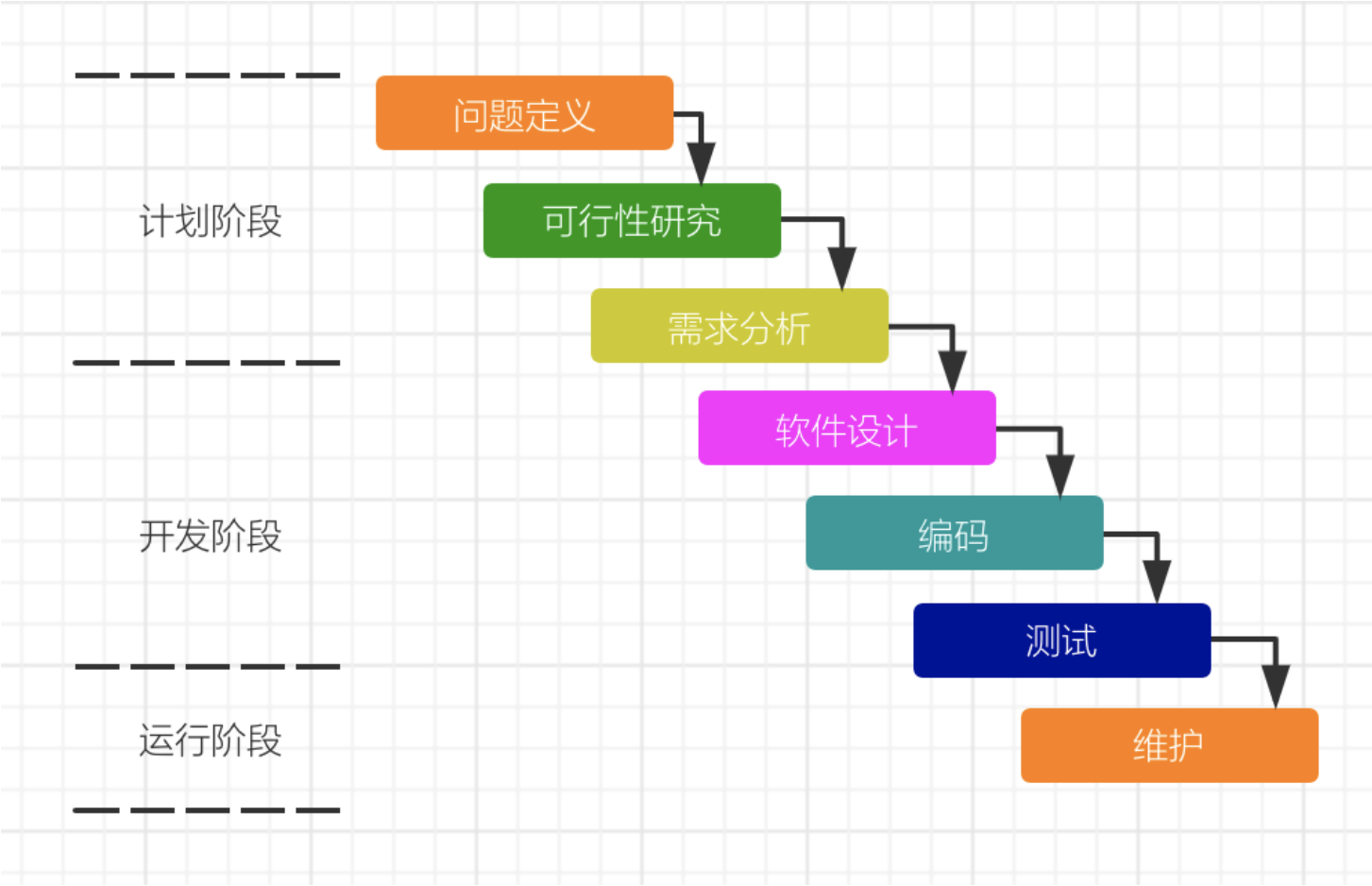
常见项目流程

- 瀑布模型
- "V"模式开发模型
- 原型模型
- 螺旋模型
- 迭代模型
- 敏捷开发模式

瀑布开发模型



瀑布模型是将软件生存周期的各项活动规定为按固定顺序而连接的若干阶段工作，形如瀑布流水，最终得到软件产品。

1970年温斯顿·罗伊斯（Winston Royce）提出了著名的“瀑布模型”，直到80年代早期，它一直是唯一被广泛采用的软件开发模型。

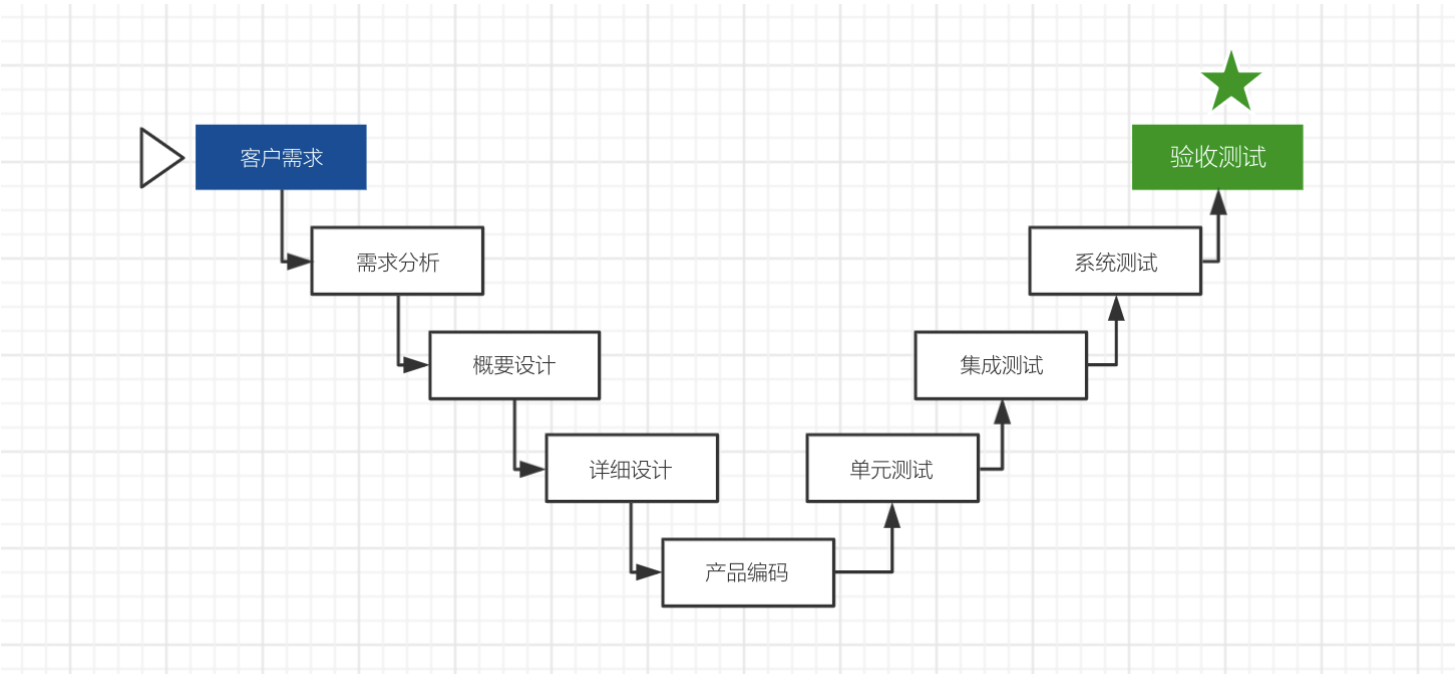
核心思想：瀑布模型核心思想是按工序将问题化简，将功能的实现与设计分开，便于分工协作，即采用结构化的分析与设计方法将逻辑实现与物理实现分开。将软件生命周期划分为制定计划、需求分析、软

件设计、程序编写、软件测试和运行维护等六个基本活动，并且规定了它们自上而下、相互衔接的固定次序，如同瀑布流水，逐级下落。

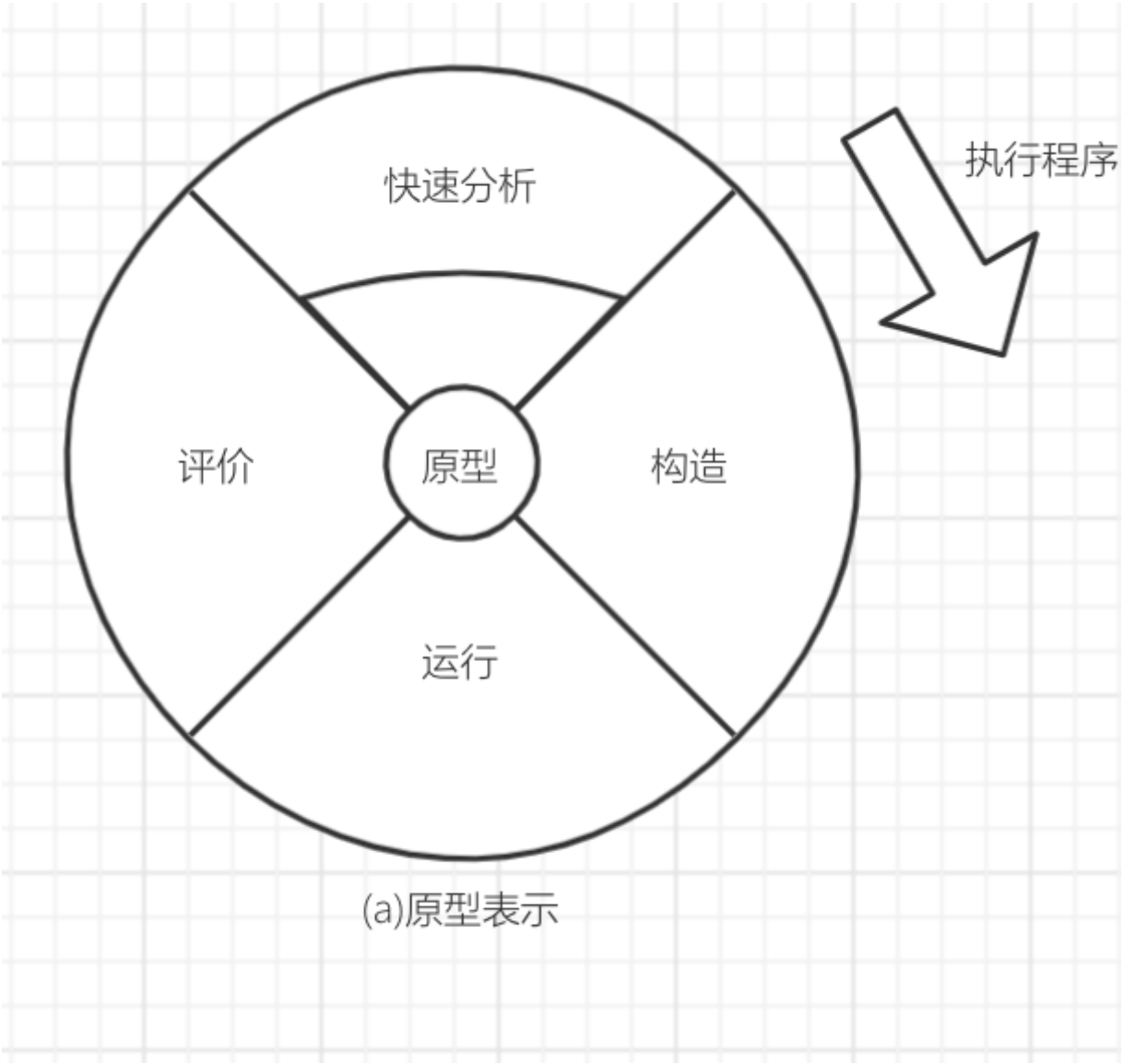
重要地位：它提供了软件开发的基本框架。其过程是从上一项活动接收该项活动的工作对象作为输入，利用这一输入实施该项活动应完成的内容给出该项活动的工作成果，并作为输出传给下一项活动。同时评审该项活动的实施，若确认，则继续下一项活动；否则返回前面，甚至更前面的活动。对于经常变化的项目而言，瀑布模型毫无价值。

- 模型优点:
  1. 为项目提供了按阶段划分的检查点。
  2. 当前一阶段完成后，您只需要去关注后续阶段。
  3. 可在迭代模型中应用瀑布模型。
  4. 它提供了一个模板，这个模板使得分析、设计、编码、测试和支持的方法可以在该模板下有一个共同的指导。
- 模型缺点:
  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，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. 通过过多的强制完成日期和里程碑来跟踪各个项目阶段。
  4. 瀑布模型的突出缺点是不适应用户需求的变化。

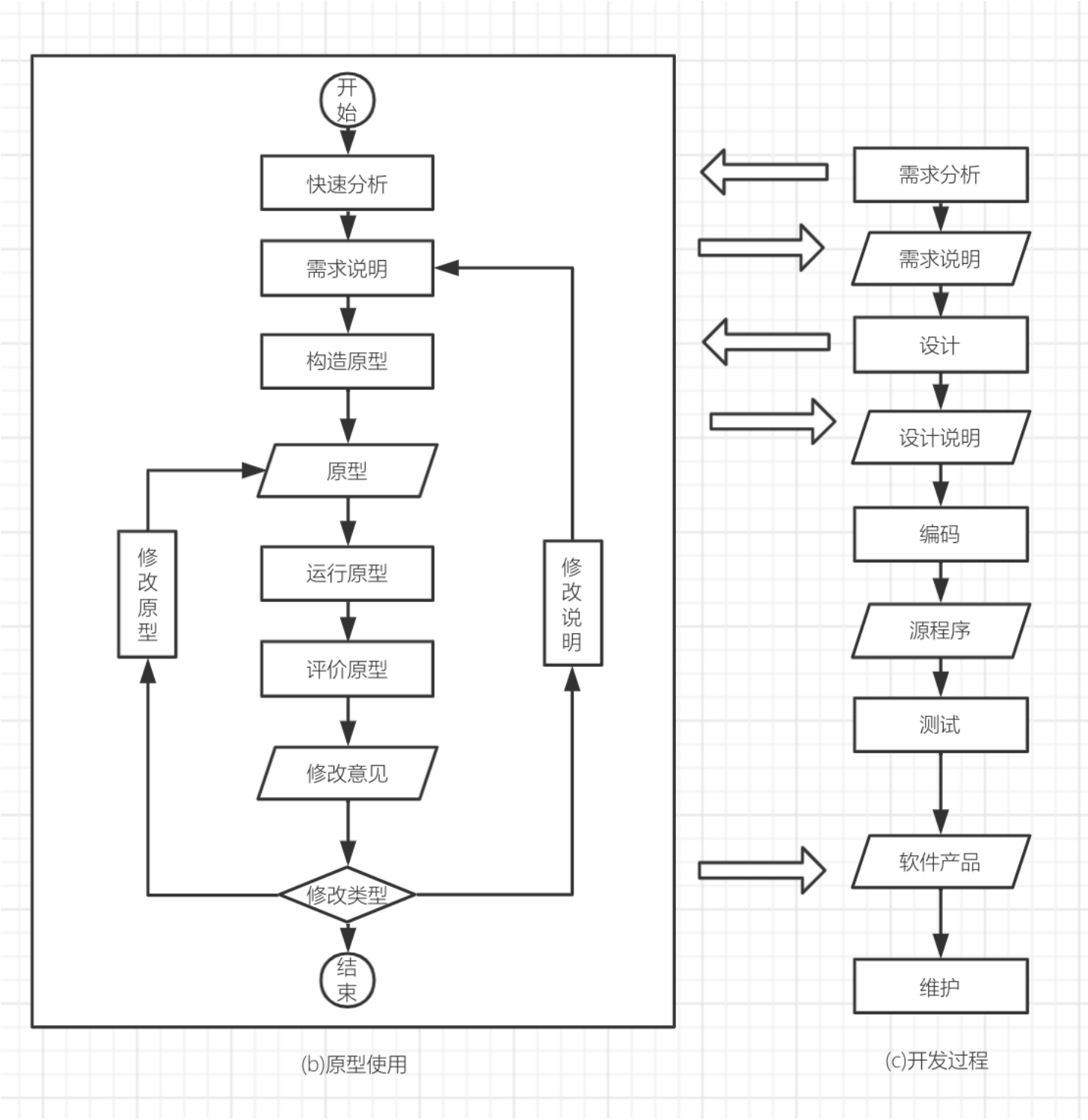
"V"模式开发模型



原型模型

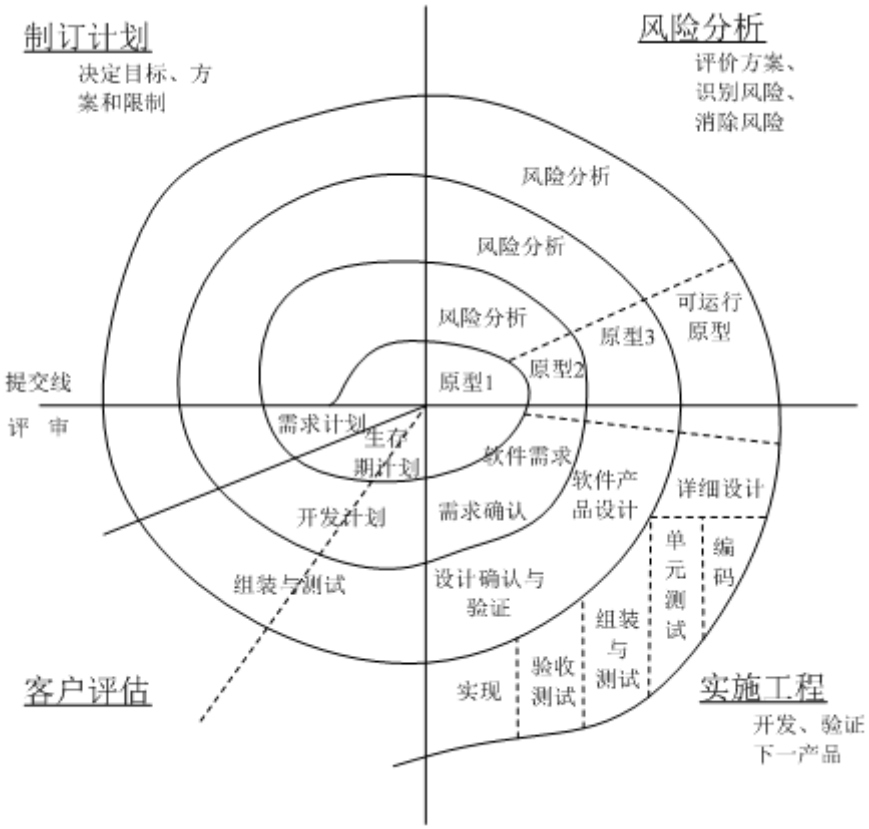


(a)原型表示



- 优点：
  - 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。
  - 这种模型适合预先不能确切定义需求的软件系统的开发。
- 缺点：
  - 所选用的开发技术和工具不一定符合主流的发展；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。
  - 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。

螺旋模型



瀑布模型和快速原型模型结合起来，强调了其他模型所忽视的风险分析，特别适合于大型复杂的系统。

螺旋模型是一种演化软件开发过程模型，它兼顾了快速原型的迭代的特征以及瀑布模型的系统化与严格监控。螺旋模型最大的特点在于引入了其他模型不具备的风险分析，使软件在无法排除重大风险时有机会停止，以减小损失。同时，在每个迭代阶段构建原型是螺旋模型用以减小风险的途径。螺旋模型更适合大型的昂贵的系统级的软件应用。

螺旋模型采用一种周期性的方法来进行系统开发。这会导致开发出众多的中间版本。使用它，项目经理在早期就能够为客户实证某些概念。该模型是快速原型法，以进化的开发方式为中心，在每个项目阶段使用瀑布模型法。这种模型的每一个周期都包括需求定义、风险分析、工程实现和评审4个阶段，由这4个阶段进行迭代。软件开发过程每迭代一次，软件开发又前进一个层次。螺旋模型基本做法是在“瀑布模型”的每一个开发阶段前引入一个非常严格的风险识别、风险分析和风险控制，它把软件项目分解成一个个小项目。每个小项目都标识一个或多个主要风险，直到所有的主要风险因素都被确定。

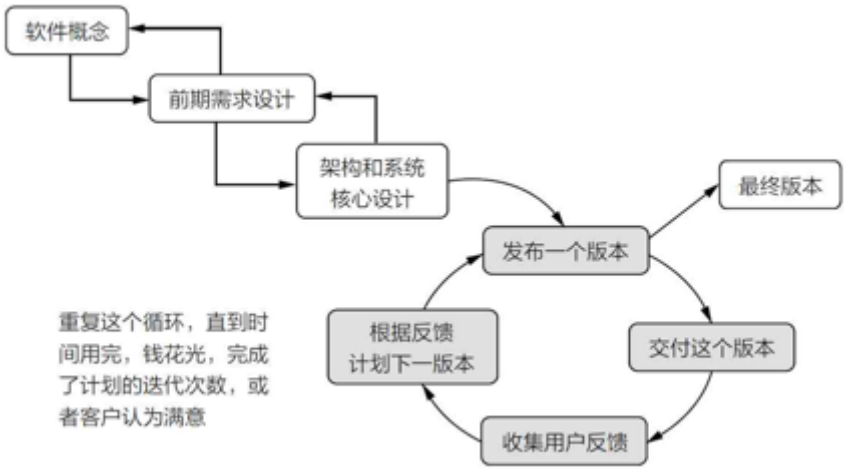
螺旋模型基本做法是在“瀑布模型”的每一个开发阶段前引入一个非常严格的风险识别、风险分析和风险控制，它把软件项目分解成一个个小项目。每个小项目都标识一个或多个主要风险，直到所有的主要风险因素都被确定。
螺旋模型强调风险分析，使得开发人员和用户对每个演化层出现的风险有所了解，继而做出应有的反应，因此特别适用于庞大、复杂并具有高风险的系统。对于这些系统，风险是软件开发不可忽视且潜在

的不利因素，它可能在不同程度上损害软件开发过程，影响软件产品的质量。减小软件风险的目标是在造成危害之前，及时对风险进行识别及分析，决定采取何种对策，进而消除或减少风险的损害。

- 模型优点:
  1. 设计上的灵活性,可以在项目的各个阶段进行变更
  2. 以小的分段来构建大型系统,使成本计算变得简单容易。
  3. 客户始终参与每个阶段的开发,保证了项目不偏离正确方向以及项目的可控性。
  4. 随着项目推进,客户始终掌握项目的最新信息，从而他或她能够和管理层有效地交互。
  5. 客户认可这种公司内部的开发方式带来的良好的沟通和高质量的产品。
- 模型缺点
  很难让用户确信这种演化方法的结果是可以控制的。建设周期长，而软件技术发展比较快，所以经常出现软件开发完毕后，和当前的技术水平有了较大的差距，无法满足当前用户需求。

迭代模型

敏捷开发模式



敏捷模型以迭代模型为理论基础。

敏捷模型的概念比较广，包含：XP、Scrum、FDD、DSDM、Crystal、ASD等等开发方法。或者说，敏捷模型是这些轻量级软件开发方法的一个统称。

模型说明

敏捷开发模式是一种从1990年代开始逐渐引起广泛关注的一些新型软件开发方法，是一种应对快速变化的需求的一种软件开发能力。它们的具体名称、理念、过程、术语都不尽相同，相对于"非敏捷"，更强调程序员团队与业务专家之间的紧密协作、面对面的沟通（认为比书面的文档更有效）、频繁交付新的软件版本、紧凑而自我组织型的团队、能够很好地适应需求变化的代码编写和团队组织方法，也更注重

做为软件开发中人的作用。
敏捷开发以用户的需求进化为核心，采用迭代、循序渐进的方法进行软件开发。在敏捷开发中，软件项目在构建初期被切分成多个子项目，各个子项目的成果都经过测试，具备可视、可集成和可运行使用的特征。换言之，就是把一个大项目分为多个相互联系，但也可独立运行的小项目，并分别完成，在此过程中软件一直处于可使用状态。

软件敏捷开发宣言中提出了以下12个原则：

1. 持续、尽早交付有价值的软件以满足客户，是我们优先要做的首要任务。
2. 拥抱需求变更，甚至是在开发的后期。敏捷过程利用变更为客户带来竞争优势。
3. 频繁交付可执行的软件，从几周到几个月，交付时间越短越好。
4. 在整个项目过程中，业务人员和开发人员必须每天在一起工作。
5. 激发每个团队成员的积极性来打造项目。为他们提供所需的环境与支持，并且信任他们可以完成工作。
6. 在一个开发团队内部最有效的传递信息的方式是面对面的交流。
7. 可执行的软件是进度的首要检验对象。
8. 敏捷过程倡导可持续发展。赞助商，开发人员和用户应该尽可能保持一致的步伐。
9. 保持关注先进的技术与设计会增强敏捷性。
10. 尽量用艺术化来简单阐述未完成的工作是很有必要的。
11. 最好的架构，需求，和设计出自于自我组织管理的团队。
12. 每隔一段时间，回顾反思如何让团队变得更高效，并相应地调整其行为。

- 模型优点
    1. 迭代快，开发周期短；
    2. 不再耗费大量的时间来写文档，而是人与人面对面交流，只写一些必要的文档；
    3. 分工详细，每天都输出成果，客户能够看得到，会信任项目团队；
    4. 沟通多，容易发现问题，同时能够激起团队的协作、奋斗；
- 模型缺点
    1. 人与人之间的信任是非常重要的环节，但是这个比较难完成，技术团队的成员可能技术能力差别大，同时也有互相竞争，又或者是项目团队的成员有所保留，不愿意这样的沟通；
    2. 团队在开发期间的任务多、压力大，需要时刻保持“兴奋”，一般很难做到。