* Finding stakeholders of the system is the most important thing for catching requirements T

无解……

* Object-oriented requirements analysis focused on to do the right thing T

P143 需求和面向对象分析重点关注学习做正确的事。而后续的设计工作强调正确地做事。

* Use Case is a collection of related success and failure scenarios that describe an actor using a system to support a goal T

P47用例就是一组相关的成功和失败场景集合，用来描述参与者如何使用系统来实现其目标。

* A timer may be a special actor of the system T

参与者是某些具有行为的事物，可以是人、计算机系统或组织。参与者可以是另一个计算机系统、一个计时器、一个传感器等。任何一个功能性需求，都有参与者启动。

* In UP, a suitable iteration length is two to six weeks timebox T

P17大部分迭代方法建议迭代时间在2-6周之间

* The Waterfall Process(model) represents a mainstream approach for software development F

瀑布模型是20世纪60-70年代所推崇的，如今并非主流的开发方式。

* In the UP, requirements are categorized according to the FURPS+ model T

P42在统一过程中，需求按照FURPS+模型进行分类，含义如下：功能性functional、可用性usability、可靠性reliability、性能performance、可支持性supportability、可实现性、接口、操作、包装、授权（+）

* Design emphasizes a conceptual solution (in software and hardware) that fulfills the requirements, rather than its implementation T

P5 设计强调的是满足需求的概念上的而解决方案，而不是其实现。

* Most requirements analysis during the inception phase F

P37大多数需求分析是在细化elaboration阶段进行的，并且伴以具有产品品质的早期变成和测试。

* Showing a report object in a domain model is always useful. F

报表对象（report objects）一般来说在领域模型中显示其他信息的报表并没有意义，因为其所有信息都是源于或复制于其他信息源的，这是排除的理由；另一方面，就业务规则而言，他有特殊的作用。所以要根据具体实例来分析，并非总是有用的。

* Operation Contract focus on data processing flow F我猜

操作契约使用前置和后置条件的形式，描述领域模型里对象的详细变化，并作为系统操作的结果，帮助理解系统功能。为了提高开发质量和效率，分析与设计人员会尽可能定义准确的系统API文档，并由系统开发工具doclet自动生成合格的操作契约，在UP中一般在细化阶段写操作契约。

* Generalization is equal to Inheritance in domain model F

P188泛化与继承是否相同，在领域模型概念视角的类图中，是不一样的。对于领域模型，超类是超集而子类是子集。

* The lower layer has the more general services F我猜

课本上说低层会有可复用的功能

* In Agile modeling, the purpose of modeling (sketching UML, …) is primarily to document F

敏捷建模的目的：主要是用于理解和沟通，而不是构建文档

* The same UML diagram can be used in multiple perspectives T

同一种UML图可以用于多种透视图perspectives

* SSD can involving other external systems. In general, they place at the right of the ':system' box T我猜

SSD展示了直接与系统交互的外部参与者、系统（作为黑盒）以及有参与者发起的系统事件这个，位于右侧

* A domain model is a visual representation of conceptual classes or real-situation objects in a domain T

课本概念：领域模型是对领域内的概念类或者现实世界中对象的可视化表示。

* Operations are usually assumed private if no visibility is given T

操作的可见性，不填则默认是私有的



* In UP, do design in Elaboration phases, do coding and testing in Construction phases. F

初始阶段inception：大题的构想、业务案例、范围和模糊评估；细化阶段elaboration：已精化的构想、核心架构的迭代实现、高风险的解决、确定大多数需求和范围以及进行更为世纪的评估；构造阶段construction对遗留下来的风险较低和比较简单的元素进行迭代实现，准备部署；移交阶段transition进行beta测试和部署。因此细化阶段不是需求或设计阶段，而是迭代实现核心架构并解决高风险问题的阶段。

* Drawing UML is a only reflection of making decisions about your design F

显然不对 也是我猜的

* Operation Contract focus on the changes in the state of objects in the domain model T

操作契约使用前置和后置条件的形式，描述领域模型里对象的详细变化，并作为系统操作的结果，帮助理解系统功能。

* 'Find Product Help' with underline in a use case text means emphasis（突出、强调）F

当一个用例引用另一个用例，被引用的那个用例会被加以下划线

补充：用例是文本文档，而非图形；用例建模主要是编写文本的活动而非制图。

编写用例文本比画用例图更重要

* A Vision with less than 10 features is desirable T

P86设想文档中包含的特性最好少于10个，因为更多的特性不能被快速掌握。

* The candidate use case 'Log In' can pass usefull Test, for we use it every day F

EXM???? PML的PPT上原话：“Log In：Boss not happy if this is all you do all day!”

* Software Engineering are the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software T

IEEE对于SE的定义：将系统化的、严格约束的、可量化的方法应用于软件的开发、运行和维护，即将工程化应用于软件

* At start of the iteration, Spend days drawing UML before coding T 我猜

每个迭代周期都需要进行需求分析、设计、实现等等。

* 'The system writes the sale to a database' in Use Case text is good F

反正pml的PPT上写撰写黑盒样例时不要这么写

* An end user can find the operation process in UML Use Case Diagram F

用例图能够展示系统边界、位于系统边界之外的事物以及系统如何被使用，不包括这些操作的具体过程。

* Interface is a UML classifier T

UML类元（classifier）是描述行为和结构特性的模型元素，他们是对众多UML元素的繁华，这些元素包括类、接口、用例和参与者。其中类和接口最为常用。

* In elaboration, we do not need deliver a runnable product to user for testing F

细化阶段需要对核心、有风险的软件架构进行编程和测试、发现并稳定需求的主体部分、规避主要风险