System Analysis and Design

L06. Writing Use Cases

Topics

- Guidelines to Writing Use Cases
- How to Find Use Cases
- Applying UML: Use Case Diagrams
- Evolve Use Cases Across the Iterations (within UP)

Guidelines to Writing Use Cases

Essential Style Use Cases

- Most programs are dependent upon a particular user interface.
- However, avoid constraining your program too early:
  - "The user keys an ID and password into a dialog box and presses the OK button."
  - "The user identifies himself to the system." ← better
  - The latter allows for biometric ID, keyin, etc.
- **Essential Style Use Cases**
  - Focus on the essence, or basic idea, not the details of implementation 关注基本思想，不要讨论实现细节.

建议：

Write in a Essential (UI-Free) Style

- **Essential Style** Writing
  - Write use cases in an essential style
  - keep the user interface out and focus on actor intent.
  - Essential style contrasts with concrete style.
- Avoid **Concrete Style** During Early Requirements Work.
  - **Concrete style:** user interface decisions are embedded in the use case text.
- Write **terse** use cases. Delete "noise" words.

书写简明扼要的用例.

Write Black-Box Use Cases

- **Black-box use cases** do not describe the internal workings of the system, its components, or design.
- Rather, the system is described as having *responsibilities*
- Black-Box, what
  - "The system records the sale"
- not Black-Box, how
  - "The System writes the sale record to a database"

Take an Actor and Actor-Goal Perspective

- What are use cases? (RUP)
  "A set of use-case instances, where each instance is a sequence of actions a system performs that yields an observable result of value to a particular actor."
- Focus on the users, asking about their goals
- Understand what the actors consider a valuable result

HOW TO FIND USE CASES

Finding Use Cases

- Use cases are defined to satisfy the goals of the primary actors. Hence, the basic procedure is:
  1. Choose the system boundary 选择系统边界
  2. Identify the primary actors 确定主参与者
  3. Identify the goals for each primary actor 确定每个参与者的目的
  4. Define use cases that satisfy these goals 依据目标定义用例

- Of course, in iterative and evolutionary development, not all goals or use cases will be fully or correctly identified near the start. It's an evolving discovery.

Step 1: Choose the System Boundary

- If the definition of the boundary of the system under design is not clear,
  - it can be clarified by further definition of what is outside (the external primary and supporting actors).
- Once the external actors are identified, the boundary becomes clearer.
- For example
  - Is the complete responsibility for payment authorization within the system boundary?
  - No, there is an external payment authorization service actor.

Steps 2 and 3: Find Primary Actors and Goals

- It is artificial to strictly linearize the identification of primary actors before user goals;
- In a requirements workshop, people brainstorm and generate a mixture of both.
- Sometimes, goals reveal the actors, or vice versa. 互相揭示
- ***Guideline:*** Brainstorm the primary actors first, as this sets up the framework for further investigation.

Questions to Help Find Actors and Goals

In addition to obvious primary actors and goals, the following questions help identify others that may be missed: 下列的问题可以帮助找出其它可能遗漏的参与者和目标.

- Who starts and stops the system?
- Who does user and security management?
- Who does system administration?
- Is “time” an actor because the system does something in response to a time event?
- How are software updates handled?
- Who gets notified of problems?

How to Organize the Actors and Goals?

There are at least two approaches:

1. As you discover the results, draw them in a use case diagram, naming the goals as use cases.
2. Write an actor-goal list first, review and refine it, and then draw the use case diagram.

How to Organize the Actors and Goals? Actor-Goal List

A section in the UP Vision artifact.

Actor	Goal
Cashier	Process Sales Process rentals Handle returns Cash in Cash out
Manager	Start up Shut down
System administrator	Add/Modify/Delete users Manage security Manage System tables
…	…

Why Ask About Actor Goals Rather Than Use Cases?

- Actors have goals and use applications to help satisfy them.
- The viewpoint of use case modeling is to find these actors and their goals, and create solutions that produce a result of value.
- This is slight shift in emphasis for the use case modeler.
- Imagine we are together in a requirements workshop. We could ask either:
  - "What do you do?" (roughly a task-oriented question) or,
  - "What are your goals whose results have measurable value?"
- Prefer the second question.

面向任务

第二个问题更关注业务的价值，以及人们希望的系统功能.

Who is the Primary Actor?

- Why is the cashier, and not the customer, a primary actor in the use case *Process Sale*?
- Depends upon the system boundary 取决于我们确定的系统边界

Other Ways to Find Actors and Goals? Event Analysis 事件分析

- Another approach is to identify external events. 识别外部事件
- What are they, where from, and why?
  Often, a group of events belong to the same use case. For example:

External Event	From Actor	Goal/Use Case
enter sale line item	Cashier	process a sale
enter payment	Cashier or Customer	process a sale
…		

用事件分析 找出更多的 actors and goals

Step 4: Define Use Cases

- In general, define one use case for each user goal.
- Name the use case similar to the user goal. Start the name of use cases with a verb. 用例与用户目标对应.
  For example, Goal: process a sale; Use Case: *Process Sale*.
- A common exception to one use case per goal is to collapse CRUD separate goals (create, retrieve, update, delete) into one CRUD use case, idiomatically called *Manage <X>*. 特例：多用户目标对应一个用例.
- For example, the goals "edit user," "delete user," and so forth are all satisfied by the *Manage Users* use case.

用测试帮助我们找到有用的用例.

What Tests Can Help Find Useful Use Cases?

- Which is a Valid Use Case?
  - Negotiate a supplier contract
  - Handle returns
  - Log in
  - Move pieces on a game board

哪一个更可能为有效的用例?

- An argument can be made that all of these are use cases *at different levels*, depending on the system boundary, actors, and goals.

与层次级别有关!

The Boss test

老板测试

- Your boss asks, "What have you been doing all day?" You reply: "Logging in!" Is your boss happy?
- If not, the use case fails the Boss Test, which implies it is not strongly related to achieving results of measurable value.
- It may be a use case at some low goal level, but not the desirable level of focus for requirements analysis.

低层次的目标，可以构成用例，但不是需求分析层用例

- That doesn't mean to always ignore boss-test-failing use cases.
  User authentication may fail the boss test, but may be important and difficult.

例外情况

Other Tests

- **The Elementary Business Process test:** EBP测试
  Task performed by
  - one person
  - at one place
  - at one time
  - in response to a business event

  that adds value and leaves data in a consistent state.

  也许不过苛刻，强调了可度量的业务价值。
- Focus on use cases that reflect EBPs. 请关注反映EBP的用例
- **The size test:** 尺寸测试
  Fully dressed is 3-10 pages.

Applying the Tests

- Negotiate a Supplier Contract
  - Much broader and longer than an EBP. Could be modeled as a *business* use case, rather than a system use case.
- Handle Returns
  - OK with the boss. Seems like an EBP. Size is good.
- Log In
  - Boss not happy if this is all you do all day!
- Move Piece on Game Board
  - Single step fails the size test

Reasonable Violations of Tests

- Although the majority of use cases should satisfy the tests, exceptions are common.
- It is sometimes useful to write separate subfunction-level use cases representing subtasks or steps within a regular EBP-level use case. 子用例
- Some subfunction may fail boss test but is complex and useful.

APPLYING UML

Applying UML: Use Cases Diagrams

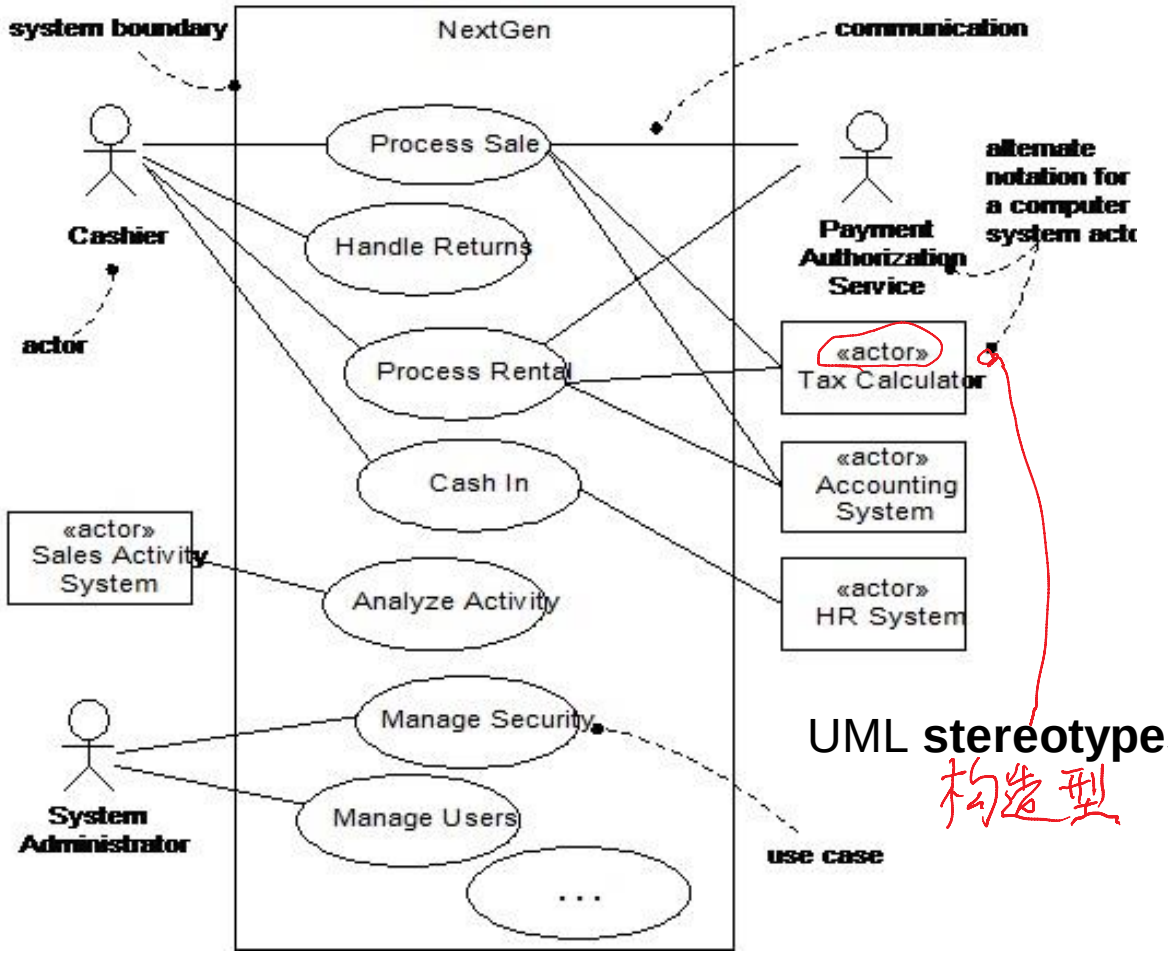
- Use case diagrams and use case relationships are secondary in use case work.
- Use cases are text documents. Doing use case work means to write text.
- Do not use diagrams as a substitute for thinking.

Use Cases Diagrams as Context Diagram

- A use case diagram is an excellent picture of the system context;
- It makes a good **context diagram**, showing
  - the boundary of a system,
  - what lies outside of it,
  - and how it gets used.
- It serves as a communication tool that summarizes
  - the behavior of a system and its actors.

A simple diagram can add clarity

The UML provides use case diagram notation to illustrate the names of use cases and actors, and the relationships between them

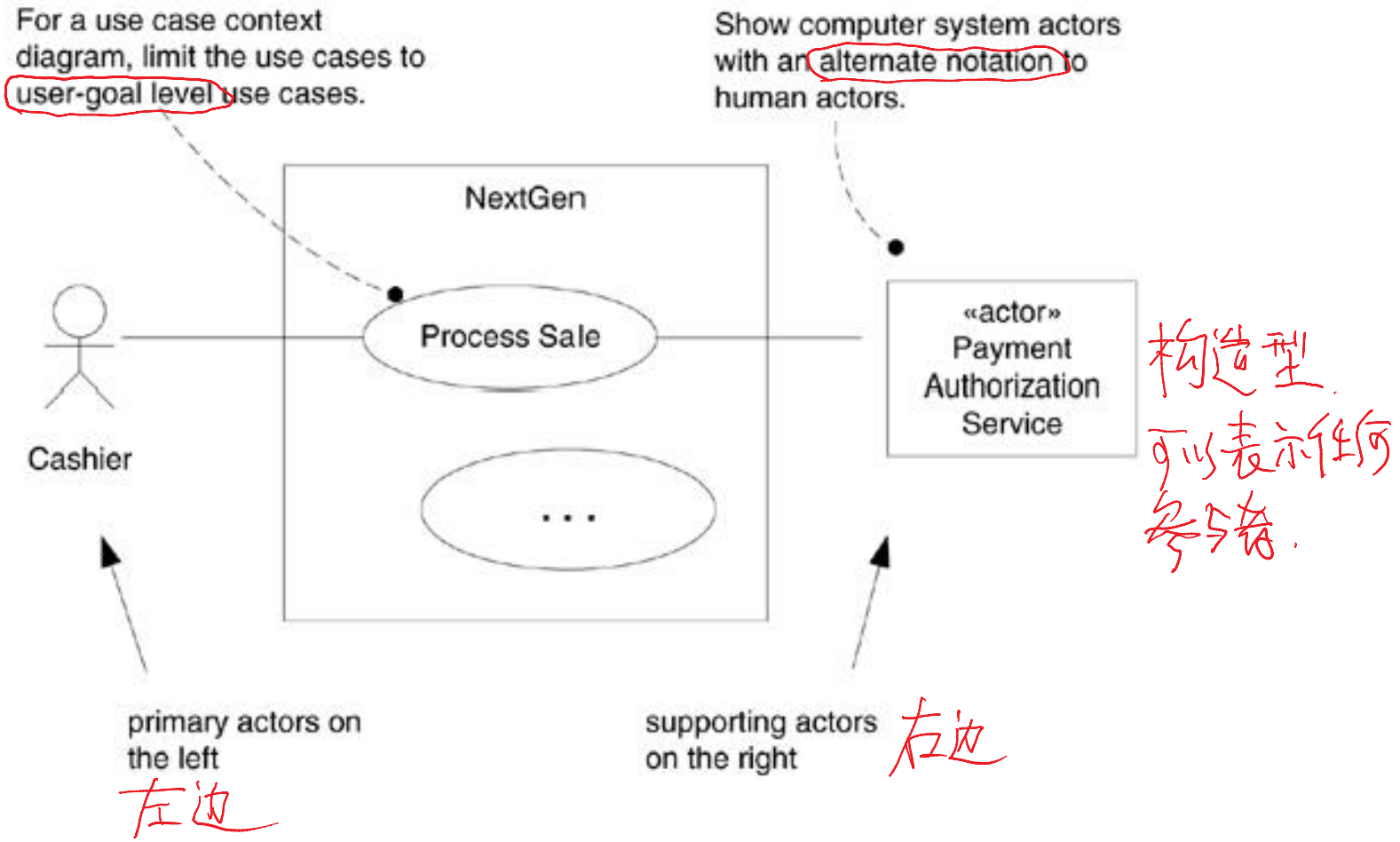


Guideline: Draw a simple use case diagram in conjunction with an actor-goal list.

Notation Suggestions

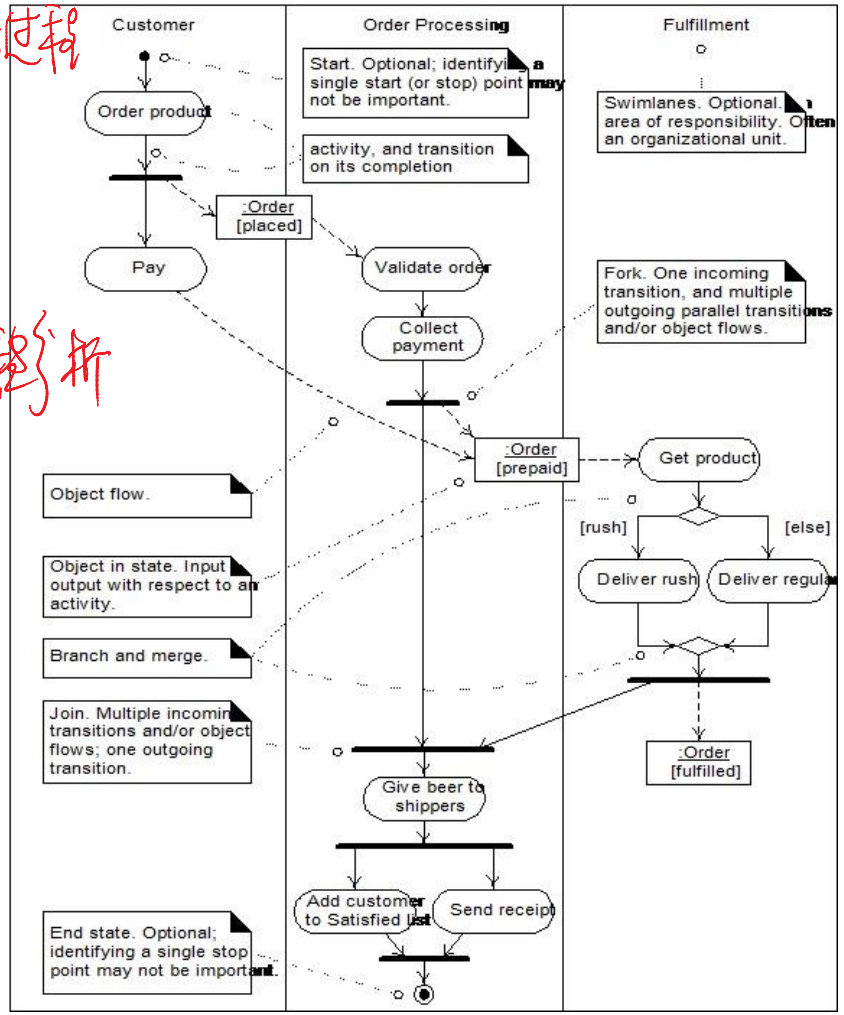
不要太过看中用例图，尽量让其短小简单。

Guideline: Downplay Diagramming, Keep it Short and Simple



Applying UML: Activity Diagrams

- **activity diagrams**: to visualize workflows and business processes.
- use cases involve process and workflow analysis,
- activity diagrams can be a useful alternative or adjunct to writing the use case text,
- especially for *business* use cases that describe complex workflows involving many parties and concurrent actions.



在环境中描述需求

Requirements in Context

- Use cases can organize a set of requirements in the context of a typical use of the system 用例可以组织需求.
- Replace *low-level feature lists* with use cases 一致简洁.
- However, use cases are not the only necessary requirements artifact. 但用例并非唯一的需求描述工件.
- Non-functional requirements, report layouts, domain rules, and other hard-to-place elements are better captured in the UP Supplementary Specification.

UP的附加说明文档包括非功能性需求等.

在环境中描述需求

Requirements in Context

- High-level feature lists (*system features,* added to a Vision document) can usefully summarize system functionality. 可以使用高层特征列表来总结系统功能
- Some applications need feature-driven viewpoint
  - don't create use cases for these
  - use cases do not really fit

} 此时仍可以用特征列表.

Evolve Use Cases Across the Iterations

Use Cases Driven Development

用例驱动的开发

Use cases are central to the UP iterative methods. The UP encourages **use-case driven development** . This implies:

- Functional requirements are in use cases (the Use-Case Model) 用例模型
- Use cases are an important part of iterative planning. 迭代计划
  - The work of an iteration is, in part, defined by choosing some use case scenarios, or entire use cases.
  - And use cases are a key input to estimation.
- Use-case realizations drive design 用例实现驱动的设计
  - The team designs collaborating objects and subsystems in order to realize the use cases.
- Use cases influence organization of user manuals 使用手册的结构
- Testing corresponds to use case scenarios 基于用例的测试
- UI shortcuts may be created for most common scenarios

UI菜单选项基于用例场景

Evolve Use Cases Across the Iterations

to develop Requirements 一个样例

Discipline	Artifact	Incep 1 week	Elab 1 4 weeks	Elab 2 4 weeks	Elab 3 3 weeks	Elab 4 3 weeks
Requirements	Use-Case Model	2-day requirements workshop. Most use cases identified by name, and summarized in a short paragraph. Pick 10% from the high-level list to analyze and write in detail. This 10% will be the most architecturally important, risky, and high-business value.	Near the end of this iteration, host a 2-day requirements workshop. Obtain insight and feedback from the implementation work, then complete 30% of the use cases in detail.	Near the end of this iteration, host a 2-day requirements workshop. Obtain insight and feedback from the implementation work, then complete 50% of the use cases in detail.	Repeat, complete 70% of all use cases in detail.	Repeat with the goal of 80-90% of the use cases clarified and written in detail. Only a small portion of these have been built in elaboration; the remainder are done in construction.

When should UP artifacts be created?
Sample UP Artifacts and Timing

Discipline	Artifact	Incep.	Elab.	Const.
	Iteration	I1	E1..En	C1..Cn
Business Modeling	Domain Model		s	
Requirements	Use-Case Model	s	r	
	Vision	s	r	
	Supplementary Specification	s	r	
	Glossary	s	r	
Design	Design Model		s	r
	SW Architecture Document		s	