

软 件 工 程



西南交通大学电气工程学院

1.1 软件危机

1.2 软件工程

1.3 软件生命周期

1.4 软件过程

各类过程模型的适用场合

- **瀑布模型的适用场合**：系统需求明确且稳定、技术成熟、工程管理较严格的场合，如军工、航天、医疗。
- **原型模型的适用场合**：客户不清楚系统的具体输入输出，开发者不确定算法效率、计算机交互的方式等。
- **增量模型的适用场合**：软件开发中需求可能发生变化、具有较大风险、或者希望尽早进入市场的项目。
- **Rational模型**：适合大项目大团队。
- **敏捷模型**：适用于需求模糊且经常改变的情况，适合商业竞争环境下的项目。

如何选择软件过程模型

- ❑ 软件过程模型是不断发展的
- ❑ 各种软件过程模型各有优缺点和适用场合
- ❑ 不同软件往往需要不同软件过程模型
- ❑ 选用时不必拘泥于某种模型
- ❑ 可组合多种模型
- ❑ 可根据实际创造新的模型

1、瀑布模型也称为经典生命周期模型（ ）

- ☐ A 是增量的模型
- ☐ B 是迭代的模型
- ☐ C 强调设计的模型
- ☒ D 是顺序的模型

提交

2、统一过程（RUP）（ ）

- ☒ A 是迭代和增量式模型
- ☐ B 适用于所有软件工程
- ☐ C 不追求架构的稳定
- ☐ D 是不强调文档的

提交

3、快速原型模型的主要特点之一是（ ）

- ☐ A 开发完毕才见到产品
- ☐ B 开发完毕后才见到工作软件
- ☐ C 及早提供全部完整的软件产品
- ☒ D 及早提供工作软件

提交

4、开发软件所需高成本和产品的低质量之间有着尖锐的矛盾，这种现象称做（ ）。

- ☐ A 软件工程
- ☒ B 软件危机
- ☐ C 软件周期
- ☐ D 软件产生

提交

5、以下**不属于**软件特点的是（ ）

- ☐ A 软件具有复杂性
- ☒ B 软件产品存在磨损问题
- ☐ C 软件具有不可见性
- ☐ D 软件费用不断增加

提交

6、下面**不是**软件工程3个要素的是（ ）

- ☐ A 过程
- ☐ B 方法
- ☒ C 环境
- ☐ D 工具

提交

□ 软件开发中经常出现的问题：

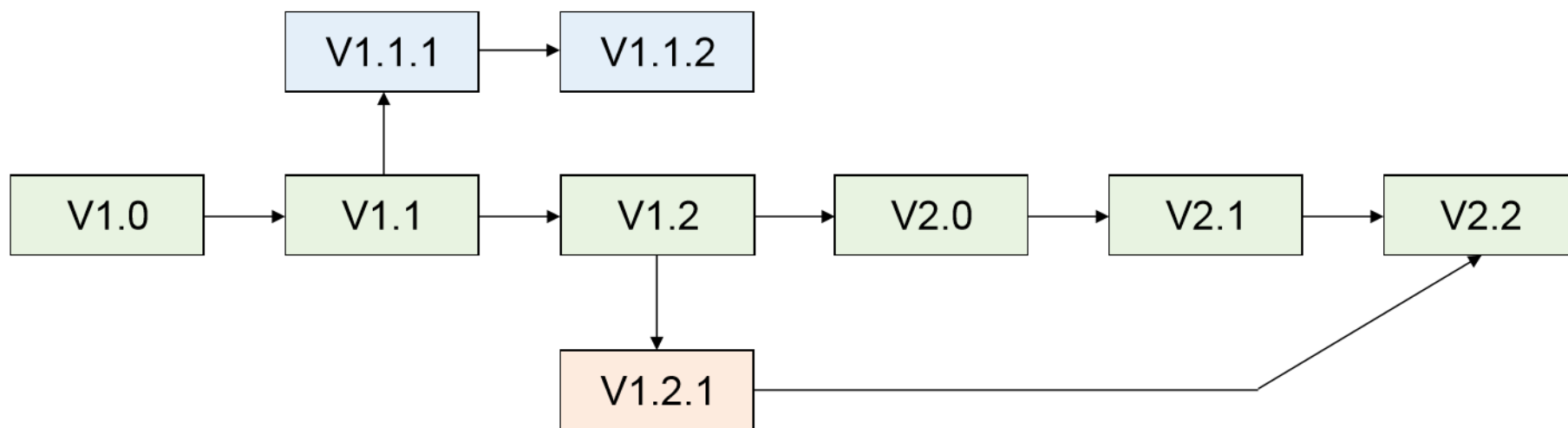
- 找不到某个文件的历史版本
- 开发人员使用错误的版本修改程序
- 开发人员未经授权修改代码或文档
- 人员流动，交接工作不彻底
- 无法重新编译某个历史版本
- 因为协同开发或异地开发，版本变更混乱

软件配置管理

- **软件配置管理：**一种标识、组织和控制修改的技术，它作用于整个软件生命周期，保证所有配置项的完整性、一致性和可跟踪性。
- **软件配置项：**配置管理的对象，软件工程过程产生的所有信息项。主要包括：文档数据、源代码、目标代码、可执行代码、数据库、测试用例、相关产品（如开发工具等）。

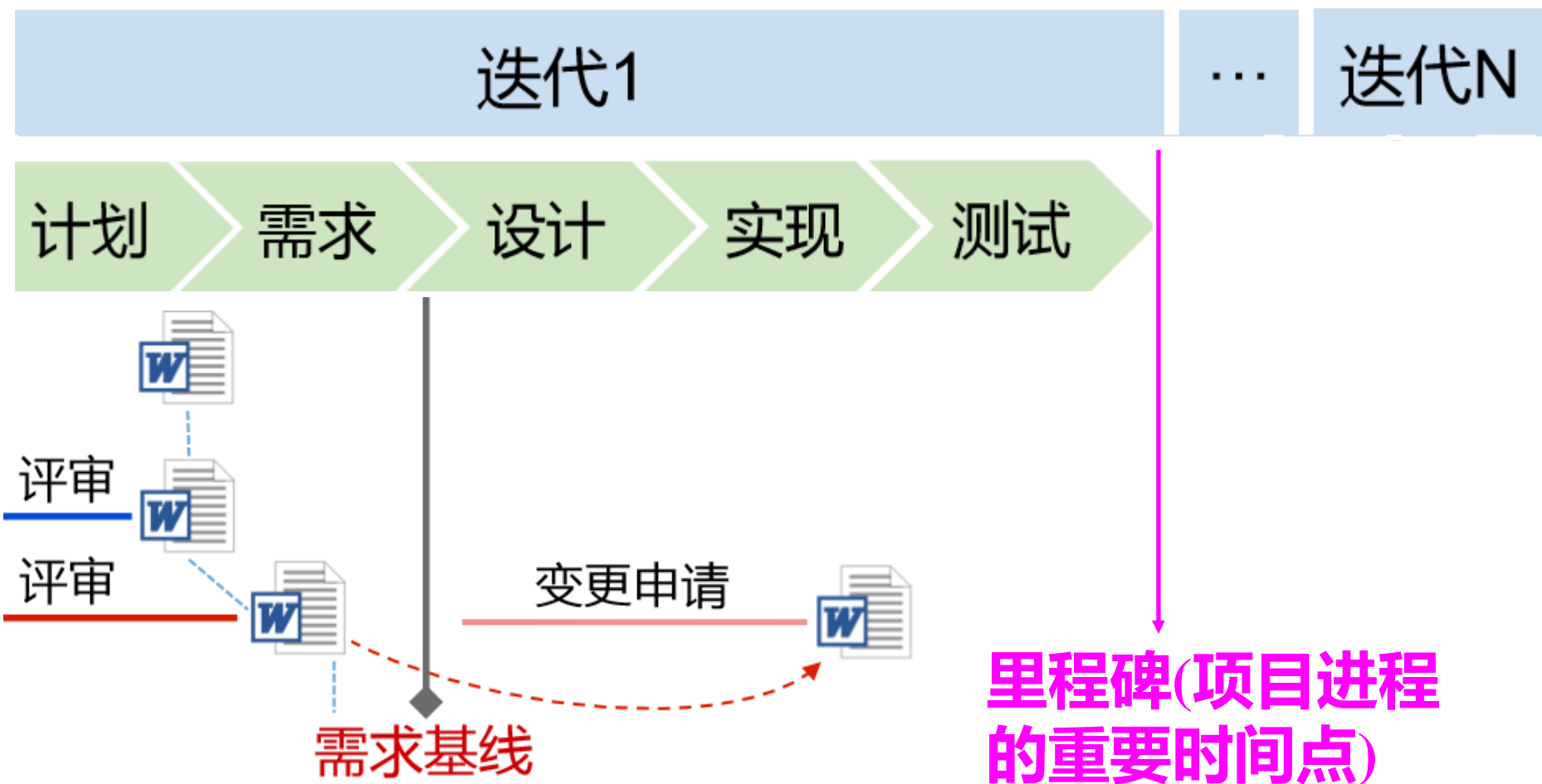
版本

- **版本：**明确定义的时间点上某个配置项的状态。
- **版本管理：**是软件配置管理的核心内容，是对系统不同的版本进行标识和跟踪的过程，从而保证软件技术状态的一致性。



基线

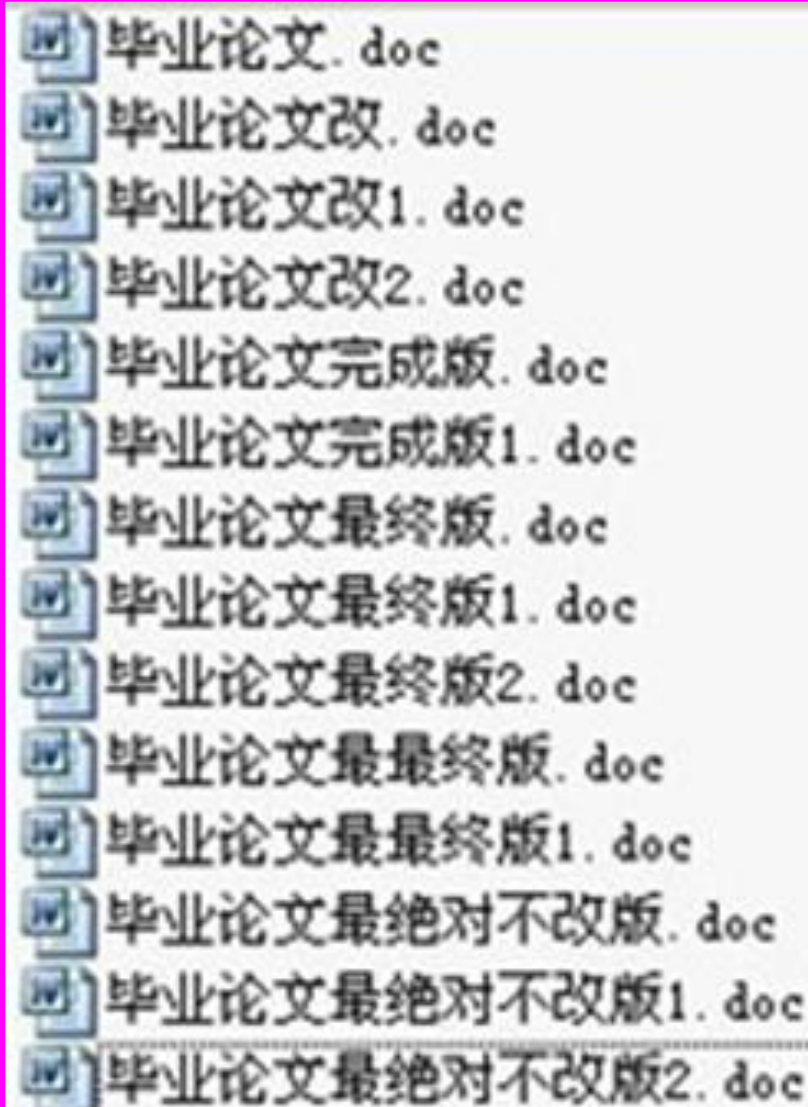
□ **基线**：软件配置项的一个稳定版本，是进一步开发的基础，只有通过正式的变更控制过程才能改变



版本控制问题

为什么要版本控制 

手工管理麻烦且混乱



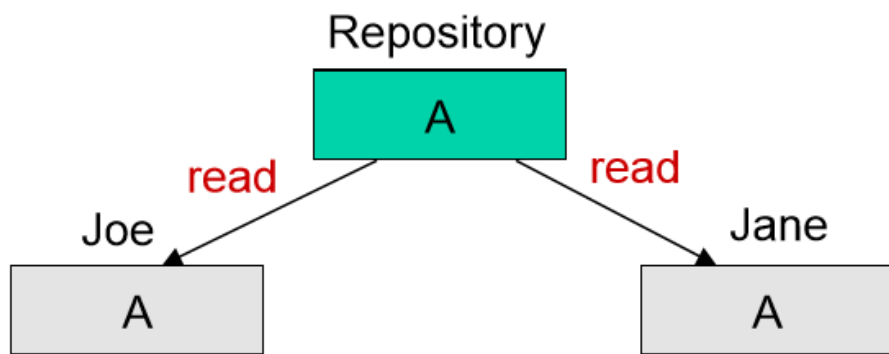
- 毕业论文.doc
- 毕业论文改.doc
- 毕业论文改1.doc
- 毕业论文改2.doc
- 毕业论文完成版.doc
- 毕业论文完成版1.doc
- 毕业论文最终版.doc
- 毕业论文最终版1.doc
- 毕业论文最终版2.doc
- 毕业论文最最终版.doc
- 毕业论文最最终版1.doc
- 毕业论文最绝对不改版.doc
- 毕业论文最绝对不改版1.doc
- 毕业论文最绝对不改版2.doc

版本控制的优点

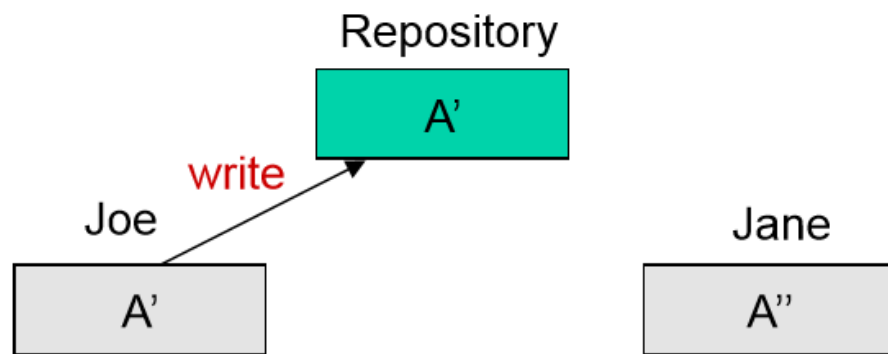
- ❑ 可以将某个文件回溯到之前的状态，甚至将整个项目都回退到过去某个时间点的状态。
- ❑ 可以比较文件的变化细节，查出最后是谁修改了哪个地方，从而找出问题出现的原因。
- ❑ 如果整个项目中的文件被改的改删的删，也照样可以轻松恢复到原先的样子
- ❑

版本控制问题

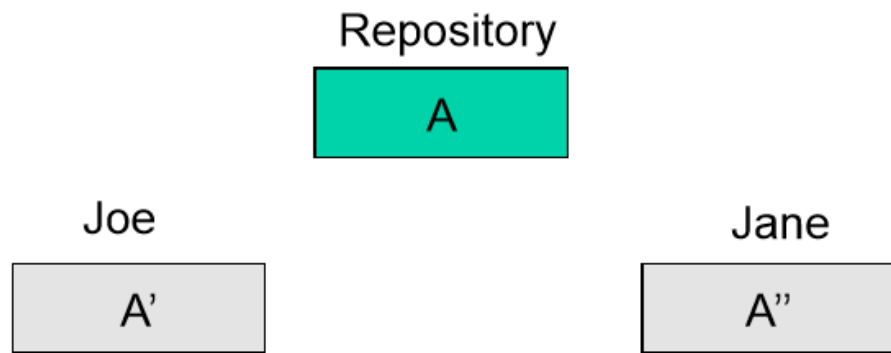
如果两人同时修改同一个代码文件，就会出现代码覆盖问题



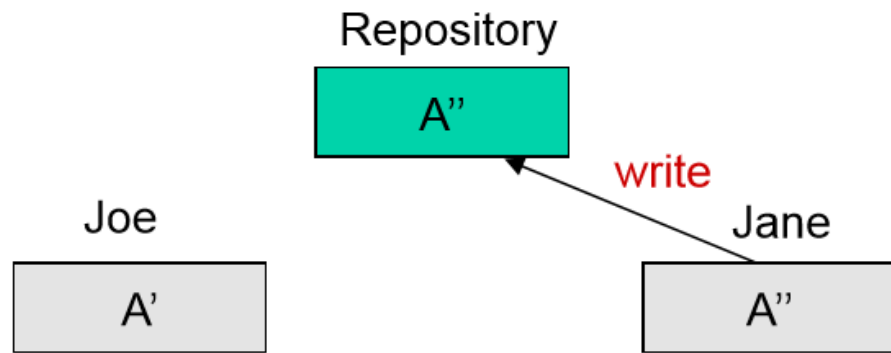
1. 两个程序员读取同一个文件



3. Joe首先发布其版本

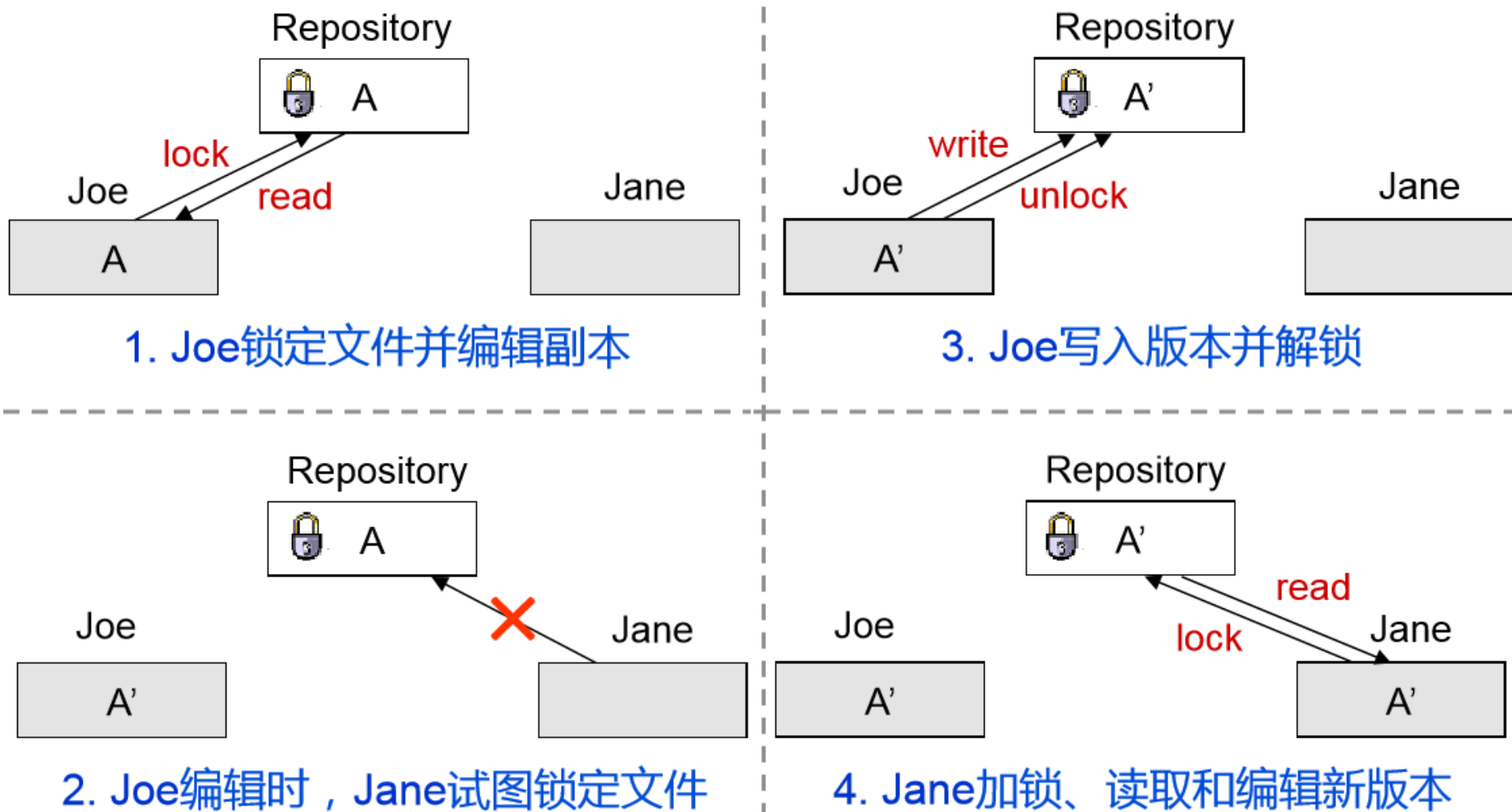


2. 两个程序员开始编辑文件副本

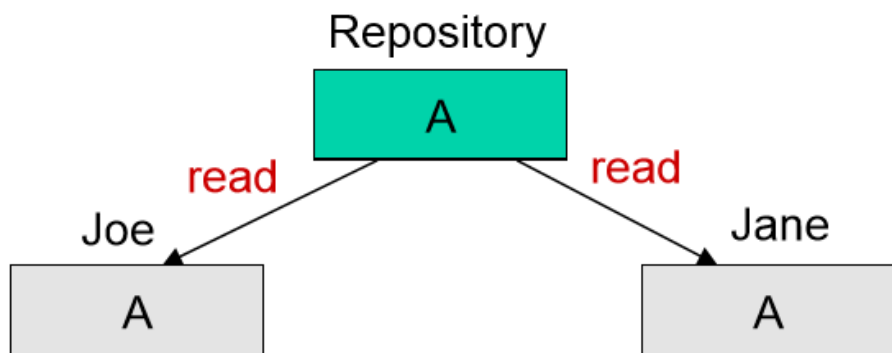


4. Jane意外地覆盖了Joe的版本

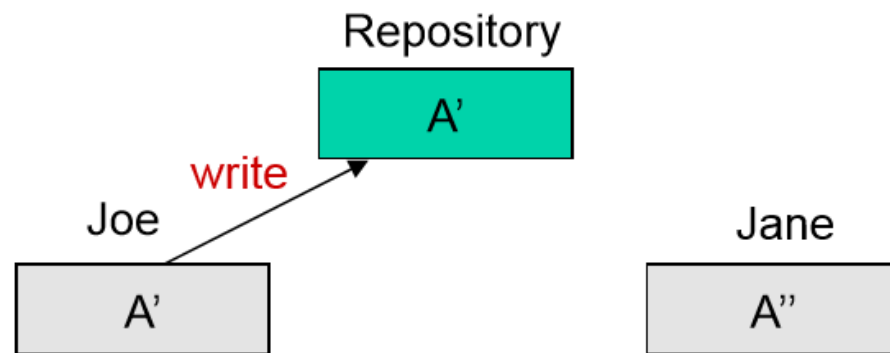
版本控制：独占工作模式



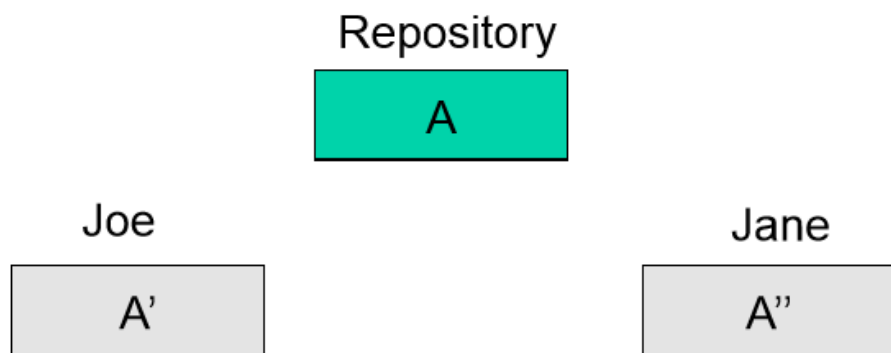
版本控制：并行工作模式



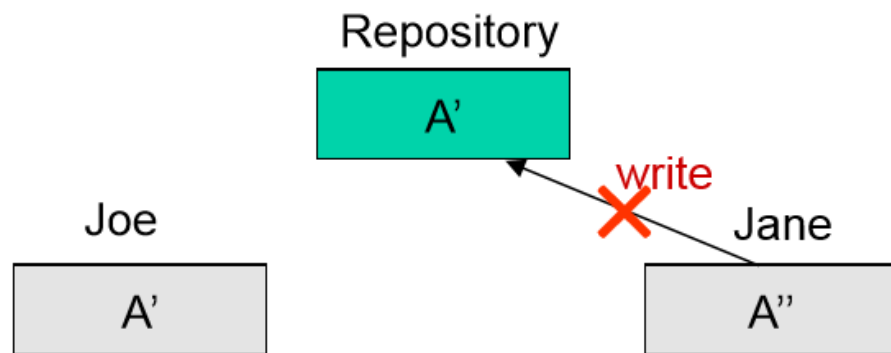
1. 两个程序员复制同一个文件



3. Joe首先发布其版本

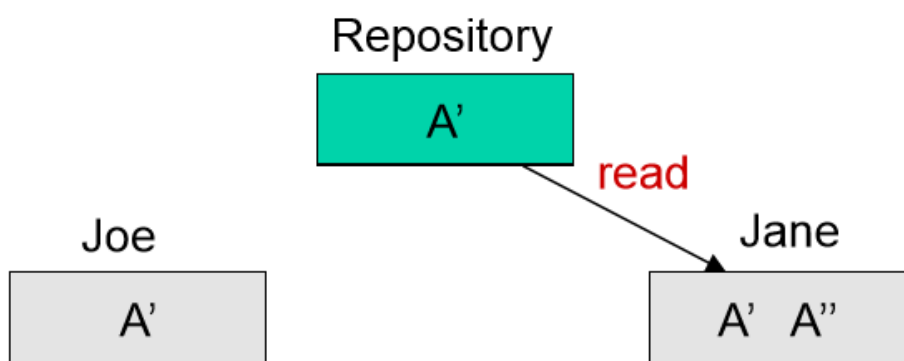


2. 两个程序员开始编辑文件副本

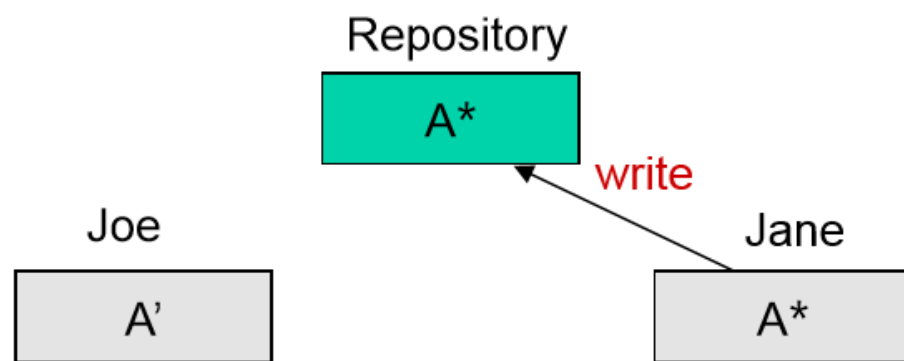


4. Jane发布时出现out-of-date错误

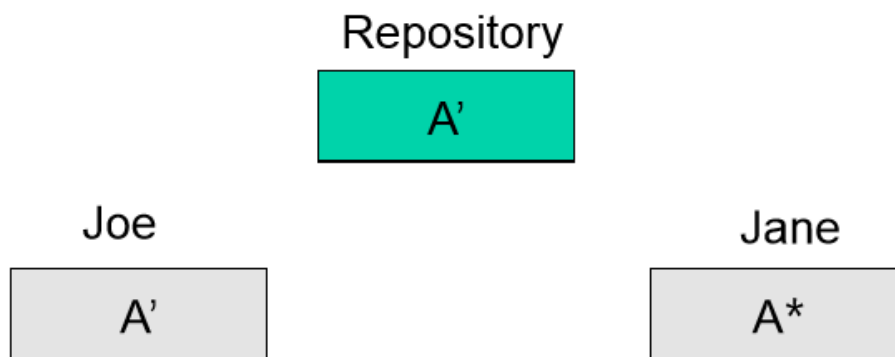
版本控制：并行工作模式



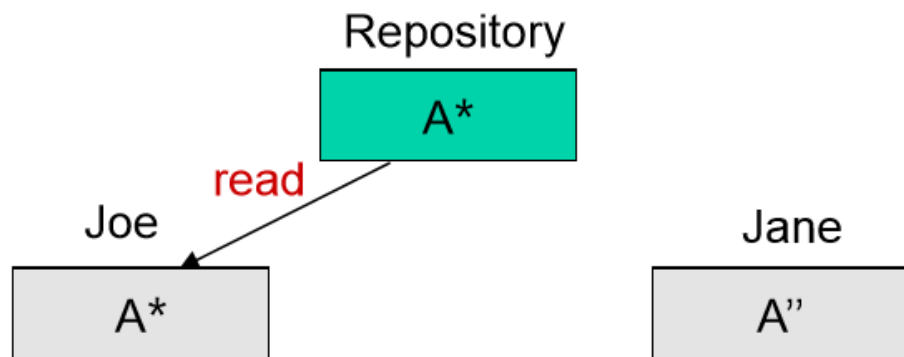
5. Jane将自己文件与最新版本比较



7. 发布合并版本



6. 创建新的合并版本



8. 两个程序员更新了各自的变化

软件配置管理工具



Rational ClearCase 是IBM公司的一款重量级软件配置管理工具，包括版本控制、工作空间管理、构建管理、过程控制，支持并行开发与分布式操作。



Subversion (SVN) 是一个开源的版本控制系统，支持可在本地访问或通过网络访问的数据库和文件系统存储库，具有较强而且易用的分支以及合并功能。



Microsoft Visual Sourcesafe 是微软公司推出的一款支持团队协同开发的配置管理工具，提供基本的文件版本跟踪功能，与微软的开发工具实现无缝集成。



Git 是一个开源的分布式版本控制工具，作为Subversion的升级版可以支持分布式异地开发，提供加密的历史记录，以变更集为单位存储版本历史，支持标签功能。

什么是Git?

□ Git是目前世界上最先进的**分布式版本控制系统**

□ 开发者: Linux之父 Linus Torvalds

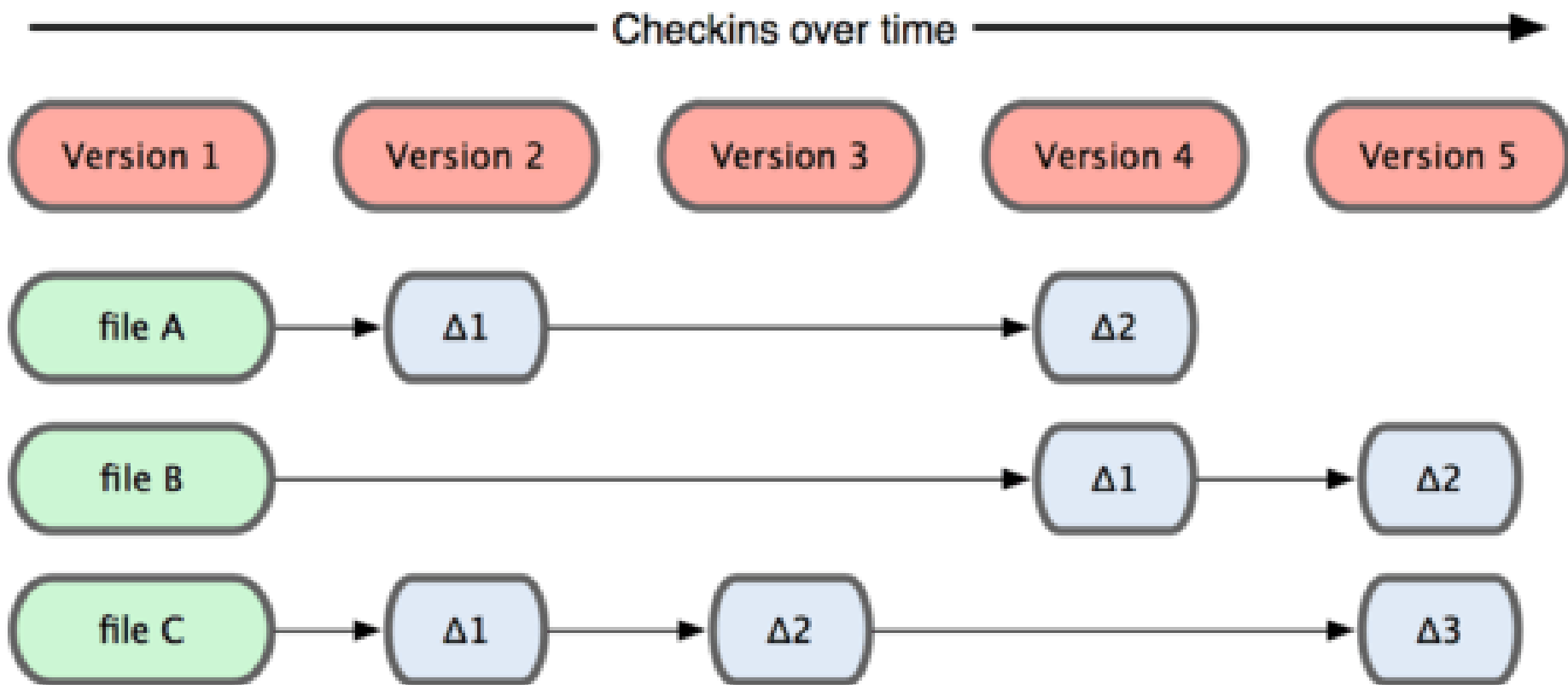


- Linux内核社区原本使用的是BitKeeper的商业化版本控制工具。2005年, 因为社区内有人试图破解BitKeeper协议, BitMover公司收回了免费使用BitKeeper的权力。
- Linus原本可以出面道个歉, 继续使用BitKeeper, 然而并没有。。。Linus大神仅用了**两周时间**, 自己用C写了一个分布式版本控制系统, 于是**Git诞生了!**

Git的思想和基本工作原理

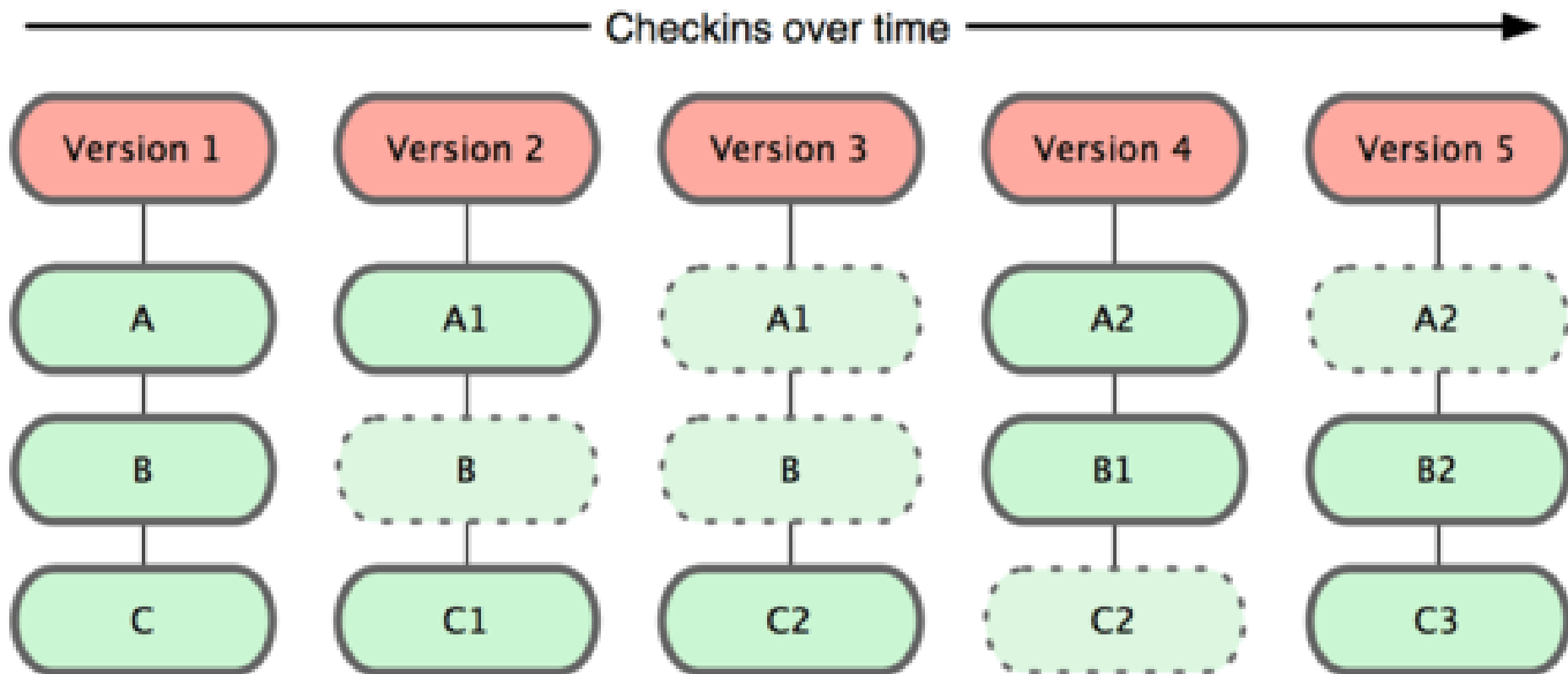
一、直接记录快照，而非差异比较

其他版本控制系统关心文件内容的具体差异，记录有哪些文件作了更新，以及都更新了哪些行的什么内容。



Git的思想和基本工作原理

Git每次提交更新时，会纵览一遍所有文件的指纹信息，若文件没有变化，Git不会再次保存，只对上次保存的快照作一链接



二、近乎所有操作都是本地执行

- Git 中的绝大多数操作都只需访问本地文件和资源，不用联网。Git 在本地磁盘上保存着所有当前项目的历史更新。
 - 如果想要看当前版本的文件和一个月前的版本之间有何差异，Git 会直接从本地数据库取出一个月前的快照和当前文件作一次差异运算，而不用请求远程服务器，或是把老版本的文件拉到本地来作比较。
 - Git 在本机可以频繁提交更新，需要上传到远程仓库时才需要联网。

三、三个工作区域

- Working Directory(**工作区**): 编辑、修改文件
- Staging area(**暂存区**): 暂存已经修改的文件
- Git repository(**Git仓库**): 最终确定的文件保存到仓库, 成为一个新版本。

每个项目都有一个.git目录保存元数据和对象数据库。每次克隆镜像仓库的时候, 实际拷贝的就是这个目录里的数据。

主文件夹 SourceCode Coursera-ML-AndrewNg-Notes				
名称	大小	类型	修改日期	
code	10 项	文件夹	2018年05月07日 星期一 14时18分22秒	
docx	8 项	文件夹	2018年05月07日 星期一 14时18分22秒	
html	11 项	文件夹	2018年05月07日 星期一 14时18分22秒	
images	323 项	文件夹	2018年05月07日 星期一 14时18分22秒	
markdown	12 项	文件夹	2018年05月07日 星期一 14时18分22秒	
ppt	19 项	文件夹	2018年05月07日 星期一 14时18分22秒	
srt	115 项	文件夹	2018年05月07日 星期一 14时18分22秒	
.git	11 项	文件夹	2018年05月07日 星期一 14时18分22秒	
branches	0 项	文件夹	2018年05月07日 星期一 14时06分29秒	
hooks	11 项	文件夹	2018年05月07日 星期一 14时06分29秒	
info	1 项	文件夹	2018年05月07日 星期一 14时06分29秒	
logs	2 项	文件夹	2018年05月07日 星期一 14时18分21秒	
objects	2 项	文件夹	2018年05月07日 星期一 14时06分29秒	
refs	3 项	文件夹	2018年05月07日 星期一 14时18分21秒	
config	206 字节	纯文本文档	2018年05月07日 星期一 14时18分21秒	
description	73 字节	纯文本文档	2018年05月07日 星期一 14时06分29秒	
HEAD	23 字节	纯文本文档	2018年05月07日 星期一 14时18分21秒	
index	62.9 KB	未知	2018年05月07日 星期一 14时18分22秒	
packed-refs	180 字节	纯文本文档	2018年05月07日 星期一 14时18分21秒	
...			2018年05月07日 星期一 14时18分22秒	

1. 工作区, 不包括 .git 目录

在电脑里能看到的目录

2. 版本库

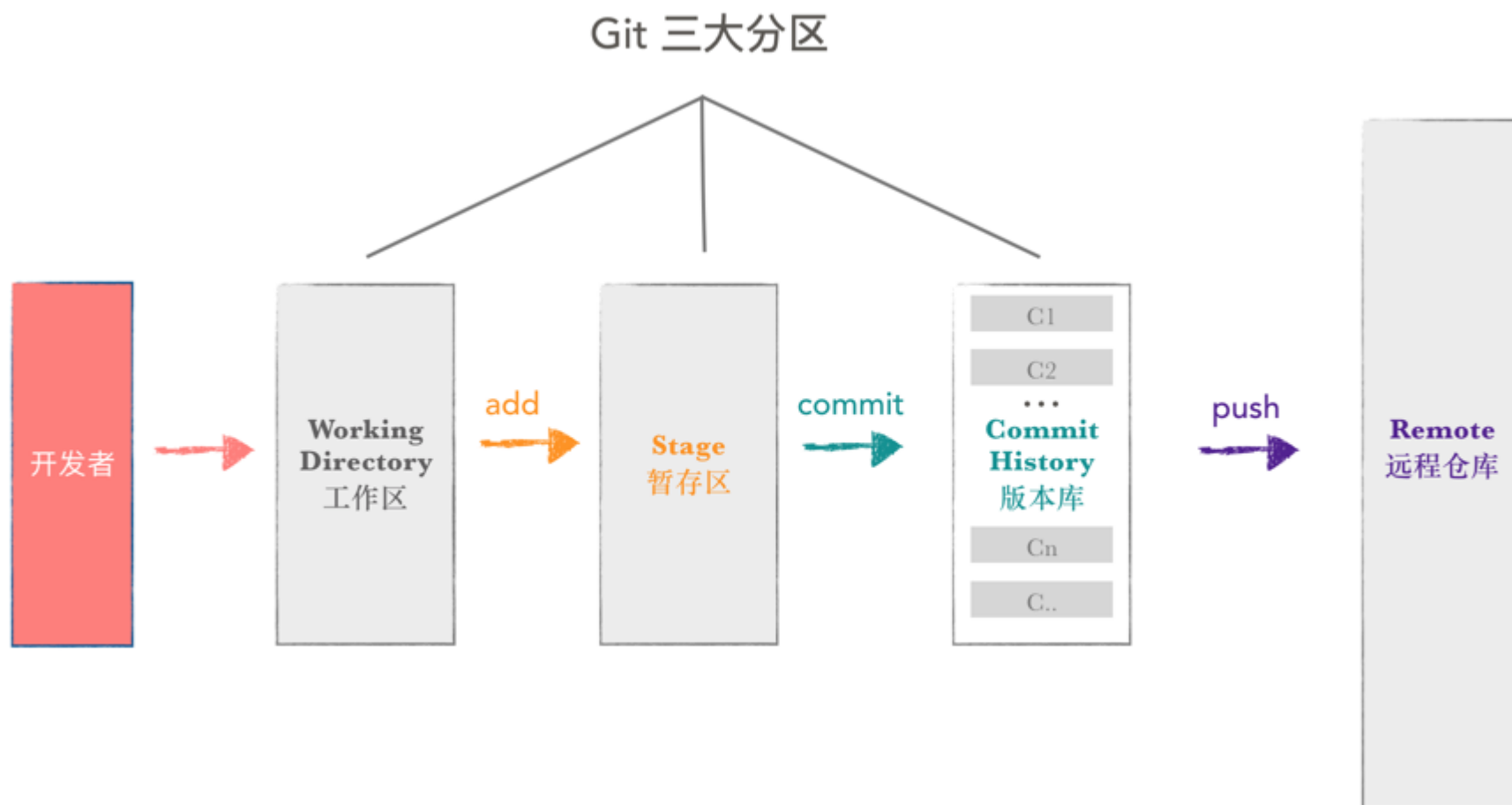
工作区里的一个隐藏目录 .git

3. 暂存区

一般存放在 .git 目录下的 index 文件 (.git/index) 中, 所以暂存区有时也叫作索引 (index)。

Git的思想和基本工作原理

各区之间的数据传递流程示意图



什么是GitHub ?

- GitHub是全球最大的开源社交编程及代码托管网站，因为只支持Git作为唯一的版本库格式进行托管，故名GitHub，诞生于2008年，目前拥有超过3600万的开发者用户。
- GitHub主要有下面两个功能：
 - 项目管理平台
 - 开源社交平台

2018年6月微软以 75 亿美元收购 GitHub

- **GitHub托管各种GIT资源版本库(repository), 分为private和public:**
 - **Public 资源版本库不限制个数, 免费。**
 - **Public 资源版本库任何人可见, 组内人员可提交。**
 - **Private 资源版本库收费, 但edu的邮箱可以免费申请有个数限制的资源版本库。**
 - **Private资源版本库组内人员可见, 可提交**

课程设计启动任务

- **完成组队**，课后我会在群上发布共享Excel文档，请各组在上面填写个人信息（**可初步分配成员角色**）
- **选题**（鼓励同学们自己选题，**可以几个组做同一个题**，但互相之间**不要干扰和抄袭**，需要有自己的想法，最后的成品允许代码不全，当然代码完成多的组会有奖励分）
- **安装Git**，**注册GitHub**，建立各组的项目远程仓库，**可申请学生账户**，**免费使用有限的私有仓库**。

3.1 需求分析的任务

3.2 与用户沟通获取需求的方法

3.3 分析建模与规格说明

3.4 实体联系图（不讲）

3.5 数据规范化（不讲）

3.6 状态转换图

3.7 其他图形工具

3.8 验证软件需求

需求(requirement)分析基本任务是：准确回答：系统必须做什么？——对目标系统提出完整、准确、清晰、具体的要求。


需求分析的结果是系统开发的基础，关系到工程的成败和软件产品的质量。

什么是需求?

小P刚加入一个项目组，项目经理安排他做需求分析，小P有点不乐意：“需求有什么好分析的啊？客户要什么就给什么呗，简直浪费我这个人才！”

一天，客户打电话说：“我要一只羊”。小P一听，太简单了，写下了需求：“XX客户需要一只羊。”小P叫小Q去处理这件事，小Q觉得很简单，抓了一只羊就送过去了。

结果客户的投诉第二天就来了，项目经理训斥小P，小P觉得委屈：“我是按照客户的要求做的呀，怎么就错了呢？”

小P为什么挨批 

什么是需求？

客户的要求非常不明确

客户的要求： 我要一只羊？

活的？死的？品种？毛色？养？吃？羊玩具？羊的标志？……

应该发现客户要求不明确的问题，进行分析，实时询问客户：什么时间、地点要这只羊？为什么要这只羊？用这只羊做什么？处理这只羊的过程是什么？

客户回答：请设计一个羊的标志作为公司的徽标，挂在公司入口处，10天后使用。标志具体要求……，**这才是客户要求的羊**

为什么要做需求分析？

有同学说，我把**用户的愿望百分之百地实现**了，这不就行了么？不用搞那么多分析啊、故事啊、讨论啊、文档啊.....

请看漫画“他满意了吗”



在长时间一丝不苟地实现之后，得到了**和用**
户要求一模一样的产品！用户满意吗？

需求工程师要做什么？

一天，软件工程师、计算机科学家和数学家在圆明园相遇，他们看到湖面上游来一只黑天鹅...



在软件工程师的模型世界中，将天鹅建模为黑色。

$$\forall x, Swan(x) \rightarrow Black(x)$$

在计算机科学家的模型世界中，仅将一部分天鹅建模为黑色。

$$\exists x : Swan(x) \wedge Black(x)$$

需求工程师要做什么？

数学家的模型则是：在圆明园，有一个湖上，存在至少一只天鹅，它的一面是黑色。

$$\exists x \exists y \exists z \exists u, Swan(x) \wedge Lake(y)$$
$$\wedge SwimOn(x, y)$$
$$\wedge At(y, OldSummerPalace)$$
$$\wedge Black(OneSide(x))$$

- 软件工程师做出尽可能简化问题复杂度的假设
- 计算机科学家要找出不失一般性的解决方法
- 数学家追求对问题描述的精确性

1. 获取和引导需求

需求可以来自用户、各种管理机构、软件企业本身、技术团队本身……

2. 分析和定义需求

3. 验证需求

4. 在软件产品的生命周期中管理需求

在软件的生命周期中，需求在发生变化，技术在发展，团队成员的能力也在提高，需要不断对需求进行重新审核并调整

3.1 需求分析的任务

一、确定对系统的综合要求

功能需求

最核心的需求。要求产品必须实现的所有功能。

性能需求

系统必须满足的定时约束或容量约束，包括响应时间、信息量速率、主存容量、磁盘容量、安全性等方面的需求。

可靠性和可用性需求

可靠性需求定量地指定系统的可靠性。可用性与可靠性密切相关，它量化了用户可以使用系统的程度。

3.1 需求分析的任务

一、确定对系统的综合要求

出错处理需求

说明系统对环境错误应如何响应。

接口需求

描述应用系统与它的环境通信的格式。如用户接口需求、硬件接口需求、软件接口需求、通信接口需求等。

约束

描述系统应遵守的限制条件。如精度、工具、语言、标准、平台等。

3.1 需求分析的任务

一、确定对系统的综合要求

逆向需求

说明软件系统不应该做什么。应选取能澄清真实需求且可消除可能发生的误解的那些逆向需求。

将来可能提出的需求

明确列出那些虽然不属于当前系统开发范畴，但是据分析将来很可能会提出来的要求，以便系统将来能方便可能的扩充和修改。

3.1 需求分析的任务

二、分析系统的数据要求

复杂的数据由许多基本的数据元素组成，数据结构表示数据元素之间的逻辑关系。可利用数据字典或图形工具辅助描绘数据结构。

三、导出系统的逻辑模型

综合上述分析结果可导出系统的逻辑模型，可用数据流程图、ER图、状态转换图、数据字典和主要的处理算法描述。

四、修正系统开发计划

根据分析过程获得的对系统的更深入更具体的了解，修正以前制定的开发计划。

需求获取

对需求顶层目标的提问

对业务流程的提问

对需求优先级的提问

对需求验收标准的提问

需求获取过程中，最困难的不是记录用户需求，而是与用户探讨磋商，**发现真正要解决的问题，确定适用的方案。**

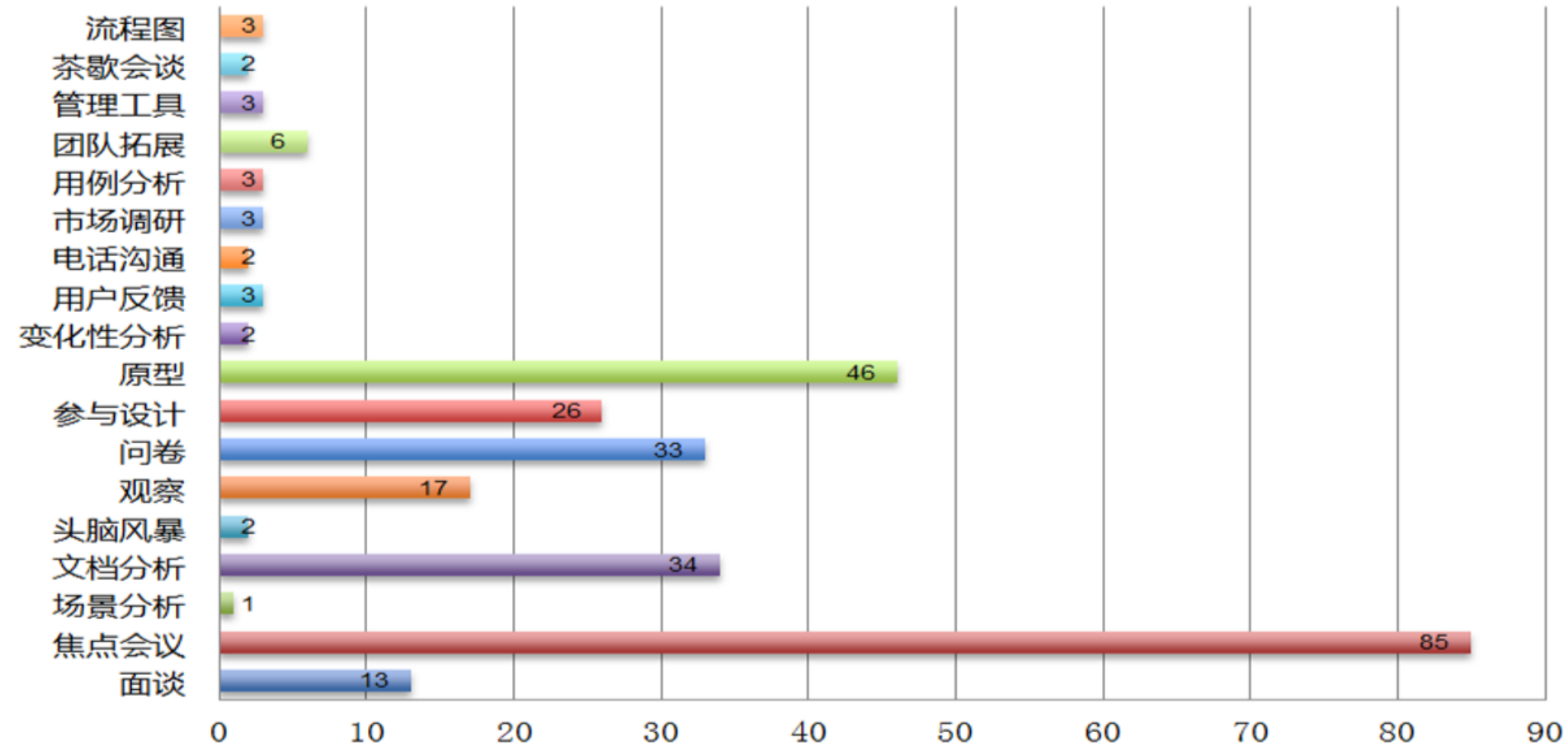
— Steve McConnell



3.2 获取需求的方法

- 深入面谈
- 问卷调查
- 会议
- 典型用户和典型场景
- 观察商业过程和工作流
- 原型法
- 文档分析
- 建模
-

软件需求获取方法应用现状调研结果



3.2 获取需求的方法

一、深入面谈

最早开始使用的获取用户需求的技术，也是迄今为止仍然广泛使用的需求分析技术。

□ 访谈中注意：

- 关注用户行为背后的原因
- 避免让用户成为设计师
- 避免讨论技术细节
- 避免诱导性问题
- 鼓励讲故事

深入了解用户的背景、心理、需求等，效果往往取决于主持面谈的团队成员的能力

3.2 获取需求的方法

- 深入面谈也可以用在某一特定领域，例如软件的用户可用性和用户界面，称为**软件可用性研究**。
- 此类研究着重探究用户使用软件时有哪些困难，并如何改进软件，让软件更好用。

常用的方法是请用户来完成一些任务，然后软件项目成员可以在一旁观察，也可以隐蔽在单向玻璃后，或通过录像观察。这时候让用户使用的软件不一定是自己公司开发的软件，也可以使用别的软件，从而找出此类软件的问题，以及用户潜在的需求。

3.2 获取需求的方法

二、用户调查问卷

向用户提供事先设计好的问题，让用户回答。

- 如用户浏览某网站时，弹出对话框，打断用户的思路，要求用户回答几个问题。用户是否会心不在焉，乱点一气？

□ 调查者的一些常见错误

■ 问题定义不准确

如你用哪个搜索引擎？用户可能提供多个合理答案：最近使用的；最喜欢的但未必最常用的（可能最喜欢的搜索引擎访问不了）。定义不准确的问题会让用户困惑，我们也许能收集到很多答案，但仍然无法准确了解用户的想法。

3.2 获取需求的方法

■ 使用含糊不清的形容词、副词描述时间、数量、频率、价格等

例如：最近、有时、经常、偶尔、很少、很多、很贵、很便宜。

这些词语对不同用户和在不同的语境中有不同的意义。

■ 让用户花额外的努力来回答问题

例如：请问你全家平均每人每年下载多少手机应用软件？用户很难在短时间内得出准确的答案。

■ 问题带有引导性的倾向

例如：用户普遍认为，搜索引擎A收录了许多侵犯版权的资料而拒绝承认错误，搜索引擎B则赢得用户信任，你会选择A或B？

■ 问题涉及用户隐私、用户所在公司的商业机密或细节等

3.2 获取需求的方法

□ 用户调查问卷的问题可采用以下这些方式：

■ 全开放式问题

例如，你对手机上的日程管理软件的期望是：_____

这种问题能让用户畅所欲言，但是整理和量化比较困难。

■ 二项选择题

用户只回答**是/否**即可。这类问题便于统计处理，容易分析，但用户没有进一步阐明理由的机会，难以反映意见与程度的差别，了解的情况也不够深入。

3.2 获取需求的方法

□ 用户调查问卷的问题可采用以下这些方式：

■ 多项选择题

■ 顺位选择题

例如，您选择手机背单词软件的主要考虑因素是
(按照优先级填写A、B、C……)：

A. 词汇量

B. 能记录进度

C. 能定制单词表

D. 支持与PC同步

E. 支持英语四级等专门词库

F. 支持发音

3.2 获取需求的方法

三、会议

- 与目标用户的代表，项目的利益相关者开会讨论用户想要什么，用户对软件的评价等等。
- 缺陷：
 - 一群人在一起，往往大家会出于讨好其他人的心理来发表意见，避免不一致的意见或冲突。
 - 参与讨论的人士表达能力也会有差异，有可能会出出现一些善于表达的人士控制讨论议程的倾向。
 - 讨论者对不熟悉的事物不能表达有价值的想法。

3.2 获取需求的方法

三、会议

□ 缺陷:

- 讨论者容易受到主持人有意或无意的影响。
- 研究者往往从不同意见中挑选最符合自己利益的那些条目，然后对外号称这就是大家的共识。

要求会议的组织者有很强的组织能力，能让不同角色都充分表达意见，并如实地总结这些意见。