

Copy of Methodology

<Block>:1

q1: 软件工程的复杂性

q2: 解决问题的策略

q3: 软件工程的角色

q4:用例图

Part 2: <PPT1>

q3软件工程师的角色

Role of software engg

- A bridge from customer needs to programming implementation
- **Customer:** Requires a computer system to achieve some business goals by user interaction or interaction with environment in a specified manner
- **SE:** To understand how the system-to-be(系统开发) needs to interact with the user or the environment so that customer's requirement is met and **design** the software-to-be (软件开发 to-be未来的)
- **Programmer's Task:** To implement the software-to-be designed by the software engineer

Monet-machine problem example:

In this model, ATM machine = system-to-be environment = Bank's remote datacenter

Bank customer = customer

q2解决问题的策略

Problem-solving Strategy

- Divide-and-conquer(First law)
 - Identify the logic part of system that each solves a part of the problem
 - Know steps in the process how it is currently done
 - Result : A model of the problem domain(domain model)

Software Development Method

- Method :Write down the problem → think hard → write down the answer
- Waterfall: finish this step before moving to the next
 - Each activity confined to its phase
 - Unidirectional no way back
 - Finish the phase before moving to next
- Iterative + Incremental: Develop increment of functionality and repeat in feedback loop
- Agile: Continuous user feedback essential; feedback loops on several levels of granularity (粒度, 间隔尺度)

q3 用例图

Problem Domain

Elements:

- System to be developed
- Actors
 - Agents external to the system that interact with it
- Concepts/Objects
 - Agents working inside the system to make it function
- Use cases
 - Scenarios(情景) for using the system,

Software Measurement

- What to measure?
 - Project → for budgeting and scheduling
 - Product → for quality assessment

Work Estimation Strategy

- Make initial guess for a little part of work
 (sizing The problem).
- Do a little work to find out how fast you can go
- Make correction on your initial estimate
- Repeat until no corrections are needed or work is completed

Exponential Cost of Estimation:(34)

- $y \rightarrow$ accuracy $x \rightarrow$ Estimation cost
-

Random

- First law of software engineering: Software is willing to learn problem domain
- Specifying software problems and solutions is like cartoon strip writting. Most of us are not artists, so we will use something less exciting: UML symbols
- Second law of software Engineering: Software should be written for people first
- Judging feasibility or quality of a design requires great deal of (一大笔) domain knowledge (领域知识)
-


Concept Maps

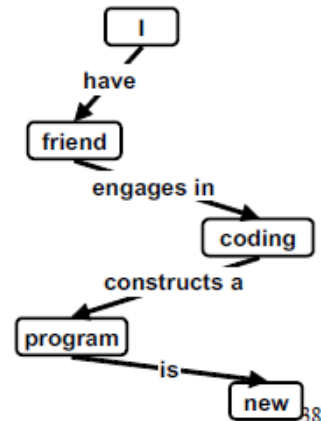
Useful tool for problem domain description

SENTENCE: "My friend is coding a new program"

translated into propositions

Proposition	Concept	Relation	Concept
1.	I	have	friend
2.	friend	engages in	coding
3.	coding	constructs a	program
4.	program	is	new

 Search the Web for Concept Maps



<Block >:2

q3: Scrum中的角色

Part1: <PPT 3>

Method of agile to solve the problem

New method is Agile Software Development Methodology

- Scrum
- Extreme Programming
- Adaptive Software Development(ASD)
- Dynamic System Method(DSDM)

A statement of Values (价值陈述)

- Individual and interactions over processes and tools
- Working software over comprehensive document
- Customer collaboration(合作) over contract negotiation(合同谈判) 超过合同谈判的用户合作
- Responding to change over following a plan

什么是Scrum?

Scrum是一种敏捷软件开发的方法学，用于迭代式增量软件开发过程。Scrum在英语是橄榄球运动中争球的意思。

Scrum

Scrum is an agile process that allows us to focus on delivering the highest business value in the shortest time.

It allows us to rapidly and repeatedly inspect(检查) actual(目前的) working software (every two weeks to one month)

The Business (公司) set the priorities, and teams self-manage to determine the best way to deliver(交付) highest prioritized feature

Every two weeks to a month anyone can see real working software and decide to release it as is or continue to enhance for another iteration

Scrum Frame work

- Role: Product Owner, Scrum Master, Team
- Ceremonies:
 - Sprint Planning,
 - Sprint review,
 - Sprint Retrospective(adj. 回顾的)
 - Daily Scrum Meeting

- Artifacts: Product Backlog, Sprint Backlog,

Word ex: What is the sprint?

- A month-long iteration(迭代), which is incremented a product functionality
- No outside influence with the Scrum team during the Sprint
- Each Sprint begin with a daily scrum meeting

Word ex: What is Daily Scrum

Daily Scrum is a meeting in which members make commitments(承诺) to each other and to the Scrum Master.

- Parameters
 - Daily
 - 15-minutes
 - Stand-up
 - Not for problem solving
- Three questions
 - What did you do yesterday
 - What will you do today
 - What obstacles(障碍) are in your way?

Word ex: Sprint Review Meeting

- Team presents It accomplished during the sprint
- Typically <takes the form of> (以..形式存在着) a demo of new features or underlying architecture
- Informal
- Participants
 - Customers

- Management
- Product Owner
- Team

Word ex: Sprint Retrospective Meeting

- Scrum Team only
- Feedback meeting
- Three questions
 - Start
 - Stop
 - Continue

Word ex: Product Backlog

- A list of all desired work on the project
- List is prioritized by Product Owner
- Spreadsheet
- Created during the Sprint Planning Meeting

q3:Scrum中的角色

The role in a development process

- Product owner:
 - Define the feature of the product
 - Decide the release date and content
 - Be responsible for the profitability of the product
 - Order the priorities of the feature
 - Accept or reject result
- Scrum Master

- Represent management to the project
 - Responsible for enacting(颁布) Scrum Value and practices
 - Removes impediments
 - Ensure the team is fully functional and productive
 - Enable close cooperation across all roles and functions
 - Shield the team from external interference
 - Scrum Team
 - Typically 5-10 people
 - Cross-functional
 - QA, Programmers, UI designer, etc.
 - Members should be full-time(全职)
 - Team are self-organizing(自组织)
 - Membership(成员身份) can change only between sprints
-

<Block >:3

q1: 什么是需求

q2: 需求产生的三步

q3: 为什么要acceptance teset

q4:如何产生一个需求

q5:分析需求需要哪些步骤

q6:三种需求类型

q7:什么是验收测试，这包括哪些内容

Part2: <PPT 4>

Software Requirement

q1:什么是需求

What is requirement?

A requirement specifies the business functions that the user will be able to perform using the system-to-be in different “situations” or “contexts”, and the kind of experience the user will have during this work

– Other concerns, such as how the system will manage the resources (computing, network, ...), how the system will manage and protect user’s data, etc

q2 :需求产生的三步

Requirement process

Requirement gathering → Requirement analysis → requirement specification

- Requirements gathering
 - helps the customer to define what is required:
 - Requirements analysis
 - Refining and modifying the gathered requirements
 - Requirements specification
 - Documenting the system in a formal manner.
-

为什么要acceptance test?

Practical Requirement Engineering

- Test your idea in practice and use the result in further work, iterating through these creative and evaluative steps until a solution is reached
 - Define the criteria(条件, 标准) for measuring the success (“**acceptance tests**”)
 - Avoid random trial-and-error by relying on domain knowledge (from publications or customer expertise)
-

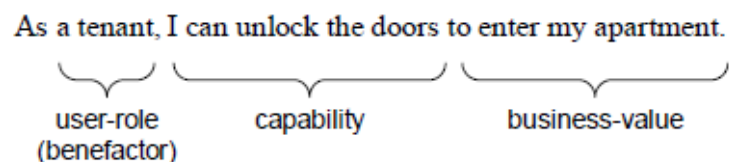
如何产生一个需求

How to derivate a system requirement

- Problem detected:
difficult access or unwanted intrusion
(plus: operating household devices and
minimizing living expenses)
- Analysis of the Causes:
 - User forgets to lock the door or turn off the devices
 - User loses the physical key
 - Inability to track the history of accesses
 - Inability to remotely operate the lock and devices
 - Intruder gains access
- System Requirements: based on the selected
cause

Requirements as user stories:

- 4-11



User story Requirement 4-12

q5:分析需求需要的步骤

Requirement Analysis Activities

- Not only refinement of customer requirements but also feasibility and how realistic
- Needs to identify the points where business policies need to be applied
 - **Explicit identification of business policies (BP) is important for two reasons**

- Making the need for BP explicit allows involving other stakeholders in decision about the solutions to adopt—Helps to involve others, particularly the customer, in decision making about each policy to adopt
 - Making the need for BP explicit allows involving other stakeholders in decision about the solutions to adopt—Helps to involve others, particularly the customer, in decision making about each policy to adopt
 - 4-19 example
-

q6:三种需求类型

Type of requirement

- Functional Requirements
 - Non-functional requirements
 - FURPS+
 - Functionality, Usability , Reliability, Performance, Supportability
 - User interface requirement
 - Don't waste time on creating a interface
-

q7:什么是验收测试，并且包括什么

Acceptance Test

Define the criteria(条件，标准) for measuring the success

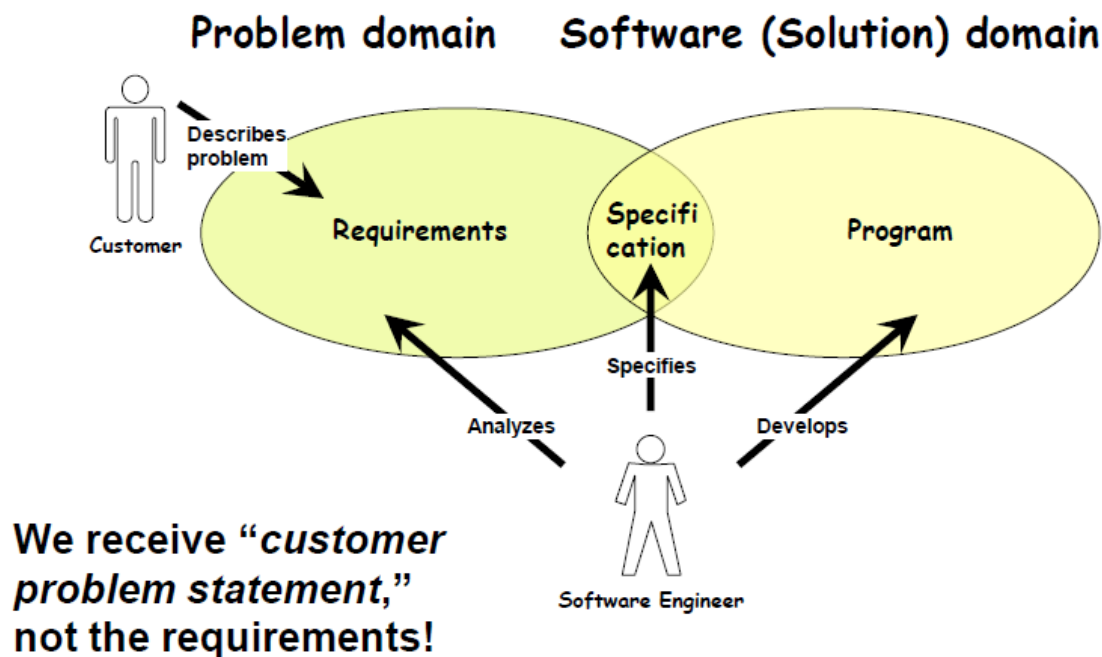
- An acceptance test describes whether the system will pass or fail the test, given specific input values
- Conducted by the customer throughout the project, not only at the end
- Each requirement describes for a given “situation” (i.e., system inputs), the output or behavior the system will produce

- An **acceptance test** specifies a set of scenarios for determining whether the (part of the) system meets the customer requirements
- An **acceptance test case** specifies, for a given “situation” or “context” (defined by current system inputs), the output or behavior the system will produce in response

Agile methods for effort estimation and work Organization

- Size points assigned to each user story
 - Total work size estimate:
 - Total size = $\sum(\text{points-for-story } i) \quad i=1 \dots N$
 - Velocity(=productivity) estimated from experience
 - Estimate the work duration = point size/velocity
-

Random



<Block >:4

q1: 什么是需求

q2: 需求产生的三步

q3: 为什么要acceptance teset

q4:如何产生一个需求

q5:分析需求需要哪些步骤

q6:三种需求类型

q7:什么是验收测试，这包括哪些内容

Part3: <PPT 5> System Architecture

Why we want to decompose system

- Tackle complexity by "divide-and-conquer"
 - See if some parts already exist can be reused
 - Focus on creative parts and avoid "reinventing the wheel"
-

Definition of Software Architecture Defination

- **Software architecture: a set of high-level decisions that determine the structure of the solution**
 - Principal decisions made throughout the development and evolution of a software system
- Decisions to use well-known solutions that are proven to work for similar problems
- Software Architecture is not a phase of development
 - Does not refer to a specific product of a particular phase of the development process (labeled “high-level design” or “product design”)