Software development process

Wu Yangfan 13211236
Zhejiang Normal University
Software Project Management

Abstract

This article gives a brief about software development process. This article may help you realize the process of software development. software development process is a splitting of software development work into distinct phases (or stages) containing activities with the intent of better planning and management.

Introduction

Software development process is software design idea and method of general process, including the function and implementation of algorithm and method of software design, the general structure design and software module design, programming and debugging, alignment and testing, and write, submit application and a series of operations.

In software engineering, a software development 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.

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

History

The software development methodology (also known as SDM) framework didn't emerge until the 1960s. According to Elliott (2004) the systems development life cycle (SDLC) can be considered to be the oldest formalized methodology framework for building information systems. The main idea of the SDLC has been "to pursue the development of information systems in a very deliberate, structured and methodical way, requiring each stage of the life cycle—from inception of the idea to delivery of the final system—to be carried out rigidly and sequentially" within the context of the framework being applied. The main target of this methodology framework in the 1960s was "to develop large scale functional business systems in an age of large scale business conglomerates. Information systems activities revolved around heavy data processing and number crunching routines".

Methodologies, processes, and frameworks range from specific proscriptive steps that can be used directly by an organization in day-to-day work, to flexible frameworks that an organization uses to generate a custom set of steps tailored to the needs of a specific project or group. In some cases a "sponsor" or "maintenance" organization distributes

an official set of documents that describe the process. Specific examples include:

1970s

Structured programming since 1969

Cap Gemini SDM, originally from PANDATA, the first English translation was published in 1974. SDM stands for System Development Methodology

1980s

Structured systems analysis and design method (SSADM) from 1980 onwards

Information Requirement Analysis/Soft systems methodology

1990s

Object-oriented programming (OOP) developed in the early 1960s, and became a dominant programming approach during the mid-1990s

Rapid application development (RAD), since 1991

Dynamic systems development method (DSDM), since 1994

Scrum, since 1995

Team software process, since 1998

Rational Unified Process (RUP), maintained by IBM since 1998

Extreme programming, since 1999

2000s

Agile Unified Process (AUP) maintained since 2005 by Scott Ambler

Disciplined agile delivery (DAD) Superseded of AUP

Estimate

Estimation is foundation for all project planning activities

Estimation Steps:

1. Description of product scope
2. Decomposition of problem into set of smaller problems
3. Each sub problem is estimated using historical data, software metrics and experience (from past projects) as guides.
4. Problem complexity and risks are considered before final estimate is made.

Models

Software Process Models:

1. Waterfall Model :

   classical
   one-shot approach
   effective control
   limited scope of iteration
   long cycle time
   not suitable for system of high uncertainty

2. V Model:

   Additional validation process introduced
   Relate testing to analysis and design
   Loop back in case of discrepancy

3. Spiral Model

   Evolutionary approach
   Iterative development combined with risk management
   Risk analysis results in "go, re-do, no-go" decision
   Four major activities
   (1)Planning
   (2)Risk analysis
   (3)Engineering
   (3)Evaluation

4. Prototyping Model
5. Phased Development Model
   (1)incremental development model
   (2)iterative development model
6. Operational Specification Model
7. Transformation Model

Figure 1.

Schedule

The project schedule is a calendar that links the tasks to be done with the resources that will do them.
(1)Before a project schedule can be created, the project manager must have a work breakdown structure (WBS) and estimates.

(2)The schedule is part of the project plan

*e-mail: 452180789@qq.com

Software project management(2016-2017)
Zhu shengqi
Zhejiang Normal university
Supervisor Dr.Kenwright

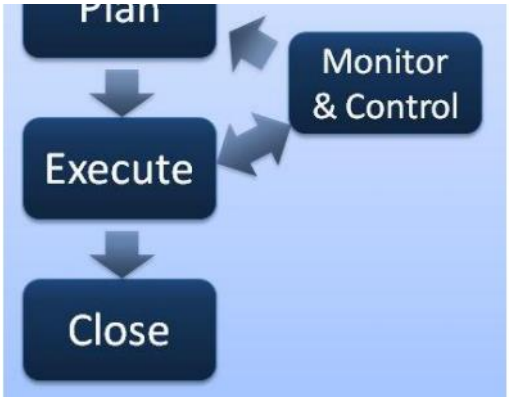

Figure 2:

Building the project schedule

Allocate resources

(1)For each task in the WBS, one or more resources must be assigned
(2)Choose person or people for each task based on qualifications, familiarity and availability ωTake overhead into account when calculating the duration of each task

Identify dependencies

(1)A task has a dependency if it involves an activity, resource or work product which is subsequently required by another task

(2)Tasks may have dependencies because they require the same resource

Every dependency has a predecessor, or a task that must be begun, in progress, or completed, for another task to begin

(1)Identify the type of predecessor for each dependency.

Create the schedule

(1)Most project schedules are represented using a Gantt chart

(2)The Gantt chart shows tasks, dependencies and milestones using different shapes

Optimize the schedule

(1) The critical path is the sequence of tasks that represent the minimum time required to complete the project. • If a task is only on the critical path when delaying that task will delay the project. • Allocating resources to tasks on the critical path will reduce the project schedule; allocating them to other tasks will have less effect.

(2)A resource is over-allocated if more than 100% allocated to multiple tasks simultaneously • If any resource is over-allocated, it means that there is a dependency between two tasks which was not discovered. • When this happens, the schedule is guaranteed to be inaccurate. Find and fix over-allocated resources.

Conclusion

Reference

Kent Beck, Extreme Programming, 2000.