瀑布模型

瀑布模型（waterfall model）是一个经典的模型，也称为传统模型（conventional model），它是一个理想化的生存期模型，如图 3- 2 所示。它要求项目所有的活动都严格按照顺序自上而下执行，一个阶段的输出是下一阶段的输入，如同瀑布流水，逐级下落。瀑布模型没有反馈，一个阶段完成后，一般不返回——尽管实际的项目中要经常返回上一阶段。瀑布模型是一个比较“老”的模型，甚至有些过时，但在一些小的项目中还是经常用到的。

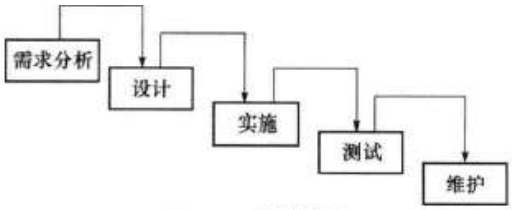

图 3-2 瀑布模型

瀑布模型的优点：

1）简单、易用、直观。

2）开发进程比较严格，一个进程接着一个进程进行。

3）模型执行过程中需要严密控制。

4）允许基线和配置早期接受控制。

5）为项目提供了按阶段划分的检查点，当前一个阶段完成后，只需要关注后续阶段。

瀑布模型的缺点：

1）在软件开发的初期阶段就要求做出正确、全面、完整的需求分析对许多应用软件来说是极其困难的。

2）由于开发模型是线性的，模型中没有反馈过程，用户只有等到整个过程的末期才能见到开发成果，从而增加了开发风险。

3）一个新的项目不适合瀑布模型，除非在项目的后期。

4）用户直到项目结束才能看到产品的质量，用户不是渐渐地熟悉系统。

5）不允许变更或者限制变更。

6）早期的错误可能要等到开发后期才能发现，进而带来严重后果。

瀑布模型的适用范围：适合于软件需求很明确的软件项目，即一般适用于功能明确、完整、无重大变化的软件系统的开发，即

1）在项目开始前，项目的需求已经被很好地理解，也很明确，而且项目经理很熟悉为实现这一模型所需要的过程。

2）解决方案在项目开始前也很明确。

3）短期项目可以采用瀑布模型。

瀑布模型的使用说明：

1）开发前，要进行概念开发和系统配置开发，概念开发主要是确定系统级的需求，提交一份任务陈述。系统配置开发需要确定软件和硬件的情况。

2）开发中，需进行需求过程、设计过程、实施过程。

3）开发后，需进行安装过程、支持过程、维护过程等。

V 模型

V 模型是由 Paul Rook 在 1980 年提出的，是瀑布模型的一种变种，同样需要一步一步进行，前一阶段任务完成之后才可以进行下一阶段的任务，如图 3-3 所示。这个模型强调测试的重要性，将开发活动与测试活动紧密地联系在一起。每一步都将比前一阶段进行更加完善的测试。

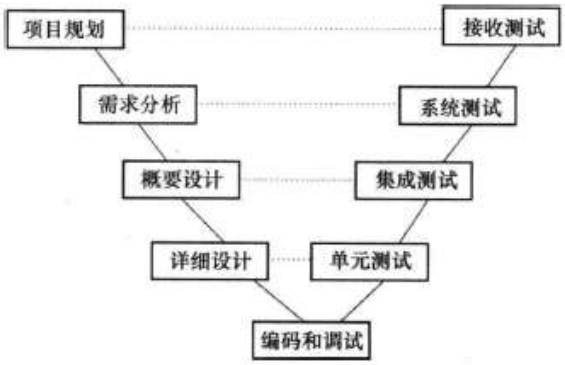

图 3-3 V 模型

一般，大家对测试存在一种误解，认为测试是开发周期的最后一个阶段。其实，早期的测试工作对提高产品的质量、缩短开发周期起着重要作用。V 模型正好说明了测试的重要性，它是与开发并行的，例如，单元测试对应详细设计，集成测试对应概要设计，系统测试对应需求分析。V 模型体现了全过程的质量意识。

V 模型的优点：

1）简单易用，只要按照规定的步骤一步一步执行即可。

2）强调测试过程与开发过程的对应性和并行性。

3）开发进程比较严格，执行过程需要严密控制。

4）允许基线和配置早期接受控制。

5）为项目提供了按阶段划分的检查点，当前一个阶段完成后，只需要关注后续阶段。

V 模型的缺点：

1）软件开发的初期阶段就要求做出正确、全面、完整的需求分析。

2）软件项目的实现方案需要很明确。

3）不能存在变更。

V 模型的适用范围：

1）项目的需求在项目开始前很明确。

2）解决方案在项目开始前也很明确。

3）项目对系统的性能安全很严格，如航天飞机控制系统、公司的财务系统等。

V 模型的使用说明：使用 V 模型，要求开发的全过程是严格按照顺序进行的，一个阶段的输出是下一个阶段的输入。同时，注意图 3-3 中虚线对应过程的并行考虑，例如，在需求分析阶段，应该有系统测试的准备；在概要设计阶段，应该有集成测试的准备；在详细设计阶段，应有单元测试的准备等。

快速原型模型

快速原型模型是在需求阶段快速构建一部分系统的生存期模型，实现客户或未来用户与系统的交互，而且用户或客户可以对原型进行评价，这些反馈意见可以作为进一步系统修改的依据。通过逐步调整原型使其满足客户的要求，开发人员可以确定客户的真正需求是什么；开发人员对开发的产品有时与客户不一致，因为开发人员更关注设计和编码实施，而客户更关注于需求。因此，如果开发人员快速构造一个原型将会很快与客户就需求达成一致。

快速原型模型通常从最核心的方面开始，向用户展示完成的部分，然后根据用户的反馈信息继续开发原型，并重复这一过程，直到开发者和用户都认为原型已经足够好，然后在此基础上开发客户满意的软件产品，交付作为最终产品的原型，如图 3-4 所示。

图 3-4　原型模型

快速原型模型以逐步增加的方式进行开发，以便于随时根据客户或最终用户的反馈来修正系统，在需求变化很快的时候，或者用户很难提出明确需求的时候，或者开发人员对最佳的架构或算法没有把握的时候，渐进原型特别有用。但是，原型模型是以牺牲项目的可控制性来换取较多的客户反馈及较好的过程可视性的。由于原型的功能和特性会随着用户的反馈而经常发生变化，因此较难确定产品的最终形态。

快速原型模型的优点：

1）可以克服瀑布模型的缺点，减少由于软件需求不明确带来的开发风险。

2）用户根据快速构建的原型系统的优缺点，给开发人员提出反馈意见。

3）根据反馈意见修改软件需求规格，以便系统可以更正确地反映用户的需求。

4）可以减少项目的各种假设及风险等。

快速原型模型的缺点：

1）需求定义之前，需要快速构建一个原型系统。

2）所选用的开发技术和工具不一定符合主流的发展。

3）快速建立起来的系统结构加上连续的修改可能会导致产品质量低下。

4）使用这个模型的前提是要有一个展示性的产品原型，因此在一定程度上可能会限制开发人员的创新。

快速原型模型的适用范围：

1）项目的需求在项目开始前不明确。

2）需要减少项目的不确定性的时候。

快速原型模型的使用说明：

1）用户和开发人员根据初始需求共同开发一个项目规划。

2）用户和开发人员利用快速分析技术共同定义需求规格。

3）设计者构建一个原型系统。

4）设计者演示这个原型系统，用户来评估性能并标识问题。

5）用户和设计者一起来解决标识的问题，循环这个过程，直到用户满意为止。

6）详细设计可以根据这个原型进行。

7）原型可以用代码或者工具来实施。

增量式模型

增量式模型（incremental life cycle model）是假设需求可以分段，成为一系列增量产品，每一增量可以分别开发。首先构造系统的核心功能，然后逐步增加功能和完善性能的方法就是增量式模型。增量式模型在各个阶段并不交付一个可运行的完整产品，而是交付满足客户需求的一个子集的可运行产品。整个产品被分解成若干个构件，开发人员逐个构件地交付产品。在使用增量式模型时，第一个增量往往是实现基本需求的核心产品。核心产品交付用户使用后，经过评价形成下一个增量的开发计划，它包括对核心产品的修改和一些新功能的发布。这个过程在每个增量发布后不断重复，直到产生最终的完善产品。增量式模型如图 3-5 所示。

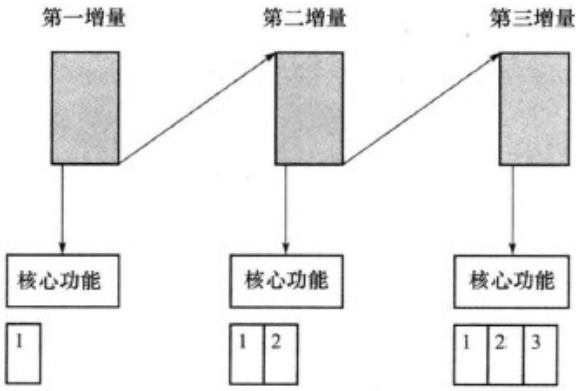

图 3-5　增量式模型

增量式模型的优点：

1）软件开发可以较好地适应变化，客户可以不断地看到所开发的软件，从而降低开发风险。

2）可以避免一次性投资太多带来的风险，首先实现主要的功能或者风险大的功能，然后逐步完善，保证投入的有效性。

3）可以更快地开发出可以操作的系统。

4）可以减少开发过程中用户需求的变更。

增量式模型的缺点：

1）由于各个构件是逐渐并入已有的软件体系结构中的，因此加入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。

2）在开发过程中，需求的变化是不可避免的。增量式模型的灵活性可以使其适应这种变化的能力大大优于瀑布模型和快速原型模型，但一些增量可能需要重新开发，从而使软件过程的控制失去整体性（如果早期开发的需求不稳定或者不完整）。

增量式模型的适用范围：

1）进行已有产品升级或新版本开发，增量式模型是非常适合的。

2）对于完成期限要求严格的产品，可以使用增量式模型。

3）对于所开发的领域比较熟悉而且已有原型系统，增量式模型是非常适合的。

4）对于市场和用户把握不是很准，需要逐步了解的项目，可采用增量式模型。

增量式模型的使用说明：使用增量式模型时，首先构建整个系统的核心部分，或者具有高风险的部分功能，这部分功能对项目的成功起到重要作用。通过测试这些功能以决定它们是否是项目所需要的，这样可以排除后顾之忧，然后逐步地增加功能和性能，循序渐进，避免太快，增加功能的时候应该高效而且符合用户的需要。

渐进式阶段模型

对于软件项目来讲，可以将大的项目划分成几个小项目来做，将周期长的项目划分成几个明确的阶段。“化繁为简，各个击破”是解决复杂问题的一个方法。例如，一个5年完成的项目可以分成5个阶段，每年提交一个版本，无形中的感觉是工作时间缩短了，工作量变小了，尽管实际中的工作量没有减少。开发过程中反复和阶段提交是比较合理的过程，渐进式阶段模型恰恰体现了这些特征。它也是近来比较流行的生存期模型。渐进式阶段模型如图3-6和图3-7所示。

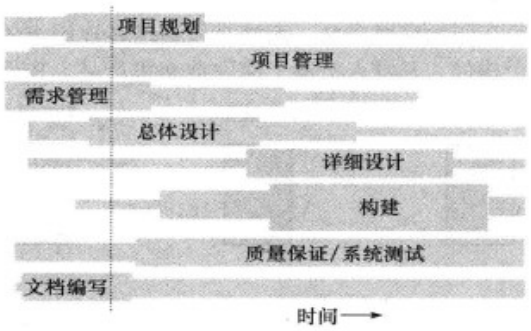

图3-6　渐进式模型

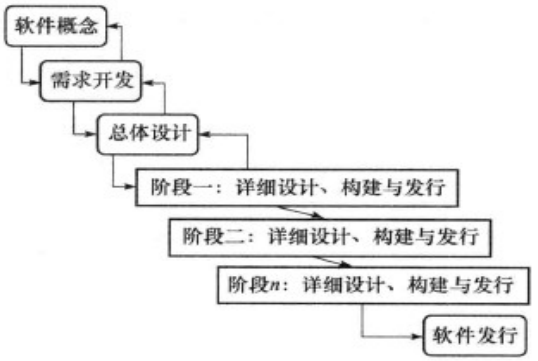

图3-7　阶段式模型

渐进式阶段模型体现渐进式过程和阶段提交的模式，从图3-6可以看出，“项目规划”、“项目管理”、“需求管理”、“总体设计”、“详细设计”、“构建”、“质量保证/系统测试”、“文档编写”等过程是贯穿项目始终的，只是各个阶段的任务量不同而已，“项目规划”开始任务多，而后每个阶段的工作量逐渐变少了，“项目管理”是从始至终都有的，而“需求管理”开始任务多，然后逐步变少，“总体设计”也是在需求快结束的时候开始，然后逐步减少。这个模型强调将项目中开发的软件分阶段完成，每个阶段可以提交不同版本的产品，而不是一次性完成，如图3-7所示。选择渐进式阶段模型的项目团队先进行软件概念分析，汇集、分析需求，再完成架构设计。这些工作通过积极的风险管理与精心规划，朝着消除风险的目标前进。在每个实施阶段，进行详细设计、实施、测试、修正等，而且每个阶段都建立可能推出的产品。这样项目经理可以提早得到明确的进度报告，而不是类似“完成80%”、“还剩20%”等模棱两可的报告。产品的动态结果比任何书面报告更能精确反应项目状况。渐进式阶段模型也称为渐进式迭代模型，每个阶段就是一个迭代过程。

渐进式阶段模型的优点：

1）阶段式提交一个可运行的产品，而且每个阶段提交的产品是独立的系统，如图3-8所示。

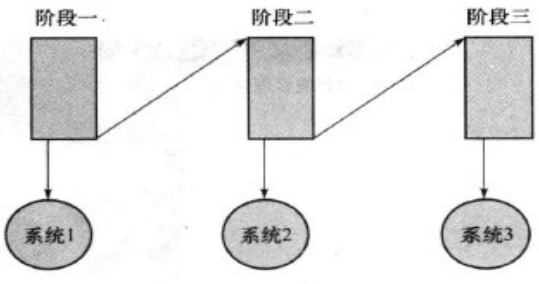

图3-8　渐进式系统

2）关键的功能更早出现，可以提高开发人员和客户的信心。

3）通过阶段式产品提交，可以早期预警问题，避免后期发现问题的高成本。

4）通过阶段式提交可以运行的产品来说明项目的实际进展，减少项目报告的负担。

5）阶段性完成可以降低估计失误，因为通过阶段完成的评审，可以重新估算下一阶段的计划。

6）阶段性完成均衡了弹性与效率，提高开发人员的效率和士气。

渐进式阶段模型的缺点：

1）需要精心规划各个阶段的目标。

2）每个阶段提交的是正式版本，所以工作量会增加。

渐进式阶段模型的适用范围：从本质上讲，渐进式阶段模型可以适用于任何规模的项目，但是需要不断提交新的版本，因此渐进式阶段模型主要适用于中型或大型项目，是目前软件开发中常采用的模型。采用这个模型可以随时看到项目的未来。

敏捷生存期模型

由于高新技术的出现及技术更迭越来越快，产品的生命周期日益缩短，企业要面对新的竞争环境，抓住市场机遇，迅速开发出用户所需要的产品，就必须实现敏捷反应。与此同时，业界不断探寻适合软件项目的开发模式，其中，敏捷软件开发（agile software development）模式越来越得到大家的关注和采用。

敏捷开发是一个灵活的开发方法，用于在一个动态的环境中向干系人快速交付产品。其主要特点是关注持续的交付价值，通过迭代和快速的用户反馈管理不确定性和应对变更。

2001 年年初，许多公司的软件团队陷入了不断增长的过程的泥潭，一批业界专家聚集在一起概括出了一些可以让软件开发团队具有快速工作、响应变化能力的价值观和原则，他们称自己为敏捷联盟。在随后的几个月中，他们创建出了一份价值观声明，即敏捷开发宣言：

- 个体和交互胜过过程和工具。
- 可以工作的软件胜过面面俱到的文档。
- 客户合作胜过合同谈判。
- 响应变化胜过遵循计划。

敏捷软件开发是一种面临迅速变化的需求快速开发软件的能力，是对传统生存期模型的挑战，也是对复杂过程管理的挑战；是一种以人为核心、迭代、循序渐进的开发方法；是一种轻量级的软件开发方法。

图 3-9 是敏捷组织提出的敏捷开发模型的整体框架，其核心价值观是敏捷开发宣言。下面重点介绍 Scrum、XP（eXtreme Programming）、OpenUP 3 个敏捷实践。

图 3-9　敏捷开发模型的整体框架