问题：

1. Методологии разработки ПО. Унифицированный процесс.

软件开发方法论。 统一流程

1. Требования и их категоризация. Атрибуты требований.

要求及其分类。 需求属性

1. Язык UML.

UML语言

1. Прецеденты использования. UseCase-диаграммы - состав, виды связей.

用例。 用例图 - 组合、连接类型

"Методологии разработки ПО" 是指在软件开发过程中使用的一种系统性和有组织的方法或框架。它们包括一系列规定和指导原则，用于指导软件团队在从需求分析、设计、编码、测试到部署和维护等阶段中进行软件开发工作。

不同的软件开发方法论有不同的特点和优势，可以根据项目的性质、规模和团队的组成选择合适的方法。以下是一些常见的软件开发方法论：

1瀑布模型（Waterfall Model）：这是一种线性的、顺序性的软件开发方法，各个开发阶段按照严格的顺序进行，要求在一个阶段完成后才能进入下一个阶段。

2敏捷开发（Agile Development）：这是一种灵活的、迭代的软件开发方法，强调团队合作、快速反馈和持续改进。常见的敏捷方法包括Scrum、Kanban、XP等。

3原型化（Prototyping）：这是一种通过创建原型来进行软件开发的方法，原型可以是初步的、简化的版本，用于快速验证和演化需求。

4增量式和迭代式开发（Incremental and Iterative Development）：这是一种将软件开发过程划分为多个增量和迭代的阶段进行开发的方法，每个阶段都可以产生一个可用的部分软件。

5融合模型（Hybrid Model）：这是一种将多个软件开发方法组合使用的方法，根据项目的需要和团队的情况来选择不同的方法。

这只是一些常见的软件开发方法论，实际上还有许多其他的方法论和框架，每个方法都有其独特的优点和适用场景。选择合适的软件开发方法论对于项目的成功和效率至关重要。

统一过程：是一种现代的软件开发过程模型。它是基于构件（Component-based）的，即所构造的软件系统是由软件构件通过明确定义的接口相互链接所建造起来的。并且它使用统一建模语言（Unified Modeling Language，UML）来制定系统的所有蓝图。

定义：

用户解决问题或者达到目标所需的条件或者权能

系统或者系统部件要满足合同，标准，会犯或者其他正式规定文档所需具有的条件或者权能

业务需求：

反映了组织机构或者客户对系统，产品高层次的目标要求，在文档中说明

用户需求：

文档描述了用户使用产品需要完成的任务，在Use case(使用实例)中说明

系统需求：

定义了开发人员必须实现的软件功能，以及实现系统所受到的约束。系统需求是的开发人员能完成他们的任务，从而满足了业务需求。这一部分需求在软件需求规格说明书中说明。

功能级需求：

则是基于用户级需求进一步拆分，完成功能列表的输出，即这一级需求可以导入到研发中。就是把具体的用户需求，变成软件的功能要求。 比如客户要把交通事故照片通过app发给保险公司。这是用户需求。 那么功能需求就是在这个模块下，要具有提交报险事故照片功能，上传现场照片。如果再具体下去，就是界面交互图。 现在互联网公司一提产品管理，需求设计，基本就是UX。 需求过于碎片化。

**功能性需求**（**functional requirement**）为一[软件工程](https://www.wikiwand.com/zh-sg/%E8%BB%9F%E9%AB%94%E5%B7%A5%E7%A8%8B)用语，功能需求定义一个软件系统或组件的[功能](https://www.wikiwand.com/zh-sg/%E5%8A%9F%E8%83%BD_(%E8%BB%9F%E9%AB%94%E5%B7%A5%E7%A8%8B))，也是一个系统需提供的功能及服务[[1]](https://www.wikiwand.com/zh-sg/%E5%8A%9F%E8%83%BD%E9%9C%80%E6%B1%82" \l "citenoteqhu1)。功能可以用一组输入、行为及输出的组合来表示。功能需求可以是计算、技术细节、资料处理或其他说明系统希望达成功能的内容。功能需求会以[非功能性需求](https://www.wikiwand.com/zh-sg/%E9%9D%9E%E5%8A%9F%E8%83%BD%E6%80%A7%E9%9C%80%E6%B1%82)（或是质量需求）为其基础，后者会描述设计或实現时的限制条件（例如性能需求、保安性或可靠度等）。

非功能性需求：

在系统工程及需求工程中，**非功能性需求**（**Non-functional requirement**）是指依一些条件判断系统运作情形或其特性，而不是针对系统特定行为的[需求](https://www.wikiwand.com/zh-sg/%E9%9C%80%E6%B1%82%E5%88%86%E6%9E%90)。和非功能性需求相对的是[功能需求](https://www.wikiwand.com/zh-sg/%E5%8A%9F%E8%83%BD%E9%9C%80%E6%B1%82)，后者会定义系统特定的行为或[功能](https://www.wikiwand.com/zh-sg/%E5%8A%9F%E8%83%BD_(%E8%BB%9F%E9%AB%94%E5%B7%A5%E7%A8%8B))。非功能性需求也可以视为为了满足客户业务需求而需要符合，但又不在功能性需求以内的特性。

一般会在系统设计（英语：Systems design）中详细列出实现功能需求的计划，而会在系统架构中详细列出实现非功能性需求的计划。一般而言，功能需求会定义系统的行为，而非功能性需求会定义系统的特性。

非功能性需求一般会称为系统的“质量”，有时也会称为“限制”、“质量属性”、“质量目标”、“质量服务需求”或“非行为性的需求”[[1]](https://www.wikiwand.com/zh-sg/%E9%9D%9E%E5%8A%9F%E8%83%BD%E6%80%A7%E9%9C%80%E6%B1%82" \l "citenote1)。有许多非功能性需求的英文都是以“ility”结尾，例如稳定性（stability）及可移植性（portability），因此非功能性需求有时也称为“ilities”。

非功能性需求可以分为以下的二类：

1. 执行质量（Execution qualities），可以在系统运作时观察到的质量，例如保安性及易用性等。
2. 发展质量（Evolution qualities），和软件系统结构及开发过程有关的质量，例如[软件可测试性](https://www.wikiwand.com/zh-sg/%E8%BB%9F%E9%AB%94%E5%8F%AF%E6%B8%AC%E8%A9%A6%E6%80%A7)、可维护性、可扩展性、可伸缩性（scalability）等[[2]](https://www.wikiwand.com/zh-sg/%E9%9D%9E%E5%8A%9F%E8%83%BD%E6%80%A7%E9%9C%80%E6%B1%82" \l "citenoteWiegers032)[[3]](https://www.wikiwand.com/zh-sg/%E9%9D%9E%E5%8A%9F%E8%83%BD%E6%80%A7%E9%9C%80%E6%B1%82" \l "citenoteYoung013)。

作为功能性需求的补充，她通常描述了系统展现给用户的行为和执行的操作等。必须满足标准和合约。

3.UML is a modeling tool for object-oriented design, independent of any specific programming language

UML mainly has the following functions:

1 Build visual models of software systems.

2 Create components for software systems.

3 Documenting a software system

There are three main models in UML system development:

1 Functional model

2 object model

3 dynamic model

UML consists of several parts such as Model Element, Diagram, View and General Mechanism.

UML 是面向对象设计的建模工具，独立于任何特定的编程语言

UML主要有以下功能：

1 建立软件系统的可视化模型。

2 为软件系统创建组件。

3 记录一个软件系统

UML系统开发主要有以下三种模型：

1 功能模型

2 对象模型

3 动态模型

UML 由模型元素、图表、视图和通用机制等几个部分组成。

4.UseCase is just a description of system functions, but a Use Case describes a part of the entire system function.

In the development process using UML, requirements are expressed in Use Cases

Use Case只是对系统功能的描述，但是一个Use Case描述的是整个系统功能的一部分。

在使用UML的开发过程中，需求用Use Cases表达

The usecase diagram embodies the functions that the system can provide to the actor and the connections between these functions.

you need to grasp three elements: Actor, Use Case and Relationship.

Actors ***1*** do not belong to the system structure, but are located outside the system ***2*** represent a class of roles rather than a specific object ***3*** not have to be a person, but can also be another external system, environment, etc

Each use case represents a type of function that the system can provide, and an oval is used in UML to represent a use case.

The relationship between use cases only needs to include three types, namely extension, inclusion and inheritance

用例图体现了系统可以提供给参与者的功能以及这些功能之间的联系。

你需要掌握三个要素：Actor、Use Case和Relationship。

参与者 1 不属于系统结构，但位于系统之外 2 代表一类角色而不是具体的对象 3 不一定是人，也可以是另一个外部系统、环境等

每个用例代表系统可以提供的一类功能，在UML中用一个椭圆来代表一个用例。

用例之间的关系只需要包括三种类型，即扩展、包含和继承