北京科技大学
University of Science and Technology Beijing

软件工程

软件项目管理

管理学基础

- 计划：预先筹划，包括对未来环境进行分析，以此确定组织活动的目标，并对目标的实施进行具体的规划和安排，计划职能是管理的首要职能，它指明了组织活动的方向，并保证各项活动有序进行

- 组织：为保证计划的顺利实现，而对组织系统的设计，及对个单位各成员在工作执行中的分工协作关系进行合理的安排

- 领导：管理者利用组织赋予的职权和自身的所拥有的影响力去指挥影响和激励他人为实现组织目标，而努力工作的管理活动过程

- 控制：按既定目标和标准对组织的活动进行监督、检查，发现偏差，采取纠正措施，使工作能按计划进行，或适当调整计划以达到预期目的

软件项目管理

- 管理内容（4P）
  - 人员：如何通过吸引、培养、激励、部署和聘用开发人员以改进软件开发（人员组织）
  - 产品：明确产品的目标、边界、约束（需求分析）
  - 过程：制定计划，确定开发过程（估算，计划，软件过程）
  - 项目：监督和控制软件项目的进展和质量（质量管理和配置管理）

北京科技大学
University of Science and Technology Beijing

估算

估算

估算

- 概念
  - 在需求已经明确的基础上，对软件规模、开发的成本和工作量进行估算，作为制定计划的依据

如何衡量软件的规模?

基于代码行数的估算

- 基本思想
  - 根据多个开发人员的经验，对待开发系统（模块）的代码行数进行估计，计算加权平均

$$LOC = \frac{\overline{min} + 4 \cdot \overline{likely} + \overline{max}}{6}$$

$\overline{min}$	表示最小规模的平均值
$\overline{likely}$	表示最有可能规模的平均值
$\overline{max}$	表示最大规模的平均值

基于代码行数的估算

$$LOC = \frac{\overline{min} + 4 \cdot \overline{likely} + \overline{max}}{6}$$

	min	likely	max
人员1	2300	3000	3500
人员2	2800	3400	4000
人员3	2000	3200	4300
估计	3183		

基于代码行数的估算

- 缺点
  - 源程序仅是软件配置的一个成分，用它的规模代表整个软件的规模似乎不太合理
  - 用不同语言实现同一个软件所需要的代码行数并不相同
  - 不适用于非命令式/过程式语言。

基于功能点的估算

- 基本思想
  - 基于软件信息域的特征和技术复杂性的经验关系来计算功能点

$$FP = I \times (0.65 + 0.01 \times \sum_{i=1}^{14} F_i)$$

实数

I表示信息域特征【软件本身的复杂性】
Fi表示技术复杂性

基于功能点的估算

- 信息域特征（I）
  - 外部输入数（EI）：每个外部输入源于一个用户，或从另一个应用系统中传送过来。它提供了面向应用系统的数据和控制信息。输入常用于更新内部逻辑文件。输入应与独立计数的查询（EQ）相区分开来。
  - 外部输出数（EO）：每个外部输出应从应系统中导出，并为用户提供信息。外部输出指的是报告、屏幕、错误消息等，不对报告中的单独数据项进行分开计数。

基于功能点的估算

- ~~信息域特征~~

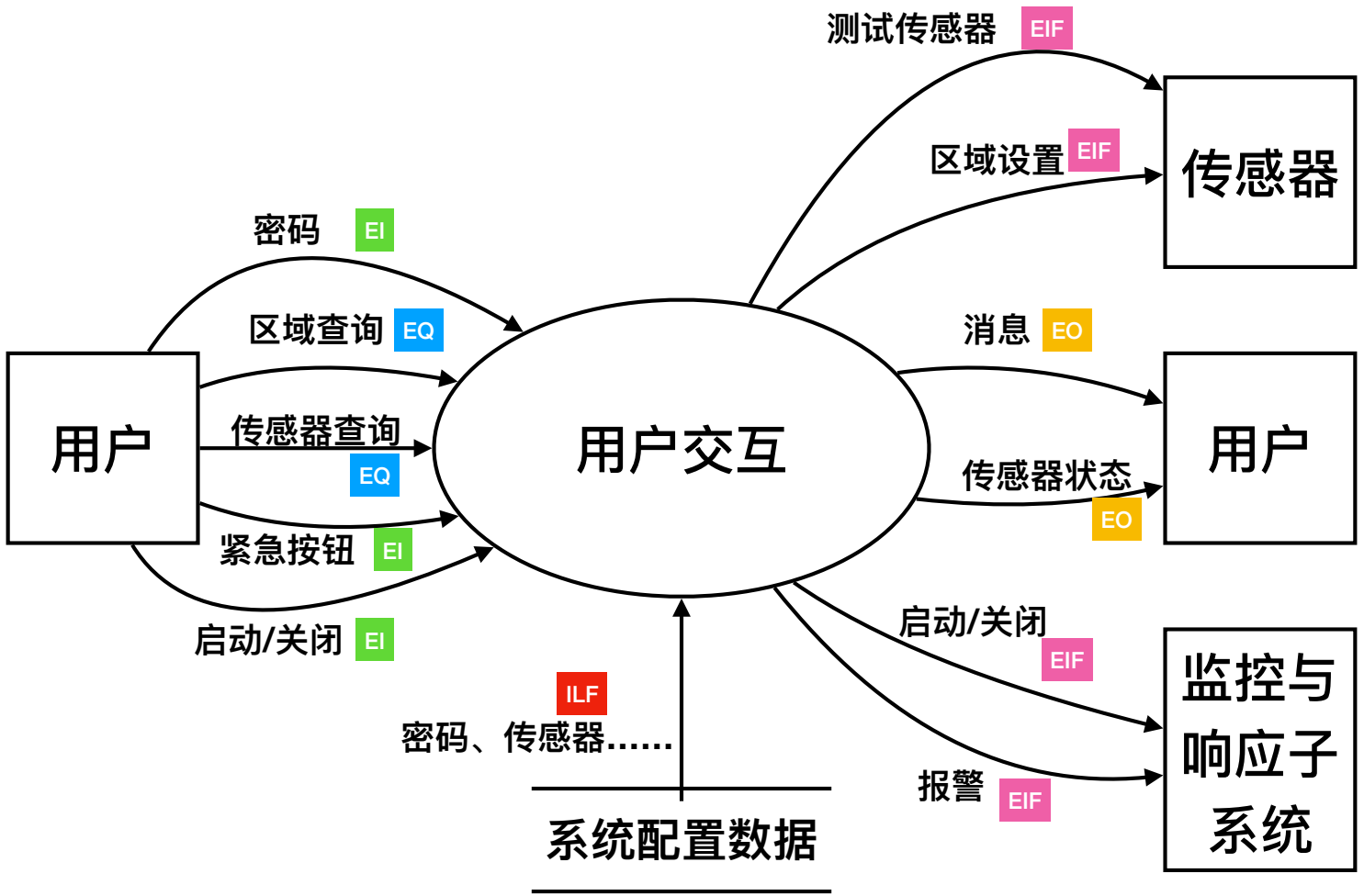
- 外部查询数（EQ）：一个外部查询定义为一个在线输入。其结果是以在线输出的方式产生的某个即时软件响应

- 内部逻辑文件数（ILF）：每个内部逻辑文件是系统中内的数据逻辑分组，可能是一个数据库或者一个独立文件

- 外部接口文件数（EIF）：每个外部接口文件是系统外部的数据逻辑分组，包括与外系统的接口

基于功能点的估算

EI-3
EQ-2
EO-2
ILF-1
EIF-4



基于功能点的估算

EI-3
EQ-2
EO-2
ILF-1
EIF-4

信息域值	计数	加权值			加权计数
		简单的	中等的	复杂的	
EI	3	3	4	6	9
EO	2	4	5	7	8
EQ	2	3	4	6	6
ILF	1	7	10	15	7
EIF	4	5	7	10	20
总计(I)	50				

基于功能点的估算

- 技术复杂性（F，每项取值0~5） 14种

1. 系统需要可靠的备份和恢复吗?
2. 从应用系统中传输信息和将信息传输到应用系统时需要专门的数据通信吗?
3. 存在分布式处理功能吗?
4. 性能是关键的吗?
5. 系统将运行在一个现有的、使用紧张的操作环境中吗?
6. 系统需要在线数据项吗?
7. 在线数据项需要对多个屏幕或操作建立输入事务吗?

基于功能点的估算

- 技术复杂性（F，每项取值0~5）

8. ILF被在线更新吗?
9. 输入、输出、文件或查询复杂的吗?
10. 内部处理是复杂的吗?
11. 所设计的代码是可复用的吗?
12. 安装包括在设计中吗?
13. 系统设备不同组织中的多个安装而设计的吗?
14. 因为系统是为便于变更和易于用户使用而设计的吗?

基于功能点的估算

- 功能点估算
  - 假设技术复杂性的值是46，根据公式，对于50个信息域特征值的系统，具有的功能点是：

$$50 \times (0.65 + 0.01 \times 46) = 51.11$$

工作量估计

- 工作量单位：人月，人年等
- 基本方法
  - 获得规模估计值后，根据经验公式计算工作量
  - 例如：
    - 规模是2000行代码，人均生产力是2500行/月，则工作量是0.8人月
      - 但计算并非这么简单（为什么）

工作量估计

- 单变量估计模型

$$E = A + B \times (e_v)^C$$

非线性

A,B,C是常数，ev是估算变量（LOC或FP）

$E = 5.2 \times (KLOC)^{0.91}$	Walston-Felix
$E = 5.5 + 0.73 \times (KLOC)^{1.16}$	Bailey-Basili
$E = 3.2 \times (KLOC)^{1.05}$	Boehm
$E = 5.288 \times (KLOC)^{1.047}$	Doty
$E = -91.4 + 0.355 \times FP$	Albrecht-Gaffney
$E = -37 + 0.96 \times FP$	Kemerer

COCOMO2模型

多因素估计模型

- COCOMO2（构造性成本模型）
  - Boehm与1981年提出，1996年更新
  - http://csse.usc.edu/tools/cocomoii.php

$$a \times KLOC^{b} \times \prod_{i=1}^{17} f_i$$

a,b是模型常数，fi是成本因素

17种

COCOMO2模型

- 成本因素
  - 每个成本因素都根据它的重要程度和对工作量影响大小被赋予一定数值（称为工作量系数）

		甚低	低	正常	高	甚高	特高
产品因素	1 要求的可靠性	0.75	0.88	1.00	1.15	1.39	
	2 数据库规模		0.93	1.00	1.09	1.19	
	3 产品复杂度	0.75	0.88	1.00	1.15	1.30	1.66
	4 要求的可复用性		0.91	1.00	1.14	1.29	1.49
	5 需要的文档量	0.89	0.95	1.00	1.06	1.13	
平台因素	6 执行时间约束			1.00	1.11	1.31	1.67
	7 主存约束			1.00	1.06	1.21	1.57
	8 平台变动		0.87	1.00	1.15	1.30	
人员因素	9 分析员能力	1.50	1.22	1.00	0.83	0.67	
	10 程序员能力	1.37	1.16	1.00	0.87	0.74	
	11 应用领域经验	1.22	1.10	1.00	0.89	0.81	
	12 平台经验	1.24	1.10	1.00	0.92	0.84	
	13 语言和工具经验	1.25	1.12	1.00	0.88	0.81	
	14 人员连续性	1.22	1.10	1.00	0.89	0.81	
项目因素	15 使用软件工具	1.24	1.12	1.00	0.86	0.72	
	16 多地点开发	1.25	1.10	1.00	0.92	0.84	0.78
	17 要求的开发进度	1.29	1.10	1.00	1.00	1..00	

COCOMO2模型

- a值的确定：经典值是3.0，需根据实际情况调整
- b值的确定：由5个分级因素Wi决定，每个Wi取值范围0~5
  - 项目先例性，开发灵活性，风险排除度，项目组凝聚力，过程成熟度

$$b = 1.01 + 0.01 \times \Sigma_{i=1}^{5} W_i$$

开发时间估计

- 问题
  - 工作量估计的结果是X人月，有Y个人，是否意味着开发时间是X/Y?
  - 错误！忽略了人员的培训、沟通成本
- 开发时间估计

$$T = 2.5E^{0.35} \quad \text{Walston-Felix}$$

$$T = 3.0E^{0.33+0.2\times(b-1.01)} \quad \text{COCOMO2}$$

$$T = 2.4E^{\frac{1}{3}} \quad \text{Putnam}$$

开发时间的估计

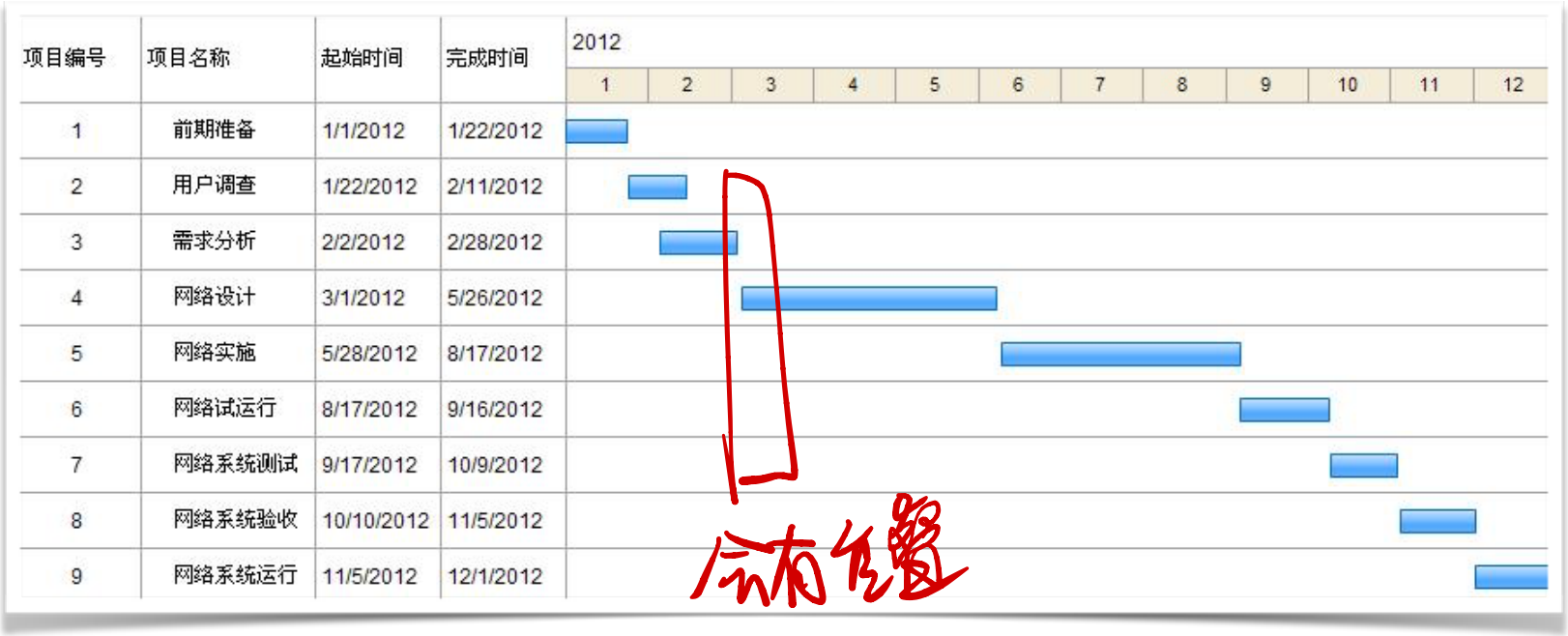
- 通过人力换时间是不可取的
  - 当小组变得更大时，每个人需要用更多时间与组内其他成员讨论问题、协调工作，因此增加了通信开销。
  - 如果在开发过程中增加小组人员，则最初一段时间内项目组总生产率不仅不会提高反而会下降。这是因为新成员在开始时不仅不是生产力，而且在他们学习期间还需要花费小组其他成员的时间。
  - Boehm根据经验指出，软件项目的开发时间最多可以减少到正常开发时间的75%。

北京科技大学
University of Science and Technology Beijing

进度安排

进度安排

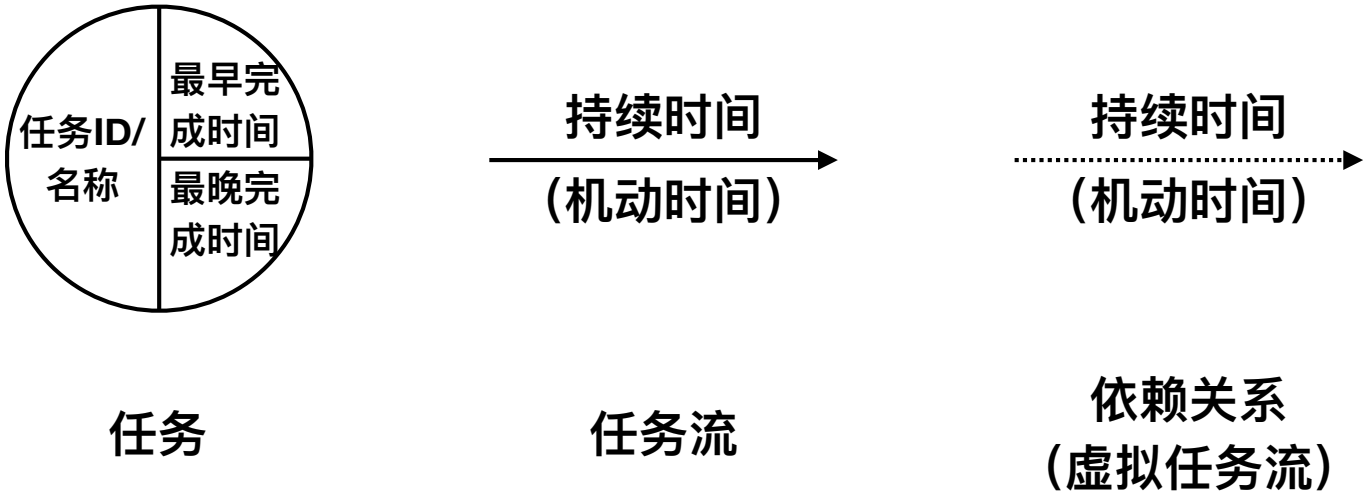
- 甘特（Gantt）图
  - 刻画任务和任务的时间安排



项目编号	项目名称	起始时间	完成时间
1	前期准备	1/1/2012	1/22/2012
2	用户调查	1/22/2012	2/11/2012
3	需求分析	2/2/2012	2/28/2012
4	网络设计	3/1/2012	5/26/2012
5	网络实施	5/28/2012	8/17/2012
6	网络试运行	8/17/2012	9/16/2012
7	网络系统测试	9/17/2012	10/9/2012
8	网络系统验收	10/10/2012	11/5/2012
9	网络系统运行	11/5/2012	12/1/2012

进度安排

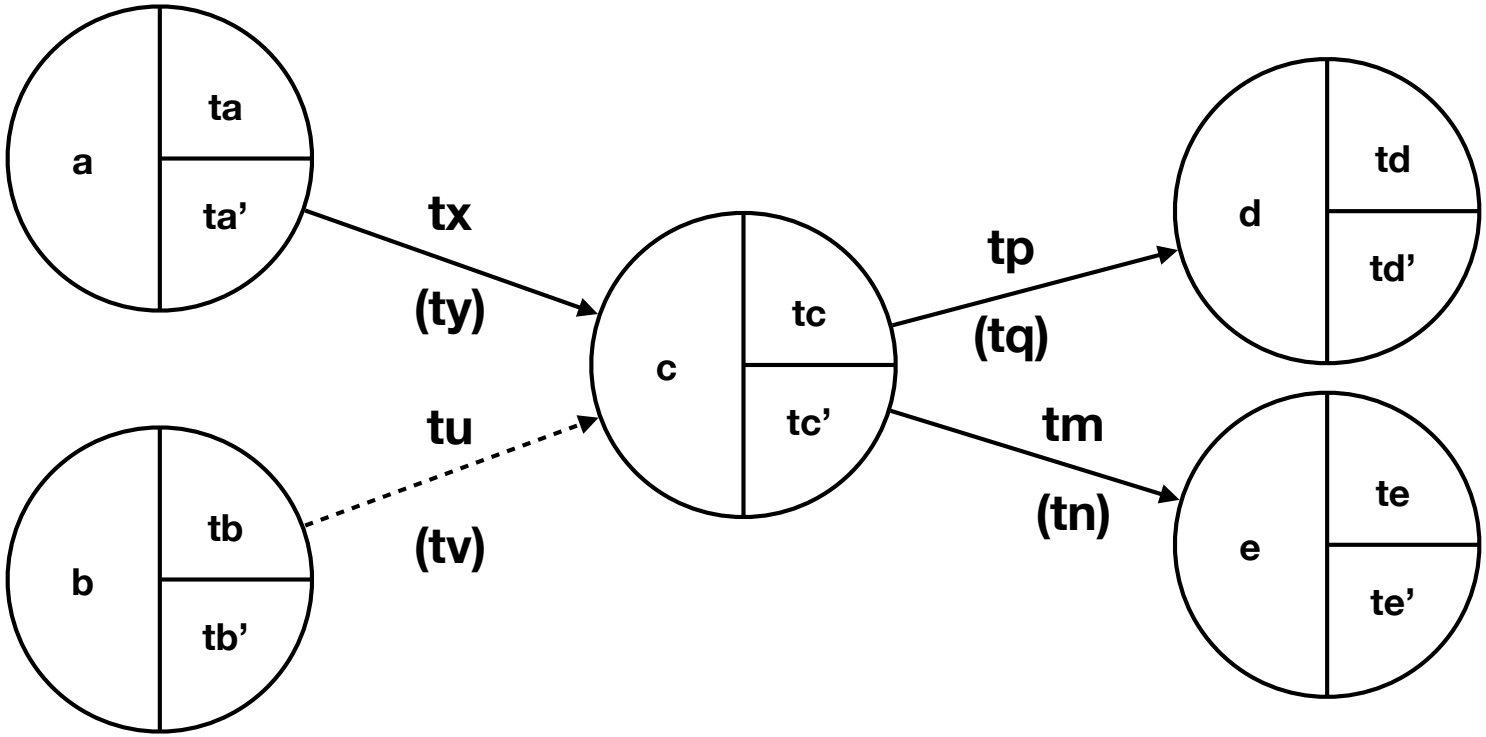
- 任务网络
  - 刻画子任务之间的相互依赖关系



进度安排

- 任务网络建立过程
  - 定义任务、任务流和依赖关系
  - 标注持续时间
  - 起始任务的最早完成时间和最晚完成时间为0
  - 从起始任务开始计算所有任务的最早开始时间
  - 终止任务的最晚完成时间等于最早完成时间
  - 从终止任务的开始计算所有任务的最晚完成时间
  - 计算机动时间

进度安排



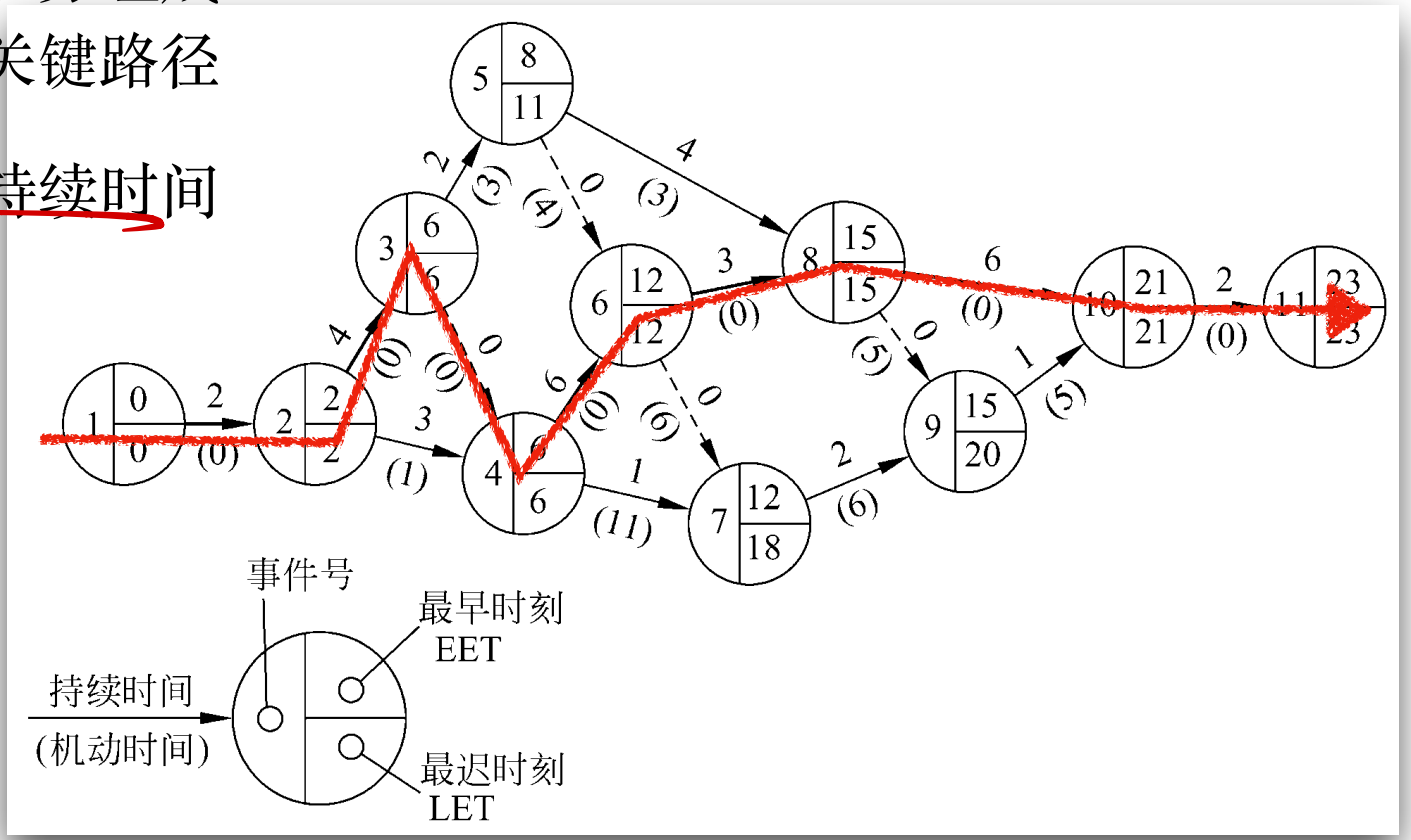
tc=max(ta+tx,tb+tu)　　ty=tc'-ta-tx

tc'=min(td'-tp,te'-tm)　　tv=tc'-tb-tv

关键路径

进度安排

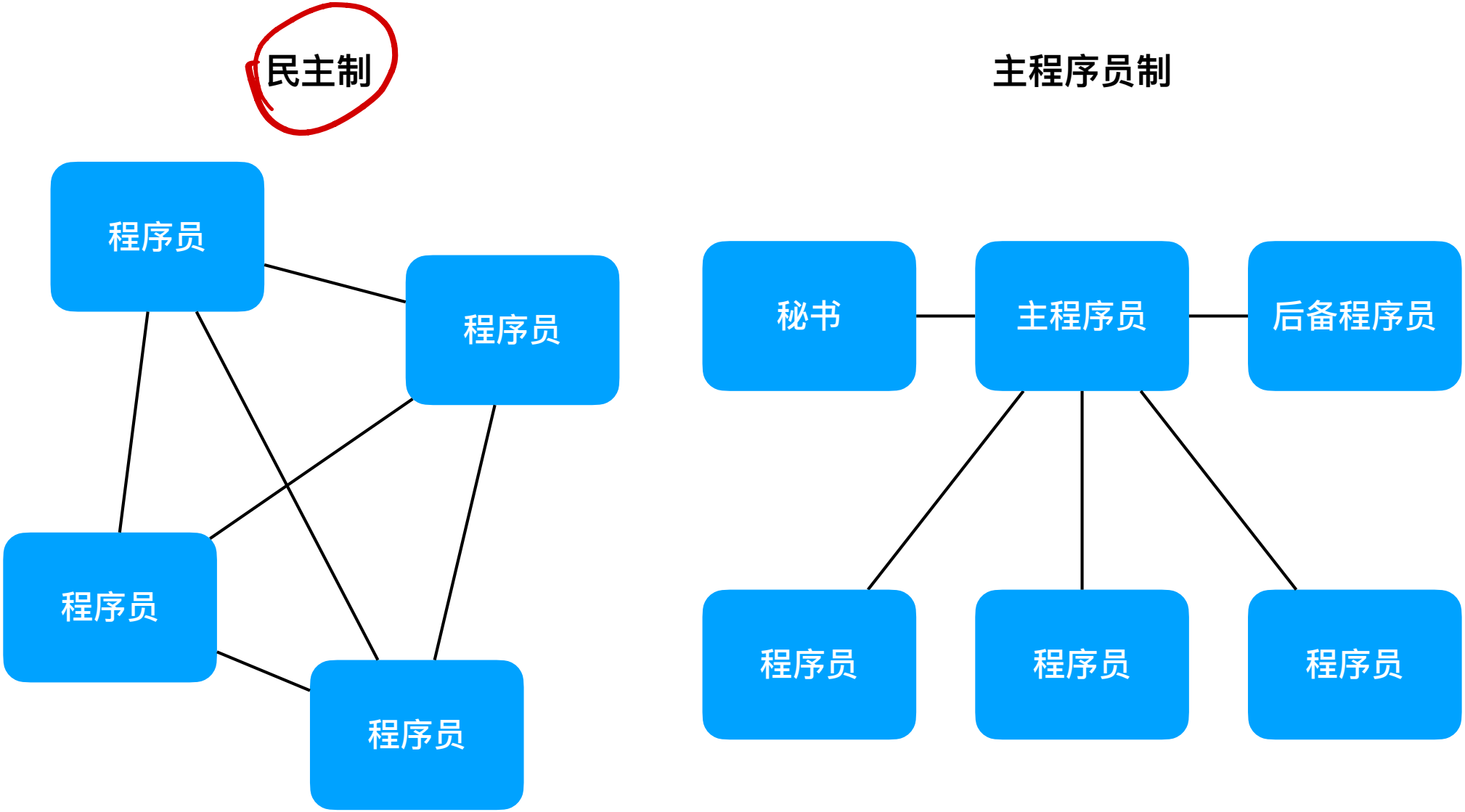
- 任务网络的关键路径
  - 最早完成时间和最晚完成时间相等的任务组成任务网络中的关键路径
  - 关键路径上的持续时间必须严格遵守



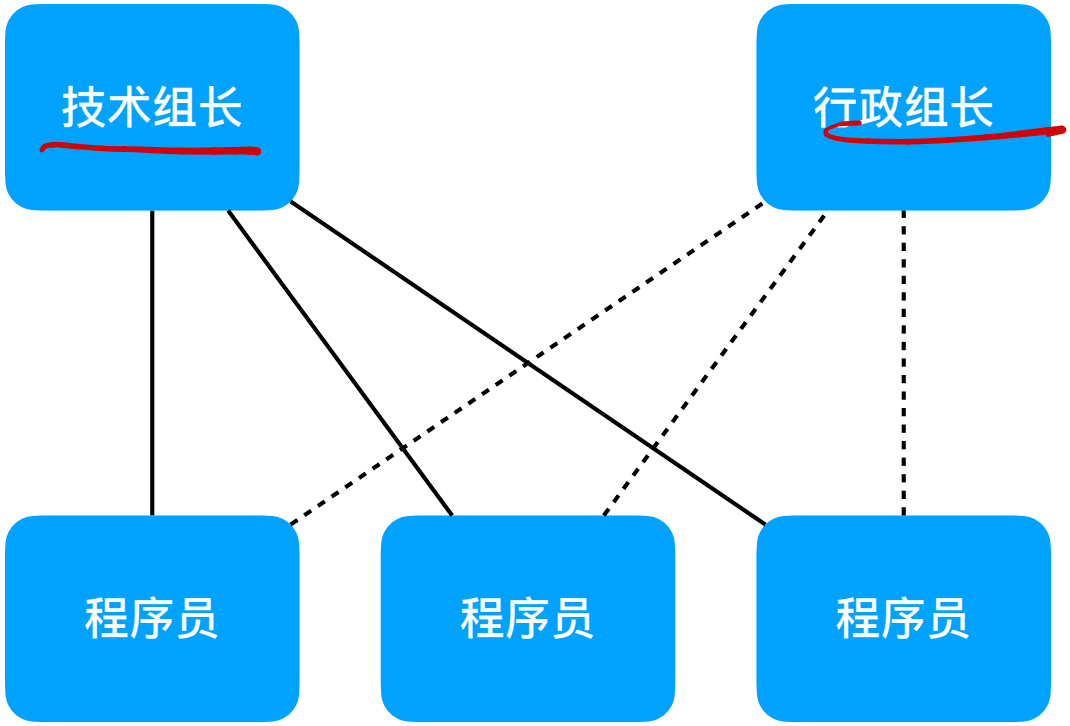
北京科技大学
University of Science and Technology Beijing

人员组织

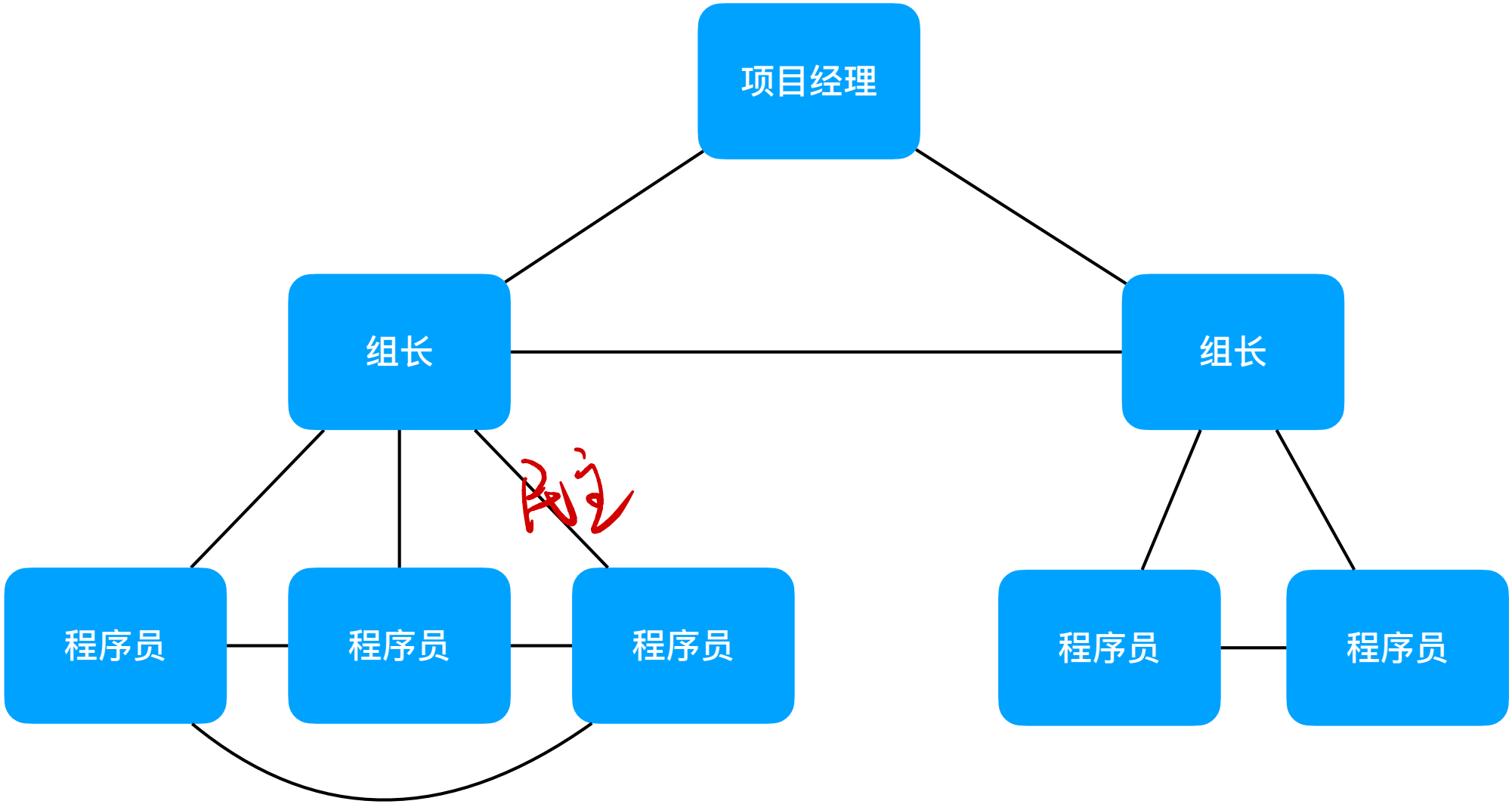
早期的组织方式



现代程序员小组



项目经理制



北京科技大学
University of Science and Technology Beijing

软件配置管理
（变更管理）

软件配置管理

- 主要目标
  - 标识变更
  - 控制变更
  - 正确地实现变更
  - 向相关人员报告变更
- 软件配置的构成（内容）
  - 代码
  - 文档
  - 数据

软件配置管理

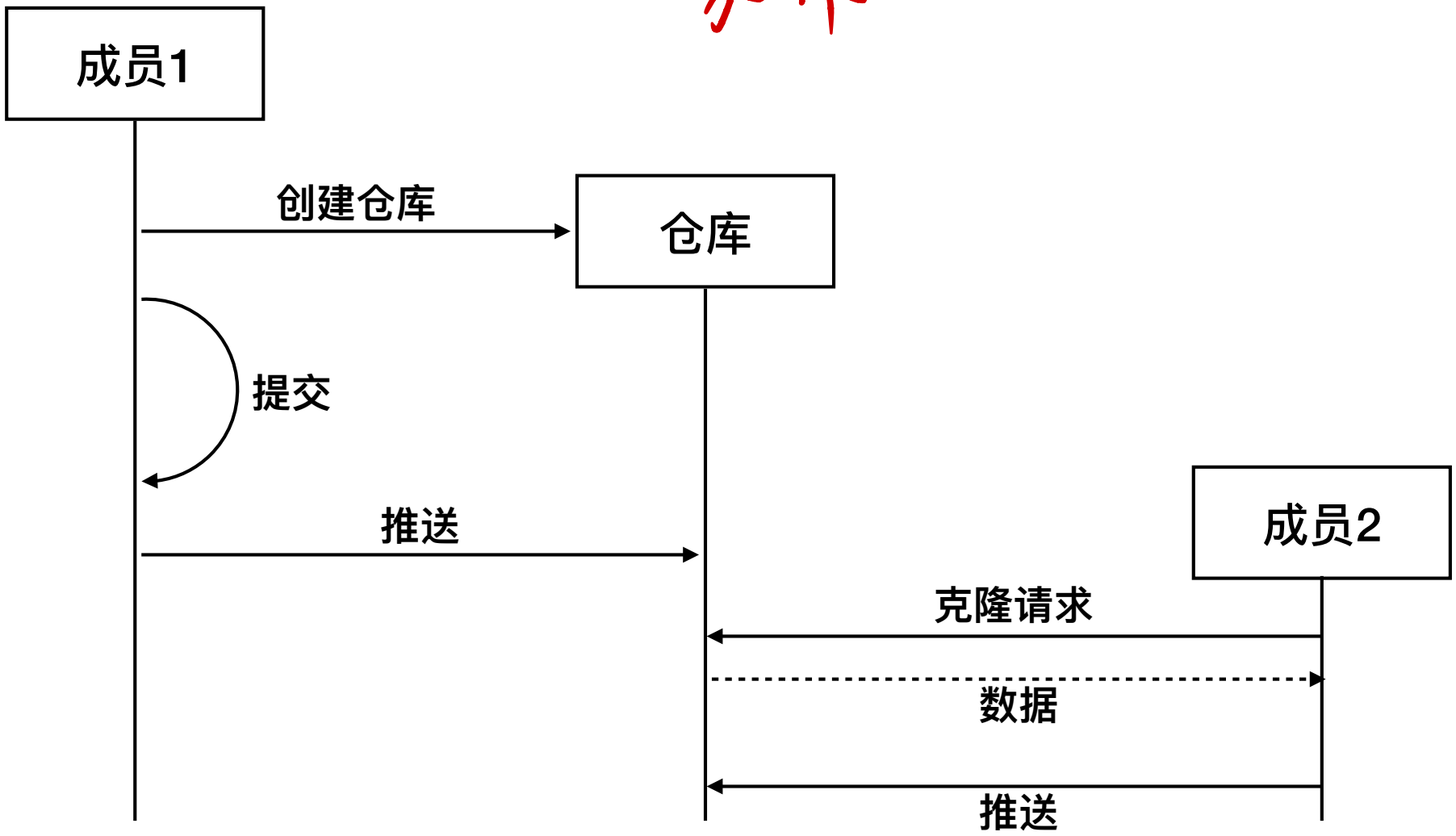
- 基线
  - IEEE 610.12-1990：已经通过正式评审和批准的规格说明或产品，它可以作为进一步开发的基础，并且只有通过正式的变更控制规程才能修改它
- 工具
  - Git
  - SVN
  - CVS

pod ? review

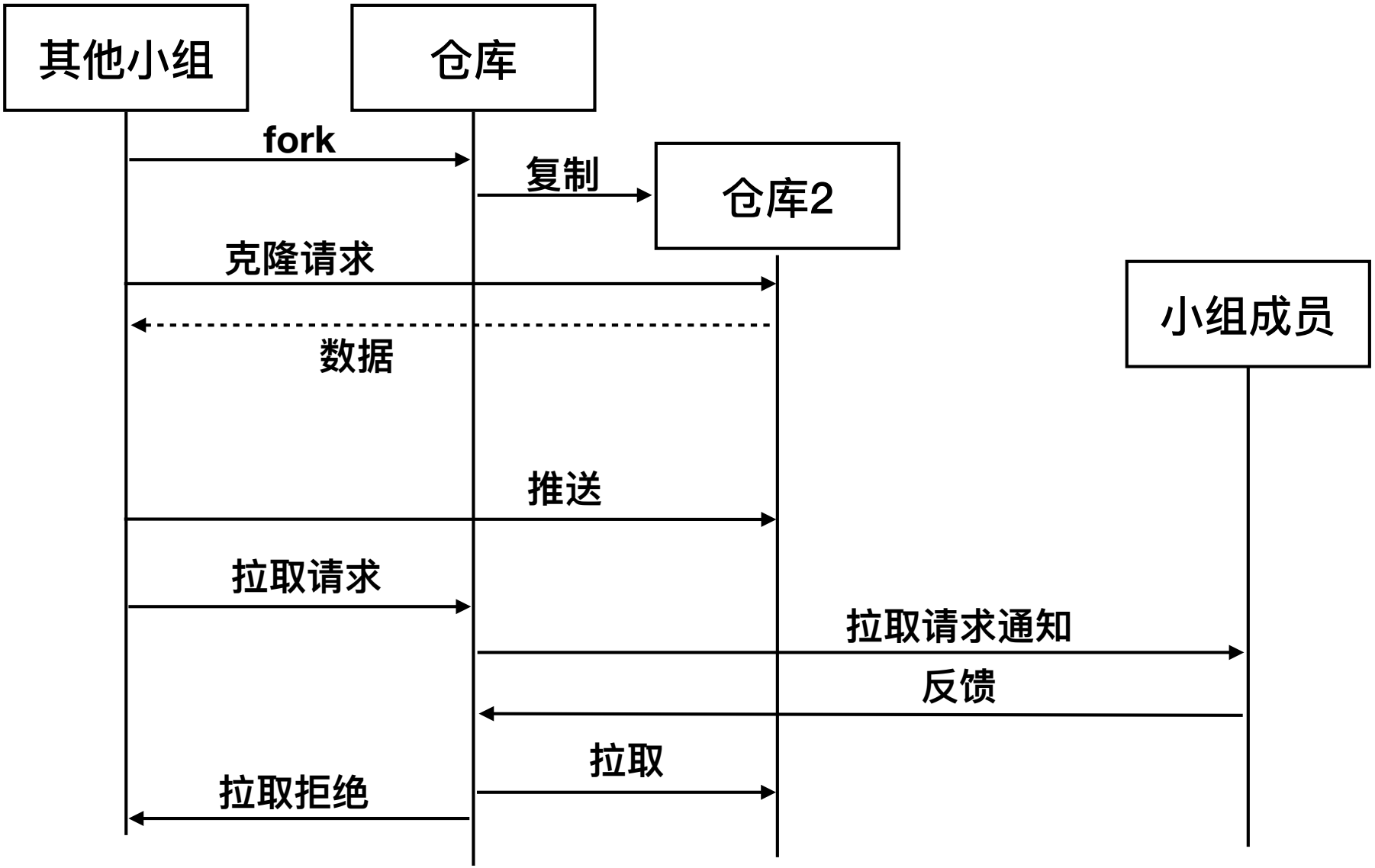
软件配置管理

- 基本过程
  - 标识——对所管理的配置对象命名
  - 版本控制——管理在软件过程中所创建的配置对象的不同版本
  - 变更控制——对需求变更进行评估，并决定是否接受变更
  - 配置审核——对变更后的软件进行审核，判断该变更是否遵循了相应的开发规范
  - 报告——撰写变更报告，主要回答①发生了什么事？ ②谁做的这件事？③这件事是什么时候发生的？④它将影响哪些其他事物？

Git 使用



Git



北京科技大学
University of Science and Technology Beijing

能力成熟度模型

能力成熟度模型

- 美国卡内基梅隆大学软件工程研究所在美国国防部资助下于20世纪80年代末建立的能力成熟度模型（capability maturity model，CMM），是用于评价软件机构的软件过程能力成熟度的模型。

- 能力成熟度模型的基本思想是，由于问题是由我们管理软件过程的方法不当引起的，所以新软件技术的运用并不会自动提高软件的生产率和质量。

- 能力成熟度模型有助于软件开发机构建立一个有规律的、成熟的软件过程。改进后的软件过程将开发出质量更好的软件，使更多的软件项目免受时间和费用超支之苦。

- 软件过程包括各种活动、技术和工具，因此，它实际上既包括了软件开发的技术方面又包括了管理方面。CMM的策略是，力图改进对软件过程的管理，而在技术方面的改进是其必然的结果。

能力成熟度模型

- CMM通过定义能力成熟度的5个等级
  - 初始级（又称为1级）
  - 可重复级（又称为2级）
  - 已定义级（又称为3级）
  - 已管理级（又称为4级）
  - 优化级（又称为5级）

初始级

- 软件过程的特征是无序的，有时甚至是混乱的。几乎没有什么过程是经过定义的（即没有一个定型的过程模型），项目能否成功完全取决于开发人员的个人能力。

- 处于1级成熟度的软件机构，其过程能力是不可预测的，其软件过程是不稳定的，产品质量只能根据相关人员的个人工作能力而不是软件机构的过程能力来预测。

可重复级

- 软件机构建立了基本的项目管理过程（过程模型）,可跟踪成本、进度、功能和质量。已经建立起必要的过程规范，对新项目的策划和管理过程是基于以前类似项目的实践经验，使得有类似应用经验的软件项目能够再次取得成功。

- 达到2级的一个目标是使项目管理过程稳定，从而使得软件机构能重复以前在成功项目中所进行过的软件项目工程实践。

已定义级

- 软件机构已经定义了完整的软件过程（过程模型），软件过程已经文档化和标准化。所有项目组都使用文档化的、经过批准的过程来开发和维护软件。这一级包含了第2级的全部特征。

- 处于3级成熟度的软件机构的过程能力可以概括为，无论是管理活动还是工程活动都是稳定的。

- 软件开发的成本和进度以及产品的功能和质量都受到控制，而且软件产品的质量具有可追溯性。这种能力是基于在软件机构中对已定义的过程模型的活动、人员和职责都有共同的理解。

已管理级

- 软件机构对软件过程（过程模型和过程实例）和软件产品都建立了定量的质量目标，所有项目的重要的过程活动都是可度量的。

- 该软件机构收集了过程度量和产品度量的方法并加以运用，可以定量地了解和控制软件过程和软件产品，并为评定项目的过程质量和产品质量奠定了基础。这一级包含了第3级的全部特征。

优化级

- 处于5级成熟度的软件机构的过程能力可以概括为: 软件过程是可优化的。
- 这一级的软件机构能够持续不断地改进其过程能力，既对现行的过程实例不断地改进和优化，又借助于所采用的新技术和新方法来实现未来的过程改进。

北京科技大学
University of Science and Technology Beijing

关于考试

基本安排

- 返校后统一进行考试
  - 间隔周期长，一定要注意复习
- 考核范围
  - 软件工程基本概念、软件过程、软件需求预可行性分析、结构化方法（数据流图、模块结构图、详细设计的各种工具）、面向对象方法、编码原则、软件度量、软件测试（测试的概念和策略、边界值、等价类、路径测试、软件可靠性）、软件维护的概念、项目管理

北京科技大学
University of Science and Technology Beijing

关于课程设计

基本安排

- 依托gitee.com高校版
  - 所有人必须学会利用git进行协同开发
- 下周一下午上课时间统一布置课设题目
  - 各班班长提前注册gitee.com，并通过邀请链接加入企业（高校）
- 建议提前商量结组方案
  - 原则上4个人一组，最多不超过6个（人数多会有惩罚系数）

关于进一步学习

持续学习

- 访问softlang.cn：
  - 网站会持续发布关于软件工程的内容
- 访问课程中心/百度盘：
  - 教学资源中会更新讲义、书籍，或提供新的练习/讲解
- 练习我和助教：
  - 加入到最新的软件工程研究题目中来
  - 课设阶段还有研究性题目，可练习我们获取/商议题目。