Development Methodologies

Prof. Dr. Josef M. Joller
jjoller@hsr.ch
SOFTWARE LIFE-CYCLE MODELS
Overview

Build-and-fix model
Waterfall model
Rapid prototyping model
Incremental model
Extreme programming
Synchronize-and-stabilize model
Spiral model
Object-oriented life-cycle models
Comparison of life-cycle models
Life-cycle model (formerly, process model)

The steps through which the product progresses

- Requirements phase
- Specification phase
- Design phase
- Implementation phase
- Integration phase
- Maintenance phase
- Retirement
Build and Fix Model

Problems
- No specifications
- No design

Totally unsatisfactory

Need life-cycle model
- “Game plan”
- Phases
- Milestones
**Characterized by**
- Feedback loops
- Documentation-driven

**Advantages**
- Documentation
- Maintenance easier

**Disadvantages**
- Specifications
  - Joe and Jane Johnson
  - Mark Marberry
Linear model

“Rapid”
Three Key Points of Prototyping

Do not turn into product

Rapid prototyping may replace specification phase—never the design phase

Comparison:
- Waterfall model—try to get it right first time
- Rapid prototyping—frequent change, then discard
Waterfall and Rapid Prototyping Models

Waterfall model
- Many successes
- Client needs

Rapid prototyping model
- Not proved
- Has own problems

Solution
- Rapid prototyping for requirements phase
- Waterfall for rest of life cycle
Somewhat controversial new approach

- Stories (features client wants)
- Estimate duration and cost of each story
- Select stories for next build
- Each build is divided into tasks
- Test cases for task are drawn up first
- Pair programming
- Continuous integration of tasks

Unusual features

- Computers are put in center of large room lined with cubicles
- Client representative is always present
- Cannot work overtime for 2 successive weeks
- No specialization
- Refactoring (improve existing system: cost / benefit?)
Microsoft’s life-cycle model

- Requirements analysis—interview potential customers
- Draw up specifications
- Divide project into 3 or 4 builds
- Each build is carried out by small teams working in parallel
- At the end of the day—synchronize (test and debug)
- At the end of the build—stabilize (freeze build)
- Components always work together
- Get early insights into operation of product
Simplified form

- Waterfall model plus risk analysis

Precede each phase by

- Alternatives
- Risk analysis

Follow each phase by

- Evaluation
- Planning of next phase
Full Spiral Model (contd)
Analysis of Spiral Model

**Strengths**
- Easy to judge how much to test
- No distinction between development, maintenance

**Weaknesses**
- For large-scale software only
- For internal (in-house) software only
Object-Oriented Life-Cycle Models

Need for iteration within and between phases
- Fountain model
- Recursive/parallel life cycle
- Round-trip gestalt
- Unified software development process

All incorporate some form of
- Iteration
- Parallelism
- Incremental development

Danger
- Infinite loop
The Rational Unified Process and the UML — developed hand-in-hand — by Rational

Contributions by major vendors
- Microsoft
- Oracle
- HP
- Texas Instruments
- IBM
- Texas Instruments
- MCI SystemHouse
- MCI SystemHouse

Standard through OMG
1. Important Features of the Iterative Approach

- Attacks risks
- Continuous integration
- Frequent, executable releases
- Continuous end user involvement
2. Manage Your Requirements

Elicit, organize, and document required functionality and constraints

Track and document tradeoffs and decisions

Business requirements are easily captured and communicated through use cases

Use cases are important planning instruments

Use Cases drives the work from analysis through test
Design, implement and test your architecture up-front!

A systematic approach to define a “good” architecture

- resilient to change by using well-defined interfaces
- by using and reverse engineering components
- derived from top rank use cases
- intuitively understandable
4. Model Software Visually

Capture the structure and behavior of architectures and components

Show how the elements of the system fit together

Maintain consistency between a design and its implementation

Promote unambiguous communication

Visual Modeling raises the level of abstraction
5. Verify Software Quality

Create tests for each key scenario to ensure that all requirements are properly implemented.

Unacceptable application performance hurts as much as unacceptable reliability.

Verify software reliability - memory leaks, bottle necks.

Test every iteration - automate test!

Software problems are 100 to 1000 times more costly to find and repair after deployment.
6. Control Changes to Software

Control, track and monitor changes to enable iterative development

**Establish secure workspaces for each developer**
- Provide isolation from changes made in other workspaces
- Control all software artifacts - models, code, docs, etc.

**Automate integration and build management**

CM is more than just check-in and check-out
Process Overview

Business Modeling
Requirements
Analysis & Design
Implementation
Test
Deployment

Configuration Management
Project Management
Environment

Inception  Elaboration  Construction  Transition

Preliminary Iteration(s)  Iter. #1  Iter. #2  Iter. #n  Iter. #n+1  Iter. #n+2  Iter. #m  Iter. #m+1
Phases in the Process

The Rational Unified Process has four phases:

- Inception - Define the scope of project
- Elaboration - Plan project, specify features, baseline architecture
- Construction - Build the product
- Transition - Transition the product into end user community
An *iteration* is a distinct sequence of activities with an established plan and evaluation criteria, resulting in an executable release (internal or external).
Each major workflow describes how to create and maintain a particular model.
In an iteration, you walk through all workflows.
Example of a Workflow
Rational Unified Process (RUP)

Trial

http://www.rational.com/products/rup/tryit/eval/gen_eval.jsp

- Username
  - jjoller@hsr.ch

- Password
  - KmVF1wP

valid approx until end of April 2002!
Conclusions

Different life-cycle models

Each with own strengths

Each with own weaknesses

Criteria for deciding on a model include

- The organization
- Its management
- Skills of the employees
- The nature of the product

Best suggestion

- "Mix-and-match" life-cycle model