软件工程综合实验基本要求

北京航空航天大学软件工程研究所
刘超

2016.3

刘超：liuchao@buaa.edu.cn
电话：(10)82317641

内容提要

- 概述
- 实验准备
- 实验1~8
- 实验总结
- 方法与实践的定义与描述
- 例1~7

概述

- 实验目的
  - 通过在限定条件下，对特定样例的实践，检验和验证特定方法的效果和特点，如有效性或局限性；比较不同方法的差异；展示样例的特性
  - 实验要求
- 实验设计
  - 实验样例：软件的初始状态，即原始材料
  - 实验条件：人员和团队，时间约束，工作环境
  - 实验方法：原理或原则，实验步骤，处理策略
  - 实验检验：检验方法与指标
  - 实验报告：实验结果分析，实验方法评价

- 实验实施
  - 学习与理解、计划与监控、建模与验证
- 结果分析
  - 分析、检验、报告

实验准备：软件项目计划

- 成立小组：3~4人
  - 责任分工
  - 小组协同
    - 小组会议：每周1~2次；计划，交流，检查
    - 网上协作：随时随地；研讨、交互
- 选择项目：开源项目
  - 2014~2015年已经做过的项目：有参考
  - 自选
- 选择方法：针对各项任务及其完成情况的分析和检验
- 制定计划：分解任务、安排进度、明确分工
- 收集资料：有关项目的资料；有关方法的文献

实验1：软件需求分析

- 实验目的
  - 学习并有效运用现有的软件需求分析方法
  - 发现并探讨解决存在的典型问题
- 实验方法
  - 学习和综合运用1~2种有效的软件需求获取、描述和分析方法，确定软件需求
    - UML：RUCM
  - 提交规范的软件需求规格说明文档，包括必要的业务逻辑模型和相关定义等

- 软件需求分析（过程要点）
  - 获取并说明软件需求
    - 当前版本满足的用户需求
    - 源代码、网上介绍、已有需求说明、使用手册等
  - 提出新需求（改进、扩展、检验）
    - 目标：典型应用场景
  - 细化并严谨地定义指定的软件需求
    - 建模 + 数据字典 + 规范说明
    - 需求分析：追踪关系
  - 需求验证
    - 完整性、一致性等
  - 需求评审（实验2.评审和实验2.复评审）
    - 检查单
  - 需求修改和完善：依据评审意见

需求工程：
需求获取、分析、验证

新要求

UML/Use Case
(RUCM)+DD➔ Req. Spec.

北航软件所

- 实验要求
  - 明确目标：典型应用场景，即用户及其业务需求
  - 主要方法：获取和分析需求的方法、策略和准则
  - 实验步骤：工作计划
  - 软件需求：功能性需求和非功能性需求
  - 标准规范：行业标准、产品规范、业务规范、文档规范
  - 来源依据：参考资料及其来源，选择策略和依据，追踪关系

讨论：软件需求及其获取和描述方法

- 什么是软件需求？
  - 区别：用户需求、软件功能
- 需求分析方法和过程？
  - 如何获得和定义（或描述）软件需求？
    - 来源和依据；特征和特性（模型）
  - 如何确认？
    - 原则、策略、验证
  - 如何更正、完善和演化？
- 现有方法的能力与不足？适用性？

实验2：软件需求评审

- 实验目的
  - 学习并有效运用现有的软件需求评审方法，发现并探讨解决存在的典型的问题
- 实验方法
  - 学习和综合运用1~2种有效的软件需求评审方法
  - 定义提交软件需求评审检查单
  - 采用规范方法进行软件需求评审
    - 提交软件问题报告
  - 需求修订与复评审
    - 提交软件需求评审报告

- 要求：
  - 明确目标：软件需求满足用户需求，文档符合规范要求
  - 主要方法：软件需求评审和验证的方法、策略
  - 检查单：检查要点
  - 评审和验证报告：评审和验证结果，需求问题
  - 标准规范：行业标准、产品规范、业务规范、文档规范
  - 来源依据：参考资料及其来源，选择策略和依据，追踪关系

- 网上互评审
  - 每个同学独立评审所有其他组提交的需求规格说明书和相关资料
  - 评分：0-10分
  - 评审意见：软件问题报告
- 会议评审
  - 课堂评审
  - 所有师生
  - 小组介绍
  - 其他小组提问
  - 老师提问
  - 小组总结：归纳问题

北航软件所

讨论

- 软件问题检查单
  - ？
- 软件问题报告单
  - ？

- 网上评分（0-10分）
  - 内容完整性
    - 核心/基本：目标一致/符合
    - 特殊（异常处理）、辅助
  - 说明易理解
    - 清楚、易理解
    - 具体、详尽（细化）
  - 模型准确性
    - 规范、合理
    - 严谨、准确
  - 扩展、改进
    - 目标：明确、合理、可行
    - 内容：具体、充实、可展现

北航软件所

参考资料

- Kotonya, *Requirements Engineering: Methods and Techniques*
- Klaus Pohl (德)著，需求工程:基础、原理和技术，机械工业出版社，2012
- Nancy G. Leveson，Requirements Engineering for Software Integrity and Safety，2002
- (英) Ian Sommerville, Pete Sawyer著，需求工程: a good practice guide，机械工业出版社，2003
- Sven J. K'ORNER,Torben Brumm，Natural Language Specification Improvement With Ontologies, International Journal of Semantic Computing, Vol. 3, No. 4 (2009) 445–470

北航软件所

实验3：软件测试需求分析

- 实验目标
  - 覆盖测试
    - 需求（场景）覆盖、功能点覆盖、语句/分支覆盖
  - 问题域估计与测试
    - 发现问题
  - 快速构建一个演示原型
    - 功能/非功能特性、缺陷修复验证
- 方法定义和选择
  - 测试需求建模
  - 测试用例设计原则和策略
  - 测试工具和测试自动化

- 软件测试（过程要点）
  - 测试需求定义和测试用例设计
    - 软件测试需求模型（RTCM）
    - 软件测试需求规格说明书
  - 测试需求和测试用例评审（实验4.测试需求和测试用例评审）
  - 选择并学习使用测试工具，生成并执行测试脚本
    - Junit，Javascript, Perl，etc
  - 充分性分析与增强测试
    - 覆盖准则
    - 问题域分析
  - 测试结果评审（实验4.测试结果评审）
    - 软件问题分析

实验4：软件测试评审

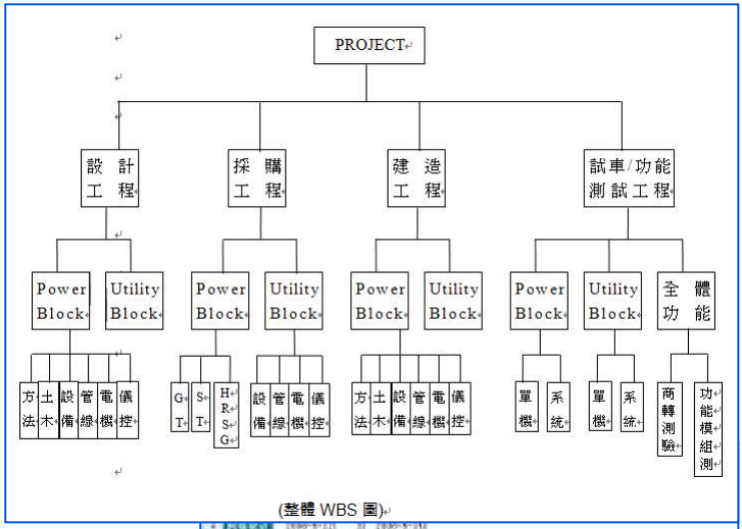
- 类似于软件需求评审方法和要求
- 评审对象
  - 软件测试需求规格说明书
  - 软件测试用例
  - 软件测试结果，软件问题报告及其修复结果
- 评审依据
  - 检查单

实验5：软件产品更新与展示

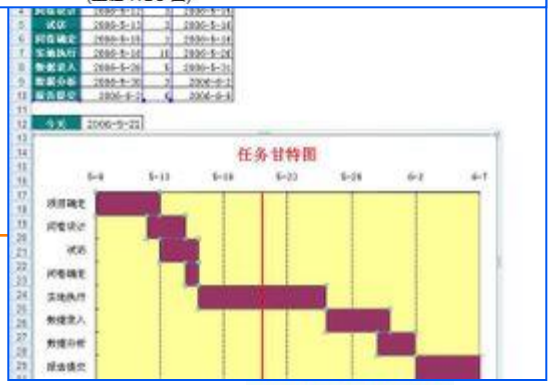
- 实验选择
  - 版本更新：软件新功能和新特性的需求和测试
  - 产品比较：同类产品的需求对比和模拟展示
  - 产品扩展：产品扩展和模拟演示
  - 产品应用：典型应用分析与模拟演示
- 比如
  - 加入或改进算法
  - 加入新属性或参数
  - 基于现有基本功能的组合，实现或增加新功能
  - 展示和对比功能及其特性的优势和差异

实验6：软件进度计划与控制

- 实验方法与工具
  - 任务分解和分配：WBS（工作分解结构）
  - 进度计划：GANTT
  - 进度监控：任务细化与调整
  - 小组协同：
    - 小组会议研讨
      - 0.5~1.5小时
      - 主题明确
      - 待定问题及负责人列表
    - 网上交流
      - 每日报告
      - 组长汇总



(整體 WBS 圖)



实验7：工作量估计与统计分析

- 实验方法与工具
  - 软件复杂度分析
    - 代码行（KOL）、圈复杂度
    - 模型复杂度：模型图数量、模型元素数量
    - 功能点：
    - 文档页数
  - 人员能力估计
  - 各项工作的工作量估计（人时）
    - 学习、分析和理解时间估计
    - 个人承担的各项具体任务的时间估计
    - 会议时间估计
  - 工作日志
    - 工作项：日期、起止时间、人时数、说明

北航软件所

工作量度量与评估

- 每日/周（计划或进行或完成）的工作：
  - 工作编号、工作名称、类型
  - 工作复杂度度量/评价
  - 工作量（完成量，小时数）
  - 新增/调整的工作
  - 完成质量度量/评价
  - 相关的软件问题报告（评审、测试）
  - 修改情况（次数、修改量、修改工作量等）

参考文献

- [美]Barry W.Boehm等著，李师贤等译，软件成本估算—COCOMO II模型方法，机械工业出版社
- 中华人民共和国行业标准，软件研发成本度量规范（SJ/T 11463-2013），中华人民共和国工业和信息化部，2013

北航软件所

实验8：配置管理

- 实验方法与工具
  - SVN
  - 配置管理计划
  - 配置管理责任分配
  - 配置管理监督

实验总结

- 实验准备
  - 做了哪些准备?是否必要?
  - 存在问题及其影响分析
- 实验1~8
  - 实验目标和方案设计合理性分析
  - 实验实施过程与结果分析
  - 存在问题及其影响分析
- 总结
  - 是否达到了预定的实验目标
  - 收获
- 建议

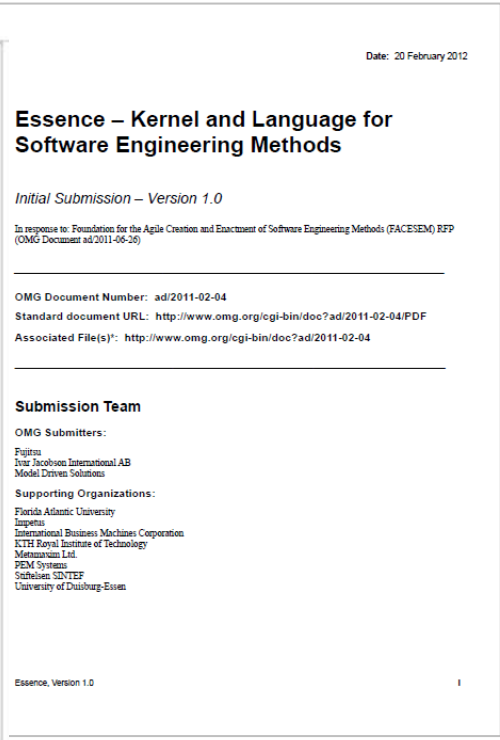
软件工程方法与实践

Dr. Ivar Jacobson

Dr. Bertrand Meyer

Dr. Richard Soley
Chairman & CEO, OMG

- 软件工程的本质与内核是什么？
- 要解决的问题
  - 基本问题及其方法？
  - 方法的选择与运用？
  - 实践的总结与提炼？
  - 方法与实践的定义？



Date: 20 February 2012

Essence – Kernel and Language for Software Engineering Methods

Initial Submission – Version 1.0

In response to: Foundation for the Agile Creation and Enactment of Software Engineering Methods (FACESEM) RFP (OMG Document ad/2011-06-26)

OMG Document Number: ad/2011-02-04
Standard document URL: http://www.omg.org/cgi-bin/doc?ad/2011-02-04/PDF
Associated File(s)*: http://www.omg.org/cgi-bin/doc?ad/2011-02-04

Submission Team

OMG Submitters:

Fujitsu
Ivar Jacobson International AB
Model Driven Solutions

Supporting Organizations:

Florida Atlantic University
Impetus
International Business Machines Corporation
KTH Royal Institute of Technology
Metamaxim Ltd.
PEM Systems
Stiftelsen SINTEF
University of Duisburg-Essen

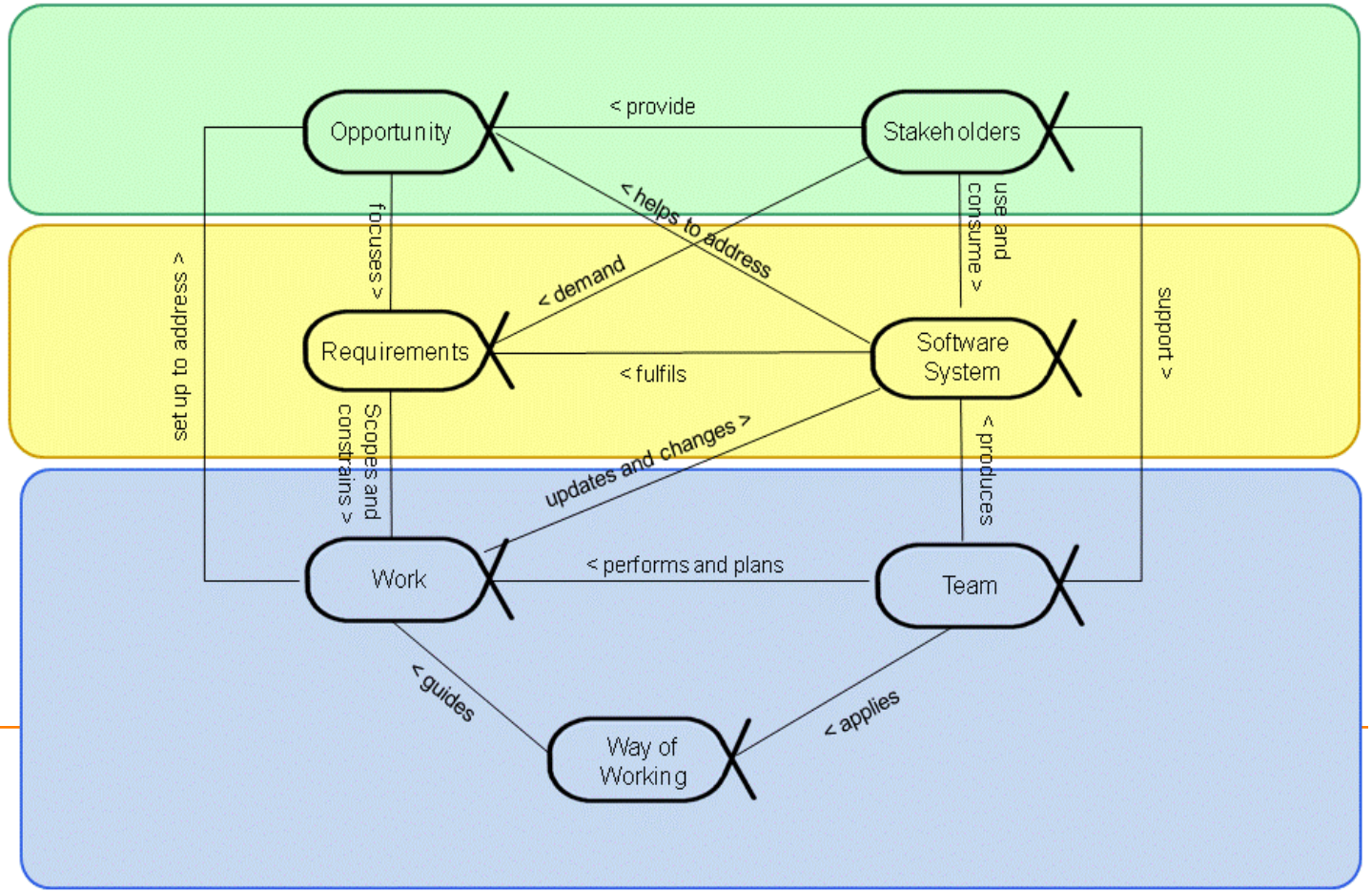
Essence, Version 1.0

SEMAT

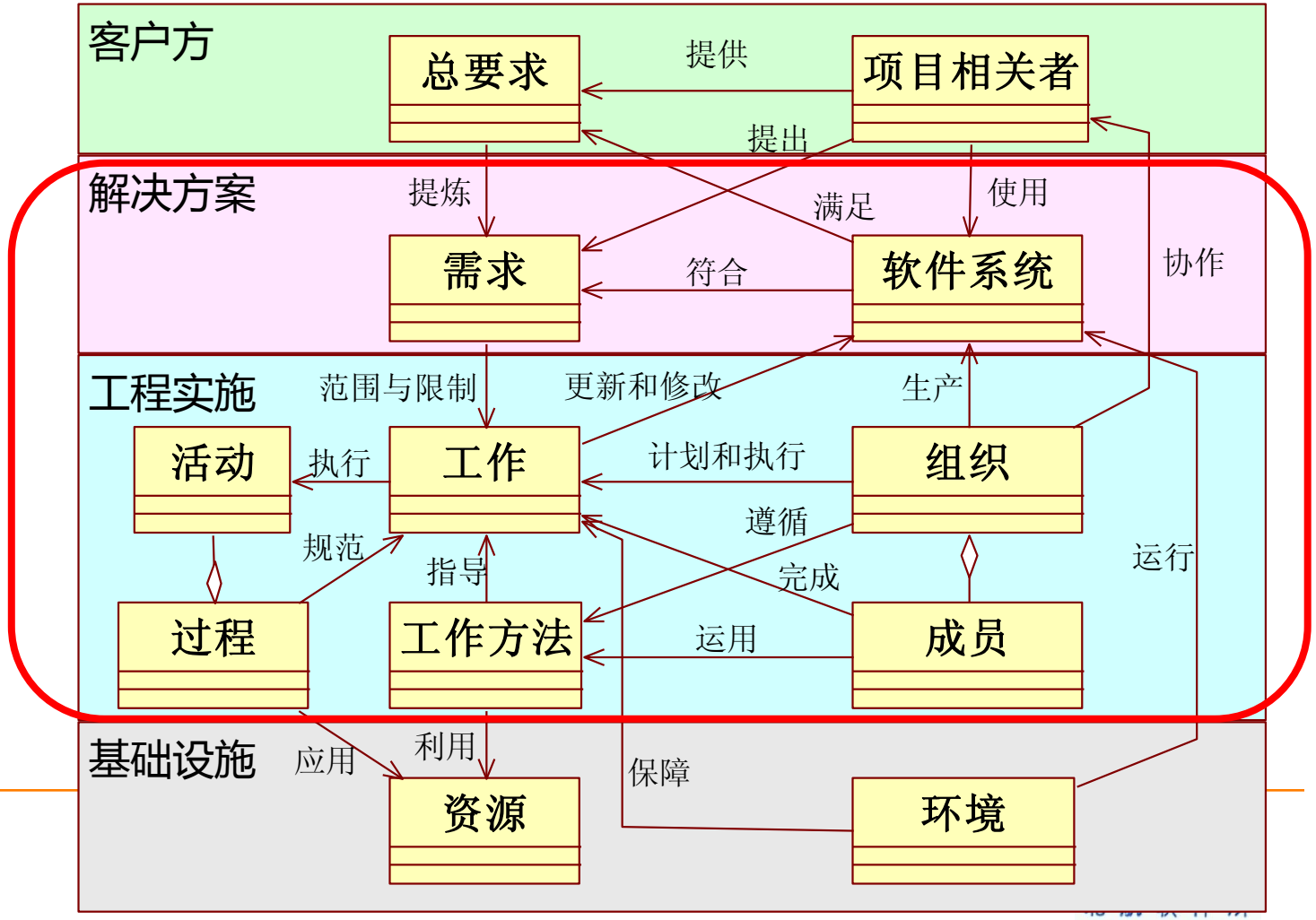
Software Engineering Method and Theory

Jacobson, Meyer, & Soley "Call for Action: The Semat Initiative" *Dr. Dobb's Journal, December 10, 2009*

- Semat
  - Alpha: “The Things to Work With”



- 软件工程过程和基础设施的要件（Seki）和结构



方法与实践的定义与描述

- 方法（method;way;means）
  - 指为达到某种目的而采取的途径、步骤、手段等。
  - 方法的含义较广泛，一般是指为获得某种东西或达到某种目的而采取的手段与行为方式
- 实践（Practice）
  - 诸多的含义，经典的观点是主观见之于客观，包含客观对于主观的必然及主观对于客观的必然
- 最佳实践（Best Practice）

有效实践

  - a method or technique that has consistently show[illegible]ved with other means, and that is used as a benchmark. In addition, a "best" p[illegible]e to become better as improvements are discovered. Best practice is considered by some as a business buzzword, used to describe the process of developing and following a standard way of doing things that multiple organizations can use.
  - Best practice is a feature of accredited management standards such as ISO 9000 and ISO 14001

方法和有效实践的定义

- **Practice**
  - A repeatable approach to doing something with a specific purpose in mind.
  - A practice provides a systematic and verifiable way of addressing a particular aspect of the work at hand. It has a clear goal expressed in terms of the results its application will achieve. It provides guidance to not only help and guide practitioners in what is to be done to achieve the goal but also to ensure that the goal is understood and to verify that it has been achieved.

Practice

- **Package: Foundation**
- **isAbstract: No**
- **Generalizations: "ElementGroup“**
- **Description**
  - A practice is a description on how to handle a specific aspect of a software engineering endeavor. A practice is an element group that names all Essence elements necessary to express the desired guidance. A practice can be defined as a composition of other practices.

Practice

- **Semantics**
  - A practice addresses a specific aspect of development or teamwork. It provides the guidance to characterize the problem, the strategy to solve the problem, and instructions to verify that the problem has indeed been addressed. It also describes what supporting evidence, if any, is needed and how to make the strategy work in real life.
  - A practice includes its own verification, providing it with a clear goal and a way of measuring its success in achievingthat goal.

Practice

- Different kinds of practices
  - Development Practices – such as practices for developing components, designing user interfaces, establishing an architecture, planning and assessing iterations, or estimating effort.
  - Social Practices – such as practices on teamwork, collaboration, or communication.
  - Organizational Practices – such as practices on milestones, gateway reviews, or financial controls.

Practice

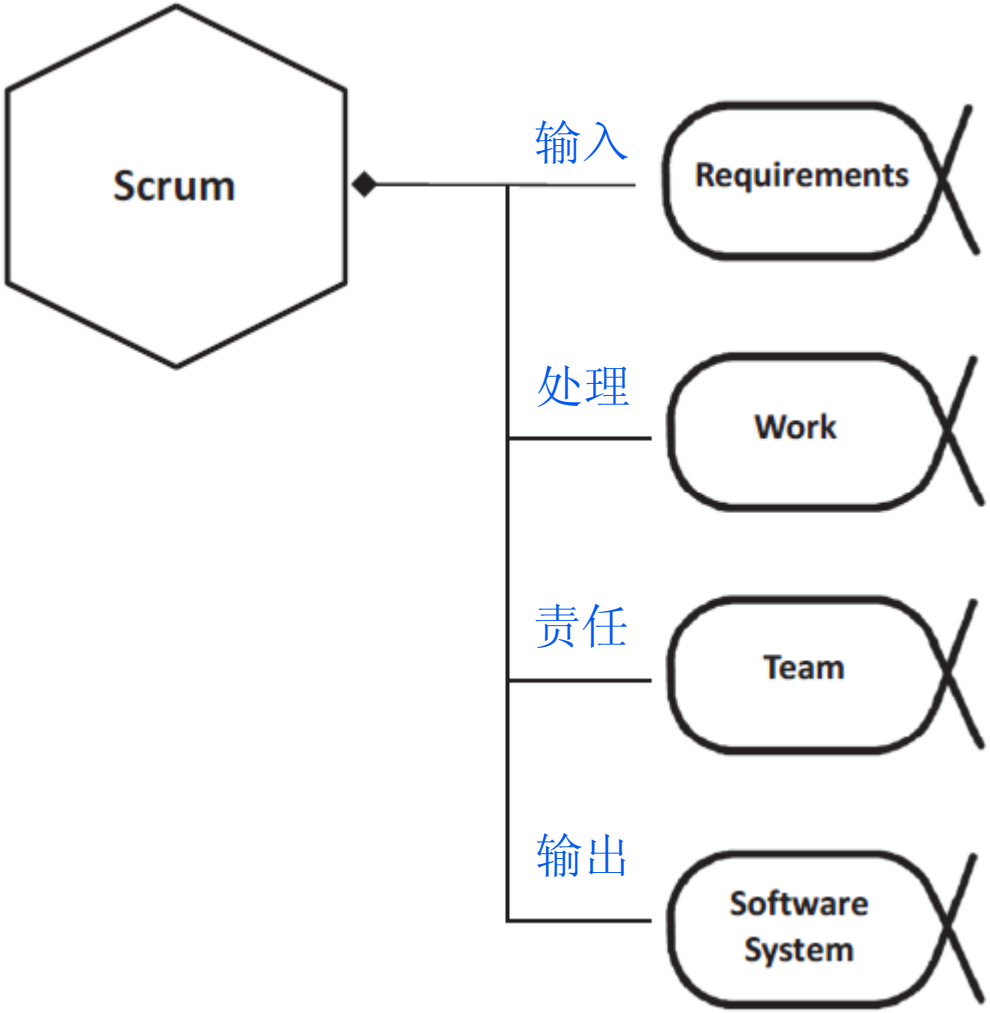
- addresses only one aspect of how to perform a development effort
- To achieve a complete description, practices can be composed
- In this way, more complete and powerful practices can be created, eventually ending up with one that describes how an effort is to be performed, i.e. **a method**.

Ex. **Scrum**[Schwaber and Sutherland 2011]

- Scrum concepts
  - Scrum team (roles)
    - Product Owner
    - Development Team (of developers)
    - Scrum Master
  - Scrum events
    - The Sprint
    - Sprint Planning Meeting
    - Daily Scrum
    - Sprint Review
    - Sprint Retrospective
  - Scrum artifacts
    - Product Backlog
    - Sprint Backlog
    - Increment

北航软件所

- Scrum Practice



- # Requirement

Definition:

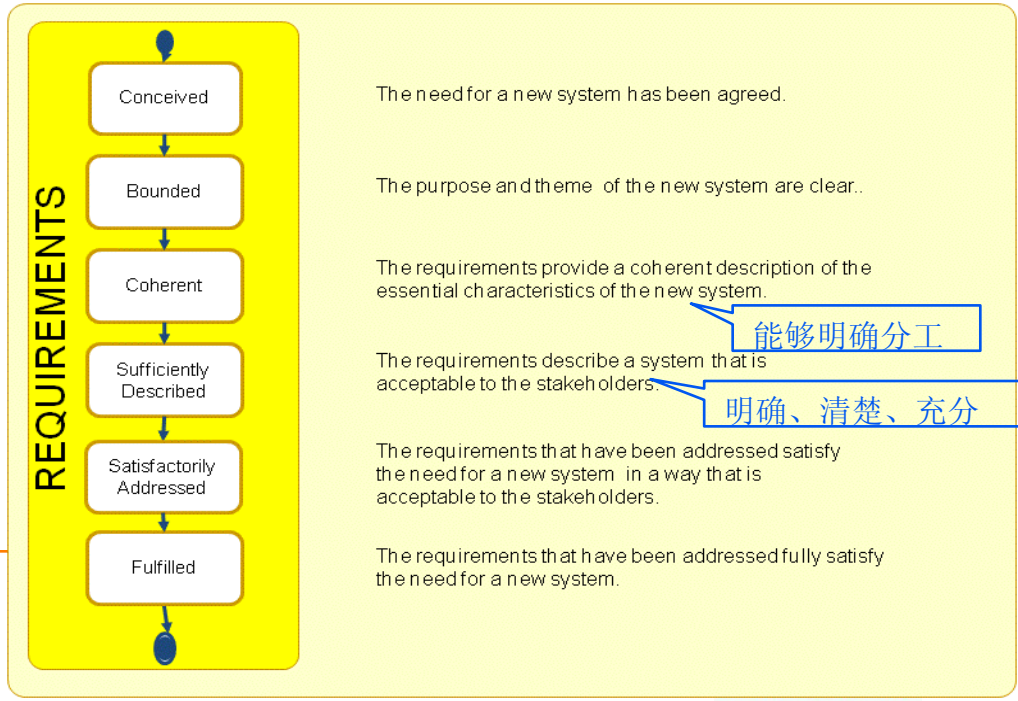
What the software system must do to address the *opportunity* and satisfy the *stakeholders*

It is important to discover what is needed from the software system, share this understanding among the stakeholders and the team members, and use it to drive the development and testing of the new system

Associations:

scopes and constrains : Work

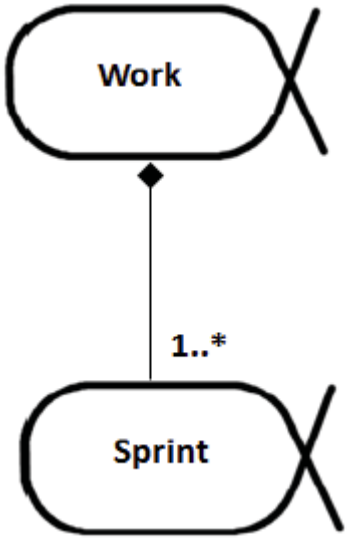
The Requirements scope and constrain the Work



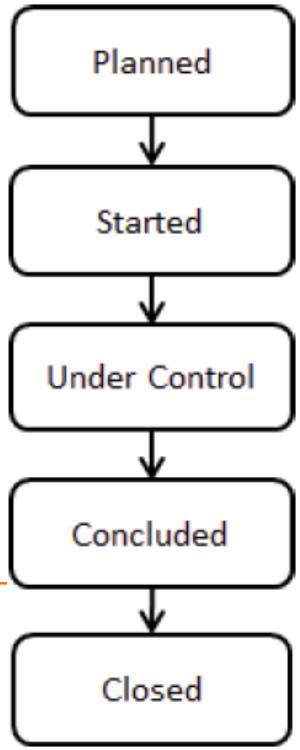
- **Work**
  - Extension of the Work alpha, called as a sub-alpha used for the duration of a development project that may cover a number of sprints.

"The heart of Scrum is a Sprint, a time-box of one month or less during which a "Done", useable, and potentially releasable product Increment is created. Sprints have consistent durations throughout a development effort. A new Sprint starts immediately after the conclusion of the previous Sprint." [Schwaber and Sutherland 2011]

Sprint sub-alpha of Work



The states of the Sprint sub-alpha

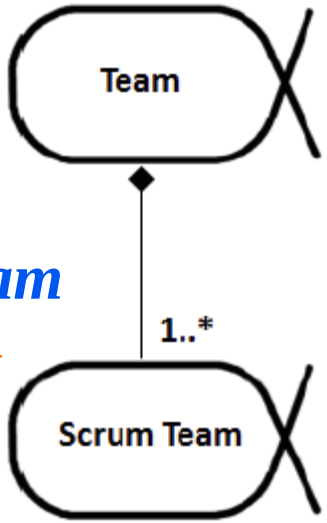


- **Team**
  - The Scrum practice relates to the Team alpha

"The Scrum Team consists of a Product Owner, the Development Team, and a Scrum Master. Scrum Teams are self-organizing and cross-functional.

- Self-organizing teams choose how best to accomplish their work, rather than being directed by others outside the team.
- Cross-functional teams have all competencies needed to accomplish the work without depending on others not part of the team.

The team model in Scrum is designed to optimize flexibility, creativity, and productivity." [Schwaber and Sutherland 2011]



Scrum Team

The states of the Scrum Team sub-alpha

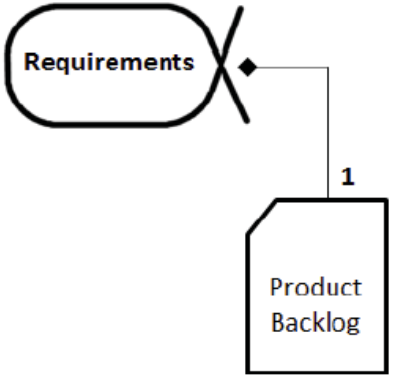
- **Work Products**
  - Product Backlog

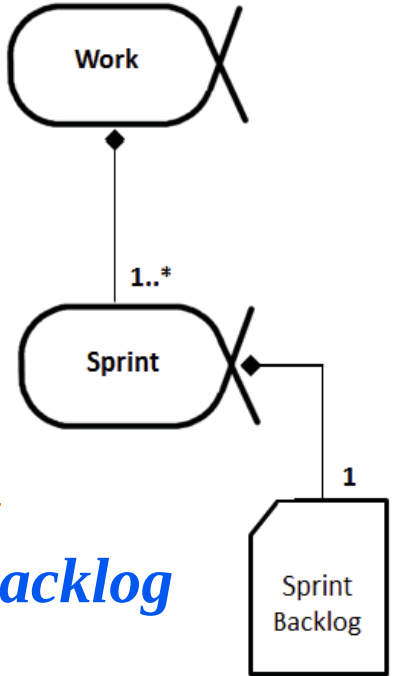
    "The Product Backlog is an ordered list of everything that might be needed in the product and is the single source of requirements for any changes to be made to the product. The Product Owner is responsible for the Product Backlog, including its content, availability, and ordering." [Schwaber and Sutherland 2011]

  - **Sprint Backlog**

    "The Sprint Backlog is the set of Product Backlog items selected for the Sprint plus a plan for delivering the product Increment and realizing the Sprint Goal. The Sprint Backlog is a forecast by the Development Team about what functionality will be in the next Increment and the work needed to deliver that functionality." [Schwaber and Sutherland 2011]

Product Backlog

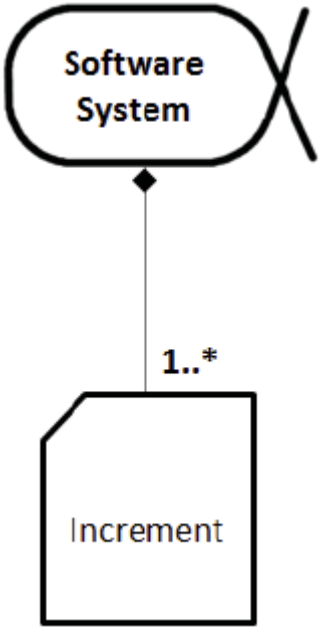




Sprint Backlog

- **Increment**
  - The Increment is associated with the Software System alpha.

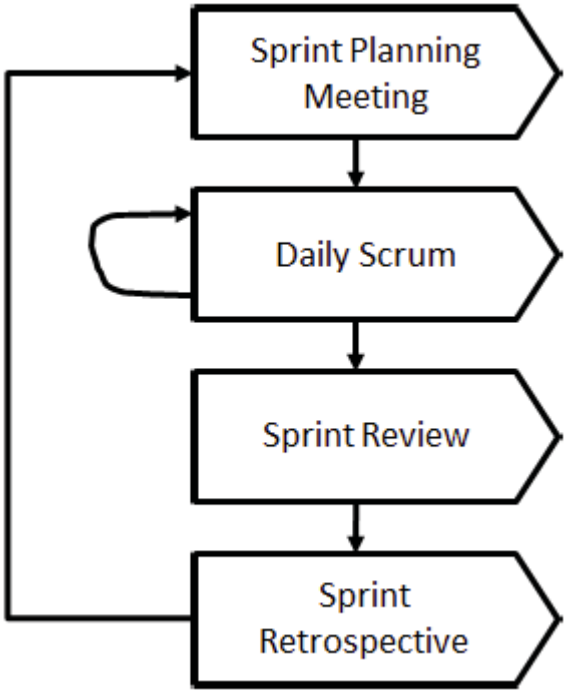
"The Increment is the sum of all the Product Backlog items completed during a Sprint and all previous Sprints. At the end of a Sprint, the new Increment must be “Done,” which means it must be in useable condition and meet the Scrum Team’s Definition of “Done.” It must be in useable condition regardless of whether the Product Owner decides to actually release it." [Schwaber and Sutherland 2011]



Textual syntax
workProduct Increment {
"The Increment is the sum of all the Product Backlog items completed during a Sprint and all previous Sprints. At the end of a Sprint, the new Increment must be “Done,” which means it must be in useable condition and meet the Scrum Team’s Definition of “Done.” It must be in useable condition regardless of whether the Product Owner decides to actually release it. (...continues...)"
}

北航软件所

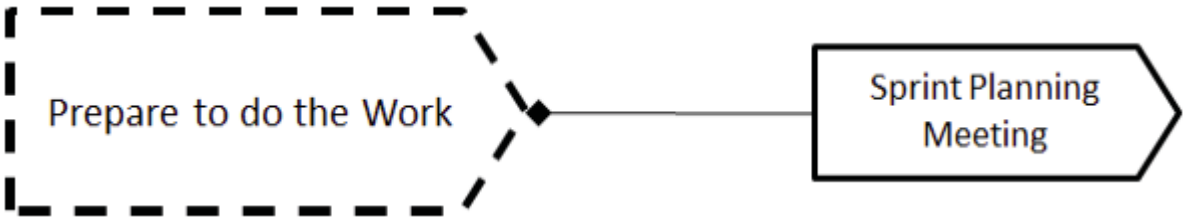
- **Activities**
  - The identified Scrum events may be mapped to corresponding activities



Scrum activities

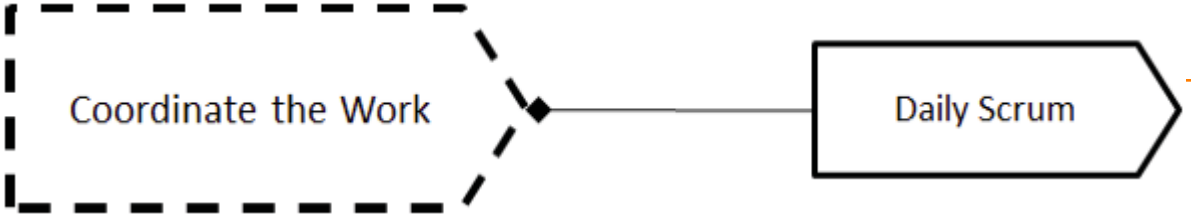
- ## Sprint Planning Meeting

"The work to be performed in the Sprint is planned at the Sprint Planning Meeting. This plan is created by the collaborative work of the entire Scrum Team. The Sprint Planning Meeting is time-boxed to eight hours for a one-month Sprint. For shorter Sprints, the event is proportionately shorter. For example, two-week Sprints have four-hour Sprint Planning Meetings." [Schwaber and Sutherland 2011]



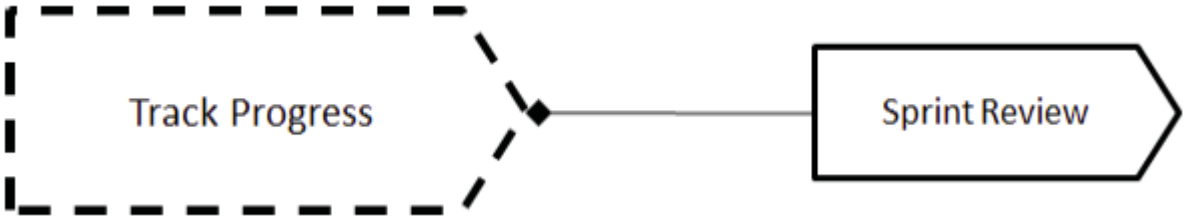
- ## Daily Scrum

"The Daily Scrum is a 15-minute time-boxed event for the Development Team to synchronize activities and create a plan for the next 24 hours. This is done by inspecting the work since the last Daily Scrum and forecasting the work that could be done before the next one." [Schwaber and Sutherland 2011]



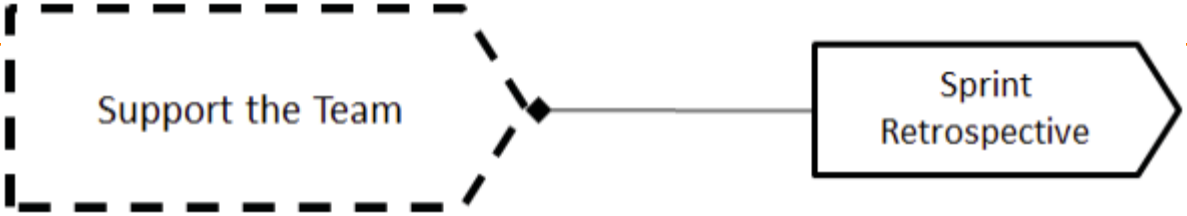
- ## Sprint Review

"A Sprint Review is held at the end of the Sprint to inspect the Increment and adapt the Product Backlog if needed. During the Sprint Review, the Scrum Team and stakeholders collaborate about what was done in the Sprint. Based on that and any changes to the Product Backlog during the Sprint, attendees collaborate on the next things that could be done. This is an informal meeting, and the presentation of the Increment is intended to elicit feedback and foster collaboration." [Schwaber and Sutherland 2011]



- ## Sprint Retrospective

"The Sprint Retrospective is an opportunity for the Scrum Team to inspect itself and create a plan for improvements to be enacted during the next Sprint. The Sprint Retrospective occurs after the Sprint Review and prior to the next Sprint Planning Meeting. This is a three-hour time-boxed meeting for one-month Sprints. Proportionately less time is allocated for shorter Sprints." [Schwaber and Sutherland 2011]



北航软件所

- **Roles**
  - Roles can be described as patterns:
    + Product Owner
    + Development Team (of developers)
    + Scrum Master
- **Textual syntax**

```
role Product Owner {
"The Product Owner is responsible for aximi
the value of the product
Development Team. Ho
widely across organizat
individuals.
(...continues...)"
}
```

```
role Development Team {
"The Development Team consists of professionals
who do the work of delivering a potentially
releasable Increment of “Done” product at the
end of each Sprint. Only members of the
```

```
role Scrum Master {
"The Scrum Master is responsible for ensuring Scrum is understood
and enacted. Scrum Masters do this by ensuring that the Scrum Team
adheres to Scrum theory, practices, and rules. The Scrum Master is a
servant-leader for the Scrum Team.
The Scrum Master helps those outside the Scrum Team understand
which of their interactions with the Scrum Team are helpful and which
aren’t. The Scrum Master helps everyone change these interactions to
maximize the value created by the Scrum Team.
(...continues...)"
}
```

例1：需求项描述

- 需求项
  - 编号：编号规则
  - 名称：命名规则
  - 简述：简述方式约定
- 用例模型：RUCM模板
  - 各项描述及其约束规则
- 相关概念及数据项定义
  - 数据词典
  - 概念/数据模型
- 其他模型
  - 状态模型、活动模型等

例2：软件评审与检查单（最佳实践）

- 评审目的
  - 采用人工方式（评审组）
  - 针对软件过程制品（文档、代码等）
  - 围绕关注点（检查单）
  - 发现问题，确认是否满足预设标准和要求
- 评审过程
  - 评审准备：检查单；资料提交和阅读
  - 评审会议：报告，提问，解答，问题记录与整理，结论
  - 问题处理
  - 复评审

- 检查单示例
  - 关于[软件需求.用例说明.RUCM规则]的检查单

编号	类别	检查项	严重性	检查要点说明
1.1	RUCM规则			
1.2				
1.3				
1.4				

北航软件所

例3：软件问题报告

- 软件评审
  - 软件问题基本属性

序号	问题概述	检查规则（检查表中规定的检查项）	问题具体说明	类别/严重性	报告人

  - 追踪关系
    + 文档：章节号、自然段/行号
    + 需求项：编号
    + 模型：模型名称及其元素/描述项

- 软件测试
  - 软件问题报告


软件问题报告
保存
退出
修复
关闭
添加注释
生成测试用例
软件问题报告
打开/再现
编号：张敏敏敏1000YS
作者：张敏
创建日期：2006-02-21
简要描述：无法从用例库向项目中导出测试用例，系统报错。
基本信息
修复信息
测试项
测试配置
自定义信息
附件
历史信息
测试版本：
20060220
测试级别：
系统测试
重现性：
总重现
严重性：
3.数据丢失或者功能失效
优先级：
2=建议修复
程序/文档名：
来源：
测试人员
问题引入阶段：
发现日期：
2006-02-21
将基本信息保存到我的配置文档
问题描述/影响分析
避开此问题的方法
附注及修改意见
1. 打开测试用例库。
2. 在测试用例库中选中一个测试用例，进行“导出用例”－“导出到项目”的操作。
3. 选择一个测试项目，并选择一个测试项，点击“确定”按钮。
问题：系统提示：向测试管理环境导出用例时出错了。

例4：软件测试需求描述

- 对应软件需求项
- 描述方法与规则
  - RTCM

例5：可追踪性的建立与维护

- 追踪关系
  - 用户需求与软件需求之间的依赖关系
  - 软件需求之间的（细化/依赖）关系
  - 软件测试需求与软件需求之间的依赖关系
  - 软件测试需求与测试用例之间的依赖（覆盖）关系
  - 软件测试用例与被测项（需求项/功能点/语句等）之间的覆盖关系
  - 软件问题报告与测试用例之间的依赖关系
  - 软件问题报告与软件问题域（修改项）之间的依赖关系

例6：任务分解说明

- 十二五核高基项目申请书（示例）

所属单位	WBS序号	子任务	负责人	目标	研究内容	考核指标
xxx	0.1.1	Linux内核体系架构分析	yyy	理解Linux内核架构的总体结构，掌握Linux内核架构的关键技术和实现细节，分析Linux内核体系架构的安全目标和安全风险	源代码分析理解技术、源代码分析理解工具、源代码架构的可视化现、操作系统内核体系架构与技术对比分析、Linux核体系架构	完成分析报告，分析覆盖Linux内核体系架构的主要内容，代码说明覆盖Linux内核体系架构源代码的全部文件
xxx	0.1.2	Linux内存管理	yyy	理解Linux内存管理的总体结构，掌握Linux内存管理的关键技术和实现细节，分析Linux内存管理的安全目标和安全风险	操作系统内存管理技术对比分析、Linux内存管理的技术与架构分析、Linux的内存管理实现细节分析	完成分析报告，分析覆盖Linux内存管理的主要内容，代码说覆盖Linux内存管理源代码的全部文件

北航软件所

- 课 题 实 施 期 任 务 汇 总 表（示例）

单位	分任务内容及WBS标号	2012年度任务				2013年度任务				2014年度任务			
		1季度	2季度	3季度	4季度	1季度	2季度	3季度	4季度	1季度	2季度	3季度	4季度
XXX	0.1.1 Linux内核体系架构分析												
	0.1.2 Linux内存管理												
	0.1.3 Android架构分析												
	0.1.4 Android内核内存管理												
	0.1.5 Dalvik虚拟机系统												

例7：质量度量

- 质量相关的分析与度量
  - 问题/缺陷：报告数量、密度/分布、改正数量
  - 审查和修改工作量
    - 发现每个缺陷的工作量
    - 改正每个缺陷的工作量

- 各阶段缺陷发现率：各阶段缺陷改正的人力花费比率

- 追踪关系表示方法
  - 依赖图
  - 依赖表
  - 依赖关系属性