The Spiral Model of Software Development and Enhancement

Barry W. Boehm, TRW Defense Systems Group
1988

Outline

- Introduction
- Previous Models
- The Spiral Model
- TRW-SPS Application
- Advantages and Difficulties
- Risk Management
- Conclusions
- Future of the Spiral Model
- Discussion

A Risk-Driven Approach

- Different idea of software development.
- How does this project affect the developers and the clients?
- How does each step in the project affect its overall development?
- Not used in previous development models.
  - Usually code-driven or document-driven.

Software Process Model

- Used to determine the order of the stages and to establish the transition criteria.
  - What do we do next?
  - How long shall we continue doing it?
- Provides guidance between the different phases of a project.
- As opposed to a software methodology.

Previous Software Process Models

- An evolution of models
  - Code & Fix
  - Stagewise & Waterfall
  - Evolutionary Development
  - Transform Model

Code & Fix

- First, elementary model
- Write code now; fix it later
- No planning involved
- Problems:
  - Code is poorly structured.
  - The software developed was usually a poor match for the users' needs.

Stagewise & Waterfall

- Born out of the shortsightedness of the Code & Fix model.
  - need for a design phase, requirements phase, and a testing phase.
- First used to develop SAGE (Semi-Automated Ground Environment), an early warning system for the Cold War era.

Stagewise

- A development process of successive phases.
  - Phases included operational plan, operational specs, coding specs, coding, parameter testing, assembly testing, shakedown, system evaluation.
- Underwent two refinements in 1970.
- Now referred to as the Waterfall Model.

Waterfall Model

- Introduced:
  - Feedback loops across multiple stages: Validation and verification steps.
  - Prototyping via a “build it twice” step alongside of requirements and design.
- Difficulties exposed even as revisions were made to the model.
  - Required elaborated documents. (Document-driven)
  - Led to pursuing stages of development in the wrong order.

Waterfall Model

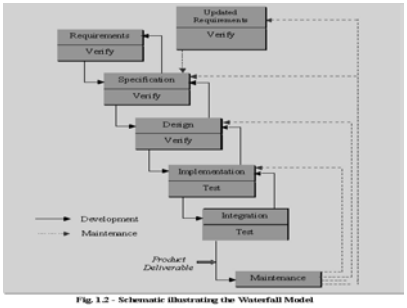


Fig. 1.2 - Schematic illustrating the Waterfall Model

Evolutionary Development

- Evolution of the system in directions based on experience.
- Provides rapid initial operational capability.
- “I can’t tell you what I want, but I’ll know it when I see it.”
- Flexible, yet uncertain approach.

Evolutionary Development

- Problems:
  - No formal design phase (same problem as Code & Fix).
  - One bad assumption – the unplanned paths “will” be compatible.
  - Hard-to-change code resulted.
  - Many problems when new software was incrementally replacing old software.

Transform Model

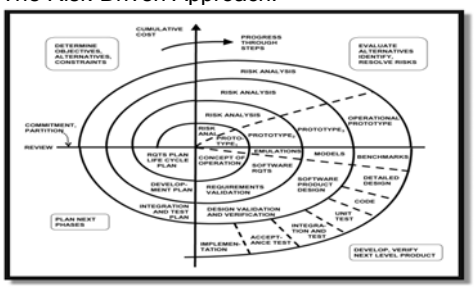
- The solution to the Evolutionary Model.
- The Transform Model contains:
  - A formal specification.
  - Automatic transformation of the specifications into code.
  - An iterative loop (for improved performance).
  - An outer iterative loop (for adjustments).
- Modifications are made to the specifications.

Transform Model

- Difficulties arose, as in the other models.
- The automatic transformation isn't so easy.
- Unplanned paths still can cause a problem (i.e. the Evolutionary Model's bad assumption).
- A knowledge-base-maintenance problem would result. Problem of choosing the best option at each transformation point.

Spiral Model

- The Risk-Driven Approach.



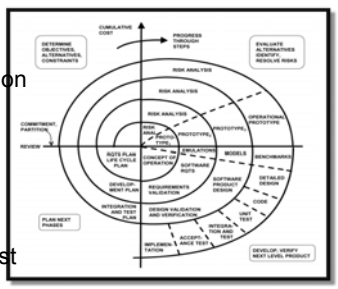
Spiral Model

- A different approach born out of the evolution of the Waterfall Model.
- Encompasses the previous models as special cases, and can make use of a combination of models.
- Risk analysis asks, " What are the areas of uncertainty, and what is the probability that they will slow the progress of development?"

A Typical Cycle

- *Risk Analysis*
- Prototype
- Design/Validation
- Planning
- *Alternatives?*
- And repeat

- Measure of Cumulative Cost and Progress



Initiating/Terminating the Process

- Initiating the process
  - Hypothesize that a particular operational mission(s) can be improved by software effort.
- *Test this hypothesis throughout.*
- Terminating
  - If a spiral violates its hypothesis then the spiral is terminated.
  - Otherwise it ends with the installation of a new or modified software product.

Cycle Requirements

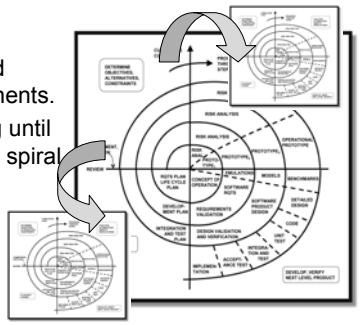
- If alternatives or uncertainties are found they must be resolved.
- Risk-driven "subsetting" allows a mixture of other software process models, as necessary, until a high-risk situation is resolved.
  - Specification-oriented (Transform or Stagewise)
  - Prototype-oriented (Waterfall)
  - Automatic-transformation oriented (Transform)
  - Simulation-oriented (Evolutionary)

Cycle Requirements

- Each cycle is completed by a review by the people concerned with the project.
- Plans for the next cycle should be introduced.
- With each succeeding level in the spiral the level of detail increases.

Overlapping Spirals

- Necessary for alternatives and parallel components.
- Stop everything until the break-away spiral is complete???
- Problems?



Applied to TRW-SPS

- An example of how the Spiral model works in a large system.
- Software Productivity System (SPS) – a group of integrated software development tools for use within TRW, as well as for other clients.
- Spiral Model rounds
  - Rounds correspond to a level in the spiral.
  - In this case, a Round 0 was needed to determine initial feasibility of the TRW-SPS project, but is not necessary for all projects.

TRW-SPS – Round 0

- Round 0: Feasibility study -- Very Basic.
  - Man-months: 5 part-time for 2 months (2MM)
  - Objectives: Significantly increase software productivity.
  - Constraints: Costs; within context of TRW culture.
  - Alternatives: Change in management, personnel, technology, facilities.
  - Risks: May be no high-leverage improvements.
  - Risk resolution: Surveys, cost models.
  - Risk resolution results: Some alternatives infeasible; significant gains can result (double in 5 yrs).
  - Plan for next phase: 6 part-time for 6 months(12MM); more analysis; develop operations.
  - Commitment: Fund next phase. (>>Next Round)

Round 1

- Round 1: Concept of Operations – More detail.
  - Man-months: 12 man-months, 6 people
  - Objectives: Double software productivity in 5 years
  - Constraints: $10,000 per person investment; preference for TRW products (LAN).
  - Alternatives: Change in office, communication, terminals, tools, CPU.
  - Risks: May miss high-leverage options, LAN price/performance, workstation costs.
  - Risk resolution: Extensive surveys, LAN benchmarking, workstation pricing.
  - Risk resolution results: Operations concept; defer OS/tools selection.
  - Plan for next phase: Partition into SDE, facilities, and management; develop prototype SDE; design -to-cost team: 15 people for one year.
  - Commitment: Develop SDE, and use it. Form a representative steering group. (>>Next Round)

Round 2

- Round 2: Top-level requirements spec. – More detail yet.
  - Man-months: 1 year, 15 people
  - Objectives: User-friendly system; integrated tools; project and personnel support.
  - Constraints: Customer-deliverable SDE, reliability.
  - Alternatives: Change in OS, host target, workstations.
  - Risks: Mismatch needs and priorities, user-unfriendly system, Unix performance, workstation compatibility. (Mini-spiral spawned)
  - Risk resolution: Survey of Unix-using organizations; workstation study.
  - Risk resolution results: Use Unix based interfaces; use tools to support early phases.
  - Plan for next phase: Tools: SREM, RTT,PDL; front end: support tools; LAN: equipment, facilities.
  - Commitment: Proceed with plans.

(>>Features)

More Rounds??

- Succeeding rounds may be necessary.
- Depends on the amount of risk.
  - Ex. The risk alleviation of the RTT preliminary design specification.
- More attractive alternatives may be found.
  - Ex. The change in UDF tool requirements.

Spiral Model Features

- Balances all of the risk elements, i.e. the high-risk elements must be lowered first.
- Offers prototyping as a risk-reduction option at any stage of development.
- It allows reworks of earlier stages as more attractive alternatives are identified.
- Detail isn't necessary until detail is needed. (Round 0)

Results

- The Software Productivity System (SPS) has grown to over 300 tools and 1.3 million instructions.
- Over 25 projects have used SPS with overall productivity up 50%; most projects have doubled productivity.
- One underestimation of Unix compatibility led to some TRW projects not using SPS.

Other Models as a Subset of the Spiral Model

- Once certain risk areas are removed other models can replace the Spiral.
  - If project is low-risk in user-interface and performance requirements, but high-risk in budget and schedule (Waterfall Model).
  - If requirements are stable, i.e. a low-risk of design, and errors produce high-risk (Two-leg model).
  - If project is low-risk in budget and schedule, and high-risk in user-interface (Evolutionary Model).
  - If automated software generation is available (Transform Model).

Advantages

- ✓It promotes reuse of existing software in early stages of development.
- ✓Allows quality objectives to be formulated during development.
- ✓Provides preparation for eventual evolution of the software product.
- ✓Eliminates errors and unattractive alternatives early.

Advantages

- ✓It balances resource expenditure.
- ✓Doesn't involve separate approaches for software development and software maintenance.
- ✓Provides a viable framework for integrated hardware-software system development.

Disadvantages

- ❖Spiral model not yet complete (in 1988).
- ❖Matching to contract software
  - Internal projects have more freedom.
  - Contract software demands total control and a full picture of the deliverables in advance.
- ❖Relying on risk-assessment expertise.
- ❖Need for further elaboration of spiral model steps.
  - Milestones and specifications.
  - Guidelines and checklists.

Risk Management

- The Spiral model relies heavily on the assessment of risks.
- It provides early identification of the top risk items.
- Improper evaluation of risks may lead inexperienced developers down the wrong path. May give an illusion of progress.
- How can a group enhance their risk management skills/level?

Software Risk Management Plan

1. Identify the project's top 10 risk items.
2. Present a plan for resolving each risk item.
3. Update list of top risk items, plan, and results monthly.
4. Highlight risk-item status in monthly project reviews.
5. Initiate appropriate corrective actions.

--obvious.

Conclusions

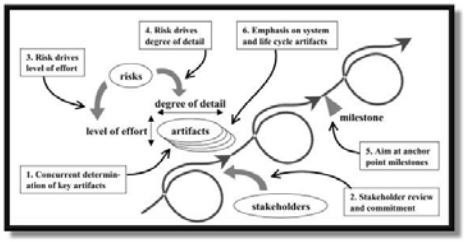
- The paper draws four conclusions:
  1. The risk-driven nature provides adaptability for a full range of software projects.
  2. The model has been successful in a large application, the TRW-SPS.
  3. The model is not yet fully elaborated.
  4. Even partial implementations of the model, such as the risk management plan, are compatible with the other process models.

The Spiral Model Today

- Still a *risk-driven* model.
- Two definitions:
  - One is a *cyclic* approach for incrementally growing a system's degree of definition and implementation while decreasing its degree of risk.
  - The other is a set of *anchor point milestones* for ensuring stakeholder commitment to feasible and mutually satisfactory system solutions.
- Used by the Department of Defense (5000.2 project).
- The Six Spiral Essential introduced by Barry Boehm in 2000.

Six Spiral Essential

• Six essential attributes which every spiral development process must incorporate.



The Spiral Model Today

- A more complete Spiral Model
  - Ability of the Spiral Model to work on external projects (stakeholder review).
  - A more elaborated method in proceeding through the spiral.
    - Use of milestones and the six “essentials”.
    - The awareness of “hazardous spiral look-alikes”, as well as possible invariants in the model.
- Spiral Model sources:
  http://www.stsc.hill.af.mil/crosstalk/2001/05/boehm.html

Discussion

- Since the Spiral Model is based on the human ability to assess risk, is the model prone to human errors?
- Is this model just a combination of all of the other software process models?
- Can we get lost in the spiral, i.e. endless levels, gigantic breakaway spirals?