单选题 1分

瀑布模型是（）。

A 适用于需求被清晰定义的情况

B 一种需要快速构造可运行程序的好方法

C 一种不适用于商业产品的创新模型

D 目前业界最流行的过程模型

南開大學 Nankai University

单选题 1分

增量模型是（）。

A 适用于需求被清晰定义的情况

B 一种需要快速构造可运行程序的好方法

C 一种不适用于商业产品的创新模型

D 已不能用于现代环境的过时模型

南開大學 Nankai University

单选题 1分

原型化模型是（）。

A 适用于需求被清晰定义的情况

B 适用于客户需求难以清楚定义的情况

C 提供一个精确表述的形式化规格说明

D 很难产生有意义产品的一种冒险模型

南開大學 Nankai University

主观题 10分

案例分析

实例一：汽车制动防抱系统

- 汽车制动防抱系统（Anti-locked Braking System，简称ABS）是一种具有防滑、 防锁死等优点的汽车安全控制系统。
- 该系统通过汽车微电脑控制，以非常快的速度精密地控制制动液压力的收放，从而 达到防止车轮抱死，确保轮胎的最大制动力以及制动过程中的转向能力，使车辆在紧急制动时具有躲避障碍的能力。

思考：这种ABS软件系统的开发，具有怎样的特点呢？

南開大學 Nankai University

单选题 1分

以下哪一种模型适合这类系统的开发?

A 瀑布模型

B 原型化模型

C 迭代式模型

D 可转换模型

南開大學 Nankai University

该系统是一个嵌入式系统，且对安全性和可靠性的要求极高。

- 一方面，对于可转换模型来说，由于数学方法具有严密性和准确性，其所交付的系统具有较少的缺陷和较高的安全性。特别适合于那些对安全性、可靠性和保密性要求极高的 软件系统，这些系统需要在投入运行前进行验证。
- 另一方面，嵌入式系统的状态数量通常有限，便于建立形式化的数学描述。

主观题 10分

案例分析

实例二：网络公开课程网站

- 某公司准备开发一个大规模在线公开课程网站，支持学校将自己的课程录像、课件 及参考资料等公布在网上，学生可以进行自主学习。
- 该系统将教育、娱乐和社交网络结合在一起，创造了一种新型的网络教育模式，对 传统的高等教育模式带了很大的冲击。

思考：这种网络公开课程网站的开发，具有怎样的特点呢？

南開大學 Nankai University

单选题 1分

以下哪一种模型适合这类系统的开发?

A 瀑布模型

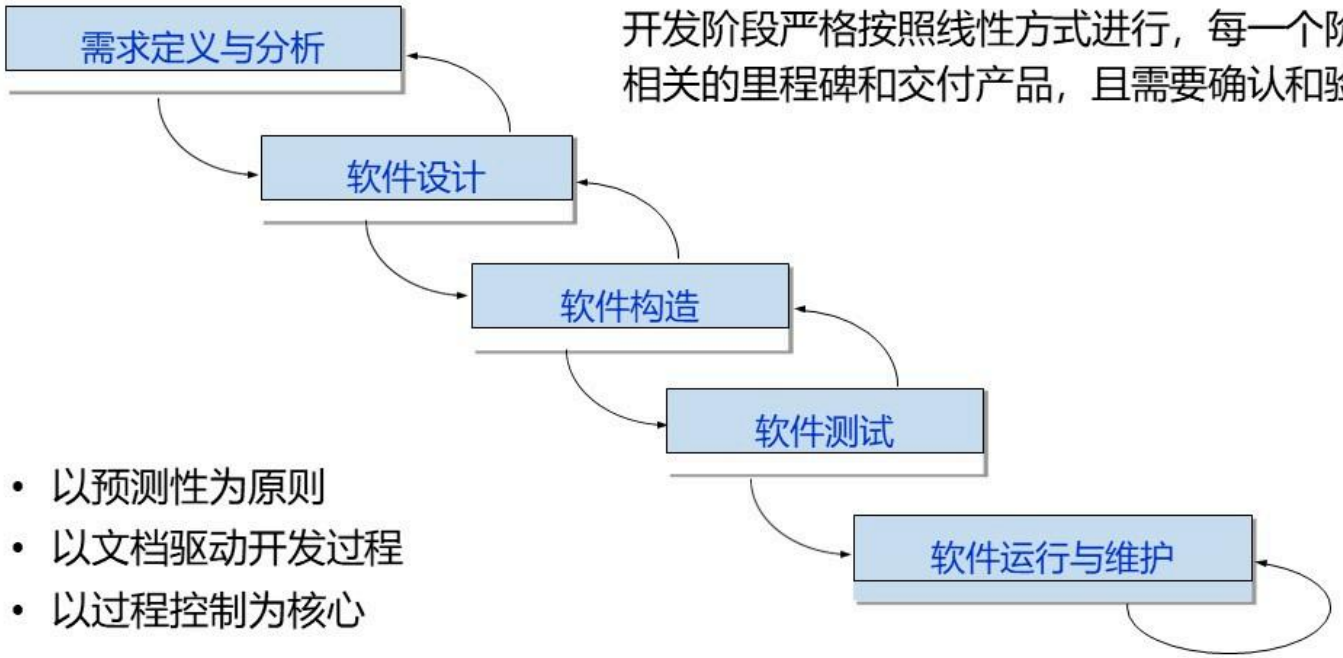
B 原型化模型

C 迭代式模型

D 可转换模型

南開大學 Nankai University

传统的软件开发模式



开发阶段严格按照线性方式进行，每一个阶段具有相关的里程碑和交付产品，且需要确认和验证。

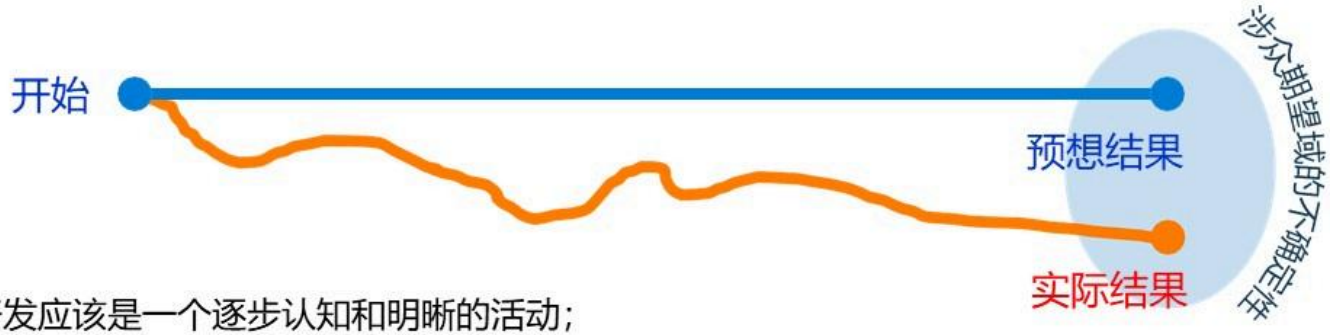
- 以预测性为原则
- 以文档驱动开发过程
- 以过程控制为核心

南开大学 Nankai University

软件开发之道

软件开发是否可以实现一个完整、详尽的计划？软件项目能否预先考虑到所有的风险？

软件项目中难以预知所有的内容和风险！！！



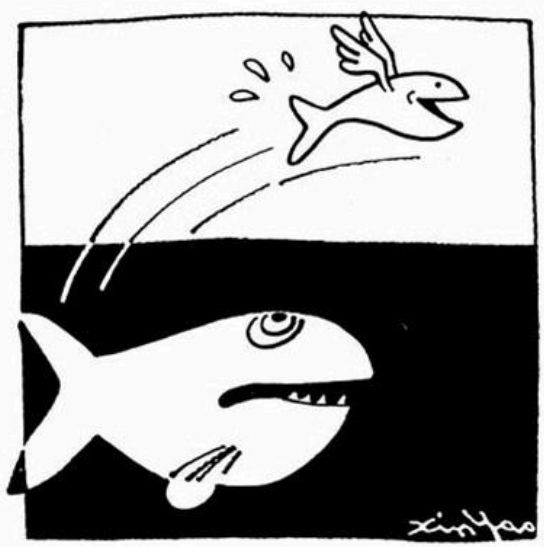
软件开发应该是一个逐步认知和明晰的活动；
软件开发中的变化是实际存在的

南开大学 Nankai University

互联网时代的软件开发

互联网产品的开发特点

- 快鱼吃慢鱼
- 版本发布成本很低
- 追求创新
- 需要快速响应用户的变化
- 需求不确定性高
- 关注用户行为



如今不是大鱼吃小鱼，而是快鱼吃慢鱼。

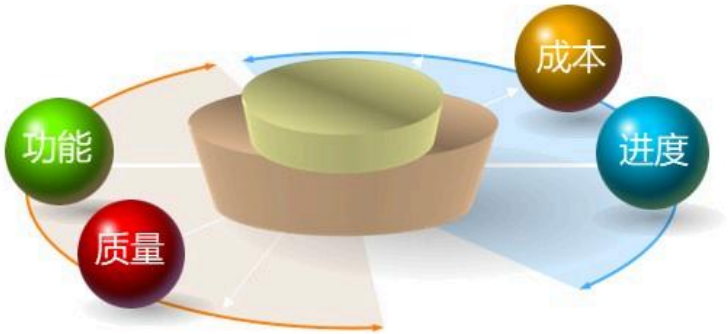
南開大學 Nankai University

轻量级软件过程模型--敏捷开发方法

WIKIPEDIA
The Free Encyclopedia

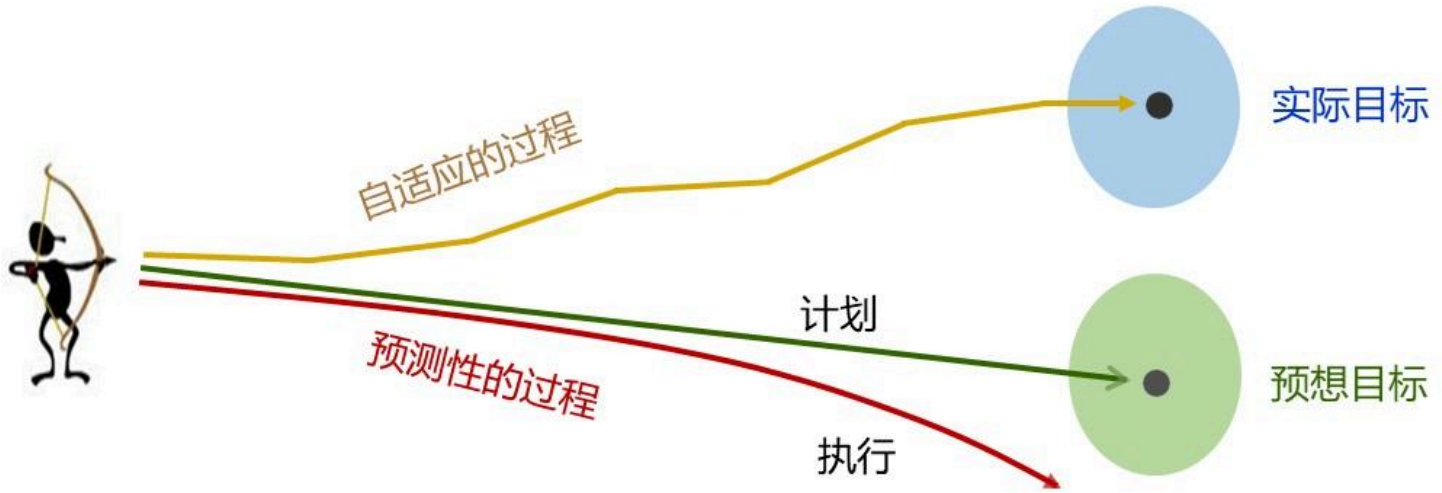
敏捷开发是一种基于更紧密的团队协作、能够有效应对快速变化需求、快速交付高质量软件的迭代和增量的新型软件开发方法。

- 更关注协作
- 更关注质量
- 更关注可工作的产品
- 更关注全才化的专才
- 基于实践而非基于理论



南開大學 Nankai University

敏捷方法：适应而非预测



- 需求是不可预测的
- 软件开发应是一个自适应的跟踪过程

南開大學 Nankai University

敏捷宣言

南開大學 Nankai University

敏捷宣言

我们正在通过亲身实践以及帮助他人实践，揭示更好的软件开发方法。通过这项工作，我们认为：

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

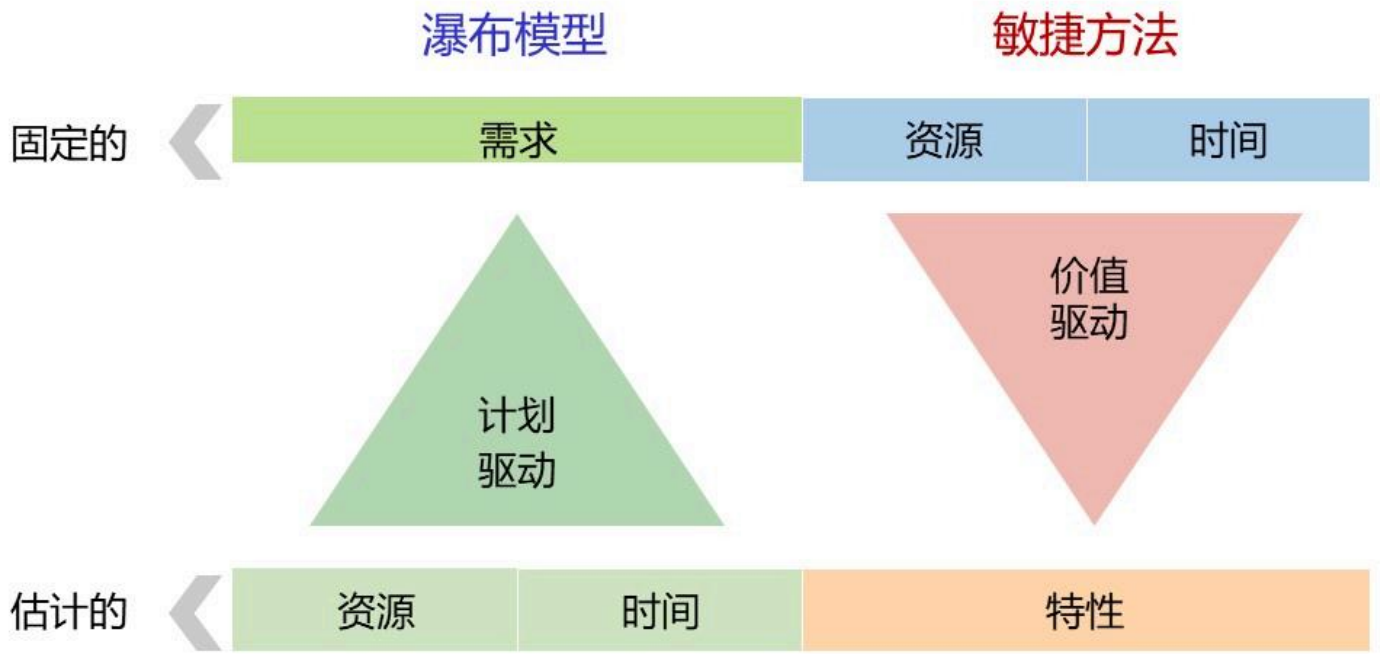
虽然右项也具有价值，但我们认为左项具有更大的价值。

敏捷宣言

- 我们的最高目标是，通过尽早和持续地交付有价值的软件来满足客户。
- 欢迎对需求提出变更——即使是在项目开发后期。要善于利用需求变更，帮助客户获得竞争优势。
- 要不断交付可用的软件，周期从几周到几个月不等，且越短越好。
- 项目过程中，业务人员与开发人员必须在一起工作。
- 要善于激励项目人员，给他们以所需要的环境和支持，并相信他们能够完成任务。
- 无论是团队内还是团队间，最有效的沟通方法是面对面的交谈。
- 可用的软件是衡量进度的主要指标。
- 敏捷过程提倡可持续的开发速度，项目方、开发人员和用户应该能够保持恒久稳定的进展速度。
- 坚持不懈地追求技术卓越和良好设计，这将提升敏捷能力。
- 要做到简单，即尽最大可能减少不必要的工作，这是一门艺术。
- 最佳的架构、需求和设计出自于自组织的团队。
- 团队要定期反省如何能够做到更有效，并相应地调整团队的行为。

南開大學 Nankai University

传统开发 vs. 敏捷开发

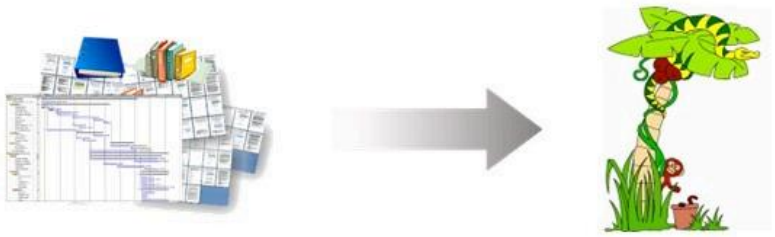


南開大學 Nankai University

传统开发 vs. 敏捷开发

好的架构（产品）是长出来的，而不是设计出来的

传统开发方法



敏捷开发方法

南開大學 Nankai University

敏捷开发方法

敏捷开发方法是一组轻量级开发方法的总称，包含很多具体的开发过程和方法，最有影响的两个方法是极限编程（XP）和Scrum开发方法。

南開大學 Nankai University

敏捷开发方法

Scrum偏重项目管理　　XP偏重编程实践

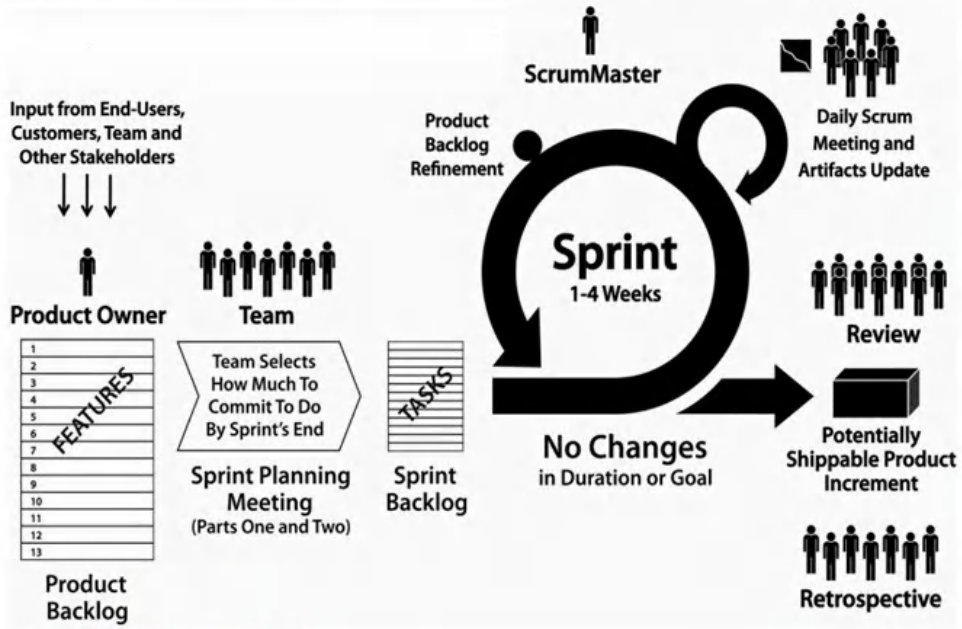
南開大學 Nankai University

Scrum方法

Scrum方法是1995年由Ken Schwaber和Jeff Sutherland博士共同提出，已被众多 软件企业广泛使用，如Yahoo, Microsoft, Google, Motorola, SAP, IBM 等。
这个单词的英文含义是橄榄球运动中的一个专业术语，表示争球的动作。

Scrum框架

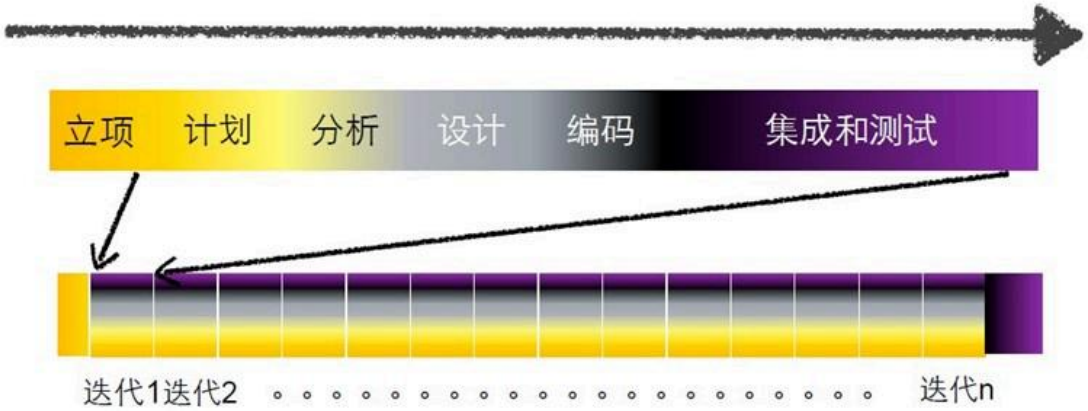
Scrum是一种**兼具计划性和灵活性**的开发过程。它把整个开发过程划分为若干个更小的迭代，每个迭代周期成为一个冲刺



- 首先，产品经理根据用户需求和市场需求，提出一个**按照商业价值进行排序**的客户需求列表，也就是**产品订单**；
- 在每一个迭代的开始，迭代规划会议要从这些产品订单中挑选一些**优先级最高的订单**，形成**一个迭代冲刺任务**；
- 一般一个冲刺其周期是**两到四周**；
- 在迭代进行中会进行**每日战略会议**，检查每日进展情况；
- 在一个迭代结束时，就会产生一个**可运行的交付版本**，由项目的多方人员决定这个版本是否达到了发布的要求。

Scrum迭代开发

迭代开发将整个软件生命周期分成多个小的迭代（一般2~4周），每一次迭代就是一个小的瀑布模型，包括需求分析、设计、实现和测试等活动，结束时都要生成一个稳定和被验证过的软件版本。



南開大學 Nankai University

Scrum迭代开发

迭代开发的关键要点：

- 每一次迭代都建立在稳定的质量基础上，并做为下一轮迭代的基线，整个系统的功能随着迭代稳定地增长和不断完善。
- 每次迭代要邀请用户代表验收，提供需求是否满足的反馈。
- 在一次迭代中，一旦团队作出承诺，就不允许变更交付件和交付日期；如果发生重大变化，产品负责人可以中止当次迭代。
- 在迭代中可能会出现“分解”和“澄清”，但是不允许添加新工作或者对现有的工作进行“实质变更”。
- 对于“分解”和“澄清”，如果存在争议，那么将其认定为变更，放到产品订单中下一次迭代再考虑。

敏捷开发应用

- ISO 9000（09版）标准将在原来八大原则的基础上新增敏捷原则
- 2000年美国军方软件开发标准（DOD 5000.2）推荐迭代为软件开发优选模式
- 2013年发布的新版PMBOK增加迭代及增量生命周期（即对应敏捷模型）

南開大學 Nankai University

单选题 1分

下面的（ ）不是敏捷开发方法的特点。

A 软件开发应该遵循严格受控的过程和详细的项目规划

B 客户应该和开发团队在一起密切地工作

C 过高度迭代和增量式的软件开发过程响应变化

D 通过频繁地提供可以工作的软件来搜集人们对产品的反馈

南开大学 Nankai University

结对编程
Pair Programming

Xu Sihan （徐思涵）
College of Computer Science
Nankai University

Slides adapted from materials by Prof. Qiang Liu (Tsing Hua University), Xin Zou (Beihang University) and Ivan Marsic (Rugers University)

南開大學 Nankai University

保证高质量代码—代码复审

- **复审前必须成功编译，并且经过测试**
- **代码风格规范**
  - 简明，易读，无二义性
  - 缩进，行宽，括号，换行，{}，命名，下划线，大小写，注释
- **流行的代码风格规范如下：**
  1. **PEP8**（Python Enhancement Proposal 8）：适用于Python编程语言的官方代码风格指南。
  2. **Google Java Style Guide**：谷歌推荐的Java代码风格指南。
  3. **Google JavaScript Style Guide**：谷歌推荐的JavaScript代码风格指南。
  4. **Airbnb JavaScript Style Guide**：由Airbnb提供的JavaScript代码风格指南。
  5. **Microsoft .NET Framework Design Guidelines**：微软推荐的.NET框架设计指南（包括C#、VB.NET等语言）。
  6. **Ruby Style Guide**：适用于Ruby编程语言的代码风格指南。
  7. **AngularJS Style Guide**：适用于AngularJS框架的代码风格指南。
  8. **WordPress PHP Coding Standards**：WordPress项目的PHP编码规范。
  9. **Facebook CSS Style Guide**：Facebook推荐的CSS代码风格指南。

保证高质量代码—代码复审

- **复审时要看：**
  - 代码规范
  - 是否每个分支都能执行到
  - 申请的资源是否在所有路径下都能被正确释放
  - 多线程调用保证线程安全，有对共享资源的保护
  - 修改后是否引入新的错误
  - 有么有其它类似的地方需要修改
  - 导致问题的根本原因是什么，如何避免
- **复审后：**
  - 更正明显错误
  - 无法快速更正的错误，创建Bug记录下来
  - 加标记todo, review, bug + 人名

代码复审核查表（一）

- **概要部分**
  - 1） 代码符合需求和规格说明么？ 2） 代码设计是否考虑周全?
  - 3） 代码可读性如何？ 4） 代码容易维护么？5） 代码的每一行都执行并检查过了吗?
- **设计规范部分**
  - 1） 设计是否遵从已知的设计模式或项目中常用的模式?
  - 2） 有没有硬编码或字符串/ 数字等存在?
  - 3） 代码有没有依赖于某一平台，是否会影响将来的移植（如Win32到 Win64）？
  - 4） 开发者新写的代码能否用已有的Library/SDK/Framework中的功能实现？在本项目中 是否存在类似的功能可以调用而不用全部重新实现?
  - 5） 有没有无用的代码可以清除?

南開大學 Nankai University

代码复审核查表（二）

具体代码部分

1） 有没有对错误进行处理？对于调用的外部函数，是否检查了返回值或处理了异常？

2） 参数传递有无错误，字符串的长度是字节的长度还是字符（可能是单/双字节）的长度， 是以0 开始计数还是以1 开始计数？

3） 边界条件是如何处理的？switch语句的default分支是如何处理的？循环有没有可能出 现死循环？

4） 有没有使用断言（Assert）来保证我们认为不变的条件真的得到满足？

5） 对资源的利用，是在哪里申请，在哪里释放的？有无可能存在资源泄漏（内存、文件、 各种GUI资源、数据库访问的连接，等等）？ 有没有优化的空间？

6） 数据结构中有没有用不到的元素？

南開大學 Nankai University

代码复审核查表（三）

- **效能**
  - 1） 代码的效能（Performance）如何？最坏的情况是怎样的？
  - 2） 代码中，特别是循环中是否有明显可优化的部分（C++中反复创建类，C#中 string 的操作是否能用StringBuilder来优化）？
  - 3） 对于系统和网络的调用是否会超时？如何处理？
- **可读性**
  - 代码可读性如何？有没有足够的注释？
- **可测试性**
  - 代码是否需要更新或创建新的单元测试？针对特定领域的开发（如数据库、网页、多线程 等），可以整理专门的核查表。

当你写的时候，就有人复审——Pair Programming

- **审查内容：**一致性、可读性、模块化、可测试性、异常处理、性能、安全性、版本控制、文档、持续集成和持续部署
- **越早发现问题越好：**代码复审越早越好
- **极限编程**：为什么不把一些卓有成效的开发方法用到极致让我们无时不刻地使用它们?
- **写的时候就有人进行代码复审：**结对编程

南開大學 Nankai University

结对编程 Pair Programming

驾驶员/领航员，两人共享一个键盘，电脑，屏幕

- 驾驶员：写设计文档，进行编码和单元测试等XP开发流程。
- 领航员：审阅驾驶员的文档；监督驾驶员对编码等开发流程的执行；考虑单元测试的覆盖率；思考是否需要和如何重构；帮助驾驶员解决具体的技术问题。领航员也可以设计 TDD 中的测试用例。
- 驾驶员和领航员不断轮换角色，不要连续工作超过一小时，每工作一小时休息15分钟。 领航员要控制时间。
- 主动参与。任何一个任务都首先是两个人的责任，也是所有人的责任。
- 只有水平上的差距，没有级别上的差异。两人结对，尽管可能大家的级别资历不同，但不 管在分析、设计或编码上，双方都拥有平等的决策权利。
- 设置好结对编程的环境，座位、显示器、桌面等都要能允许两个人舒适地讨论和工作。如果是通过远程结对编程，那么网络、语音通讯和屏幕共享程序要设置好。

南開大學 Nankai University

最合适的场景

- 降低容易犯的错误
- 新手 + 新手， 或者双方各有明显弱点
- 探索一个新的领域
- 传播知识和技能
  - 老手 + 新手 也可以
- 工具：
  - 排排坐，一个电脑
  - VS Live Share

南開大學 Nankai University

不适合的场景

- 需要深入地研究的项目，需要一个人长时间的独立钻研
- 在做后期维护的时候，如果维护的技术含量不高，只需要做有效的复审即可
- 如果验证测试需要运行很长时间，那么两个人在那里等待结果是有点浪费时间。
- 如果团队的人员要在多个项目中工作，不能充分保证足够的结对编程时间，那么成员要经 常处于等待的状态，反而影响效率。
- 关键是如何最**大限度地发挥“领航员”的作用**，如果用处不大，也就无需结对。

南開大學 Nankai University