第一部分：研讨传统软件开发过程模型与敏捷开发

**传统软件开发过程模型与敏捷开发的比较（李雨桐）**

传统软件开发过程模型 如瀑布模型 原型模型 增量模型 螺旋模型

新型软件开发模型 如敏捷开发模型

传统模型的缺点为

（1）过分强调了分阶段实施，使得开发过程各个阶段之间存在严重的顺序性和依赖性

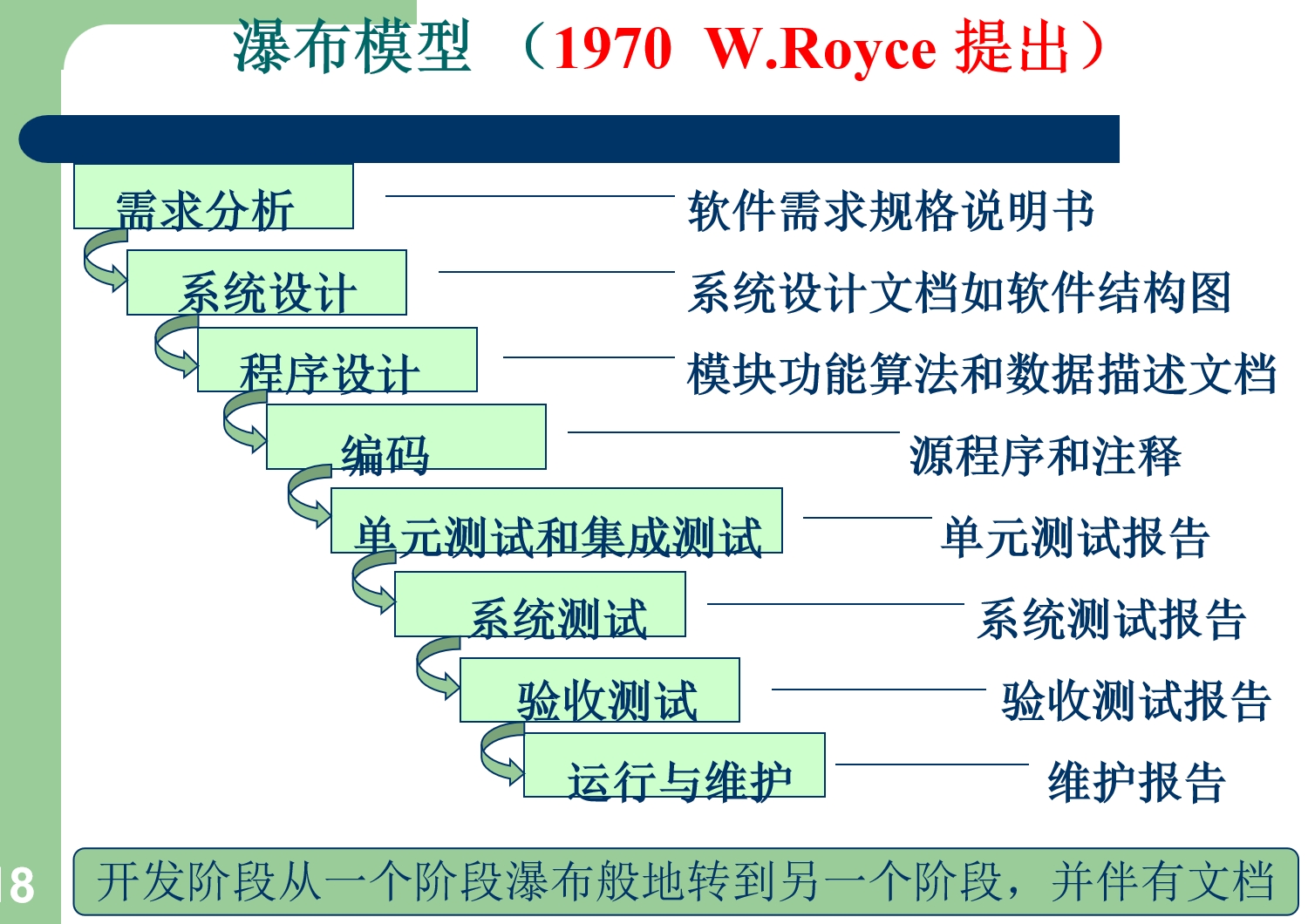
（2）思维成果的可重用性很差

（3）忽视了人在软件开发过程中的地位和作用

1. 瀑布模型

1.基本思想

瀑布模型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。其 严格强调文档，前一个阶段的输出就是下一个阶段的输入，文档是个阶段衔接的唯一信息。所以很多开发人员好象是在开发文档，而不是开发软件，因为要到开发的后期，才可以看到软件的“模样”。



2.

优点：

（1）让软件开发过程有序可控，为项目提供了按阶段划分的检查点。瀑布模型的每个阶段都有明确的任务，每个阶段都有明确的交付产物，都有相应的里程碑。这些让整个过程更可控，而且能及早发现问题

（2）当前一阶段完成后，您只需要去关注后续阶段

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。

（3）让分工协作变成可能。瀑布模型的六个阶段，也让软件开发产生相应的基础分工：项目经理、产品经理、架构师、软件工程师、测试工程师、运维工程师。

（4）质量有保障。瀑布模型每个阶段都需要交付相应的文档，而文档的撰写和评审，可以帮助在动手之前把问题沟通清楚，想清楚。瀑布模型在编码结束后，会有严密的测试，只有测试验收通过后，才能上线发布，这些措施都让软件的质量更有保障。

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）没有迭代与反馈。瀑布模型对反馈没有涉及，所以对变化的客户需求非常不容易适应，瀑布就意味着没有回头路。

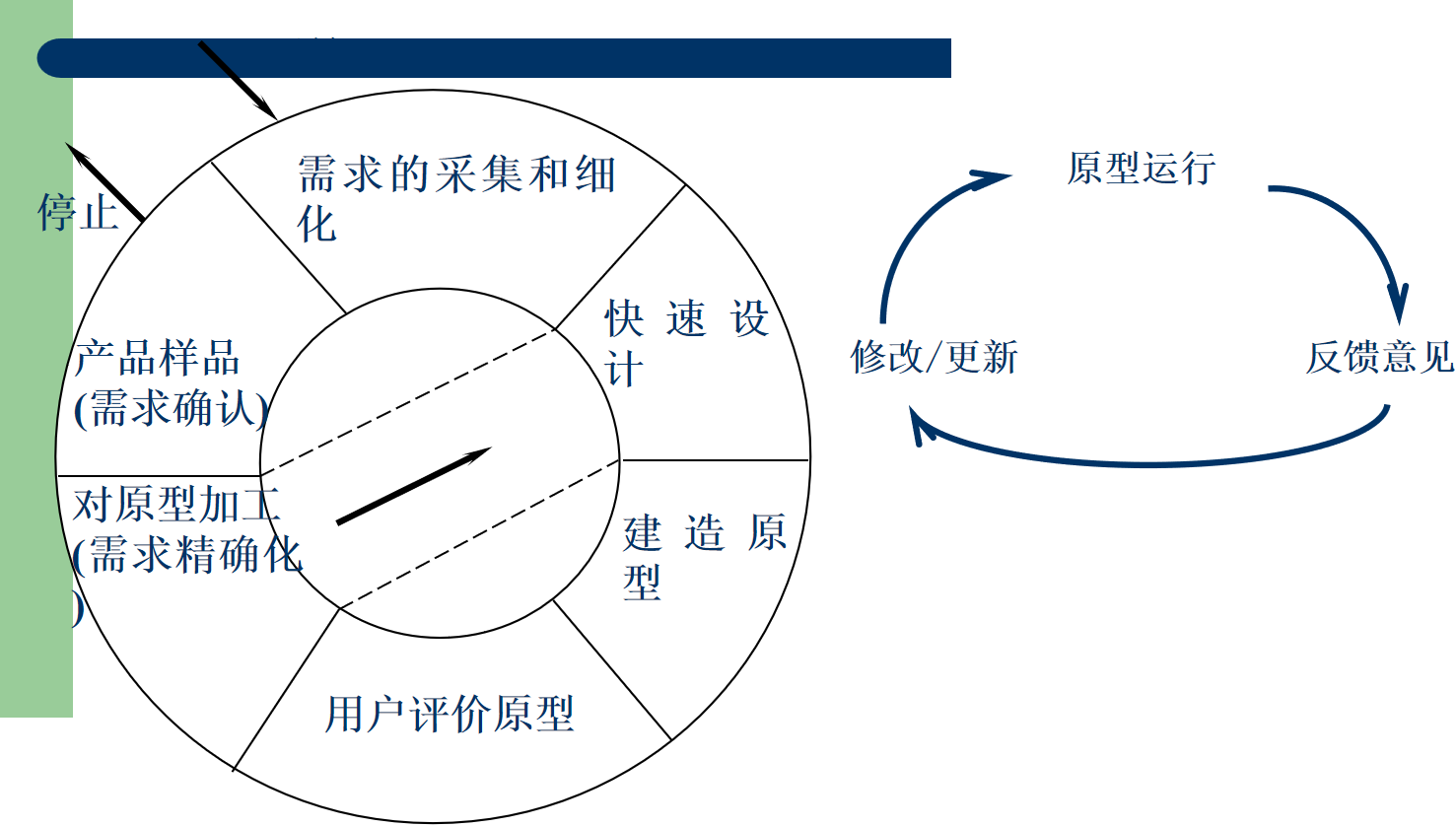
（4）通过过多的强制完成日期和里程碑来跟踪各个项目阶段。

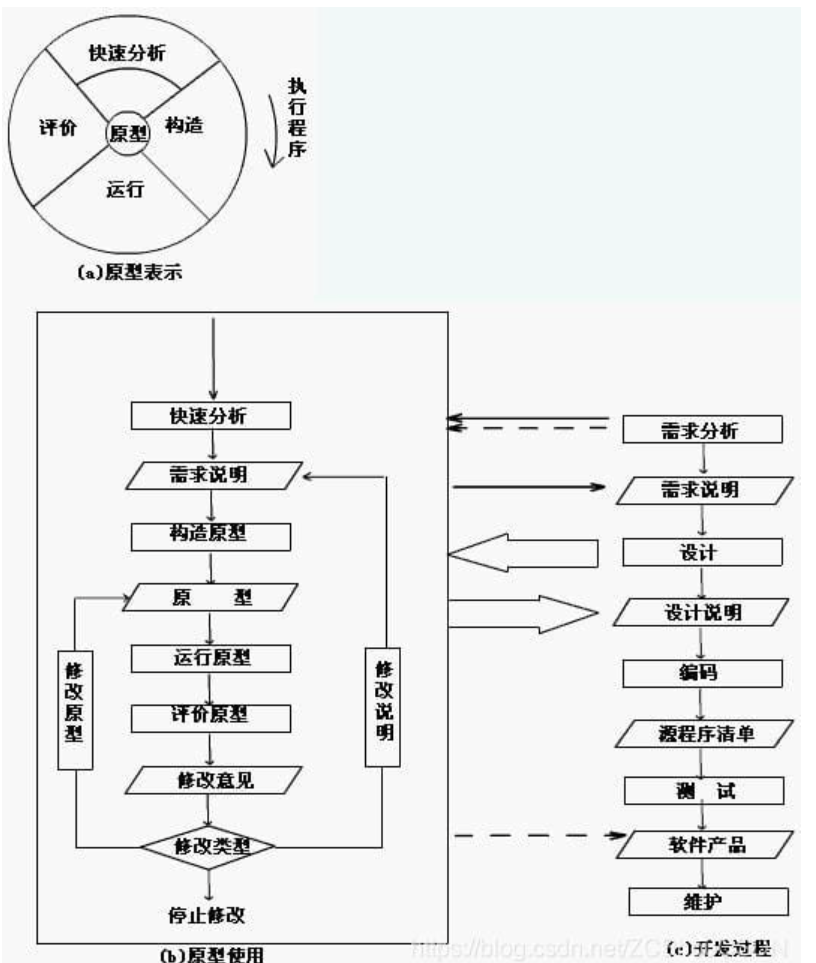
（5）瀑布模型的突出缺点是不适应用户需求的变化。

（6）瀑布模型是一种软件文档的开发，把开发者变成流水线上的机器，大量重复性的工作让编程人员提不起兴趣，工作很枯燥，没有激情，编程成了一种没有创意的机械劳动，这让一向以高科技为标志的高级程序人员大为恼火。

1. 原型模型
2. 基本思想：

快速原型模型（Rapid Prototype Model）又称原型模型，它是增量模型的另一种形式；它是在开发真实系统之前，构造一个原型，在该原型的基础上，逐渐完成整个系统的开发工作。





2.优点

（1）原型系统已经通过与用户交互而得到验证，克服瀑布模型的缺点，据此产生的规格说明可以正确地描述用户的需求。因此，在开发过程的后续阶段不会因为发现了规格说明文档的错误而进行较大的返工。

（2）开发人员通过建立原型系统已经学到了许多东西（至少知道了“系统不应该做什么，以及怎么不去做不该做的事情”），因此，在设计和编码阶段发生错误的可能性也比较小，这自然减少了在后续阶段需要改正前面阶段所犯错误的可能性。

3.缺点

（1）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，因此不适合大型系统的开发（适合开发小型的、灵活性高的系统）。

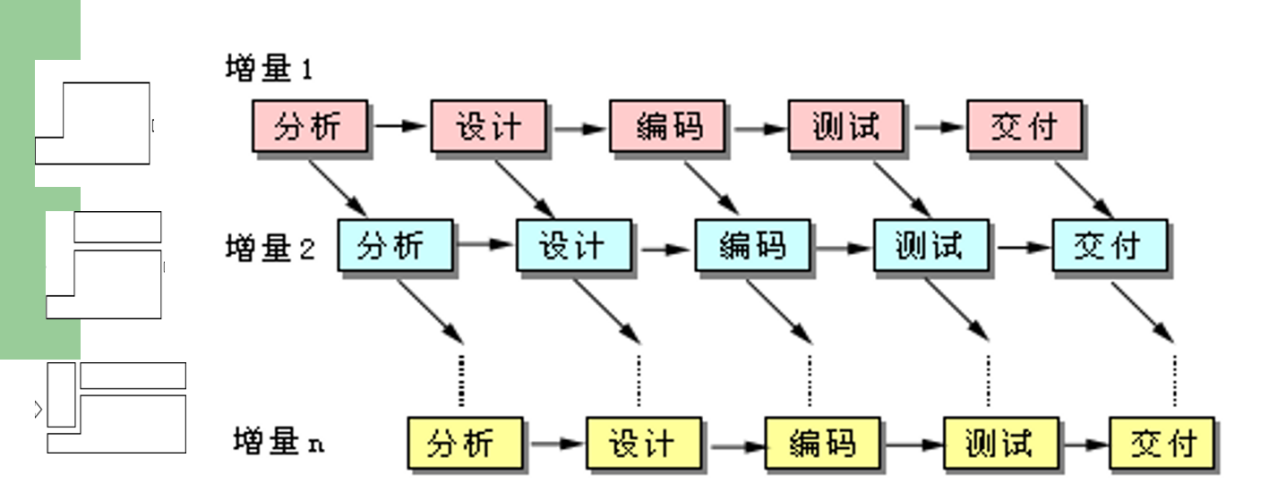
（2）使用这个模型的前提是要有一个展示性的产品原型，因此在一定程度上可能会限制开发人员的创新

（3）所选用的原型（开发技术和工具）不一定符合主流的发展

（4）快速原型模型是不带反馈环的，软件产品的开发基本上是按线性顺序进行的。

1. 增量模型
2. 基本思想

增量模型（Incremental Model）融合了瀑布模型的基本成分（重复应用）和原型实现的迭代特征，该模型采用随着日程时间的进展而交错的线性序列，每一个线性序列产生软件的一个可发布的“增量”。产品被分解为多个组件，每个组件都是单独设计和构建的。各个构件完成后逐渐并入已有的软件体系结构中。



1. 优点

（1）采用增量模型的优点是人员分配灵活，刚开始不用投入大量人力资源。如果核心产品很受欢迎，则可增加人力实现下一个增量。因此，增量能够有计划地管理技术风险。

（2）每次迭代后，应执行回归测试。在此测试期间，可以快速识别软件的故障元素，因为在任何单个迭代中很少进行更改。

（3）测试和调试比其他类型的软件开发方法更容易，因为每次迭代时所做的更改相对较小。这允许对整个产品中的每个元素进行更有针对性的、更严格的测试。

客户可以响应功能并查看产品，以了解任何需要或有用的更改。

（4）增量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

1. 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，所以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

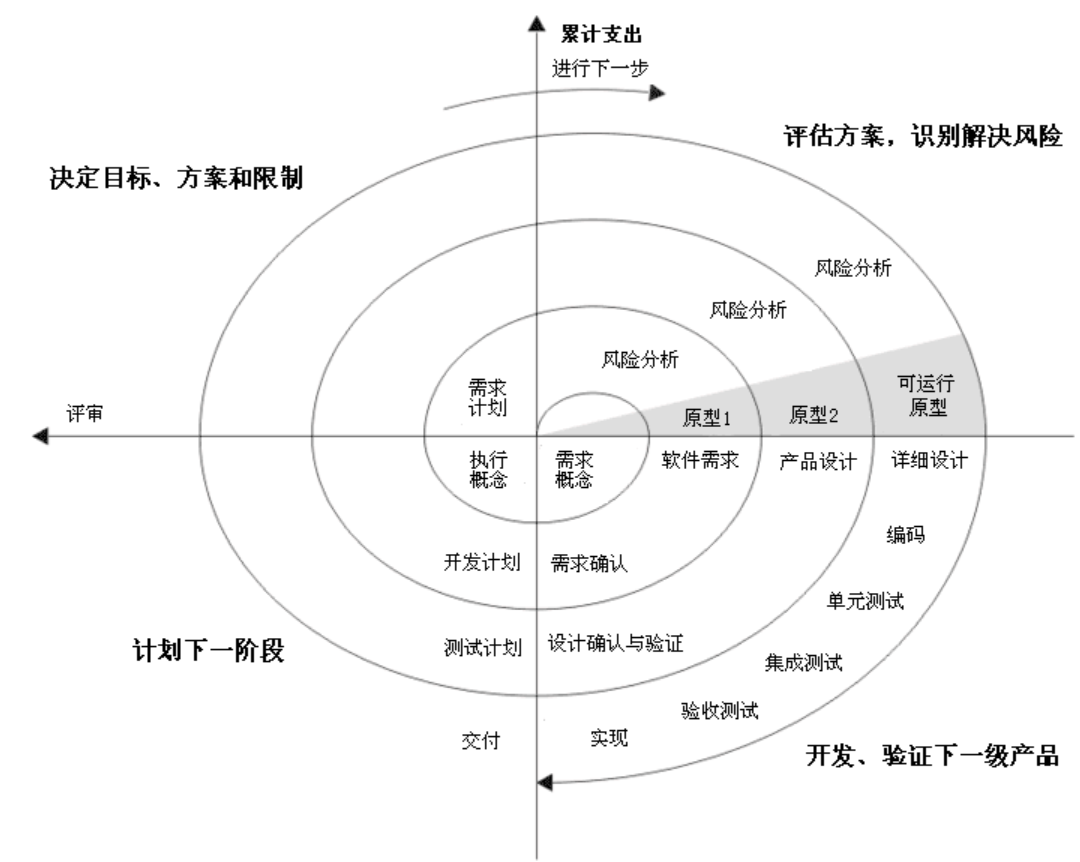
（2）在开发过程中，需求的变化是不可避免的。很容易退化为边做边改模型，从而是软件过程的控制失去整体性。

（3）如果增量包之间存在相交的情况且未很好处理，则必须做全盘系统分析

随着产品增加了其他功能，可能会出现与系统体系结构相关的问题，而早期原型中并不明显

1. 由增量产生的成本可能会超过组织的成本。
2. 螺旋模型
3. 基本思想

螺旋模型（Spiral Model）是巴利·玻姆（Barry Boehm）于 1988 年 5 月在他的文章《一种螺旋式的软件开发与强化模型》提出的。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，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。 

2.优点

（1）通过原型的创建，使软件开发在每个迭代的最初明确方向；

（2）通过风险分析，最大程度地降低软件彻底失败造成损失的可能性；

（3）在每个迭代阶段植入软件测试，使每个阶段的质量得到保证；

（4）整体过程具备很高的灵活性，在开发过程的任何阶段自由应对变化；

（5）每个迭代阶段累计开发成本，使支出状况容易掌握；

（6）通过对用户反馈的采集，与用户沟通，以保证用户需求的最大实现；

3.缺点：

（1）过分依赖风险分析经验与技术，一旦在风险分析过程中出现偏差将造成重大损失；

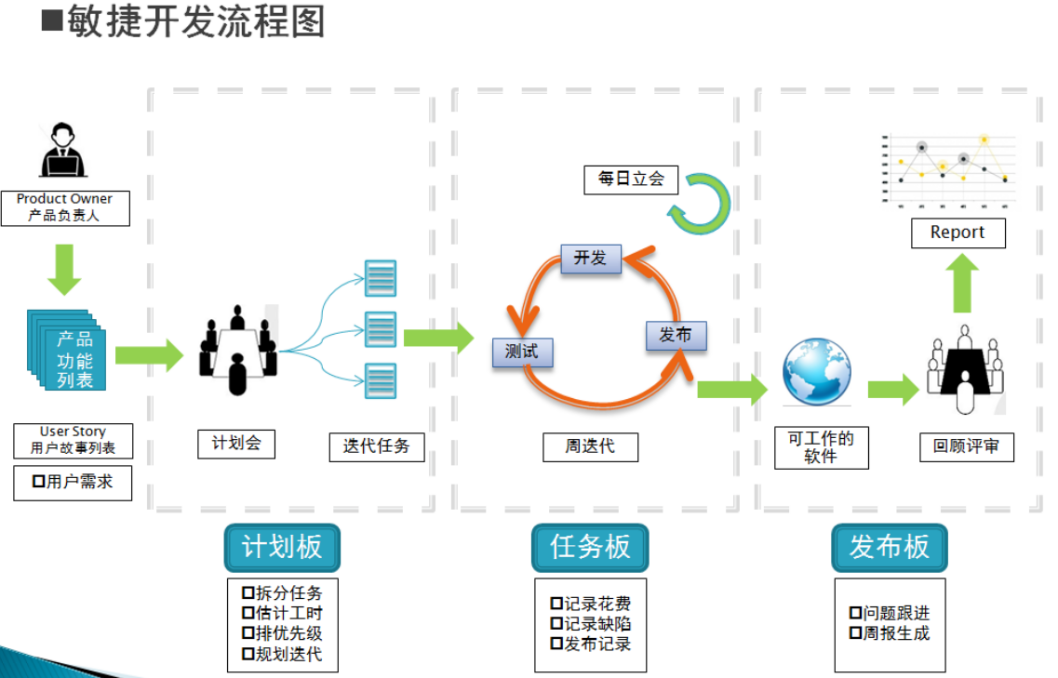
（2）过于灵活的开发过程不利于已经签署合同的客户与开发者之间的协调；

（3）由于只适用大型软件，过大的风险管理支出会影响客户的最终收益；

1. 敏捷开发模型

1.基本思想

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。



敏捷开发都是以迭代为主要方式

2.优点：

（1）采用简单计划策略，不需要长期计划和复杂模型，开发周期短

（2）在全过程采用迭代增量开发、反馈修正和反复测试的方法，能够适应用户经常变化的需求

（3）注重市场快速反应能力，客户前期满意度高

3.缺点：

（1）注重人员的沟通，忽略文档的重要性，若项目人员流动大太，给维护带来不少难度

（2）对编码人员的经验要求高，若项目存在新手比较多时，老员工比较累

**如何将敏捷开发应用于小组的项目中（张勇）**

将敏捷开发应用于外卖系统的软件工程项目中可以在敏捷开发的框架下高效地开发外卖系统软件，并不断地适应变化和提供价值。

1.明确定义项目愿景和范围：

确定外卖系统的目标和核心功能，例如用户注册、浏览菜单、下订单、支付等。

定义项目愿景，明确所要解决的问题以及期望的结果。

2.编制产品待办事项列表：

与利益相关者合作，编制优先级排序的产品待办事项列表（Product Backlog），列出所有需求和功能。

利用产品待办事项列表（Product Backlog）来管理项目的需求和功能。待办事项应该被明确定义、优先级排序，并持续更新以反映最新的需求和优先级。

利用敏捷项目管理工具（如Jira、Trello等）来管理和跟踪待办事项，确保团队和利益相关者对项目的需求有清晰的了解。

3.划分开发周期：

将整个项目划分为若干个固定长度的迭代周期（通常为2至4周），每个迭代周期称为一个迭代或冲刺（Sprint）。在迭代周期内，团队致力于完成迭代计划中确定的任务和目标。

4.迭代规划会议：

在每个迭代开始前召开迭代规划会议，团队和利益相关者一起确定在该迭代期间要完成的任务和目标。任务应该根据优先级排序并估算出工作量。

5.迭代开发：

开发团队在迭代期间完成分配的任务和功能，通常采用每日站立会议（Daily Standup Meeting）来跟踪进度、讨论问题和调整计划。

6.迭代评审会议：

每个迭代结束时召开迭代评审会议，展示和演示已完成的功能，并接受利益相关者的反馈和建议。

7.迭代回顾会议：

在迭代评审会议后，开发团队召开迭代回顾会议，回顾迭代期间的工作，讨论团队绩效、工作流程和改进点。

8.持续集成和交付：

采用持续集成和持续交付的实践，确保团队能够频繁地集成代码并交付可工作的软件产品。自动化测试、部署和发布流程可以帮助提高软件交付的速度和质量。

9.灵活应对变化：

随着项目的推进和利益相关者需求的变化，及时调整产品待办事项列表和迭代计划。

10.持续改进：

不断地通过迭代回顾会议和反馈机制来改进团队的工作流程、沟通方式和技术实践。

11.团队协作和持续改进：

重视团队协作和沟通，建立良好的团队文化和工作氛围。定期举行团队建设活动，促进团队成员之间的合作和信任。持续改进是敏捷开发的核心原则之一，团队应该不断地反思和改进工作流程和实践，以提高团队的效率和质量。

**分析小组项目中可能存在的风险（丁弘扬）**

1.技术风险：

（1）系统稳定性问题：随着用户数量的增加和订单量的增长，可能会出现卡顿，延迟或崩溃的情况，影响用户体验和餐厅的经营状况。

（2）技术和功能更新可能滞后：

（3）如果不能及时更新技术栈，可能会面临性能瓶颈，安全隐患等问题

2.运营风险：

（1）订单处理失误：由于系统流程设计不当或人为操作错误，可能导致订单处理出现延误、漏单或错单等问题，给餐厅和消费者带来损失。

配送问题：外卖系统通常涉及第三方配送服务，配送员的效率、服务态度以及配送过程中的安全问题都可能影响用户体验和餐厅声誉。

（2）数据安全风险：

用户隐私泄露：外卖系统涉及大量用户个人信息，如姓名、地址、电话号码等。如果系统存在安全漏洞或被黑客攻击，可能导致用户隐私泄露。

支付安全：外卖系统通常支持在线支付功能，如果支付环节存在安全漏洞或被恶意攻击，可能导致用户资金损失或产生法律纠纷

**细化风险管理（做出风险分级及应对预案）（刘易）**

1.外卖系统的风险管理

风险分类：

我们将根据风险的潜在影响和发生可能性对其进行分类：

（1）影响：

高： 可能对系统功能、财务或声誉产生重大影响的风险。

中： 可能导致中等程度中断或不便的风险。

低： 对系统影响最小的风险。

（2）可能性：

高： 非常可能发生的风险。

中： 发生可能性中等的风险。

低： 不太可能发生的风险。

2.风险识别和应对预案：

（1）技术风险：

系统稳定性问题：

影响： 高

可能性： 高（随着用户数量和订单量的增加而增加）

缓解措施：

实施可扩展且完善的系统架构。

进行负载测试和性能优化。

监控系统性能并主动解决瓶颈问题。

（2）技术和功能更新滞后：

影响： 中到高

可能性： 中

缓解措施：

制定定期更新计划并优先考虑关键更新。

分配资源用于持续维护和开发。

监控技术趋势并主动采用新解决方案。

（3）运营风险：

订单处理错误：

影响： 中

可能性： 中

缓解措施：

实施带有错误检查的自动化订单处理系统。

对员工进行订单处理程序培训。

与客户和餐厅建立清晰的沟通渠道。

实施订单准确性复核机制。

（4）配送问题：

影响： 中

可能性： 中

缓解措施：

优化送货路线和物流。

实施实时跟踪和通信系统。

与可靠的送货服务提供商合作并持续关注他们的绩效。

为送货人员制定明确的服务标准和培训。

实施保险政策以涵盖送货过程中可能出现的损失。

（5）客户不满意和负面评价：

影响： 中

可能性： 高

缓解措施：

提供优质的客户服务和支持。

实施反馈机制并及时解决客户问题。

关注在线评论并做出专业回应。

（6）数据安全风险：

用户隐私泄露：

影响： 高

可能性： 中

缓解措施：

实施强有力的安全措施，如加密和访问控制。

定期进行安全审计和渗透测试。

对员工进行网络安全最佳实践培训。

实施数据匿名化和最小化技术。

遵守数据隐私法规。

（7）支付安全漏洞：

影响： 高

可能性： 中

缓解措施：

使用安全可靠的支付网关。

实施欺诈检测和预防措施。

遵守支付卡行业标准。

定期审计和更新支付安全系统。

（8）商业风险：

来自其他外卖平台的竞争：

影响： 高

可能性： 高

缓解措施：

通过独特的功能和服务来区分您的平台。

与餐厅和送餐合作伙伴建立牢固的关系。

实施有效的营销和客户获取策略。

（9）法规和法律要求的变化：

影响： 中

可能性： 中

缓解措施：

了解相关法规和法律变化。

实施合规措施并根据需要调整系统。

必要时寻求法律建议。

（10）财务可持续性和盈利能力：

影响： 高

可能性： 中

缓解措施：

制定完善的商业模式和财务计划。

监控关键财务指标并优化运营以提高盈利能力。

确保资金并有效管理现金流。

（11）其他注意事项：

保险：考虑针对数据泄露、送货事故和食品安全问题等潜在风险获取适当的保险。

应急计划：制定针对关键风险的应急计划，以确保在中断情况下业务连续性。

沟通：主动与客户和合作伙伴沟通潜在风险和已采取的缓解措施。

（12）风险监控和审查：

应定期审查和更新此风险管理计划，以反映系统、业务环境和风险状况的变化。实施一个流程来监控已识别的风险并主动识别新风险。

通过实施此风险管理计划，可以最大限度地减少潜在威胁的影响，并确保外卖系统成功运营

**第二部分：阅读Scrum开发方法文档，理解Scrum过程工作模型（王攀）**

Scrum的三个主要元素：角色、活动、产出物

一、三个角色：

1. 产品负责人Product Owner（PO） 定义产品需求，负责产品成败；（需求获取、决策、讲解、验收、编写者）

2.Scrum教练 Scrum Master 协助开发团队依循Scrum的精神来开发软件，确定Scrum所规范的几个活动都有定时且正确进行；

3. 开发团队Team Member 负责软件开发；（写程序、单元测试、做计划、编写技术文档）

二、四个活动：

1. Sprint策划会议

参与人员：产品负责人、Scrum教练、开发团队

过程：上半部分确定sprint目标，挑选 product backlog、确定团队成员名单、确定完成标准，并按照优先级划分；下半部分对sprint backlog分解任务、任务估计、任务分工，确定sprint演示日期，以及每日站立会议的时间和地点。

产出物：Sprint Backlog

2. 每日站立会议

团队每天站成一圈开一个约15分钟的会议，团队成员要报告三件事情：（1）昨天做了哪些事？（2）今天准备做什么？（3）有没有遇到任何问题或阻碍？

会议目的：监督个人承诺、了解项目进展、培养团队文化

会议步骤： （1）功能测试，PO确认任务是否完成； （2）轮流汇报 （3）更新燃尽图

3. Sprint 评审会议

参与人员：开发团队、客户、管理者、产品负责人

团队向产品负责人及利益相关者展示本次sprint中已完成的每一个故事，并且产品负责人确认这些故事有没有做到他想要的程度；调查利益相关者的意见、期望及优先级，这些新的想法可能变成新的需求，移到后续的sprint中实现。

4. 回顾会议

参与人员：Scrum教练、开发团队、产品负责人如果有空，也参加

主要目的：检讨与改善软件开发流程；在会议中开发人员列举出在此sprint中有哪些开发流程是好的，要继续维持；有哪些是不好的或是没做到的，应该要改善的项目。最后团队讨论出改善行动方案，在下一个sprint（或是连续若干个sprint）中实施此改善项目。

三、三个产出物：

1. 产品清单

一个故事就是其中的一条内容。

产品清单 ：用户角色+功能+目的+优先级+验收标准

特点：

独立性： 避免故事之间的依赖性，便于优先级划分；

可协商性：故事是有弹性的，可变化；

可预测性：开发者应该能预测故事的规模及实现所需要的时间；

短小精悍：一般一个故事在一个迭代周期内可实现；

测试性：能定义出验收标准；

2. 冲刺清单

sprint backlog就是某一个sprint准备实施的故事（产品清单的子集合）；其中的故事也要经过排序。

3. 燃尽图