全国大学生软件测试大赛

http://www.mooctest.org/#/home/index

全国大学生软件测试大赛

三个分项赛：

（1）嵌入式测试；　（2）Web应用测试；（3）开发者测试。

比赛工具：

(1) 嵌入式测试采用凯云嵌入式系统测试教学实训平台；

(2) Web应用测试采用慕测平台，使用Selenium编写自动化测试脚本，使用Jmeter编写性能测试脚本；

(3) 开发者测试采用慕测平台，使用Junit，提供WebIDE和Eclipse两种答题方案。

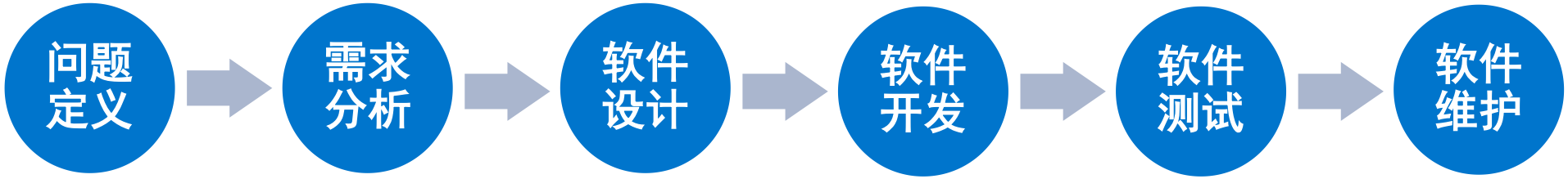
第1章 软件测试基础

章节概述/ Summary

现在已经步入了“智能化时代”，人们的工作与生活已经离不开软件，每天都会与各种各样的软件打交道。软件与其他产品一样都有质量要求，要想保证软件产品的质量，除了要求开发人员严格遵守软件开发的规范外，最重要的手段就是软件测试。本章将针对软件与软件测试的基础知识进行讲解。

1.1 软件概述

通常，可将软件生命周期划分为6个阶段。



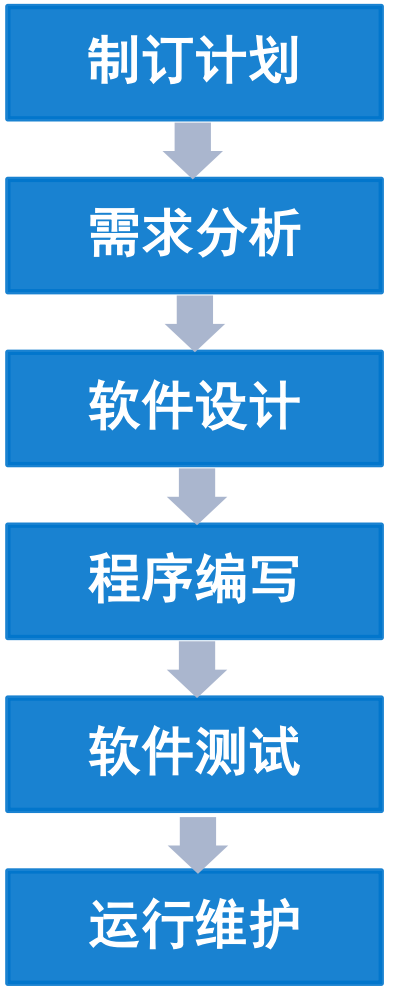
1.1.2 软件开发模型

- 熟悉6个典型的软件开发模型，能够区分这6个软件开发模型

下面根据软件开发模型的发展历史，介绍6个典型的开发模型。

1. 瀑布模型

瀑布模型为整个项目划分了清晰的检查点，当一个阶段完成之后，只需要把全部精力放在后面的开发上即可。这有利于大型软件开发人员的组织管理及工具的使用与研究，可以提高开发的效率。



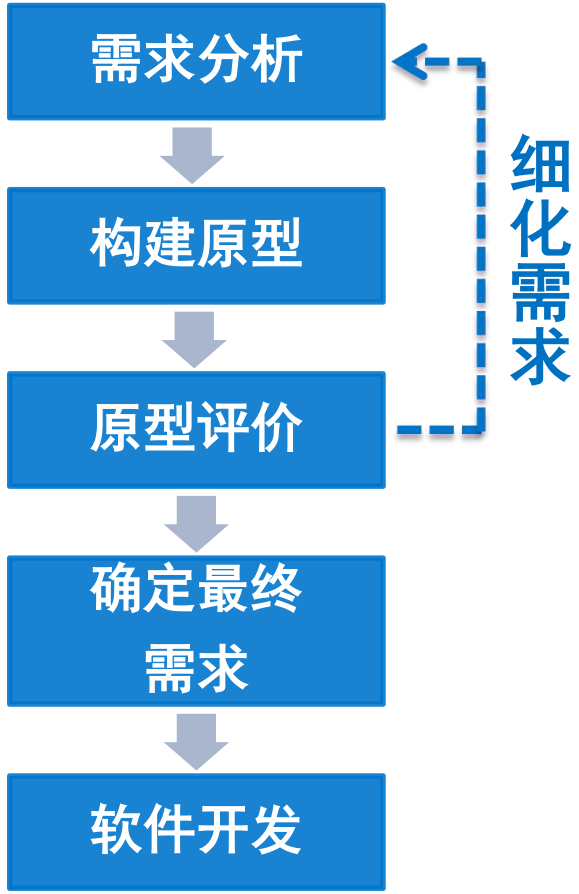
瀑布模型强调文档的作用，并要求每个阶段都要仔细验证。但是，这种模型的线性过程太理想化，已不再适合现代的软件开发模式，几乎被业界抛弃。其主要问题有以下2个方面

① 各个阶段的划分完全固定，阶段之间产生大量的文档，极大地增加了工作量。

② 由于开发模型是线性的，早期的错误可能要等到开发后期的测试阶段才能发现，从而增加了开发的风险。

2.快速原型模型

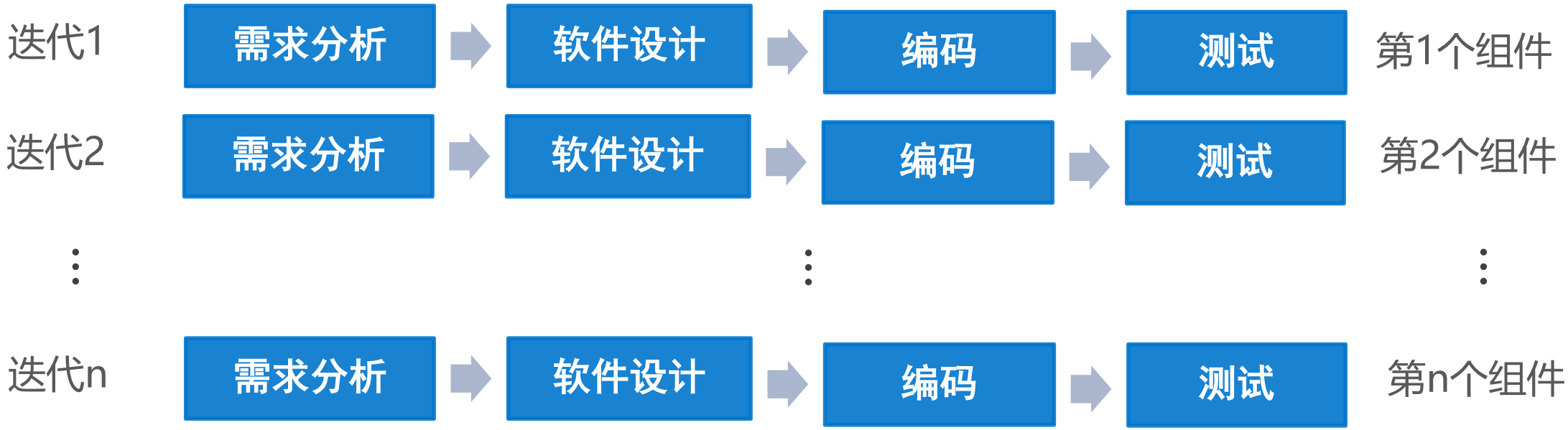
快速原型模型首先根据需求分析进行原型开发，即建造一个快速原型，实现客户或未来的用户与系统的交互，然后用户或客户对原型进行评价，进一步细化待开发软件的需求。通过逐步调整原型使其满足客户的要求，开发人员最终确定客户的真正需求是什么；最后在快速原型的基础上开发客户满意的软件产品。



与瀑布模型相比，快速原型模型规避了需求不明确带来的风险，适用于不能预先确定需求的软件项目。快速原型模型的关键在于快速构建软件原型，但准确地设计出软件原型存在一定的难度，此外，这种开发模型也不利于开发人员对产品进行扩展。

3.迭代模型

迭代模型又称为增量模型或演化模型，软件被作为一系列的增量构件来设计、实现、集成和测试，每一个构件是由多种相互作用的模块所形成的提供特定功能的代码片段构成的。它将一个完整的软件拆分成不同的组件，然后对每个组件进行开发测试，每测试完一个组件就将结果展现给用户，确定此组件的功能和性能是否满足用户需求，最终确定无误后，将组件集成到软件体系结构中。



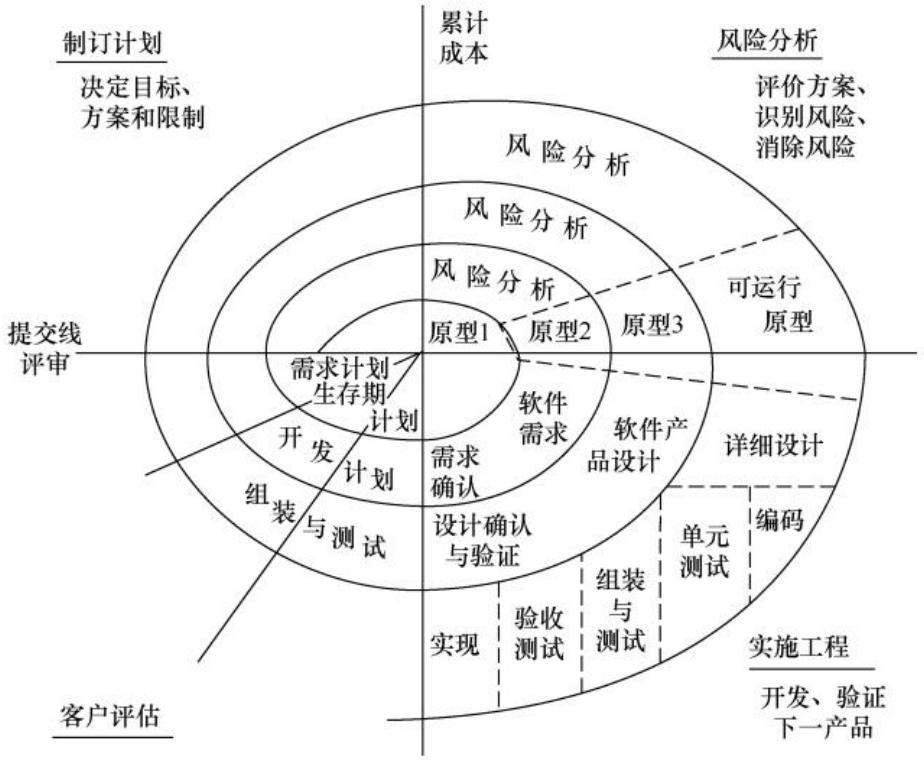
迭代开发模型可以很好地适应用户需求变更，它以逐个组件的形式交付产品，用户可以经常看到产品，如果某个组件没有满足用户需求，则只需要更改这一个组件，这降低了软件开发的成本与风险。

但是由于各个构件是逐渐并入已有的软件体系结构中的，所以加入的构件必须不破坏已构造好的系统部分，这需要软件具备开放式的体系结构。

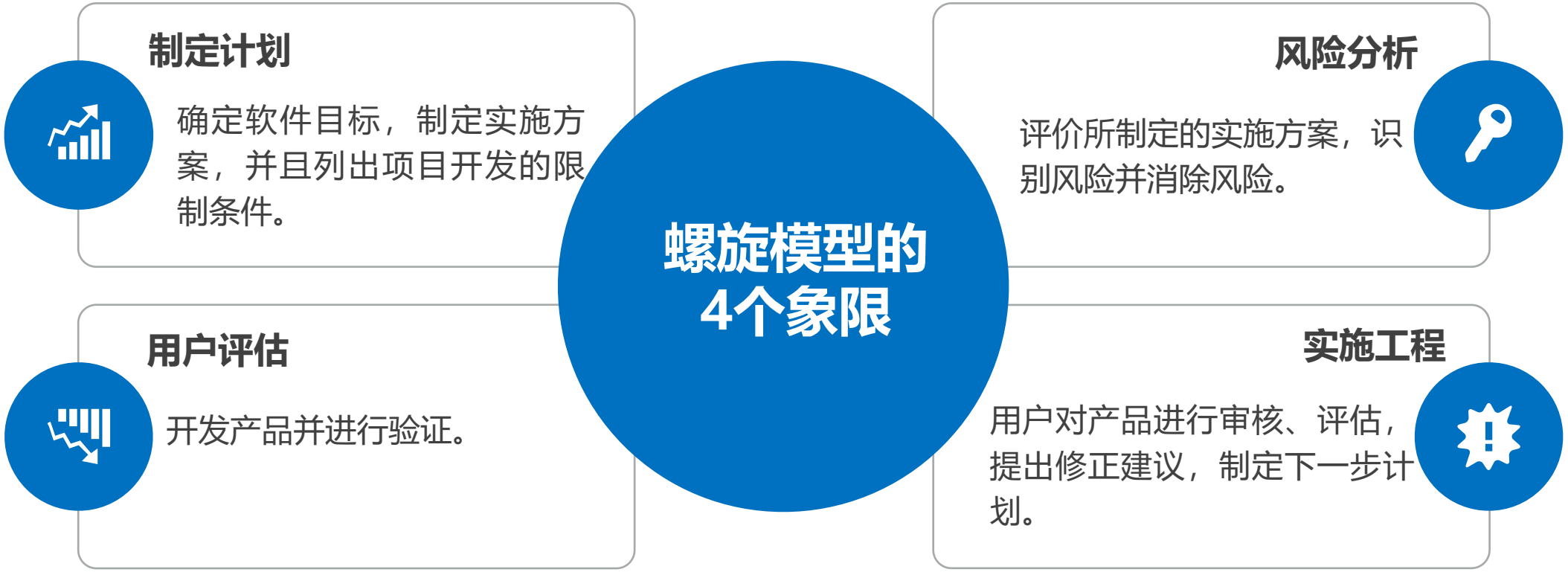
其次，在开发过程中，需求的变化是不可避免的。正是因为迭代模型以逐个组件的形式开发、修改，很容易退化为“边做边改”的开发形式，从而失去对软件开发过程的整体控制。

4.螺旋模型

螺旋模型融合了瀑布模型和快速原型模型，它最大的特点是引入了其他模型所忽略的风险分析。如果项目不能排除重大风险，就停止项目从而减小损失，这种模型比较适用于开发复杂的大型软件。螺旋模型的若干个阶段是沿着螺线方式进行的，螺旋模型的开发过程如右图所示。



1.1.2 软件开发模型



完成一个螺旋周期后，会基于客户评估和风险应对的结果，再次进入“制订计划”阶段，开启下一个螺旋周期。每一次循环，软件在功能、质量、稳定性等方面都会得到进一步提升，累计成本也会随着周期推进而增加，最终逐步构建出符合需求的成熟软件产品。

5.敏捷模型

敏捷模型是20世纪90年代兴起的一种软件开发模型，它以用户的需求进化为核心，采用循序渐进的方式进行软件开发。

敏捷模型还有一个重要的概念——迭代，即不断对产品进行细微、渐进式地改进，每改进一小部分都要进行评估，如果这部分的改进可行，再逐步扩大改进范围。在敏捷模型中，软件开发不再是线性的，开发的同时也会进行测试工作，甚至可以提前写好测试代码，因此对于敏捷模型，有“开发未动，测试先行”的说法。

相比于传统的软件开发模型，敏捷模型更注重“人”在软件开发中的作用，参与项目的各部门人员应该紧密合作、快速有效地沟通（如面对面沟通），提出需求的用户可以全程参与到开发过程中，以适应软件频繁的需求变更。为此，敏捷模型描述了一套软件开发的原则，具体如下。

- 个体和交互重于过程和工具。
- 可用软件重于完备文档。
- 用户协作重于合同谈判。
- 响应变化重于遵循计划。

对于较大的项目，参与开发的人员越多，有效沟通越困难，因此敏捷模型比较适用于小型项目的开发，而不太适用于大型项目的开发。

多学一招

敏捷模型的开发方式

敏捷模型的开发方式

Scrum

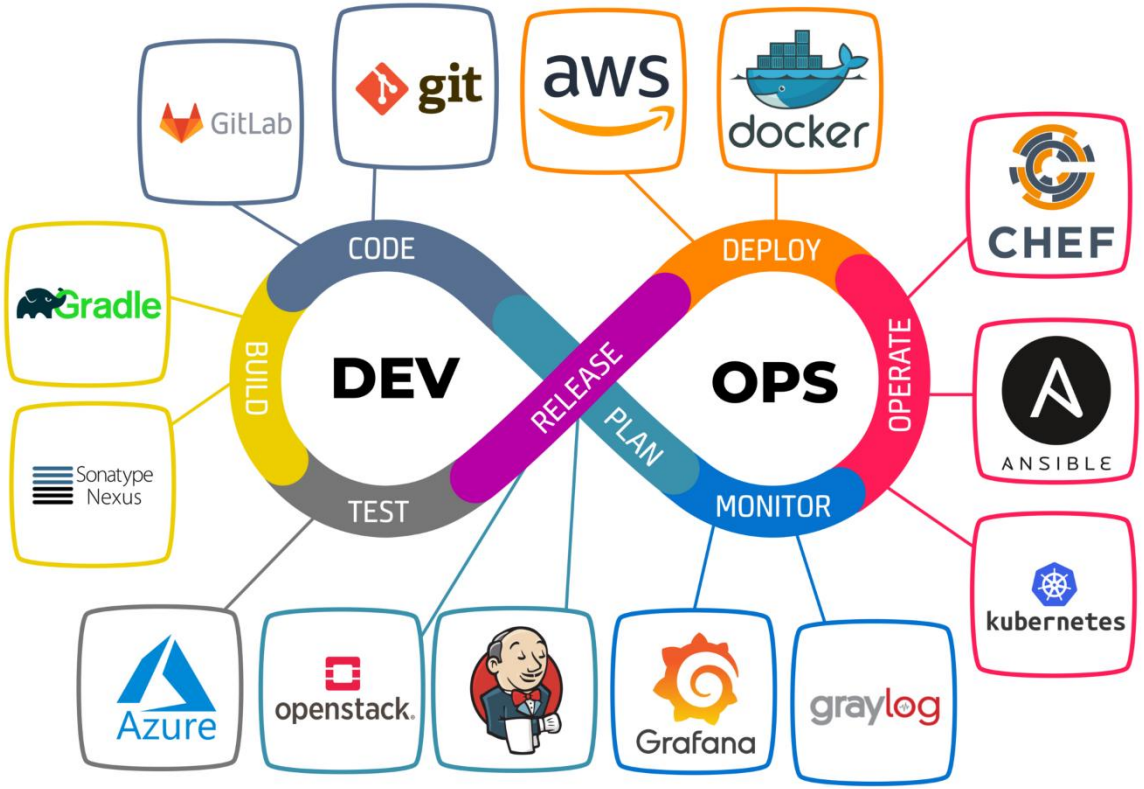
Scrum是一个开发管理框架。在使用Scrum开发方式的团队中，一般会选出一个Scrum Master（产品负责人）全面负责产品的开发过程。

Kanban

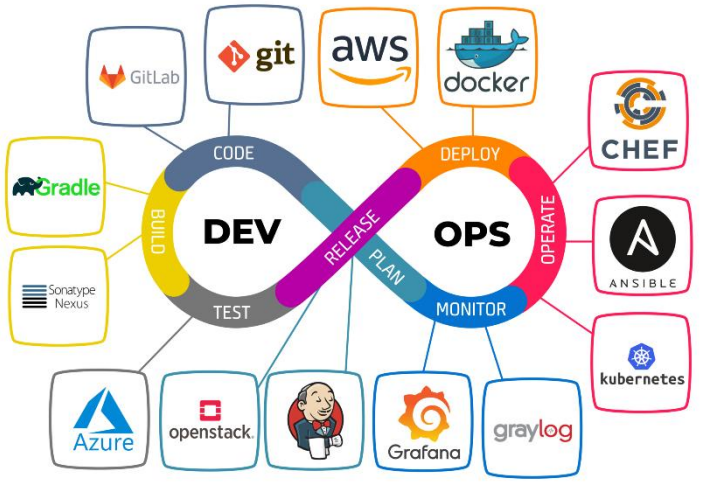
Kanban开发方式将工作细分成任务，将工作流程显示在“看板卡”上，每个人都能及时了解自己的工作任务及工作进度。

6.DevOps

DevOps 是 Development（开发）和 Operations（运维）的组合词，是一组过程、方法与系统的统称，用于促进开发、技术运营和质量保障部门之间的沟通、协作与整合，同时也是一种文化理念、实践和工具集。



生命周期



DevOps 生命周期通常由多个阶段组成，不同组织的阶段名称和顺序可能有所差异，但一般包含以下核心步骤：

- 规划(PLAN)：确定下一个版本软件的新特性和功能需求。
- 编码(CODE)：开发人员编写代码。
- 集成(BUILD)：新代码被集成到现有代码库中，然后进行测试和打包，为发布和部署做准备
- 测试(TEST)：进行各类测试，包括单元测试、集成测试、系统测试、用户验收测试等。
- 发布(RELEASE)：将经开发、测试的软件版本以正式可用状态交付给用户或投入生产环境
- 部署(DEPLOY)：将经过测试的软件部署到生产环境。
- 运维(OPERATE)：对生产环境中的软件进行监控和维护，确保其稳定运行。
- 监控与反馈(MONITOR)：持续监控软件在生产环境中的性能、用户行为等数据，收集用户反馈，根据这些信息为下一轮的开发和优化提供依据，形成一个持续改进的闭环。

工具链

- 版本控制工具：如 Git，提供分支管理、合并和重写存储库历史记录等功能，便于团队成员协作开发和管理代码版本。
- 构建工具：像 Jenkins，用于自动构建软件项目，将源代码编译成可执行的程序或软件包。
- 配置管理工具：例如 Ansible、Chef 等，用于自动化配置服务器和管理基础设施，确保环境的一致性。
- 容器化平台：以 Docker 为代表，Docker 用于创建、打包和分发应用程序及其依赖项，实现环境隔离。
- 监控和日志工具：比如 Prometheus 和 ELK Stack（Elasticsearch、Logstash、Kibana），Prometheus 用于监控系统指标，ELK Stack 用于收集、存储和分析日志数据，帮助团队了解系统运行状况和排查问题

关键特性

- 文化与沟通：强调跨职能团队之间的开放沟通和协作，打破开发、测试、运维等团队之间的壁垒，提倡所有团队成员对项目的成功共同承担责任。
- 持续集成与持续交付（CI/CD）：
  - 持续集成（CI）：允许多个开发人员频繁地将代码集成到共享存储库中，每当代码更改合并时，自动运行构建和测试流程，以确保代码的正确性和兼容性，尽早发现集成问题。
  - 持续交付/持续部署（CD）：确保软件可以随时部署至生产环境，无需大量人工干预。团队还可以选择使用功能标记，有条不紊地向用户交付新代码，而不是一次性全部发布，提高了软件开发团队的交付速度、生产力和可持续性。
- 敏捷性与灵活性：支持敏捷方法论，如 Scrum 和 Kanban 等，通过短周期的迭代开发，快速响应需求变化，实现快速迭代和适应变化的能力。

从软件质量的概念、软件质量模型、影响软件质量的因素这3个方面介绍软件质量的相关知识。

1. 软件质量的概念

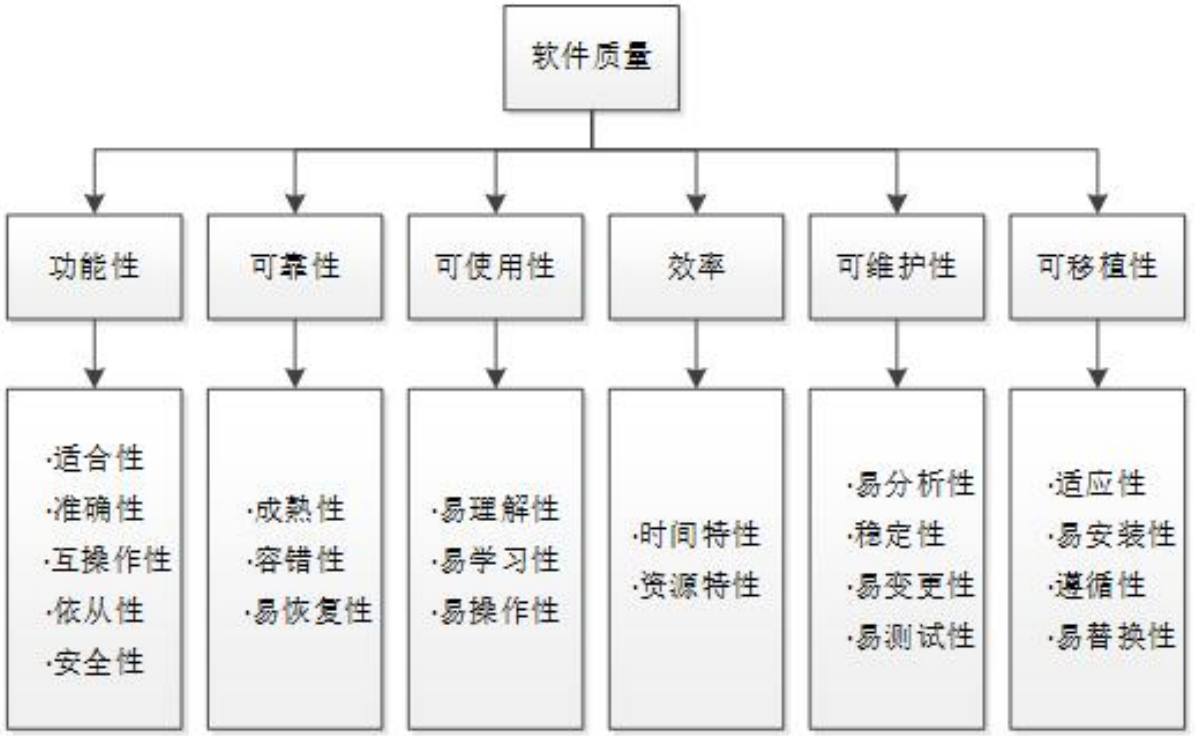
软件质量是指软件产品满足基本需求和隐式需求的程度。

满足高质量软件的3个需求

- **满足需求规定**：软件产品符合开发者明确给定的目标，并且能可靠运行。
- **满足用户基本需求**：软件产品的需求是由用户给出的，软件开发最终的目的就是满足用户基本需求，解决用户的实际问题。
- **满足用户隐式需求**：软件产品除了满足用户的基本需求外，如果还能满足用户的隐式需求，将会极大地提升用户满意度，这就意味着软件质量更高。

2. 软件质量模型

ISO/IEC 9126:1991是一个通用的评价软件质量的国际标准，它不仅对软件质量进行了定义，而且制定了软件测试的规范流程，包括测试计划的撰写、测试用例的设计等。ISO/IEC 9126:1991软件质量管理模型如下图所示。



1.1.3 软件质量概述

功能性

在指定条件下，软件产品满足用户基本需求和隐式需求的能力。

效率

在指定条件下，相对于所有资源的数量，软件产品可提供适当性能的能力。

可靠性

在指定条件下使用时，软件产品维持规定的性能级别的能力。

可维护性

指软件产品被修改的能力。修改包括修正、优化和功能规格变更的说明。

可使用性

在指定条件下，软件产品被使用、理解、学习的能力。

可移植性

指软件产品从一个环境迁移到另一个环境的能力。

ISO/IEC 25002:2024

- 兼容性：产品与其他系统或产品共存和协同工作的能力
- 安全性：保护信息和资产免受未经授权的访问、使用、披露、中断、修改或销毁的能力

3. 影响软件质量的因素

4种比较常见的影响因素

需求模糊

用户需求是不可视的，往往也说不清楚，导致产品设计、开发人员与用户之间存在一定的理解误差。

软件开发人员问题

人员流动性较大，导致软件开发前后不一致，进而影响软件质量。

软件开发缺乏规范性文件的指导

不太重视团队成员的工作规范，导致团队成员开发“随意性”比较大，影响软件质量。

缺乏软件质量的控制、管理

软件质量无法用具体的量化指标去度量，因此很少有人关心软件最终的质量。

1.2 软件缺陷管理

1.2.1 软件缺陷的概念

一般看来，满足以下的任意一种情况都可以称为软件缺陷

（1）软件未达到产品说明书中标明的功能。

（2）软件出现了产品说明书中指明的不应该出现的功能。

（3）软件功能超出了产品说明书中指明的范围。

（4）软件未达到产品说明书中指明的应达到的目的。

（5）软件难以理解和使用、运行速度慢或最终用户认为不好。

1.2.1 软件缺陷产生的原因

软件缺陷产生的原因主要有以下5点。

1. **需求不明确**

用户需求不清晰或者开发人员对需求的理解不明确，导致软件在设计时偏离用户的需求目标，造成软件功能或特征上的缺陷。用户频繁变更需求也会影响软件最终的质量。

2. **软件结构复杂**

如果软件系统结构比较复杂，很难设计出具有很好层次结构或组件结构的框架，就会导致软件在开发、扩充、系统维护上出现困难。

3. **编码问题**

开发人员水平参差不齐，再加上开发过程中缺乏有效的沟通和监督，问题累积得越来越多，如果不能逐一解决这些问题，会导致最终软件中存在很多缺陷。

4. **项目期限短**

现在大部分软件产品开发周期都相对较短，开发团队要在有限的时间内完成软件产品的开发，压力非常大，开发人员对待软件的态度是“非严重就不解决”。

5. **使用新技术**

新技术本身存在不足或开发人员对新技术掌握不精，都会影响软件产品的开发过程，导致软件存在缺陷。

1.2.2 软件缺陷的分类

从不同的角度可以将软件缺陷划分为不同的种类，具体划分如下。

按照测试种类划分

按照测试种类可以将软件缺陷分为界面缺陷、功能缺陷、性能缺陷、安全性缺陷、兼容性缺陷等。

按照缺陷的严重程度划分

按照缺陷的严重程度可以将缺陷划分为严重缺陷、一般缺陷、次要缺陷、建议缺陷。

按照缺陷的优先级划分

按照缺陷的优先级不同可以将缺陷划分为立即解决缺陷、高优先级缺陷、正常排队缺陷、低优先级缺陷。

按照缺陷的发生阶段划分

按照缺陷的发生阶段不同可以将缺陷划分为需求阶段缺陷、构架阶段缺陷、设计阶段缺陷、编码阶段缺陷、测试阶段缺陷。

1.2.3 软件缺陷的处理流程

- 熟悉软件缺陷的处理流程，能够归纳处理软件缺陷的每个环节的内容

1.2.3 软件缺陷的处理流程

软件缺陷的处理流程如下图所示。

（1）提交：测试人员发现缺陷之后，将缺陷提交给测试组长。

（2）分配：测试组长接收到测试人员提交的缺陷之后，将其移交给开发人员。

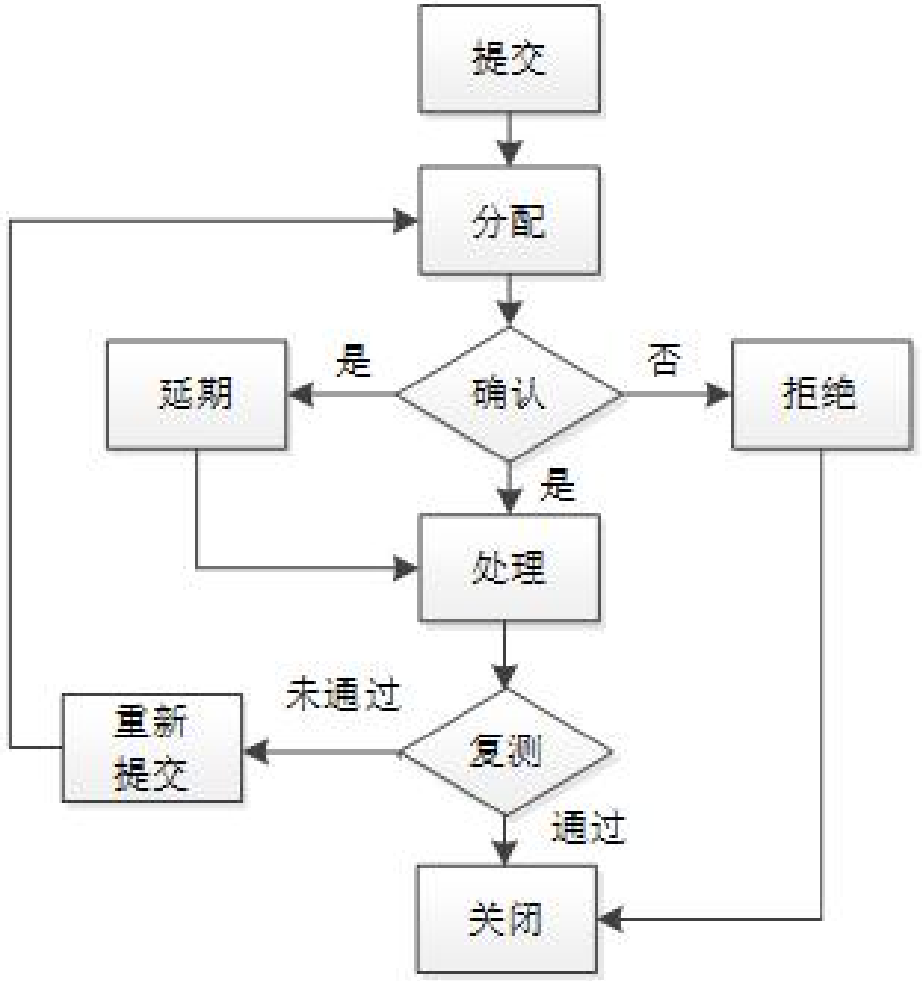
（3）确认：开发人员接收到移交的缺陷之后，会与团队甚至测试人员一起商议，确定其是否是一个缺陷。

（4）拒绝/延期：如果经过商议之后，发现其不是一个真正的缺陷，则拒绝处理此缺陷，对其进行关闭处理。如果经过商议之后，确定其是一个真正的缺陷，则可以根据缺陷的严重程度或优先级等立即处理或延期处理。

（5）处理：开发人员修改缺陷。

（6）复测：开发人员修改好缺陷之后，测试人员重新进行测试（复测），检测缺陷是否已经修改。如果未被正确修改，则重新提交缺陷。

（7）关闭：测试人员重新测试之后，如果缺陷已经被正确修改，则将缺陷关闭，整个缺陷处理完成。



多学一招

软件缺陷报告

在实际软件测试过程中，测试人员在提交软件测试时都会按照公司规定的模板将缺陷的详细情况记录下来并生成软件缺陷报告。每个公司的软件缺陷报告模板通常并不相同，但一般都会包括缺陷的ID、类型、严重程度、优先级、以及测试环境等，有时还会有测试人员的建议。

多学一招

在编写软件缺陷报告时要注意以下事项。

- 每个缺陷都有一个唯一的ID，这是缺陷的标识。
- 缺陷要有重现步骤。
- 一个缺陷生成一份报告。
- 软件缺陷报告要整洁、完整。

1.2.4 常见的软件缺陷管理工具

- 了解常见的软件缺陷管理工具，能够列举3个常用的软件缺陷管理工具

3个常用的软件缺陷管理工具

1.Bugzilla

Bugzilla是Mozilla公司提供的一款免费的软件缺陷管理工具。Bugzilla能够建立一个完整的缺陷跟踪体系，包括缺陷跟踪、记录、缺陷报告、解决情况等。

2.禅道

禅道集产品管理、项目管理、质量管理、缺陷管理、文档管理、组织管理和事务管理于一体，是一款功能完备的项目管理软件，完美地覆盖了项目管理的核心流程。

3.Jira

Jira是Atlassian公司开发的项目与事务跟踪工具，被广泛用于缺陷跟踪、用户服务、需求收集、流程审批、任务跟踪、项目跟踪和敏捷管理等工作领域。是目前比较流行的基于Java架构的管理工具。

1.3 软件测试概述

1.3.1 软件测试简介

- 了解软件测试的定义

1.3.1 软件测试定义

IEEE 的定义：

- 在特定的条件下运行系统或构件，观察或记录结果，对系统的某个方面做出评价
- 分析某个软件项以发现现存的和要求的条件之差别（即错误）并评价此软件项的特性

Glenford J. Myers提出：

（1）软件测试是程序的执行过程，目的在于发现错误。

（2）软件测试是**为了证明程序有错误，而不是证明程序无错误**。

（3）一个好的软件测试用例在于能发现至今未发现的错误。

（4）一个成功的软件测试是发现了至今未发现的错误的测试。

Bill Hetzelt 在《软件测试完全指南》中指出：

“软件测试是以评价一个程序或者系统属性为目标的任何一种活动。软件测试是对软件质量的度量。”

Test = Verification + Validation

Verification： Are we building the product right?

是否正确地构造了软件？即是否正确地做事，验证开发过程是否遵守已定义好的内容。验证产品满足规格设计说明书的一致性

Validation： Are we building the right product?

是否构造了正是用户所需要的软件？即是否正在做正确的事。验证产品所实现的功能是否满足用户的需求

1.3.2 软件测试的目的

- 熟悉软件测试的目的，能够从3个角度归纳软件测试目的

1.3.2 软件测试的目的

从软件开发、软件测试与用户需求的角度，可以将软件测试的目的归结为以下3点。

1

1.从软件开发的角度

从软件开发角度来说，软件测试通过找到的缺陷帮助开发人员找到开发过程中存在的问题，包括软件开发的模式、工具、技术等方面存在的问题与不足，从而预防缺陷的产生。

2

2.从软件测试的角度

从软件测试角度来说，主要目的是使用最少的人力、物力、时间等找到软件中隐藏的缺陷，保证软件的质量，也为以后软件测试积累丰富的经验。

3

3.从用户需求的角度

从用户需求角度来说，软件测试能够检验软件是否符合用户需求，对软件质量进行评估和度量，可为用户评审软件提供有力的依据。

1.3.3 软件质量保证定义

软件质量保证是贯穿软件项目整个生命周期的有计划的系统活动，经常针对整个项目质量计划执行情况进行评估、检查和改进，确保项目质量与计划保持一致。

软件质量保证确保软件项目的过程遵循了对应的标准及规范要求，且产生了合适的文档和精确反映项目情况的报告，其目的是通过评价项目质量建立项目达到质量要求的信心。软件质量保证活动主要包括评审项目过程、审计软件产品，就软件项目是否真正遵循已经制订的计划、标准和规程等，给管理者提供可视性项目和产品可视化的管理报告。

1.3.4 软件质量保证和软件测试的区别

- 了解软件质量保证和软件测试的区别

1.3.4 软件质量保证与软件测试的区别

软件质量保证（SQA）：

核心目标是预防质量问题，通过建立和执行一套完整的质量体系（如流程、标准、规范），确保软件开发过程符合预定的质量要求，从根源上减少缺陷的产生，最终交付高质量的软件产品。

简单说，SQA 关注 “过程正确”，通过规范过程来保证结果质量。

软件测试：

核心目标是发现质量问题，通过执行测试用例、模拟用户场景等方式，识别软件中已存在的缺陷（如功能错误、性能问题、兼容性问题等），并推动缺陷修复，确保软件在交付前尽可能消除可见问题。

简单说，测试关注 “结果验证”，通过检查产品来发现已有缺陷。

1.3.5 软件测试的分类

- 熟悉软件测试的分类，能够从不同角度归纳软件缺陷的分类

1.3.5 软件测试的分类

按照不同的分类标准可以将软件测试分为很多不同的种类。

1. 按照测试阶段分类

单元测试大多是开发人员进行的自测。

集成测试是冒烟测试之后进行的测试，用于验证软件是否满足设计需求。

逐行逐字的按照说明书的描述对软件产品进行测试，确保其符合用户的各项要求。

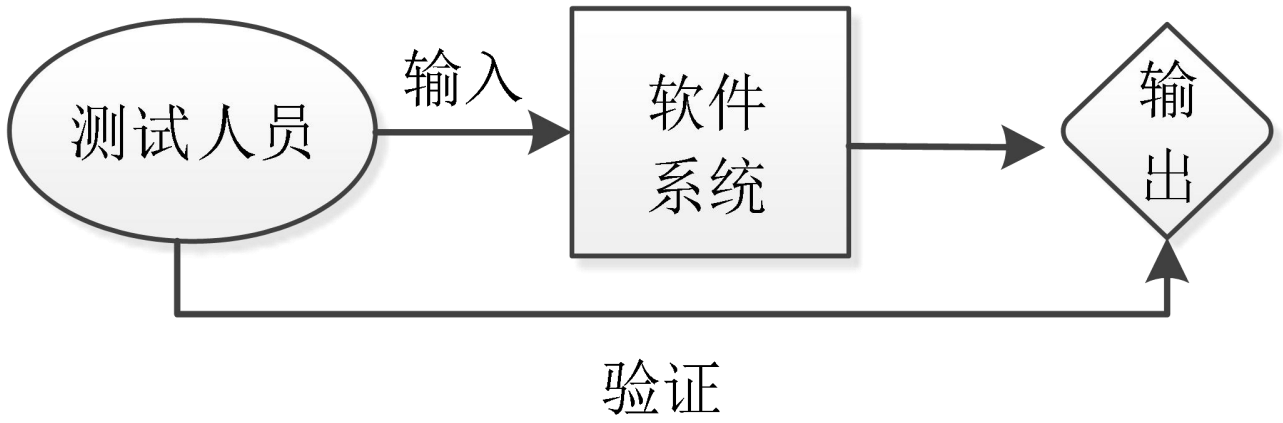
冒烟测试是指软件构建版本建立后，对系统的基本功能进行简单的测试。

将其与其他系统的成分（如数据库、硬件和操作人员等）组合在一起进行测试。

2. 按照测试技术分类

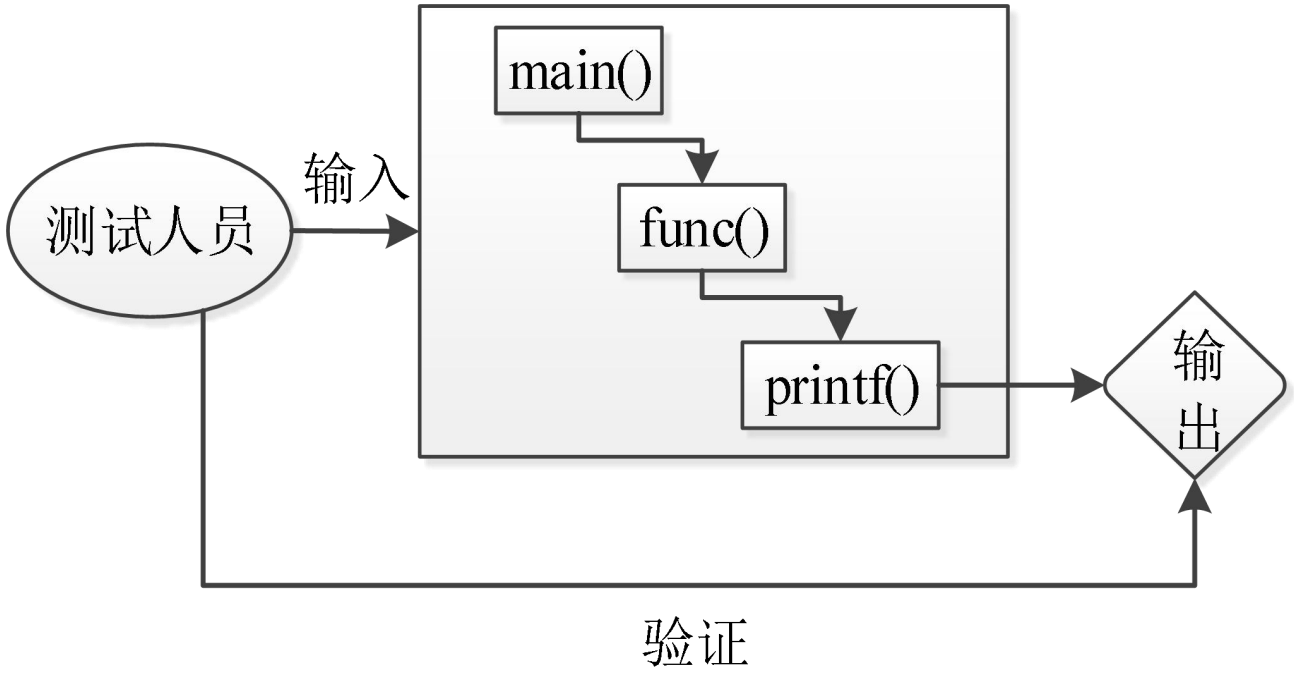
(1) 黑盒测试

黑盒测试又叫功能测试、数据驱动测试、基于需求规格说明书的功能测试，它把软件当作一个有输入与输出的“黑匣子”，只要输入的数据能输出预期的结果即可，不必关心程序内部是怎样实现的，注重于测试软件的功能性需求。



(2) 白盒测试

白盒测试又叫透明盒测试、结构测试、逻辑驱动测试或基于代码的测试，它是指测试人员了解软件程序的逻辑结构、路径和运行过程，在测试时，按照程序的执行路径得出结果。白盒测试把软件（程序）当作一个透明的“盒子”，测试人员清楚地知道从输入到输出的每一步过程。白盒测试如下图所示。



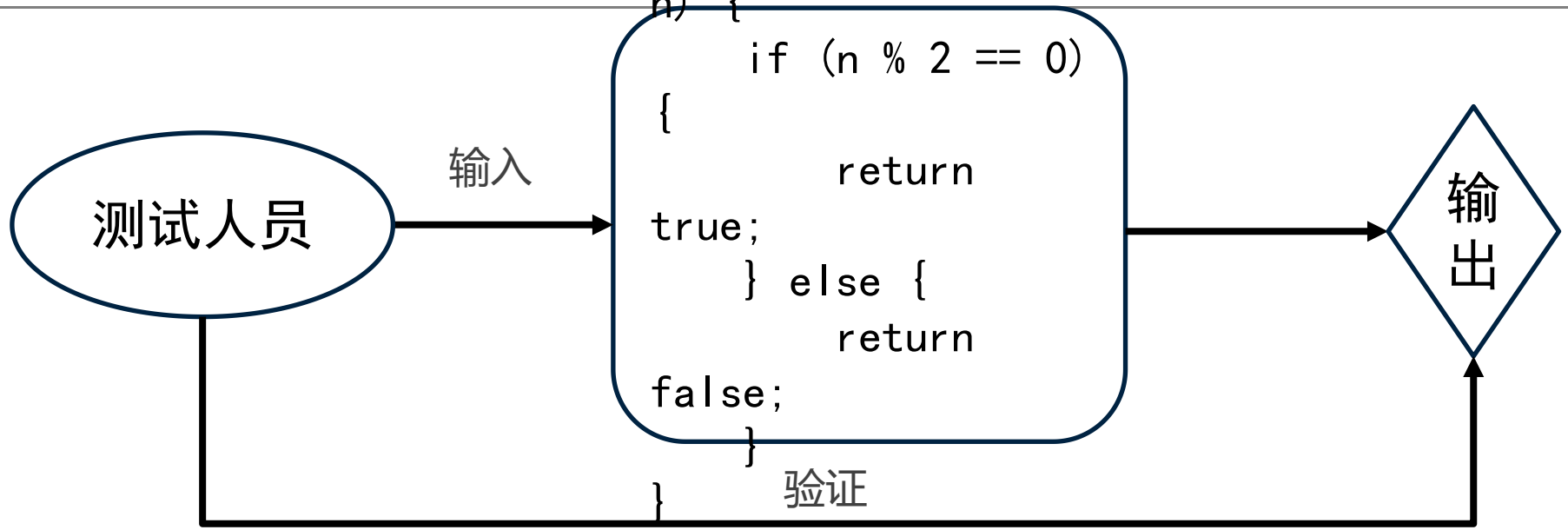
(3) 灰盒测试

灰盒测试是介于黑盒测试与白盒测试之间的一种软件测试方法，它由方法和工具组成，这些方法和工具取决于应用程序内部交互的环境。灰盒测试通常用于集成测试阶段，测试人员在使用灰盒测试方法时，不仅需要关注输入、输出的正确性，而且需要关注程序内部的情况，通常根据一些现象、事件、标志来判断内部的运行状态。

总结

相对于黑盒测试来说，白盒测试对测试人员的要求会更高一点，它要求测试人员具有一定的编程能力，而且要熟悉各种脚本语言。但是在企业中，黑盒测试与白盒测试并不是界限分明的，在测试一款软件时往往将黑盒测试与白盒测试相结合对软件进行完整、全面的测试。灰盒测试虽然没有白盒测试详细、完整，但是比黑盒测试更关注程序的内部逻辑，能够用于黑盒测试以提高测试的效率。

1.3.5 软件测试的分类



黑盒测试

- 关注点：功能是否满足需求，而不关心内部代码逻辑。
- 测试依据：需求说明文档。
- 设计方法：等价类划分、边界值分析等。
- 测试用例：
  - ✓ 输入 2 → 期望输出 true
  - ✓ 输入 3 → 期望输出 false
  - ✓ 输入 0 → 期望输出 True
  - ✓ 输入 -1 → 期望输出 false
- 特点：完全站在用户角度，只看“输入-输出”的对应关系。

白盒测试

- 关注点：程序内部结构和逻辑。
- 测试依据：源代码。
- 设计方法：语句覆盖、分支覆盖、路径覆盖等。
- 测试用例：
  - ✓ 输入 4 → 覆盖 if (n % 2 == 0) 为真分支
  - ✓ 输入 5 → 覆盖 if (n % 2 == 0) 为假分支
- 特点：测试者需要了解代码，确保所有分支、路径都被执行。

灰盒测试

- 关注点：结合黑盒和白盒，既考虑需求输入输出，又参考部分代码结构。
- 测试依据：需求说明 + 部分设计/代码信息。
- 设计方法：基于接口设计、数据库、逻辑推断。
- 测试用例：
  - ✓ 知道函数内部通过 % 运算来判断 → 设计测试 n = 2、n = -2，以检查对负数是否也正确处理。
  - ✓ 知道函数返回布尔值 → 设计输入 1000000，验证大数时性能与正确性。
- 特点：测试范围更广，兼顾用户视角与开发逻辑，常用于集成测试、系统测试阶段

3. 按照软件质量特性分类

按照软件质量特性可以将软件测试分为功能测试和性能测试。

功能测试

功能测试是指测试软件的功能是否满足用户的需求，包括准确性、易用性、适合性、互操作性等。

性能测试

性能测试是指测试软件的性能是否满足用户的需求，包括负载测试、压力测试、兼容性测试、可移植性测试和健壮性测试等。

4. 按照自动化程度分类

按照自动化程度可以将软件测试分为手工测试和自动化测试。

手工测试

手工测试是测试人员编写与执行测试用例的过程。手工测试比较耗时、费力，而且测试人员如果在疲惫状态下，很难保证测试的效果。

自动化测试

自动化测试是指借助脚本、自动化测试工具等完成相应的测试工作，它也需要人工的参与，但是它可以将要执行的测试代码或流程写成脚本，通过执行脚本完成整个测试工作。

5. 按照测试项目分类

按照测试项目可以将软件测试分为界面测试、功能测试、性能测试、安全性测试、文档测试等，其中功能测试和性能测试前面已经介绍，下面主要介绍其他几种测试。

1

界面测试

界面类测试是指验证软件界面是否符合用户需求，包括界面布局是否美观、按钮是否齐全等。

2

安全性测试

安全性测试是指测试软件在受到没有授权的内部或外部用户的攻击或恶意破坏时如何进行处理，是否能保证软件与数据的安全。

3

文档测试

文档测试以需求分析、软件设计文档、用户手册、安装手册为主，主要验证文档说明与实际软件之间是否存在差异。

6. 其他分类

还有一些软件测试无法具体归到哪一类，但在测试行业中也会经常进行这些测试，例如α测试、β测试、回归测试、随机测试等。具体介绍如下。

α测试

α测试是指对软件最初版本进行测试。测试人员记录软件最初版本在使用过程中出现的错误和问题，整个测试过程是可控的。

β测试

β测试是指对上线之后的软件版本进行测试，由用户在使用过程中发现错误和问题并进行记录，然后反馈给开发人员进行修复。

回归测试

确认原有的缺陷已经消除并且没有引入新的缺陷，这个重新测试的过程称为回归测试。

随机测试

随机测试是没有测试用例、检查列表、脚本或指令的测试，它主要根据测试人员的经验对软件进行功能和性能抽查。

多学一招

纸杯测试

“纸杯测试”是一个经典的测试案例，这是微软曾给软件测试面试者出的一道面试题，用于考察面试者对软件测试的理解和掌握程度。

测试项目：纸杯。

需求测试：查看纸杯说明书是否完整。

界面测试：观察纸杯的外观，例如表面是否光滑。

功能测试：用纸杯装水，观察是否漏水。

安全测试：纸杯是否有病毒或细菌。

可靠性测试：从不同高度扔下来，观察纸杯的损坏程度。

易用性测试：用纸杯盛放开水，检查纸杯是否烫手、纸杯是否易滑、是否方便饮用。

兼容性测试：用纸杯分别盛放水、酒精、饮料、汽油等，观察是否有渗漏现象。

1.3.5 软件测试的分类

多学一招

可移植性测试：将纸杯放在温度、湿度等不同的环境中，查看纸杯是否还能正常使用。

可维护性：将纸杯揉捏变形，看其是否能恢复。

压力测试：用一根针扎在纸杯上，不断增加力量，记录用多大力时能穿透纸杯。

疲劳测试：用纸杯分别盛放水、汽油，放置24小时，观察其渗漏情况（时间和程度）。

跌落测试：让纸杯（加包装）从高处落下，记录可使其破损的高度。

震动测试：将纸杯（加包装）六面震动，评估它是否能应对恶劣环境下的公路/铁路/航空运输等。

测试数据：编写具体测试数据（略），其中可能会用到场景法、等价类划分法、边界值分析法等测试方法。

期望输出：需要查阅国际标准及用户的使用需求。

用户文档：使用手册是否对纸杯的用法、使用条件、限制条件等进行了详细描述。

说明书测试：查看纸杯说明书的正确性、准确性和完整性。

1.4 软件测试过程模型

1.4 常见的软件测试模型

- 了解常见的软件测试模型，能够列举4个常见的软件测试模型

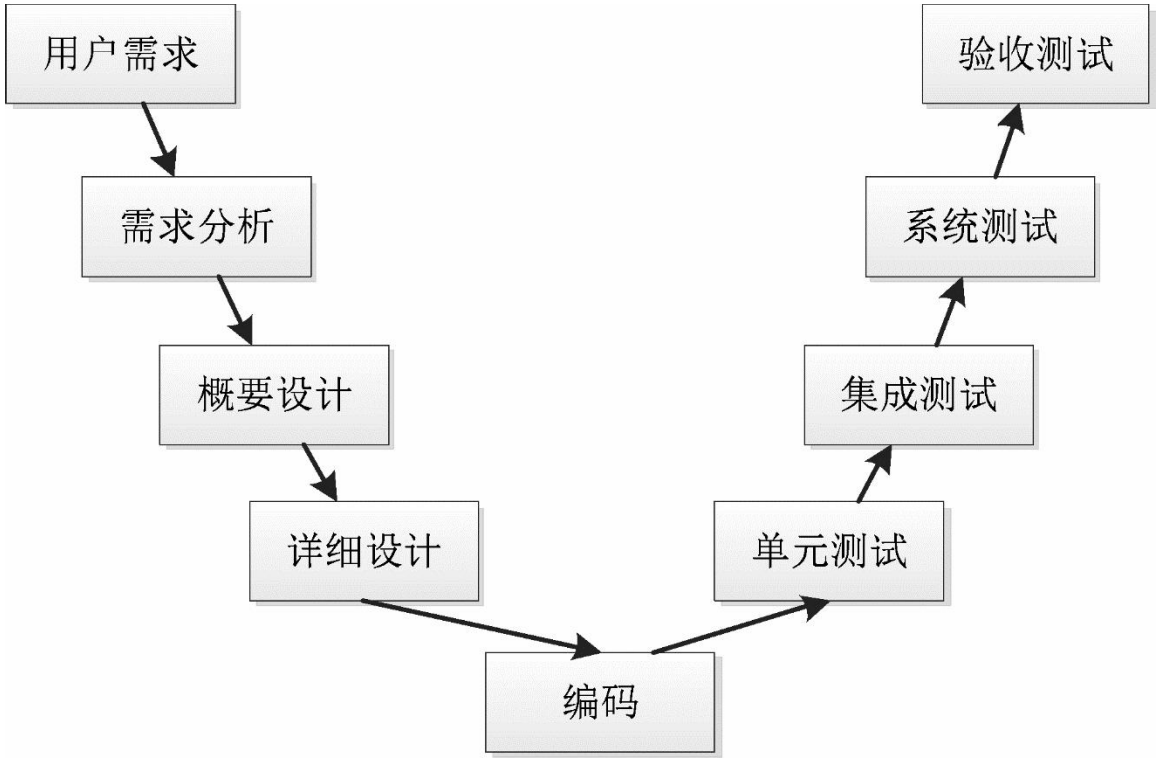
下面介绍4种比较重要的软件测试模型。

1. V模型

V模型描述了基本的开发过程和测试行为。箭头方向为时间方向，从左至右分别是开发的各阶段与测试的各阶段

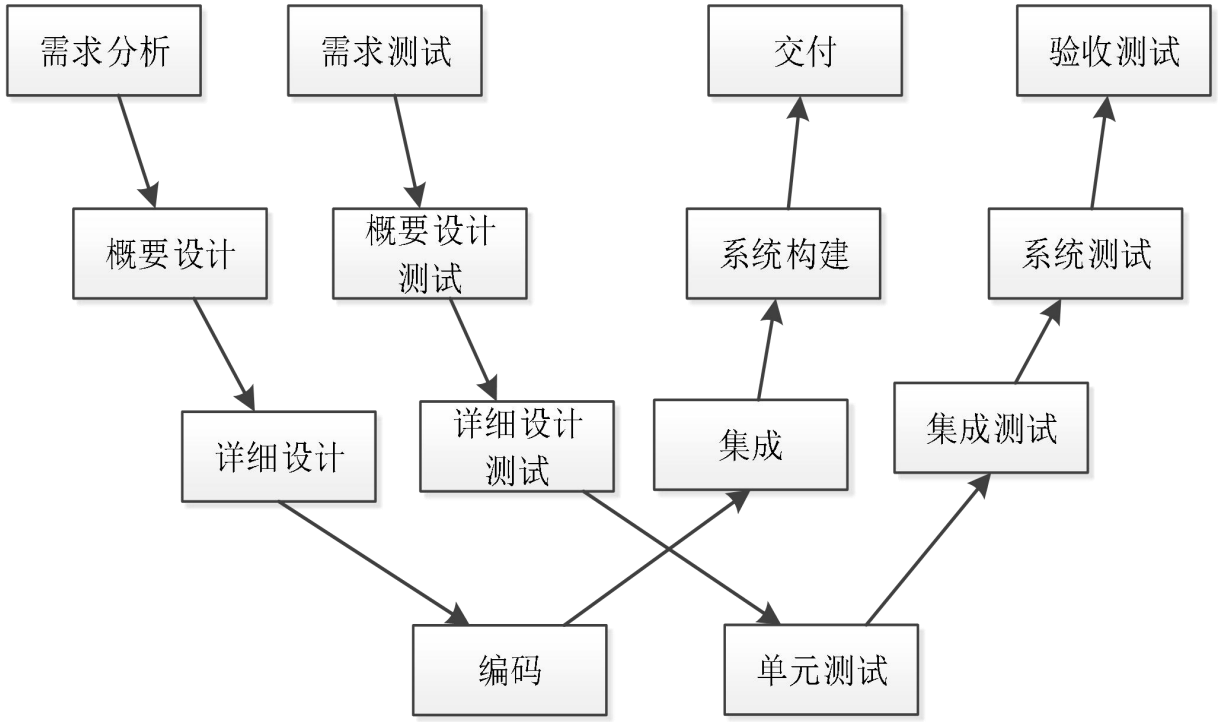
V 模型非常明确地标注了测试过程中存在的不同级别，并使测试的每个阶段都与开发的阶段相对应。

但 V 模型也存在着一定的局限，它不能发现需求分析等前期阶段产生的错误，直到编码完成之后才进行测试，因此早期出现的错误不能及时暴露。



2. W模型

W模型是由V模型演变而来的，它强调测试应伴随着整个软件生命周期。W模型如下图所示。

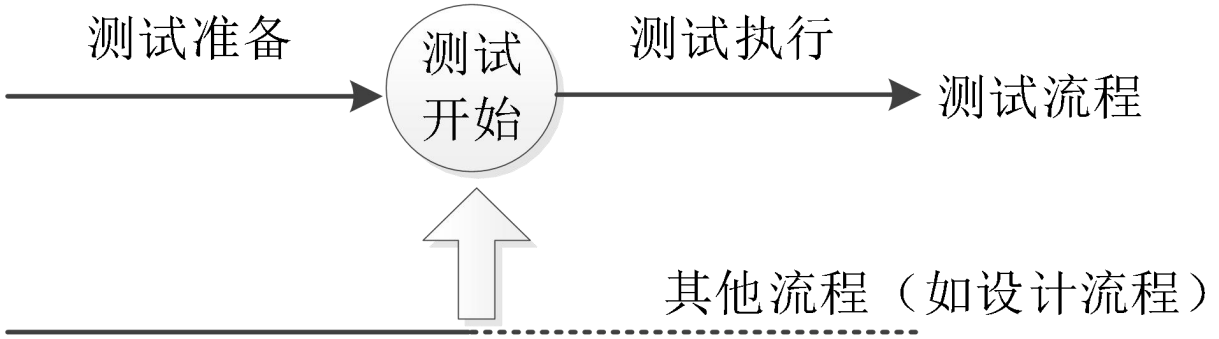


在W模型中，测试与开发是同步进行的，这样可以使开发过程中的问题及早地暴露出来。

但W模型仍存在着较为明显的问题，因为它将开发活动认定为从需求开始到编码结束的串行活动，只有在上一活动结束后才能开始下一步的行动，无法支持需要迭代、自发性以及变更调整的项目

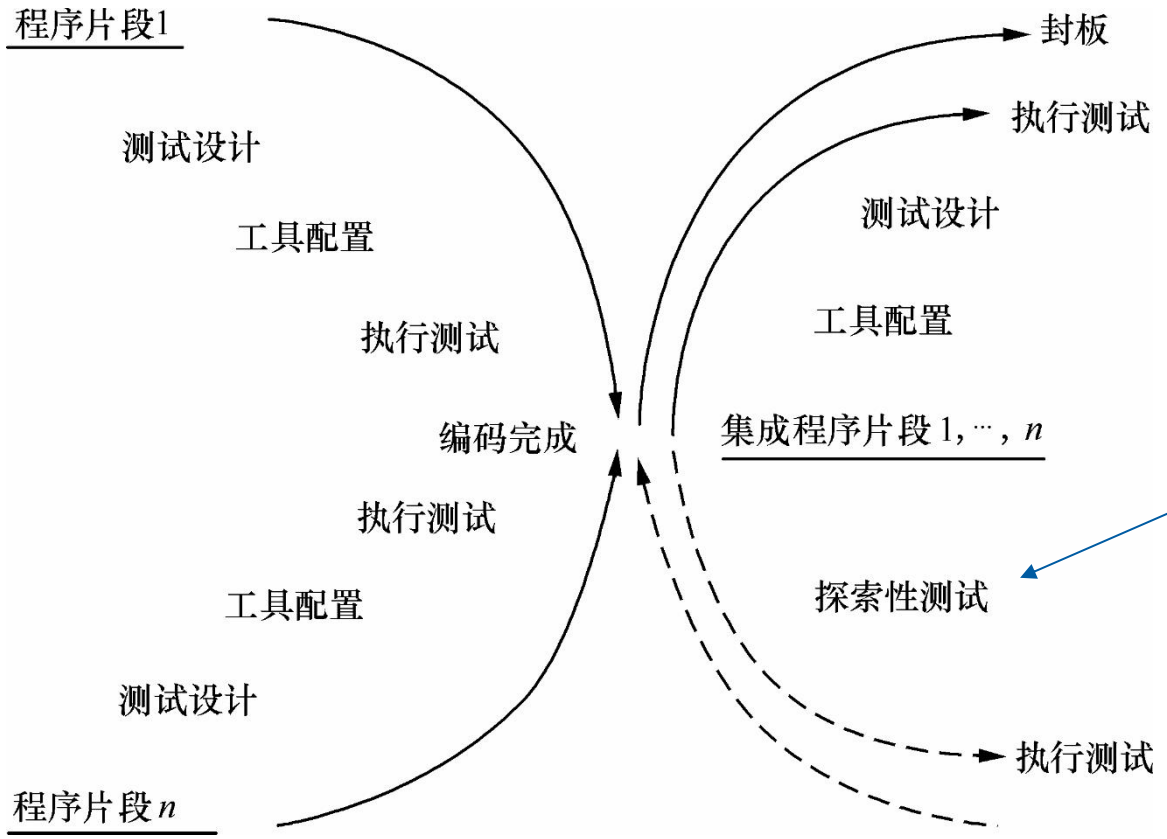
3. H模型

为了解决V模型与W模型存在的问题，有专家提出了H模型，H模型将测试活动完全独立出来，形成一个完全独立的流程，这个流程将测试准备活动和测试执行活动清晰地体现出来。H模型如下图所示。



4. X模型

X模型的设计原理是将程序分成多个片段反复迭代测试，然后将多个片段集成再进行迭代测试，X模型如下图所示。



不进行实现计划的特殊类型测试

总结

前面共介绍了4种软件测试模型，在实际测试工作中，测试人员更多的是结合W模型与H模型进行工作，软件各个方面的测试内容以W模型为准，而测试周期、测试计划和进度是以H模型为指导。X模型更多是作为最终测试、熟练性测试的模板，例如对一个业务进行的测试已经有2年时间，可以使用X模型进行模块化的、探索性的方向测试。

1.5 软件测试的原则

1.5.1 软件测试的原则

- 熟悉软件测试的原则，能够归纳软件测试的6个基本原则

1.5.1 软件测试的原则

软件测试行业公认的6个基本原则。

1.测试应基于用户需求

测试的核心目的是验证软件是否满足用户的需求，所以一切都以用户需求为依据，确保测试能有效检验软件是否符合用户期望

2.测试要尽早进行

在软件开发的早期阶段就开展测试相关工作，尽早发现潜在的问题，降低后续修复问题的成本，提高软件开发效率

3.不能做到穷尽测试

选取具有代表性的测试用例来覆盖主要的功能、场景和风险点

4.遵循GoodEnough原则

测试不需要追求绝对的完美，只要软件达到了“足够好”的质量标准，在可接受的缺陷范围内，就可以考虑结束测试

5.测试缺陷要符合“二八”定理

80% 的软件缺陷集中在 20% 的模块中，测试时应重点关注这些高缺陷率的模块，提高测试的效率和针对性

6.避免缺陷免疫

防止测试人员或开发人员对某些类型的缺陷产生“免疫”，即不能因为经常遇到某些缺陷，就忽视对它们的检测

1.5.2 软件测试停止原则

- 测试超过了预定的时间，停止测试。
- 执行了所有测试用例但没有发现故障，停止测试。
- 使用特定的测试用例方法作为判断测试停止的基础。
- 正面指出测试完成的要求，如发现并修改70个软件故障。
- 根据单位时间内查出故障的数量决定是否停止测试。

1.6 软件测试的基本流程

1.6.1 软件测试的流程

- 熟悉软件测试的基本流程，能够归纳软件测试的5个基本流程

1.6.1 软件测试的流程

不同类型的软件产品测试的方式和重点不一样，测试流程也会不一样。同样类型的软件产品，不同公司所制定的测试流程也会不一样。虽然不同软件的详细测试步骤不同，但它们所遵循的最基本的测试流程通常是一样的。

1. 分析测试需求

测试人员在制定测试计划之前需要先对用户需求进行分析，以便对要开发的软件产品有一个清晰的认识，从而明确测试对象及测试工作的范围和测试重点。在分析需求时还可以获取一些测试数据，将其作为测试计划的基本依据，为后续的测试打好基础。

分析测试需求其实也是对用户需求进行测试，测试人员可以发现用户需求中不合理的地方，例如需求描述是否完整、准确、无歧义，以及需求优先级安排是否合理等。测试人员一般会根据软件开发需求文档制作一个需求规格说明书检查列表，按照各个检查项对用户需求进行分析、校验。

软件需求规格说明书检查列表如下表所示。

序号	检查项	检查结果	说明
1	是否覆盖了用户提出的所有需求项	是【】否【】NA【】	
2	用词是否准确、语义是否存在歧义	是【】否【】NA【】	
3	是否清楚地描述了软件需要做什么以及不做什么	是【】否【】NA【】	
4	是否描述了软件的目标环境，包括软硬件环境	是【】否【】NA【】	
5	是否对需求项进行了合理的编号	是【】否【】NA【】	
6	需求项是否前后一致、彼此不冲突	是【】否【】NA【】	
7	是否清楚地说明了软件的每个输入、输出格式，以及输入与输出之间的对应关系	是【】否【】NA【】	
8	是否清晰地描述了软件系统的性能要求	是【】否【】NA【】	
9	需求的优先级是否进行了合理分配	是【】否【】NA【】	
10	是否描述了各种约束条件	是【】否【】NA【】	

2. 制定测试计划

测试工作贯穿于整个软件生命周期，是一项庞大而复杂的工作，需要制定一个完整且详细的测试计划作为指导。测试计划中一般要做好以下工作安排。

- 确定测试范围：明确哪些对象是需要测试的，哪些对象是不需要测试的。
- 制定测试策略：测试策略是测试计划中最重要的部分，它将要测试的内容划分出不同的优先级，以确定测试重点，并根据测试模块的特点和测试类型（如功能测试、性能测试）选定测试环境和测试方法（如人工测试、自动化测试）。
- 安排测试资源：通过考虑测试难度、时间、工作量等因素，对测试资源进行合理安排，包括人员分配、工具配置等。
- 安排测试进度：根据软件开发计划、产品的整体计划来安排测试工作的进度，同时还要考虑各部分工作的变化。在安排工作进度时，最好在各项测试工作之间预留一个缓冲时间以应对计划变更。
- 预估测试风险：罗列出测试工作过程中可能会出现的不确定因素，并制定应对策略。

3. 设计测试用例

测试用例（Test Case）是指一套详细的测试方案，包括测试环境、测试步骤、测试数据和预期结果。不同的公司会有不同的测试用例模板，虽然它们在风格和样式上有所不同，但本质上是一样的，都包括测试用例的基本要素。

测试用例编写的原则是尽量用最少的测试用例达到最大测试覆盖率。测试用例常用的设计方法包括等价类划分法、边界值分析法、因果图与决策表法、正交实验设计法、逻辑覆盖法等，这些设计方法会在后面的章节中陆续讲解。

4. 执行测试

执行测试是按照测试用例执行测试的过程，这是测试人员最主要的活动阶段。在执行测试时要根据测试用例的优先级进行。测试执行过程看似简单，只要按照测试用例完成测试工作即可，但实则并不如此。测试用例的数目非常多，测试人员需要完成所有测试用例的执行，每一个测试用例都可能会发现很多缺陷，测试人员要做好测试记录与跟踪，衡量缺陷的质量并编写缺陷报告。

5. 编写测试报告

测试报告是对一个测试活动的总结，包括对项目测试过程进行归纳，对测试数据进行统计，对项目的测试质量进行客观评价。一份完整的测试报告必须包含以下几个要点。

- 引言：描述测试报告编写目的、报告中出现的专业术语解释和参考资料等。
- 测试概要：介绍项目背景、测试时间、测试地点及测试人员等信息。
- 测试内容及执行情况：描述本次测试模块的版本、测试类型，使用的测试用例设计方法及测试通过覆盖率，依据测试的通过情况提供对测试执行过程的评估结论，并给出测试执行活动的改进建议，以供后续测试执行活动借鉴。
- 缺陷统计与分析：统计本次测试所发现的缺陷数目、类型等，分析缺陷产生的原因，给出规避措施等建议，同时还要记录残留缺陷和未解决问题。
- 测试结论与建议：从需求符合度、功能正确性、性能指标等多个维度对版本质量进行总体评价，给出具体、明确的结论与建议。
- 测试报告的数据是真实的，每一条结论的得出都要有评价依据，不能是主观臆断的。

多学一招

测试的准入、准出

测试准入标准如下。

（1）开发编码结束，开发人员在开发环境中已经进行了单元测试，即开发人员完成了自测。

（2）软件需求上规定的功能都已经实现。如果没有完全实现，开发人员提供测试范围。

（3）测试项目通过基本的冒烟测试，界面上的功能均已实现，符合设计规定的功能。

（4）测试项目的代码符合软件编码规范并已通过评审。

（5）开发人员提交了测试申请并提供了相应的文档资料。

多学一招

测试准出标准如下。

（1）测试项目满足用户需求。

（2）所有测试用例都已经通过评审并成功执行。

（3）测试覆盖率已经达到要求。

（4）所有发现的缺陷都记录在缺陷管理系统中。

（5）一、二级错误修复率达到100%。

（6）三、四级错误修复率达到95%。

（7）所有遗留问题都已经有解决方案。

（8）测试项目的功能、性能、安全性等都满足要求。

（9）完成系统测试总结报告。

多学一招

在测试过程中可能会出现一些意外情况导致测试工作暂停，这个暂停并不是上面所说的测试结束，而是非正常的。测试中需要暂停的情况主要包括以下4种。

（1）测试人员进行冒烟测试时发现重大缺陷，导致测试无法正常进行，需要暂停并返回开发。

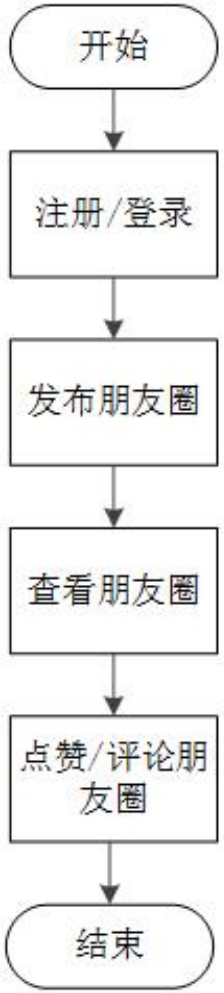
（2）测试人员进行冒烟测试时发现缺陷过多可以申请暂停测试，返回开发。

（3）测试项目需要更新调整而暂停，测试工作也要相应暂停。

（4）如果测试人员有其他优先级更高的任务，可以申请暂停测试。

1.6.2 实例：微信朋友圈功能的测试流程

微信朋友圈的功能业务流程如下图所示。



1.6.2 实例：微信朋友圈功能的测试流程

微信朋友圈功能的业务流程主要包括6个，分别是开始、注册/登录、发布朋友圈、查看朋友圈、点赞/评论朋友圈、结束。下面主要对该业务流程中的发布朋友圈功能进行测试。

(1)分析测试需求

测试人员对软件需求进行分析，并确定要测试的功能是发布朋友圈。发布的朋友圈内容主要有5种形式，分别是文字、照片、视频、文字+照片、文字+视频，假设关于这5种形式的朋友圈内容的具体要求如下。

- 文字：1~1500字。
- 照片：1~9张。
- 视频：1~15秒。

(2)制订测试计划

发布朋友圈功能的测试计划如下表所示。

软件版本	微信8.0.62版本
模块	发布朋友圈
负责人	测试组长
测试人员	测试员1、测试员2
测试时间	2025.09.01~2025.09.03
测试用例	001~008
回归测试	2025.09.10~2025.09.13

(3)设计测试用例

本次测试的重点是发布朋友圈，在设计测试用例时，需要考虑发布朋友圈的内容形式，在分析测试需求阶段可以明确发布朋友圈的内容形式主要有以下5种。

- 只发布文字。
- 只发布照片。
- 只发布视频。
- 发布文字+照片。
- 发布文字+视频。

1.6.2 实例：微信朋友圈功能的测试流程

针对前面5种形式来设计发布朋友圈功能的测试用例，如下表所示。

用例编号	测试功能	测试标题	预置条件	步骤描述	测试数据	预期结果
001	发布朋友圈	发布一段文字	1.网络连接正常； 2.成功登录微信	1.在“发现”界面点击朋友圈； 2.长按朋友圈界面右上角的相机图标； 3.输入一段文字然后点击“发表”按钮	一段字数为50的内容	发布成功
002	发布朋友圈	发布内容为空	1.网络连接正常； 2.成功登录微信	1.在“发现”界面点击朋友圈； 2.长按朋友圈界面右上角的相机图标； 3.不输入内容	内容为空	发布失败
003	发布朋友圈	发布1张照片	1.网络连接正常； 2.成功登录微信； 3.进入发布朋友圈界面	1.在发布朋友圈界面点击相机图标； 2.在相册中选择任意一张照片	1张照片	发布成功
004	发布朋友圈	发布10张照片	1.网络连接正常； 2.成功登录微信； 3.进入发布朋友圈界面	1.在发布朋友圈界面点击相机图标； 2.在相册中选择照片	10张照片	发布失败

1.6.2 实例：微信朋友圈功能的测试流程

用例编号	测试功能	测试标题	预置条件	步骤描述	测试数据	预期结果
005	发布朋友圈	发布一段15秒视频	1.网络连接正常； 2.成功登录微信； 3.进入发布朋友圈界面	1.在发布朋友圈界面点击相机图标； 2.在相册中选择一段视频	一段15秒的视频	发布成功
006	发布朋友圈	发布一段20秒视频	1.网络连接正常； 2.成功登录微信； 3.进入发布朋友圈界面	1.在发布朋友圈界面点击相机图标； 2.在相册中选择一段视频	一段20秒的视频	发布失败
007	发布朋友圈	发布文字+照片	1.网络连接正常； 2.成功登录微信； 3.进入发布朋友圈界面	1.在发布朋友圈界面点击相机图标； 2.在相册中选择照片； 3.输入文字	1.选择9张照片； 2.输入一段10字文字	发布成功
008	发布朋友圈	发布文字+视频	1.网络连接正常； 2.成功登录微信； 3.进入发布朋友圈界面	1.在发布朋友圈界面点击相机图标； 2.在相册中选择视频； 3.输入文字	1.选择一段10秒视频； 2.输入一段10字文字	发布成功

(4)测试执行

发布朋友圈功能的测试缺陷报告如下表所示。

缺陷ID	25_09_001
测试软件名称	微信
测试软件版本	8.0.62
缺陷发现日期	20250902
测试人员	测试员1
缺陷描述	该版本的发布朋友圈功能在输入内容为空的情况下，点击“发表”按钮后发布成功
附件（可附图）	附图1（链接）
缺陷类型	功能类型缺陷

缺陷严重程度	严重
缺陷优先级	立即解决
测试环境	手机信息：vivo X100; 内存：16.0GB; 系统类型：Android 15.0操作系统
重现步骤	1.在发现界面点击朋友圈; 2.长按朋友圈界面右上角的相机图标; 3.不输入内容; 4.点击“发表”按钮
备注	无

(5)编写测试报告

本次测试的测试报告可以按照如下目录编写。

一、引言

1.目的

2.术语解释

3.参考资料

二、测试概要

1.项目简介

2.测试环境

3.测试时间、地点及人员

三、测试内容及执行情况

1.测试目标

2.测试范围

3.测试用例使用情况

4.回归测试

四、缺陷统计与分析

1.缺陷数目与类型

2.缺陷的解决情况

3.缺陷的趋势分析

五、测试分析

1.测试覆盖率分析

2.需求符合度分析

3.功能正确性分析

4.产品质量分析

5.测试局限性

六、测试总结

1.遗留问题

2.测试经验总结

七、附件

1.测试用例清单

2.缺陷清单

3.交付的测试工作产品

4.遗留问题报告

本章小结

本章对软件测试的基础知识进行了讲解，首先介绍了软件相关的知识，包括软件生命周期、软件开发模型、软件质量；其次讲解了软件缺陷管理，包括软件缺陷产生的原因、分类、处理流程和常见的缺陷管理工具；然后讲解了软件测试的简介、目的、分类，软件测试与软件开发的关系、常见的软件测试模型、软件测试的原则；最后讲解了软件测试的基本流程，并且通过微信发布朋友圈的功能测试让读者简单认识了软件测试的基本流程。本章的知识细碎且独立，却是软件测试入门的必备知识，能为读者在后续章节更深入地学习软件测试打下坚实的基础。

谢谢