
          - Test, Preoperations, Validation Test
            - Operations, Maintenance, Revalidation
Animation Clip
Sequential with Versions

Pro and Con

• Pro
  – well established
  – works on quasi-static projects

• Con
  – does not scale to large projects in dynamic environments
Optimal Positioning of Methodology

- Sequential with Versions
- Sequential
Cyclical Methodology

Cyclical Methodology with Version

Version 1
Version 2
Animation Clip
Cyclical with Versions
Pro and Con

- Pro
  - feedback path

- Con
  - no governors
  - no priority
  - no conflict management
  - may diverge instead of converge
Optimal Positioning of Methodology

- Cyclical with Version
- Cyclical
- Sequential with Versions
- Sequential
Best-First Methodology

- Borrow the steady progression of the sequential methodology.
- Borrow the iterative nature of the cyclical methodology.
- Add goal focus
- Add priority (cost)
- Add Non-monotonic governor
  - change order control
Priority Function

- Each potential task is assigned a priority.
- This priority reflects the benefit to the final goal of accomplishing the task based on what has already been accomplished.
- The highest priority task is accomplished next.
Change Order Control

- Process to manage change.
- Once a component is completed to the satisfaction of the team it is placed under change order control and frozen.
- Only absolutely necessary changes are allowed.
- Changes should be seldom, well justified, and documented.
The WaterSluice Methodology

A: Analysis
D: Design
I: Implementation
T: Testing

P1: Proof of Principle
P2: Prototype
P3: Alpha and Beta
P4: Product

Product
Animation Clip
WaterSluice - Pro and Con

• Pro
  – feedback path
  – governors
  – priority
  – goal directed
  – forces converges

• Con
  – more complex
Optimal Positioning of Methodology

- WaterSluice with Versions
- WaterSluice
- Cyclical with Version
- Cyclical
- Sequential with Versions
- Sequential

Change

High

Medium

Non-Monotonic

None

Small

Medium Size

Large
Theory

- Map waterfall to breadth-first search
- Map spiral to depth-first search
- Map Watersluice to best-first search
Search Space

- Start
- Analysis
- Design
- Implementation
- Testing
- Goal
Spiral: Depth-First Search
WaterSluice: Best-First Search
## Summary of Theorems

<table>
<thead>
<tr>
<th>Methodology</th>
<th>Static Complete</th>
<th>Dynamic Complete</th>
<th>Dynamic Optimum</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>waterfall</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>good</td>
</tr>
<tr>
<td>spiral</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>good-&gt;better</td>
</tr>
<tr>
<td>WaterSluice</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>good-&gt;best</td>
</tr>
</tbody>
</table>
Other Work

- DADL - Distributed Architecture Definition Language
- Noema - Engineering paradigm
- CHAIMS - Component engineering
- Distributed Computer Environments
The Thesis

- Software Engineering
- Methodologies
- Requirements
  - two examples
- Implementation and Testing
  - C++ container classes
- Decision Making
Conclusion

The WaterSluice methodology borrows the iterative nature of the cyclical methodologies and the steady progression of the sequential methodologies and then adds priority and change order control.

The WaterSluice methodology will work best in a very dynamic environments as compared to the sequential or cyclical methodologies.