《软件开发过程与项目管理》

Software Development Process and Project Management

任课教师：范国祥

电　　话：0451-86418876-811(O)

13199561265(Mobile)

邮　　箱：fgx@hit.edu.cn

哈工大计算学部

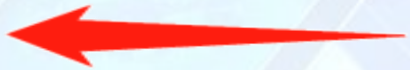
国家示范性软件学院

软件工程教研室

2023. 10

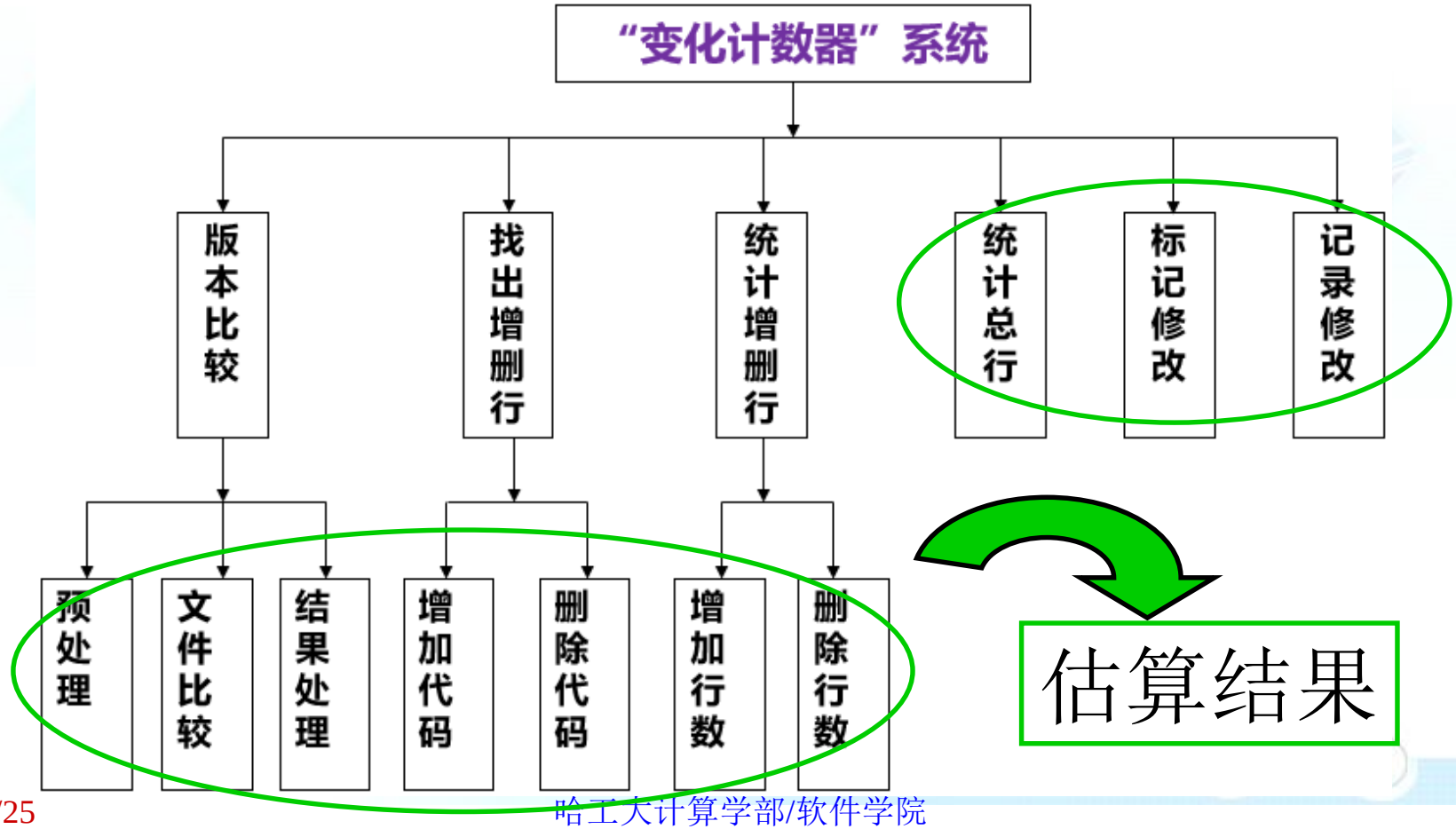
传统估算方法

1. 代码行估算法
2. 功能点估算法
3. 用例点估算法
4. 类比估算法
5. 自下而上估算法
6. 三点估算法
7. 专家估算法



自下而上估算--定义

- 利用任务分解图(WBS)，对各个具体工作包进行详细的成本估算,然后将结果累加起来得出项目总成本



自下而上估算 – 特点

- 相对比较准确，它的准确度来源于每个任务的估算情况
- 花费时间

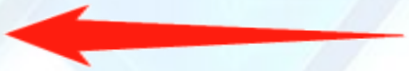
软件项目管理-成本估算

自下而上估算举例

子任务	人力	时间（月）	成本（万元）	总计（万元）
项目准备阶段	M：2/D：8/Q：1	0.5	16.5	
设计阶段	M：2/D：8/Q：2/S：1	1	39	
基础模块开发： 公共控制子系统	M：2/D：15/Q：2/S：1	1.5	90	145.5
中央会计子系统				
客户信息子系统				
基本功能模块开发： 账户管理子系统	M：2/D：12/Q：2/S：1	0.25	12.75	
出纳管理子系统	M：2/D：18/Q：2/S：1	0.5	34.5	126
凭证管理子系统	M：2/D：8/Q：2/S：1	0.25	9.75	
会计核算子系统	M：2/D：18/Q：2/S：1	0.5	34.5	
储蓄子系统	M：2/D：18/Q：2/S：1	0.5	34.5	
扩展功能模块开发： 同城业务子系统	M：2/D：15/Q：2/S：1	0.25	15	
联行业务子系统	M：2/D：18/Q：2/S：1	0.5	34.5	
内部清算子系统	M：2/D：12/Q：2/S：1	0.25	12.75	
固定资产管理子系统	M：2/D：10/Q：2/S：1	0.25	11.25	189.75
信贷管理子系统	M：2/D：18/Q：2/S：1	0.5	34.5	
一卡通业务子系统	M：2/D：18/Q：2/S：1	0.5	34.5	
中间业务子系统	M：2/D：12/Q：2/S：1	0.25	12.75	
金卡接口模块	M：2/D：18/Q：2/S：1	0.5	34.5	
现场联调	M：2/D：10/Q：2	1	42	42
总成本	503.25			

传统估算方法

1. 代码行估算法
2. 功能点估算法
3. 用例点估算法
4. 类比估算法
5. 自下而上估算法
6. 三点估算法
7. 专家估算法



三点估算法

- 基于任务成本的3种估算值(即最有可能成本、最乐观成本、最悲观成本）来加权平均计算预期成本的方法

三点估算法 – 三种估算值

- CM：Most possible Cost，最可能成本，比较现实的估算成本
- CO：Optimistic Cost，最乐观成本，最好情况预期所得到的估算成本
- CP：Pessimistic Cost，最悲观成本，最差情况预期所得到的估算成本
- CE=f(CO,CM,CP)
  CE即Expected Cost，预期成本，通过CO、CM、CP来计算

三点估算结果

(1) 三角分布: CE = (CO + CM + CP) / 3

(2) 贝塔分布: CE = (CO + 4 CM + CP) / 6

三点估算 – 例子

例: 假设 CO = 7, CP = 12, CM = 9 则:

(1) 三角分布: CE = (7 + 9 + 12) / 3 = 9.33

(2) 贝塔分布: CE = (7 + 4×9 + 12) / 6 = 9.17

传统估算方法

1. 代码行估算法
2. 功能点估算法
3. 用例点估算法
4. 类比估算法
5. 自下而上估算法
6. 三点估算法
7. 专家估算法 

专家估算法

- 由多位专家进行成本估算，一个专家可能会有偏见，最好由多位专家进行估算，取得多个估算值，最后得出综合的估算值

专家估算法 – Delphi专家估算法

1. 组织者确定一批专家，规定：这些专家互相不见面
2. 组织者发给每位专家一份软件规格说明（重点WBS）以及估算值记录表
3. 每位专家认真研究软件规格说明书，以无记名方式对该软件给出3个规模的估算值：最小值a_i、最可能值m_i、最大值b_i
4. 组织者整理并计算每位专家估算的结果：

   $E_i = (a_i + 4m_i + b_i) / 6$ 【贝塔分布算法】
5. 最终可以获得一个多数专家共识的软件规模：

   $E = (E_1 + E_2 + \ldots + E_n) / n$ 【n - 专家数量】
6. 如果各个专家的估算差异超出规定的范围（例如：15%），则需重复上述过程

Delphi专家估算法-举例

- 某多媒体信息查询系统，采用专家估算方法

专家1估值：1，8，9

$E_1 = (1 + 4*8 + 9) / 6 = 7$

专家2估值：4，6，8

$E_2 = (4 + 4*6 + 8) / 6 = 6$

估算结果 $E = (6 + 7) / 2 = 6.5$

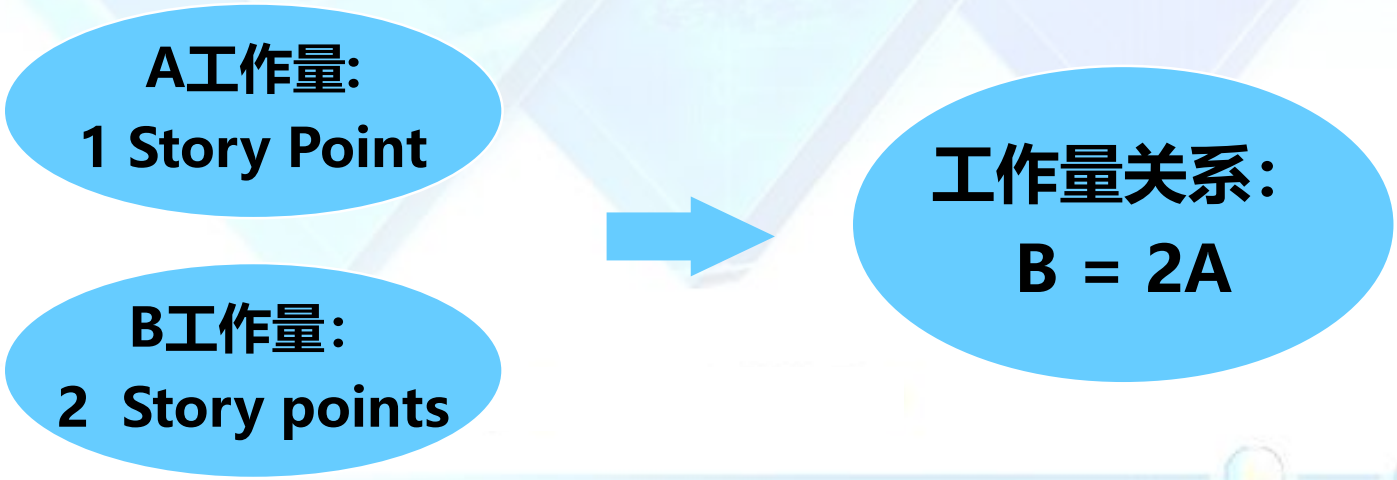
现代的敏捷估算思维

- 采用轻量级估算方法快速生成高层级估算
- 短期规划可以进行详细的估算

Story Point估算方法

- Story Point（故事点）用来度量实现一个Story 需要付出的工作量的相对估算值
- 例如：A任务工作量为1个Story Point， B任务工作量为2个Story Points，则认为：

  B工作量 = 2倍的A工作量



统一估算基准 – 参照基准

- 建立团队共同认可的估算基准值

  例如：用户注册 Story：3

- 其他 Story则以基准值做比对来定义工作量或成本

Story Point估算 – 常用的2个标准

- Fibonacci数列：

  0，1，2，3，5，8，13，21，34，55，89，…

- 2^n数列：

  0，1，2，4，8，16，32，64，128，…

Story Point估算 — Fibonacci标准的7个等级

- 0、1、2、3、5、8、13七个等级

1. 选取预估为3 Story Points 的某个Story，假设名字为A
2. 将需要预估的Story与Story A进行比较
3. 如果2个Story工作量差不多，设置该Story 的Story Point为3
4. 如果工作量略少，则设置为2
5. 如果工作量更少，则设置为1
6. 如果该Story不需要完成，则设置为0
7. 同理，如果略多/更多/再多，可以相应的设置为5/8/13
8. 如果该Story 超过13 Story Points，可以认为是Epic，可以再分解

Story Point估算 — 举例

针对《SPM需求规格》：

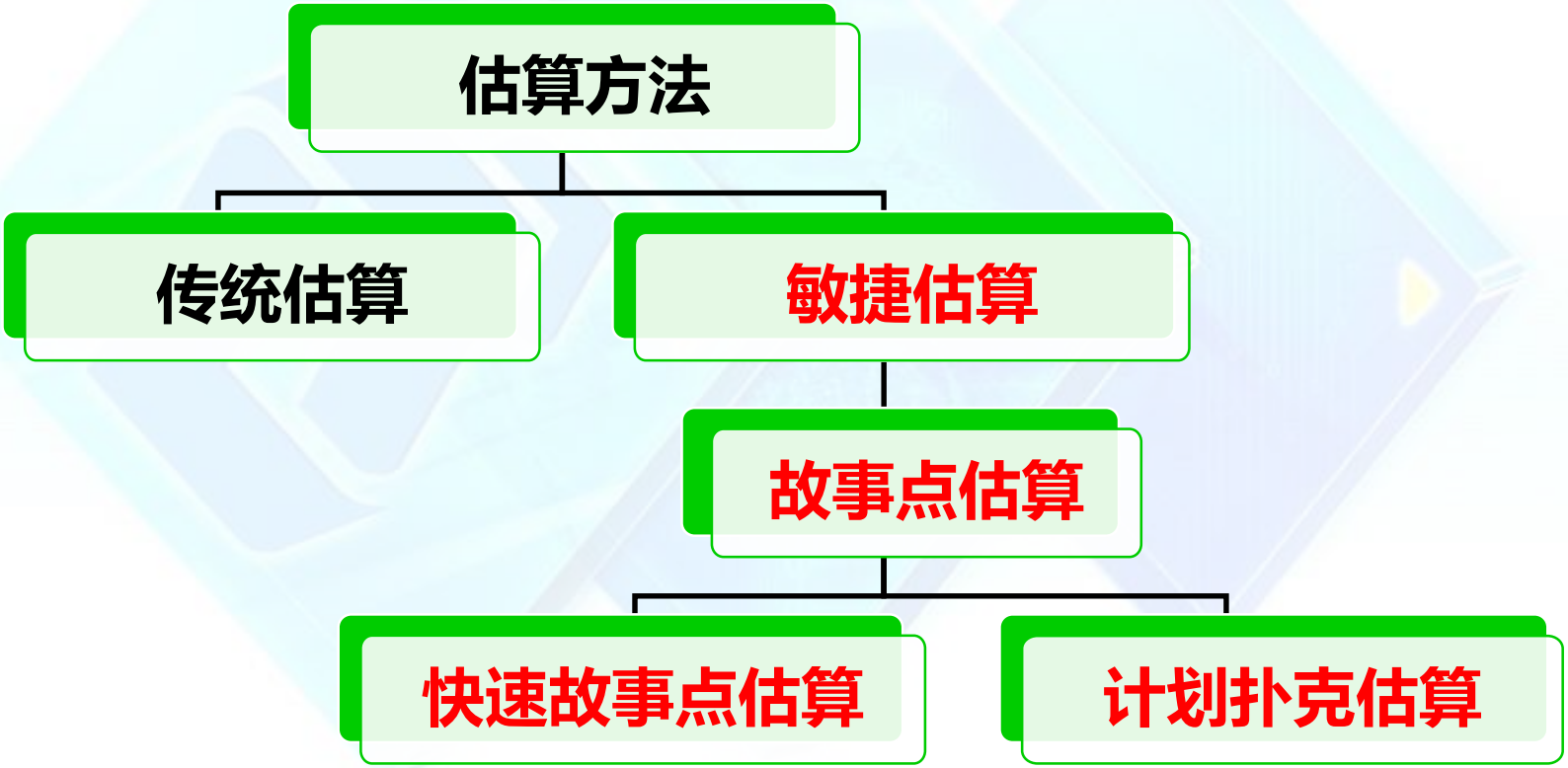
预估：注册功能为3个Story Points

则估算:

登录功能为2个Story Points

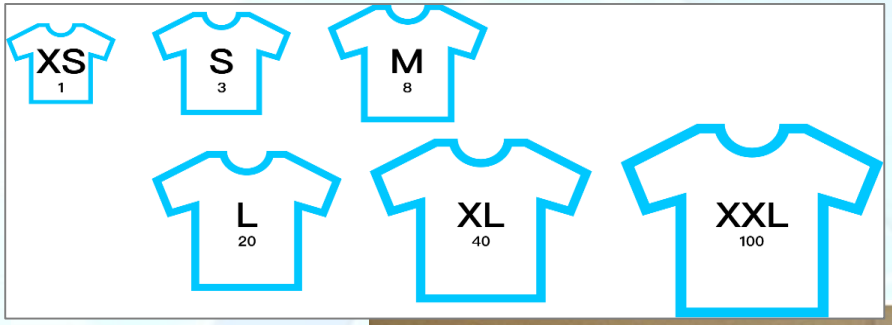
人员管理功能为5个Story Points

敏捷估算技术



快速故事点估算(Fast Story Point Estimation)（T-Shirt)

- 快速、粗略估算；过程有趣，杜绝枯燥、乏味、繁琐



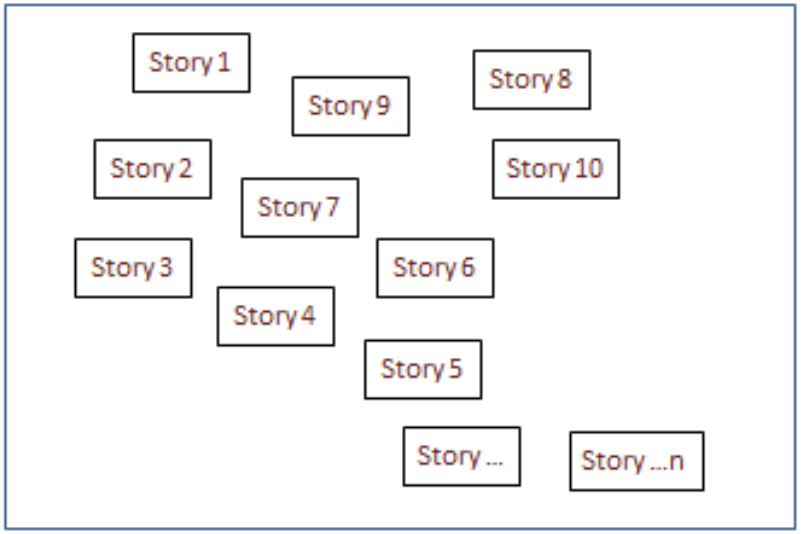
基准估算参考(类似T-Shirt尺码)

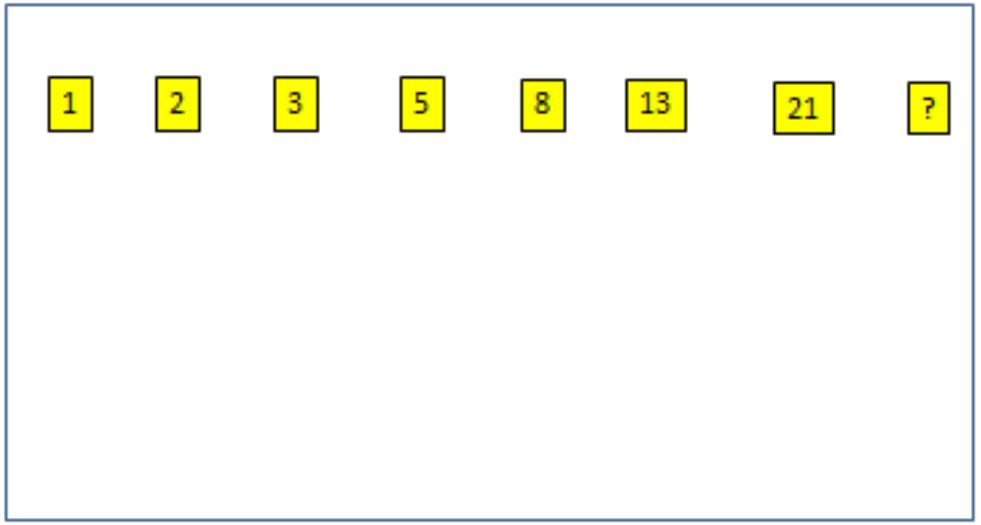
尺寸	工时	点数
很小（Tiny）	8小时（1天）	1
较小（Small）	16小时（2天）	2
中等（Medium）	24小时（3天）	3
较大（Big）	40小时（5天）	5
很大（Large）	64小时以上（8天以上）	8
巨大（Large）	104小时以上（13天以上）	13

快速故事点估算(Fast Story Point Estimation)（T-Shirt)

前提条件：

- 团队成员理解大多数用户故事的要点
- 有同地协作的团队
- 每个用户故事独立打印，必须在估算过程之前准备好，并贴在墙上
- 在墙上写下Fibonacci数列，并加上1列“？”



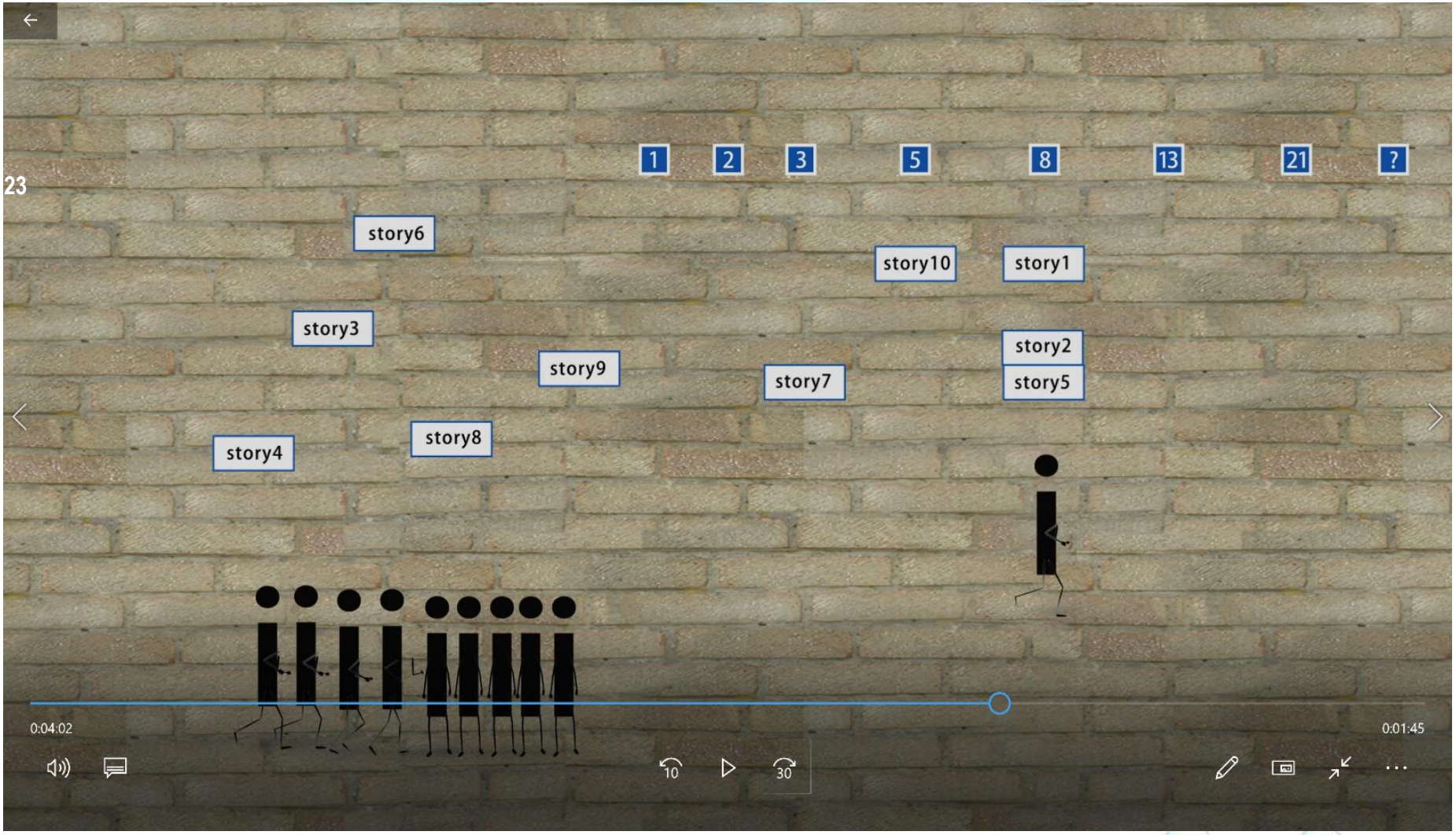


快速故事点估算(Fast Story Point Estimation)（T-Shirt)

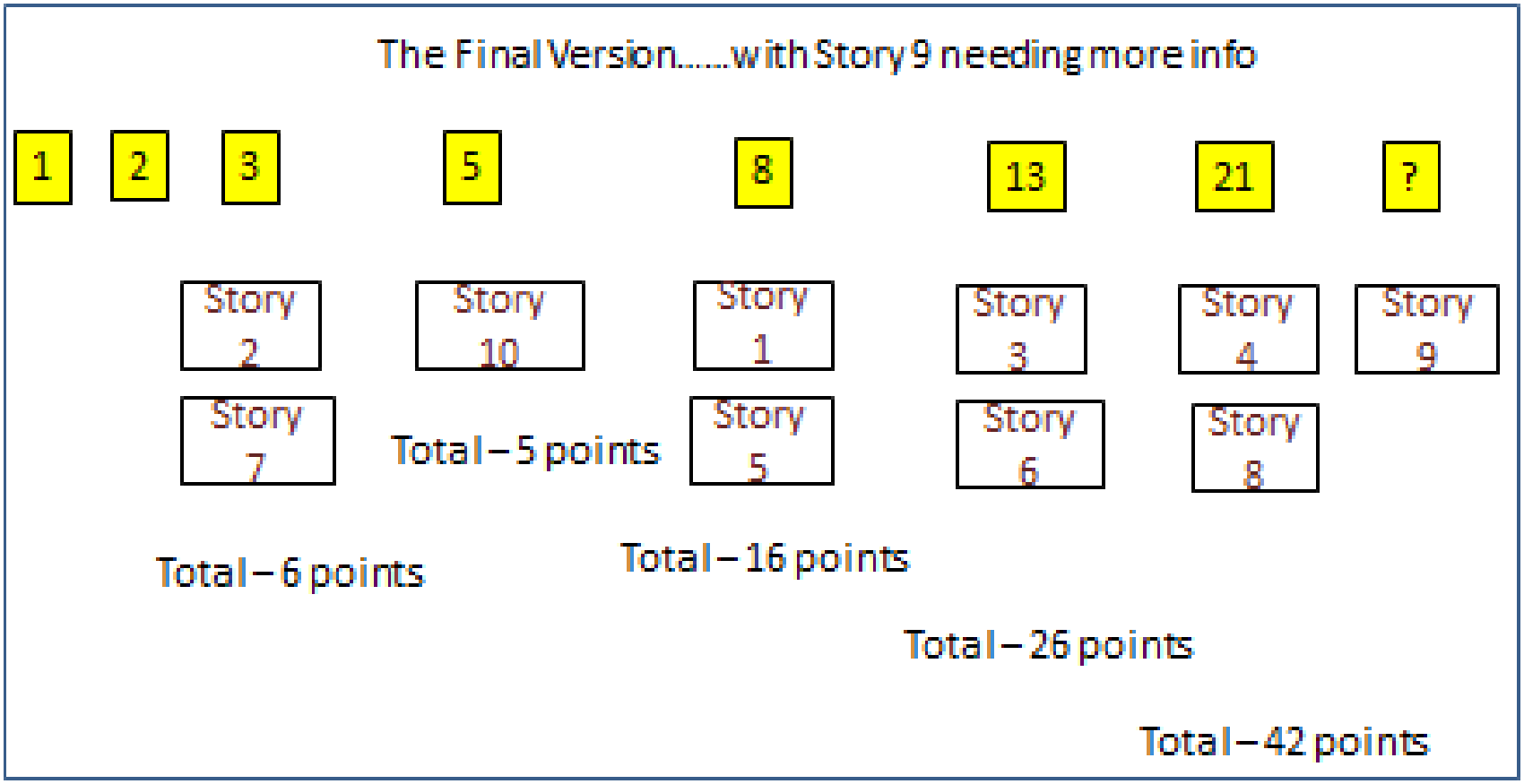
估算过程：

1. 团队人员排成一排
2. 第一名成员把一个用户故事放到他认为可以正确反映故事点值的那一列上
3. 第一名成员做完后排团队成员的最后一个位置
4. 下一个团队成员可以挪动已经摆好的用户故事，也可以选择另外的用户故事，把它挪到他认为可以正确反映故事点值的那一列
5. 继续这个过程，直到所有用户故事都摆放完毕
6. 在此循环过程中，会有用户故事在不同的估值点列中来回挪动，引导师可以把这些用户故事挪到列表的上方，最后专门讨论
7. 当大多数的故事都摆放好后，团队成员投票选择那些“问题”故事属于哪一列
8. 如果无法达成一致，把这个故事放到最高值的一列
9. 一旦团队成员对放置的故事都满意，计算每一列故事的个数，并且乘以故事点值，从而得到所有的故事点

快速故事点估算(Fast Story Point Estimation)（T-Shirt)



快速故事点估算(Fast Story Point Estimation)（T-Shirt)



Grand Total – 95 points with one outstanding story to estimate

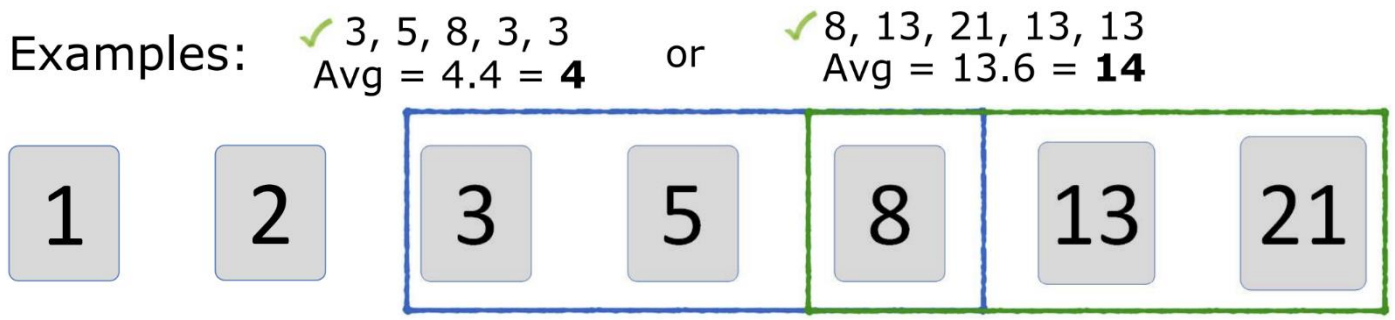
If the team's velocity is 30 points/sprint, it should take 3+ sprints to complete all stories

计划扑克估算（Planning Poker）

- 详细估算：可以设定相应规则

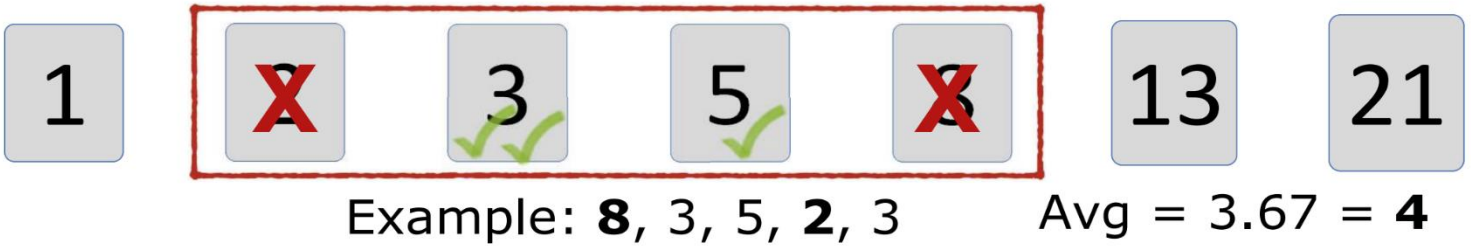
计划扑克估算 – 自设规则

- 每个人独立思考出什么牌
- 当每个人都表示准备就绪时，Scrum Master 喊“3，2，1-亮牌”
- 每个人都同时出牌
- 如果所有估算值均在三个连续数内，则取平均值



计划扑克估算 – 自设规则

- 如果所有估算值均超出了三个连续数，那么最高和最低估值的要说明理由，然后重新投票
- 可以再重复投票最多两次，直到落到三个连续值内，则取平均值
- 如果重复投票3次，仍然不能落在三个连续值内，去掉最高和最低估值，其余取平均值。



- 针对每个用户故事，都要完成上述估算过程

估算方法总结

本章小结 –核心之“道”

规模估算 → **工作量估算** → **成本估算**

规模估算	工作量估算	成本估算
代码行估算法	类比估算法	类比估算法
功能点估算法	自下而上估算法	自下而上估算法
用例点估算法	参数估算法(未讲)	参数估算法(未讲)
故事点方法	专家估算法	专家估算法
	三点估算法	三点估算法

- 传统估算 — 绝对值估算
- 敏捷估算 — 相对值估算