迭代式软件开发

软件开发生命周期（Software Development Life Cycle，SDLC）

- 软件开发生命周期指软件产品从开发到报废的生命周期，通常周期中包括了需求分析、软件设计、实现与调试、测试与验收、部署、维护等活动。
- 软件报废：不需要其功能；很难在合理的成本范围内继续演进开发原有软件。

软件开发过程

- 简称软件过程。
- 软件开发过程是指一个软件产品开发的方法，它描述了软件开发中的活动和任务。简单的说，过程就是软件开发中的一系列活动，如果能够按照这些活动进行工作，我们就可以获得预想的结果。
- “过程决定质量”

- 常见的软件开发过程模型有瀑布、迭代、螺旋模型和敏捷软件开发等，有很多分类的方法。

- 当前有关于软件开发过程的讨论与争议很多，还没有一种大家共同认可的开发过程，但多数人都同意软件过程对软件开发具有重要的指导意义，软件开发不应该是无序的、混乱的。

- Why?

  - 软件项目通常都是很复杂的系统，开发人员需要按照一定的开发模型来分解工作任务，按照一定的步骤进行工作以完成项目的开发。如果没有开发模型的存在，开发人员很容易迷失在复杂系统中，缺乏指引，陷入混乱。

创建-修补（Build-Fix）

- 编码-修补式软件开发被分成编码和修补两个阶段，它没有合理的开发计划，系统的开发由一系列仅能满足当前需要的短期决策构成，开发者往往很随意的做出决定，也不会给出设计的原因。
- 适用于软件规模小、质量要求低。学生作业。
- 问题：大规模、高质量。
  - 当增加新的功能到系统中时，因为系统中充斥了各种临时的局部设计决定，新模块的添加异常艰难。同时，系统缺陷也无处不在，并且难以修补，可能在修复一个缺陷时又带来新的缺陷。

瀑布模型

- 描述了软件开发的基本框架。
- 多数人已经认识到瀑布模型的缺陷，但瀑布模型仍然可以帮助我们认识软件开发活动，并对其提供指导。
- 瀑布模型（Waterfall Model）是由W.W.Royce在1970年最初提出的，它按时间顺序描述了一个软件项目的开发。

- 瀑布模型核心思想是根据开发活动来分解项目。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。

- 瀑布模型设计了一系列软件开发阶段，按时间顺序展开开发活动，从需求分析开始直到产品发布和维护，每个阶段都会产生循环反馈。如果在某个阶段出现了问题，那么最好“返回”上一个阶段并进行修改。项目开发过程像是从一个阶梯“流动”到下一个阶梯，这也是瀑布模型名称的由来。

- 优点：帮助开发人员理解软件开发中的活动和任务，界定清晰的软件开发检查点。

- 问题I：在实践中很少有项目能够以纯线性的方式进行，通过回到前面的阶段或修改前面某个阶段的结果是非常常见的现象，而这时带来的成本增加和系统开发的混乱是很难避免的。

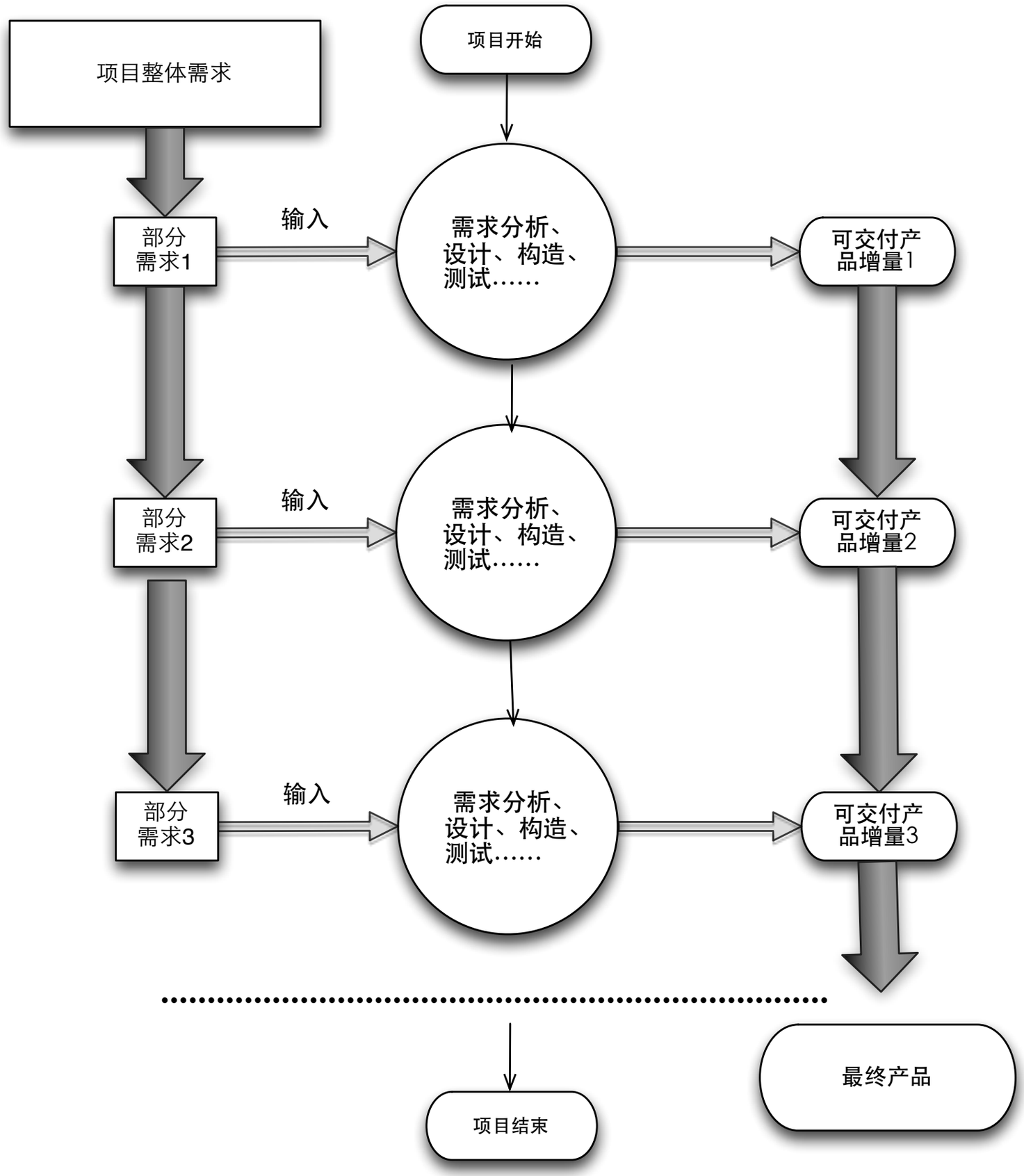
- 问题2：在瀑布模型中，开发人员很难判断其前期的工作是否正确。计划、需求分析和设计中的错误往往要到产生了测试、集成时才能够发现，而在前期，我们没有足够的信息来进行这种判断。

- 按照软件工程中的贝姆定律（Boehm's Law），在开发过程中越晚修正缺陷，代价就会越高。

- 问题3:只有到项目开发的后期才能看到能够运行的软件，这使得开发团队很难在早期和用户就需求进行验证和讨论，增加了需求误解的可能性，同时也对团队的开发士气造成了不好的影响。

迭代式软件开发

- 迭代式软件开发根据软件项目的不同功能子集来分解项目。
- 在迭代式软件开发中，整个开发工作被组织为一系列小的项目，被称为一系列的迭代。
- 每个迭代周期结束时都应该得到一个经过测试的，集成起来的基本可用的软件产品，某些少量程序缺陷的修复或用户培训可以放在最后一个迭代周期后进行。


项目整体需求
项目开始
部分
需求1
输入
需求分析、
设计、构造、
测试……
可交付产
品增量1
部分
需求2
输入
需求分析、
设计、构造、
测试……
可交付产
品增量2
部分
需求3
输入
需求分析、
设计、构造、
测试……
可交付产
品增量3
项目结束
最终产品

- 特点I.易于应对需求变更
  - 迭代式软件开发可以方便的在每一个迭代结束时修改原有的需求，以应对相应的变更。
  - 用户在看到一个迭代的结果后，会重新审视原来提出的需求，修正原有的需求或增加新的需求。

- 特点2.　提高团队士气
  - 开发人员的士气和对项目的热情部分决定了软件项目的成败。
  - 开发人员通过每次迭代都可以在短期内看到自己的工作成果，有助于他们增强信心，更好地完成开发任务。

真的是迭代式开发?

- “我们正在进行一次分析迭代，接下来会有两次设计迭代。”
- “这次迭代的代码中有很多缺陷，我们在下次迭代时修复它们。”
- 有一个判定标准就是，每一次迭代结束时，系统中的代码需要经过测试，正确的集成起来，并且达到基本可交付的产品级品质。

Why

- 测试与集成都是非常难于估算的开发活动，不能把这样的活动放到迭代开发的最后进行，否则就无法得到迭代式开发的益处。
- 每次迭代的结果是开发团队在本迭代内工作真实有效的反馈，它有助于开发团队及时调整开发计划和软件开发实践。
- 测试和集成工作应该做到：即使本次迭代的软件不进行发布，那么如果要进行发布的话，也不需要大量的工作。

开发长度

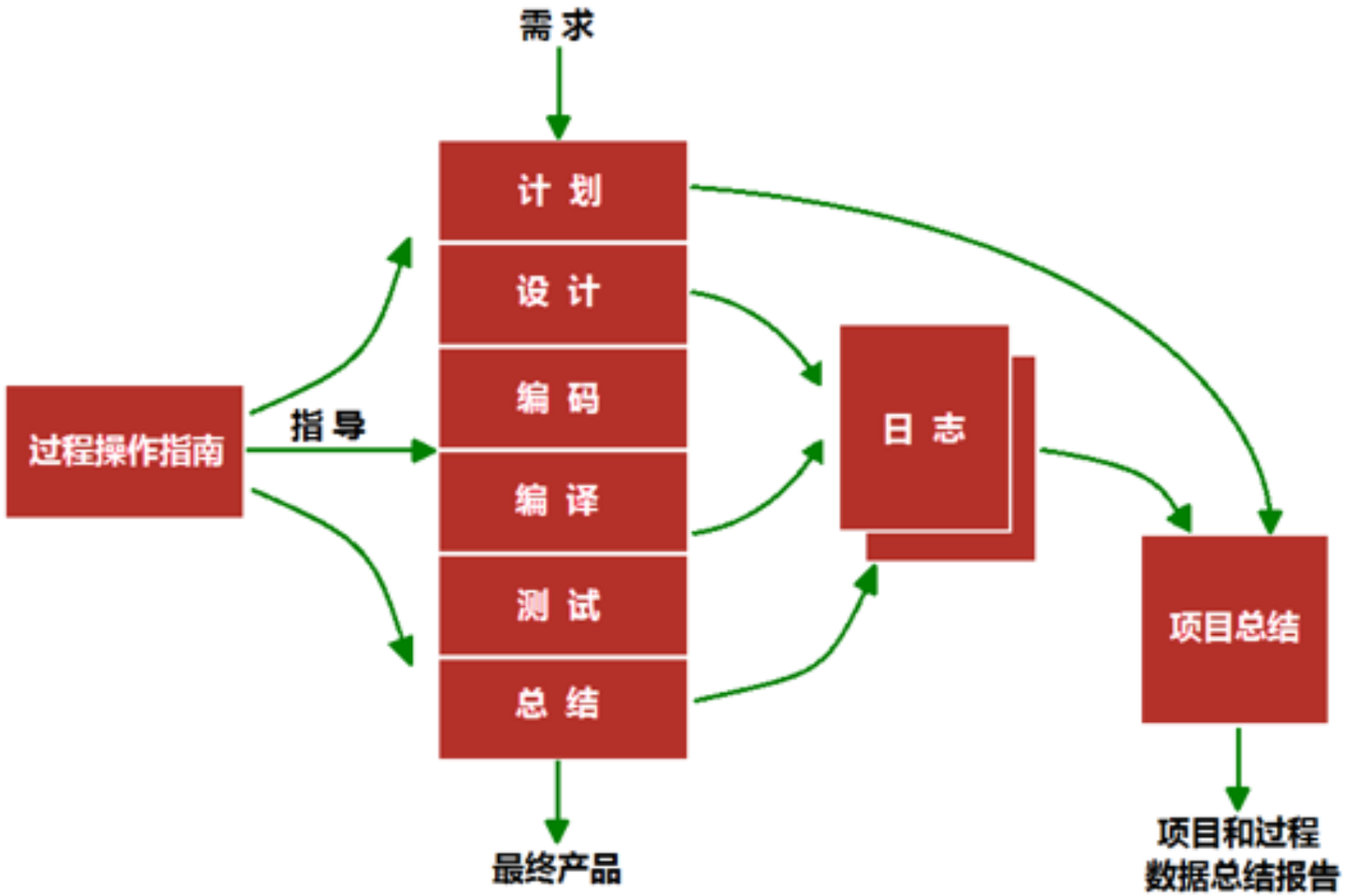
- 增加迭代时间来完成既定的项目功能。
- 固定时间长度（time boxing）的迭代周期：工作节奏、功能优先级。
- 敏捷软件开发都是迭代式开发过程。

个人软件过程

- 个人软件过程（Personal Software Process，PSP）由美国卡内基梅隆大学软件工程研究所（Software Engineering Institute，SEI）的Humphrey等开发，于1995年推出，着重于软件开发人员的个人培训，品质改善和工期估算。
-

- PSP能够提供：
  - 1、个体软件过程原则；
  - 2、软件开发工程师如何制定准确的计划；
  - 3、软件工程师为改善产品质量需要采取的措施；
  - 4、度量个体软件过程的改进；
  - 5、流程的改变对软件工程师个人能力的影响。

- PSP是包括了数据记录表格、过程操作指南和规程在内的结构化框架。



PSP原则

- “过程质量决定最终产品质量”
  - 软件系统的整体质量由该系统中质量最差的某些组件所决定。
  - 软件组件的质量取决于开发这些组件的软件工程师，更加确切的说，是由这些工程师所使用的开发过程所决定。
  - 作为合格的软件工程师，应当自己度量、跟踪自己的工作，自己管理软件组件的质量。
  - 作为合格的软件工程师，应当从自己开发过程的偏差中学习、总结，并将这些经验教训整合到自己的开发实践中，也就是说，应当建立持续地自我改进机制。

时间度量

- 一条完整的时间日志记录应该包含以下基本信息：编号、所属阶段、开始时间、结束时间、中断时间、净时间以及备注。
- “理想工作时间”，flow

缺陷度量

- 一个典型的缺陷日志包含以下几项内容：编号、发现日期、缺陷类型、注入阶段、消除阶段、消除时间、关联缺陷以及缺陷描述。

PSP缺陷类型描述

编号	缺陷类型	描述
1	Documentation	注释、提示消息错误等；
2	Syntax	拼写、标点、打印、指令格式错误等；
3	Build，Package	变更管理、库以及版本控制方面的错误等；
4	Assignment	变量的申明、重命名、域以及限制方面的错误；
5	Interface	过程调用接口、输入输出、用户接口方面的错误；
6	Checking	错误信息、不充分检验等方面的错误；
7	Data	数据结构与内容方面的错误；
8	Function	逻辑、指针、循环、递归、计算以及功能性错误；
9	System	配置、计时、内存方面的错误；
10	Environment	设计、编译、测试或者其它支撑环境的问题引发的错误。