Software Development Processes

Michael Zhu*
Zhejiang Normal University
Software Project Management

Abstract

In software engineering, a software development methodology (also known as a system development methodology, software development life cycle, software development process, software process) is a splitting of software development work into distinct phases (or stages) containing activities with the intent of better planning and management. It is often considered a subset of the systems development life cycle. The methodology may include the pre-definition of specific deliverables and artifacts that are created and completed by a project team to develop or maintain an application. (e.g., [Elliott 2004])In this article, I will introduce what is Software Development, Project Characteristics and Considerations, Software process models and Selecting process model.

Keywords: project characteristics, project management, planning, software process models, selecting process model

1 Introduction

Common methodologies include waterfall, prototyping, iterative and incremental development, spiral development, rapid application development, extreme programming and various types of agile methodology. Some people consider a life-cycle "model" a more general term for a category of methodologies and a software development "process" a more specific term to refer to a specific process chosen by a specific organization. For example, there are many specific software development processes that fit the spiral life-cycle model.

- Software Process: the set of activities, methods, and practices that are used in the production and evolution of software;
- Software Process Model: one specific embodiment of a software process architecture;
- Modelling: To provide a common understanding; To locate any inconsistencies, redundancies and omissions; To reflect the development goals and provide early evaluation; To assist the development team to understand any special situation;
- Model/Approac: Look at risks and uncertainties(are requirement well understood?are technologies to be used well understood?), Look at the type of application being built(information system? embedded system?criticality? differences between target and development environments?), Clients own requirements(need to use a particular method);

2 About Project

- Project Characteristics:
  Data oriented or control oriented system? General package or application specific? A particular type of system for which specific tools have been developed? Safety-critical system? Nature of the hardware/software environment?
- Project Considerations
  Control systems; Information systems; General applications; Specialized techniques; Hardware environment; Safety-critical systems; Imprecise requirements;
- Technical Constraints
  Type of the system to be developed; Risks and uncertainties of the project; User requirements concerning implementation;
- Technical Approach
  Selected methodology or process model(s); Development methods; Required software tools; Target hardware/software environment;
- Technical Implementation
  Development environment; Maintenance environment; Training;
- Technical Implications
  Project products and activities: effect on schedule duration and overall project effort; Financial: report used to produce costings;

3 Model

- Waterfall Model

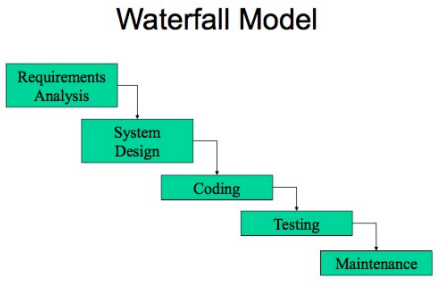

Figure 1: *Waterfall model - classical;one-shot approach effective control;limited scope of iteration long cycle time;not suitable for system of high uncertainty; Waterfall Variations - Sashimi waterfall: activities are allowed to overlap;Waterfall with sub projects: implementation of different components proceeds in parallel;Waterfall with risk reduction: an initial risk analysis; helps mitigate risks in later phases of implementation*

- V Model

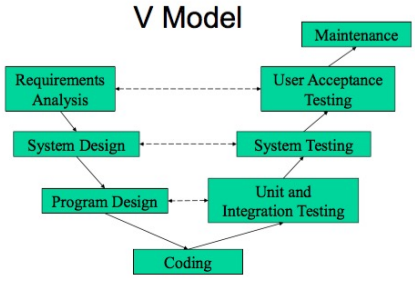

Figure 2: *V model - Additional validation process introduced; Relate testing to analysis and design; Loop back in case of discrepancy;*

*e-mail:383974871@qq.com

The article is based on work to support Software Project Management.
Software Project Management (2016/17)
Author Name: Michael Zhu
University: Zhejiang Normal University
Title: Software Development Processes
Supervisor: Dr. Kenwright

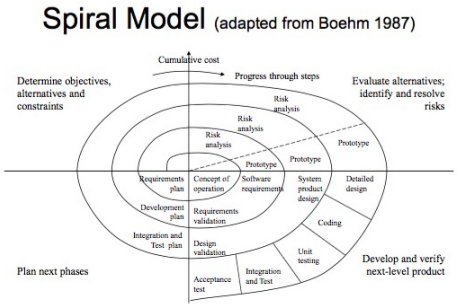

Figure 3: *Spiral Model - Evolutionary approach; Iterative development combined with risk management; Risk analysis results in go, re-do, no-go decision*

- Spiral Model

4 Prototype

- Prototyping Model
  Goals: meet (some) user requirements at an early stage; reduce risk and uncertainty; verify a design or implementation approach;
- Classification of Prototype
  Throw-away: After users agree the requirements of the system, the prototype will be discarded. Evolutionary: Modifications are based on the existing prototype. Incremental: Functions will be arranged and built accordingly.
- Prototyping Model

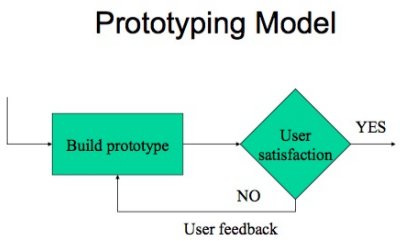

Figure 4

- Benefits of Prototyping
  Learning by doing; Improved communication; Improved user involvement; Clarification of partially-known requirements;
- Prototyping Sequences
  Requirements gathering Quick design; Prototype construction Customer evaluation Refinement; Loop back to quick design for fine tuning Product engineering;
- Benefits of Prototyping
  Demonstration of the consistency and completeness of a specification; Reduced need for documentation; Reduced maintenance costs; Feature constraint; Production of expected results for testing real system;
- Drawbacks of Prototyping
  Users sometimes misunderstand the role of the prototype; Lack of project standards possible Lack of control; Additional expense; Machine efficiency; Close proximity of developers;
- Forms of Prototypes
  Mock-ups; Simulated interaction; Partial working model;
- Prototype Products
  Human-computer interface; System functionality;
- Incremental Model Break system into small components; Implement and deliver small components in sequence; Every delivered component provides extra functionality to user;
- Iterative Model Deliver full system in the beginning; Enhance functionality in new releases

5 Overview

A particular development team may also agree to programming environment details, such as which integrated development environment is used, and one or more dominant programming paradigms, programming style rules, or choice of specific software libraries or software frameworks. These details are generally not dictated by the choice of model or general methodology. So learn it is very important.

6 Conclusion

Software development processes is a deep knowledge of the subject. It contains some development' model and we can be use it when we are developing software. If we want to develop software we must learn it well.

Acknowledgements

Thanks to Dr. Benjamin Kenwright.

References

ELLIOTT, G. 2004. *Global Business Information Technology: an integrated systems approach.* Pearson Education. 1