



西安交通大学

《软件工程经济学》

王晨旭

软件学院

软件学院行政楼A212

cxwang@mail.xjtu.edu.cn

课程目标:

软件工程经济学主要针对**软件生命周期**中的工程经济学问题，运用工程经济学的理论方法，进行**货币的时间价值分析、软件项目成本估算、软件产品定价、可行性分析、项目经济学分析、风险评估和不确定性问题决策**等，并实施恰当的商业战略，比如投资组合管理，现金流量和融资管理，评估财务绩效，并适当调整软件项目目标和战略决策。

培养学生的经济意识、项目管理能力、抽象思维能力、总结归纳能力、严谨求实的科学作风和分析计算能力。为进一步研究软件项目管理理论、过程改进理论等打下必要的基础。

目标解读：经济意识培养

经济学是什么？





客户端播放

從前，有三個人——艾伯、貝克和查理，

暫停



00:03 / 05:13



以社会竞争现象为例

收益最大化

个人利益最大能
否促使社会利益
最大化? ? ?



囚徒困境

1950年由就职于兰德公司的梅里尔·弗勒德（Merrill M. Flood）和梅尔文·德雷希尔（Melvin Dresher）提出，后由阿尔伯特·塔克(Albert W. Tucker)以囚徒方式阐述，并命名为“囚徒困境”。



囚徒困境

在一次严重纵火案发生后，警察抓到两名嫌疑人。事实上正是他们为了报复而一起放火的，但警察没有掌握足够的证据。于是，将他们隔离审问，两个人都明白警方的政策：

- 如果两人都坦白，入狱5年；
- 如果两人都不坦白，由于证据不充分，他们只会被认定为妨碍公务罪而入狱1年；
- 如果一人抵赖，而另一人坦白，则抵赖者入狱8年，坦白者无罪释放。

囚徒困境

囚徒 B

坦白

不坦白

坦白

囚徒 A

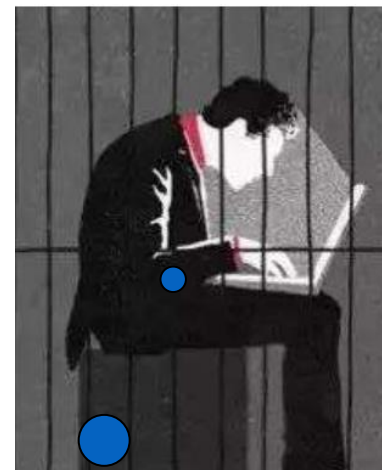
不坦白

	坦白	不坦白
坦白	-5, -5	0, -8
不坦白	-8, 0	-1, -1



此时，如果您是囚徒之一您会怎么办？

我坦白，那家伙肯定靠不住，肯定顶不住无罪释放的诱惑，会出卖我，到时候我就成了冤大头！况且如果他讲义气的话，我坦白，我无罪释放，为什么不呢？大不了两败俱伤，五年也比八年好啊



这小子，为人我清楚，八成顶不住，我不坦白就惨了！

囚徒困境

		囚徒 B	
		坦白	不坦白
囚徒 A	坦白	-5, -5	0, -8
	不坦白	-8, 0	-1, -1

- 囚徒A的选择:
若B选坦白, 则A选坦白;
若B选不坦白, 则A选坦白;
- 囚徒B的选择:
若A选坦白, 则B选坦白;
若A选不坦白, 则B选坦白;
- (坦白, 坦白) 是囚徒困境模型的必然均衡结果。但并不是最优的策略选择, 最优应为 (不坦白, 不坦白)。

囚徒困境



“囚徒困境”的思考？？

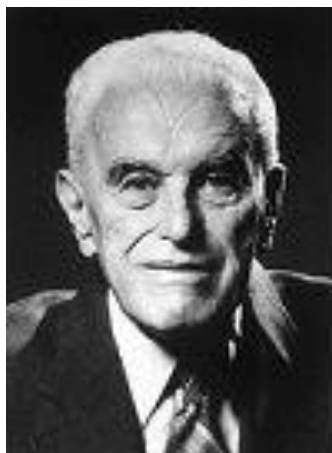
“囚徒困境”模型说明一个道理：从个人利益出发选择的最优策略，从整体看却不一定是个好的结果，也就是有时候个人利益和集体利益之间存在冲突。

囚徒困境

1994年纳什(Nash)、海萨尼(Harsanyi)、塞尔腾(Selten)三人，因为他们在非合作博弈论的研究方面所出了突出贡献，而获得诺贝尔经济学奖。



John Nash



John Harsanyi



Reinhard Selten

囚徒困境



约翰·纳什 (JOHN F. NASH)

美国人 (1928-), 由于他与Harsanyi、Selten在非合作博弈的均衡分析理论方面做出了开创性的贡献, 对博弈论和经济学产生了重大影响, 而获得1994年诺贝尔经济学奖。

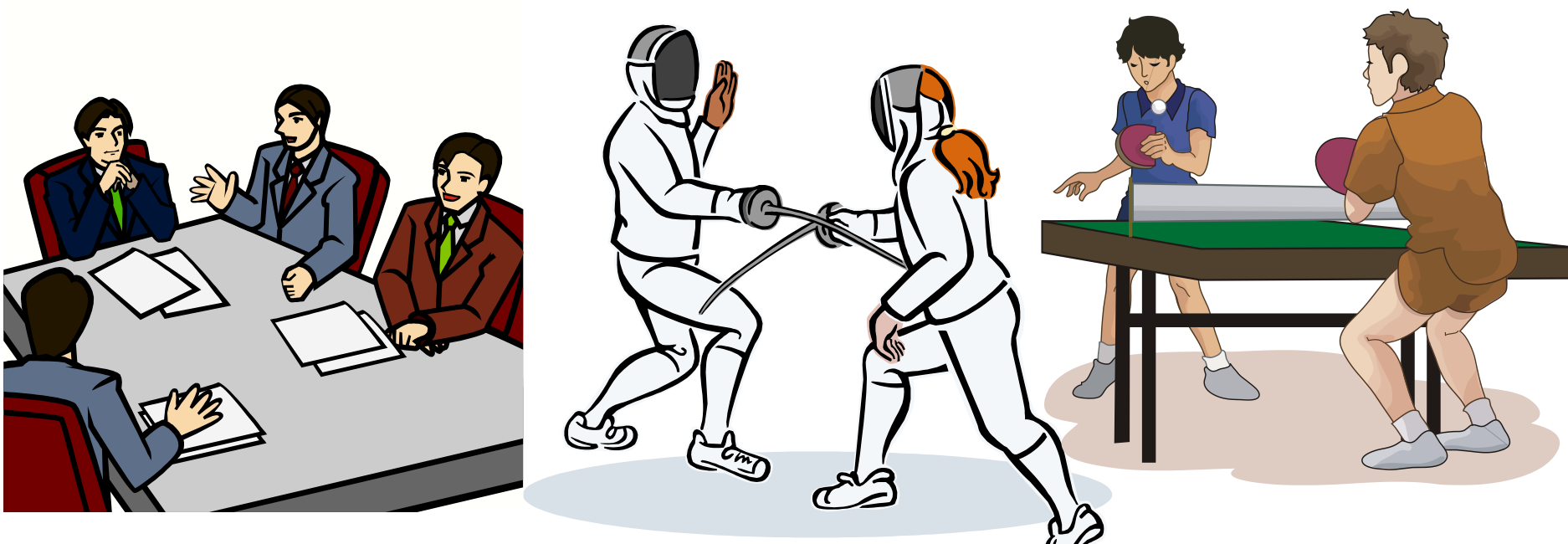
美丽
心灵



荣获8项奥斯卡提名, 最终夺得第74届奥斯卡最佳影片、最佳导演、最佳改编剧本和最佳女配角4项大奖。

囚徒困境

生活中，博弈无孔不入、无处不在：



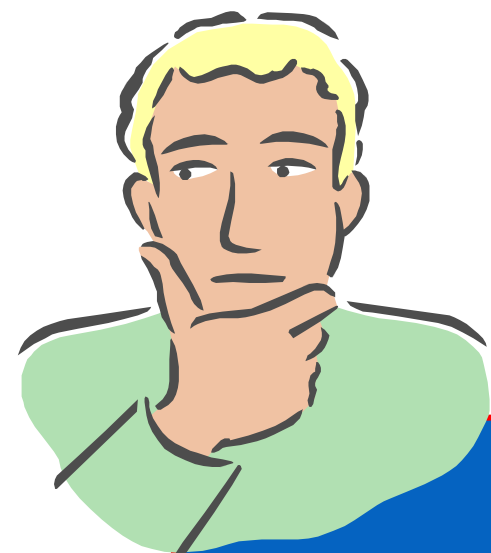
“要想在现代社会做个有价值的人，你就必须对博弈论有个大致的了解。” —— Paul Samuelson

囚徒困境

我们是不是生活中的囚徒呢？ ？ ？

内卷





讨论题：

如何看待教育内卷现象？对于个人有何启示？
软件工程中哪些体现了经济学思维？

QQ群： 软件工程经济学A班



扫描二维码加入QQ群：

- 1.发布课程公告
- 2.发布课程作业
- 3.发布课件
- 4.学生反馈

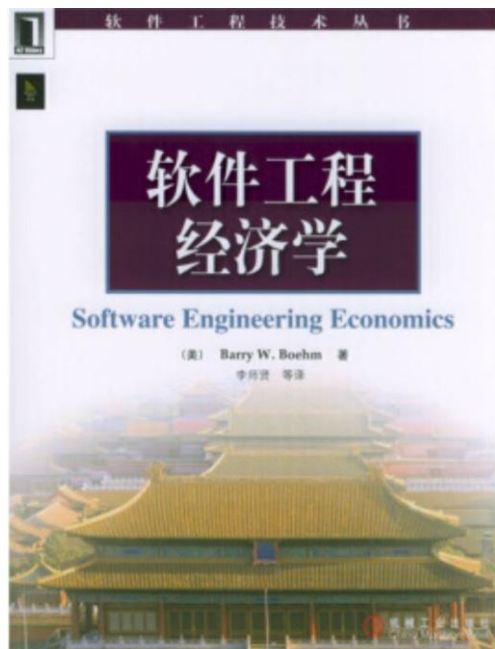
考核方式：

最终成绩=

期末考试成绩 (70%) + 平时成绩 (30%)

平时成绩=考勤+作业

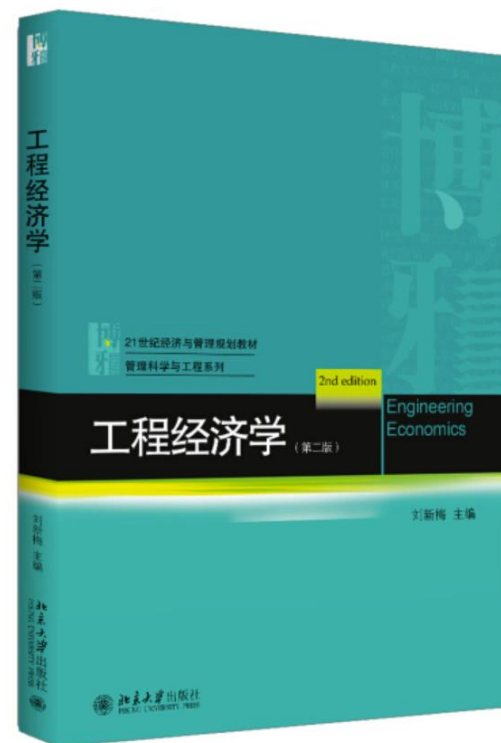
参考教材:



软件工程经济学
贝姆[美] 著
李师贤译
机械工业出版社



软件工程经济学
赵玮 著
西电出版社



工程经济学（电子版）
刘新梅 著
北京大学出版社

课程体系及学时分配

第1章 软件工程与软件工程经济学 (3学时)

第2章 软件工程经济学基础 (6学时)

第3章 软件的成本、工期与定价分析 (6学时)

第4章 软件项目的经济效益、社会效益与风险分析 (4学时)

第5章 软件生产过程经济分析 (5学时)

第6章 软件项目的进度计划制订与团队组织 (4学时)

第7章 软件测试的资源分配、进度管理与最优发行 (4学时)

第1章 软件工程与软件工程经济学

软件工程经济学 = 软件 + 工程 + 经济学

1.1 软件、软件产业与软件企业

1.2 软件工程

1.3 软件工程经济学的概念与任务

1.1 软件、软件产业与软件企业

1.1.1 软件及其分类与特点

学术界和产业界目前对于软件没有一个严格的分类标准。

目前学术界对软件的普遍性解释为：软件是计算机系统中与硬件相互依存的另一部分，它是包括程序、数据及其相关文档的完整集合。

按软件功能进行分类时的软件类别名称及其相应软件产品

表 1.1 按功能分类的软件信息表

序号	类别名称	类 别 内 涵	软 件 产 品
1	系统软件	泛指能与计算机硬件相配合,使计算机系统各个部件、相关程序和数据能协调、高效工作的软件	操作系统、数据库管理软件、设备驱动程序、文件编辑系统、系统检查与诊断软件
2	支撑软件	泛指能协助程序人员来开发软件的工具性软件和中间件,以及协助管理人员控制开发进度的工具	商业图形软件、文字/文件处理软件、C/S 开发工具、数据模型构造器、统计软件包、流程图设计软件
3	应用软件	泛指在某一特定领域内开发,为特定目标服务的一类软件	电力调度与控制软件、高速公路收费软件、银行业务系统软件、通信控制软件、导弹发射与控制软件等

按规模进行分类时的软件类别名称及其相应的产品规模、参加人数及研制周期

表 1.2 按规模分类的软件信息表

序号	类别名称	产品规模(源程序行数/行)	参加人数/人	研制周期
1	微型软件	500	1	1 周~1 月
2	小型软件	$(1\sim 2)\times 10^3$	1	1 月~6 月
3	中型软件	$(3\sim 50)\times 10^3$	2~5	1 年~2 年
4	大型软件	$(50\sim 100)\times 10^3$	8~20	2 年~3 年
5	超大型软件	$(0.1\sim 1)\times 10^6$	100~1000	4 年~5 年
6	极大型软件	$(1\sim 10)\times 10^6$	1000~5000	6 年~10 年

按软件标准化程度进行分类时的软件类别名称及其相应软件产品

表 1.3 按标准化程度分类的软件信息表

序号	类别名称	类 别 内 涵	软 件 产 品
1	标准化	可以封装发售，用户买来即可使用的软件	Windows 各版本的操作系统、Office 各版本的办公软件、瑞星安全软件等
2	半定制软件	具有相当一部分公共性功能，但在应用时还需要做一定的客户化开发工作，才能满足客户的需要	ERP 软件、财会软件、银行业务管理软件、电信业务管理系统、公路收费系统
3	软件服务	根据特定客户需求量身定制的软件，其特点是专用性强，可复用性不强	各种外包软件、系统集成服务等

第1章 软件工程与软件工程经济学

按与有关硬件或软件关联方式分类

表 1.4 按与有关硬件或软件的关联程度分类的软件信息表

序号	类别名称	类别内涵	软件产品
1	嵌入型 (Embedded) 软件	该软件要求在与其有紧密联系的硬件、软件 and 操作的限制条件下运行, 通常与某些硬件设备结合或嵌入在一起, 故对接口、数据结构及算法要求较高	航天测控系统、军事作战指挥系统、大型而复杂的事务处理系统、大型/超大型的操作系统
2	组织型 (Organic) 软件	该软件一般规模较小, 结构简单, 软件需求不那么苛刻, 受硬件的约束较少, 故开发人员应对此类软件开发目标理解充分, 使用环境熟悉	一般的数据库管理系统、操作系统、系统检查与诊断系统
3	半独立型 (Semidetached) 软件	介于上述两类要求之间	大多数事务处理系统、大型/新型数据库管理系统、简单的指挥系统、新的操作系统、大型的库存/生产控制系统

软件的特点

无形性、抽象性、可复制性和共享性

软件的生产过程（除复制外）几乎都是从头开始

创新已成为软件产品发展的动力和企业竞争的焦点

目前只能由人采用手工方式来生产

软件的成本构成与硬件产品相比，无需库存成本

市场的进入壁垒一般较低

1.1.2 软件产业及其特点

软件产业是指软件产品和软件服务相关的一切经济活动和关系的总称。

软件产业的特点

高技术、高附加价值与高效益

推动产业结构调整、产品技术改造的基础和支撑

国际化特征明显

专业化分工越来越细

规模经济效益日益明显

根据中国软件行业协会发表的《2000年中国软件产业研究报告》，软件产业包括**软件产品**和**软件服务**两大部分，其中**软件产品**分为**系统软件**、**支撑软件**和**应用软件**。

软件服务包括**信息系统集成**、**信息系统运行和维护服务**、**数据中心与资源外包服务**、**数据加工与处理服务**、**软件测试服务**、**信息系统咨询和评估服务**、**信息系统监理**、**软件与信息系统管理与人才工程化培训**等。而上述软件服务中，前三个属于软件产品的支撑与维护，后五个又可统称为软件专业化服务。

国外关于软件产业的划分与中国软件行业协会的认识有一定的差距，表1.5给出国际数据公司(International Data Corporation, IDC)关于软件产业的领域细分情况。

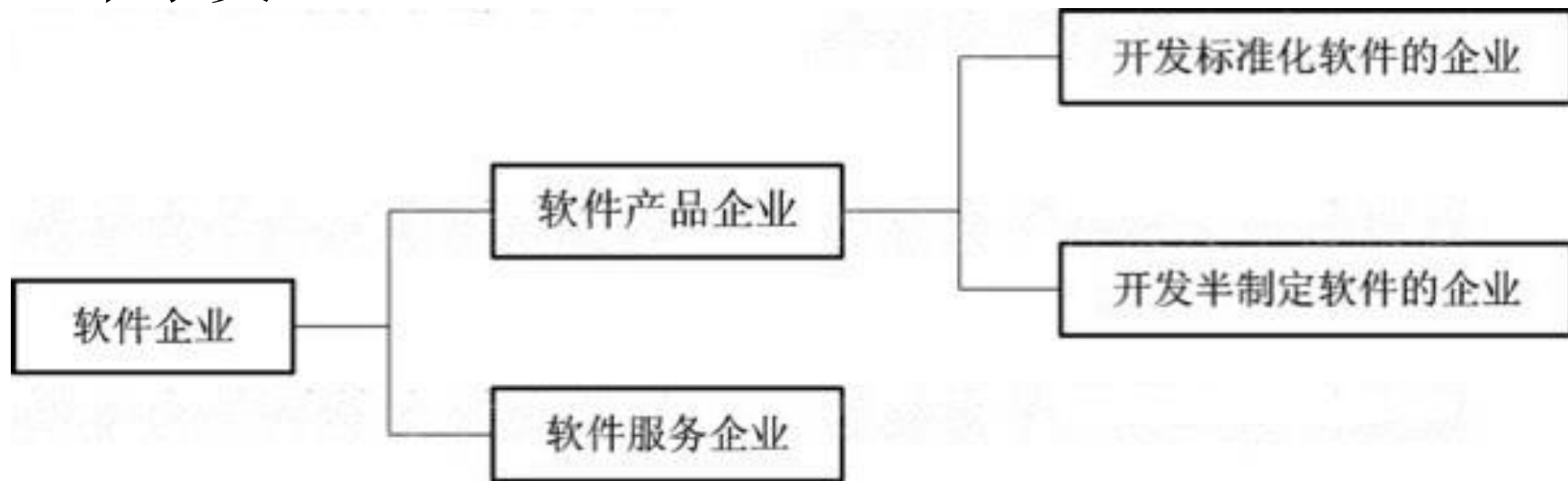
第1章 软件工程与软件工程经济学

表 1.5 IDC 的软件产业领域细分表

序号	领域类别名称	软件产品或重要功能
1	系统基础软件	系统管理软件、网络管理软件、系统安全软件、服务器软件、网络软件、中间件等
2	应用开发和配置软件	信息/数据管理软件、应用设计和构建工具软件、应用软件生命周期管理、应用服务器等
3	应用解决方案	消费应用(如家庭、游戏和娱乐等) 协作应用(如集成协作环境、消息应用等) 内容管理应用(内容及文件管理应用) 写作应用(如文字加工等) 说话及自然语言应用(翻译、自然语言处理) 企业资源管理(如人力资源管理、设备管理、财务管理等) 服务行业应用(银行、证券、保险、电信及其他公用事业) 产品供应链应用(如制造、零售、批发流通、CAD/CAM、EDA、ERP 等) 客户关系管理应用(销售支持、客户管理、呼叫中心等) 其他企业类应用(如商务业绩管理等)

1.1.3 软件企业及其管理

从市场角度出发，软件企业可分为**软件产品企业**和**软件服务企业**两类，而软件产品企业又可分为开发标准化软件的企业和开发半定制软件的企业两个子类。



企业管理是指在企业特定的生产方式下，管理者按照某些原则、程序和方法，使用一定的手段(工具、设备)，针对生产的各要素(人力、物力、设备、资金、信息)进行计划、组织、指导、协调和控制，以使其发挥最大的经济效果，达到预期的管理目标的一种筹划和过程。

软件企业管理从企业管理的分类来看，通常包括如下内容：战略管理、生产运作管理、市场营销管理、财务管理、人力资源管理、采购管理、信息管理等。上述分类管理及其管理目标与管理活动见表1.7。除上述各类管理外，通常还有项目管理、库存管理、计划管理等等。

第1章 软件工程与软件工程经济学

表 1.7 企业管理的分类与活动表

序号	管理类别	管理目标与内涵	管 理 活 动
1	战略管理	对企业一系列长远目标的管理决策	完成企业外部、内部环境分析,制定企业的总体发展规划、职能部门战略和战略控制策略
2	生产运作管理	通过对企业产品的生产运作过程的计划、组织与控制,高效率地生产出低成本、高质量的产品	完成对产品生产过程的业务流程、工艺流程、作业计划和技术方法、环境配置的设计,建立产品的质量评估体系,完成对产品生产过程的质量监控与评价
3	市场营销管理	通过对产品市场的调查分析,以支持市场营销活动的组织协调与控制	调查了解企业外部环境与消费者行为及其变化,确定产品目标市场,研究产品定价策略,完成产品的销售渠道和销售策略组合设计及其市场活动的执行与控制
4	财务管理	通过对企业资金的筹划、分析,预测和组织,协调与控制,以支持企业系统目标的完成	制定企业财务制度与财务计划,完成企业资金的筹集与纳税筹划,完成企业财务的分析、预测与风险评估,做好企业财务的内部控制(成本、现金、应收帐款等的控制)
5	人力资源管理	有效开发与合理组织企业的各类人力资源,以支持企业的技术创新与生产、销售目标	建立企业人员的激励机制与约束机制,完成企业的各类岗位工作设计与绩效评估体系,建立研究企业的薪酬分配政策与计划,完成内部员工的绩效评估、薪酬分配、员工培训和外来人员的招聘
6	采购管理	通过对产品必需的原材料、设备与技术的采购和业务外包,以支持企业的生产目标	原材料、设备、技术的市场调查、评估与选购,供应商的分析、评价与组织、协调,完成企业业务外包的合同和组织与协调
7	信息管理	通过对企业的物流、资金流和信息流的分析与组织,以最大限度地支持企业各类管理活动	建立企业的信息中心,完成企业物流、资金流和信息流的信息采购、传输、存储、加工和应用

作为一种产品，软件与硬件有很大的不同。软件是一种逻辑载体，没有具体的形状与尺寸，只有逻辑的规模和运行的效果，且这种逻辑载体由于其面向的产品需求目标的不同，故在生产过程中呈现出不同软件的各自特点和无重复性，这就导致软件产品只能一个一个地生产出来。

软件更像一个特殊的项目(Project)。而软件开发管理就可用项目管理(Project Management)的理论和方法来进行指导。

所谓**项目**，可视为在既定的资源和需求约束下，为实现某种目的而相互联系的一次性的有计划的工作任务。项目管理则是伴随着项目工作的进行而展开的，为确保项目能达到预期目标的一系列的管理方法和管理行为。

目前，软件项目管理通常包括如下9个部分：**项目综合管理、项目范围管理、项目时间管理、项目费用管理、项目质量管理以及人力资源管理、信息与配置管理、风险管理、采购管理。**

有关此9项分项管理的管理活动内容详见表1.8。在这9个分项管理中又以项目综合管理为中心，以项目范围、时间、费用、质量管理为准则展开工作，它们的关联见图1.2。

第1章 软件工程与软件工程经济学

表 1.8 项目分项管理活动表

序号	分项管理	主要管理活动
1	综合管理	项目计划的制定；项目计划执行；项目计划的更改与控制
2	范围管理	项目范围定义；项目范围规划确定；项目范围变更及其控制
3	时间管理	确定项目工作任务与顺序；项目进度计划制定与优化；项目进度的跟踪与控制
4	费用管理	项目资源需求确定；项目规模与成本估算；项目成本控制
5	质量管理	项目质量标准与评估确定；项目质量计划与跟踪；项目质量控制
6	人力资源管理	项目人力资源需求估计；项目团队组织；项目团队建设
7	信息与配置管理	项目信息的采集、存储、整理与发布；项目信息的应用分析；项目合同与文档管理
8	风险管理	项目风险辨别及其影响程度估计；项目风险分析；项目风险应对策略与控制
9	采购管理	制订项目指标与采购计划；供应商选择与指标计划执行；供应商管理

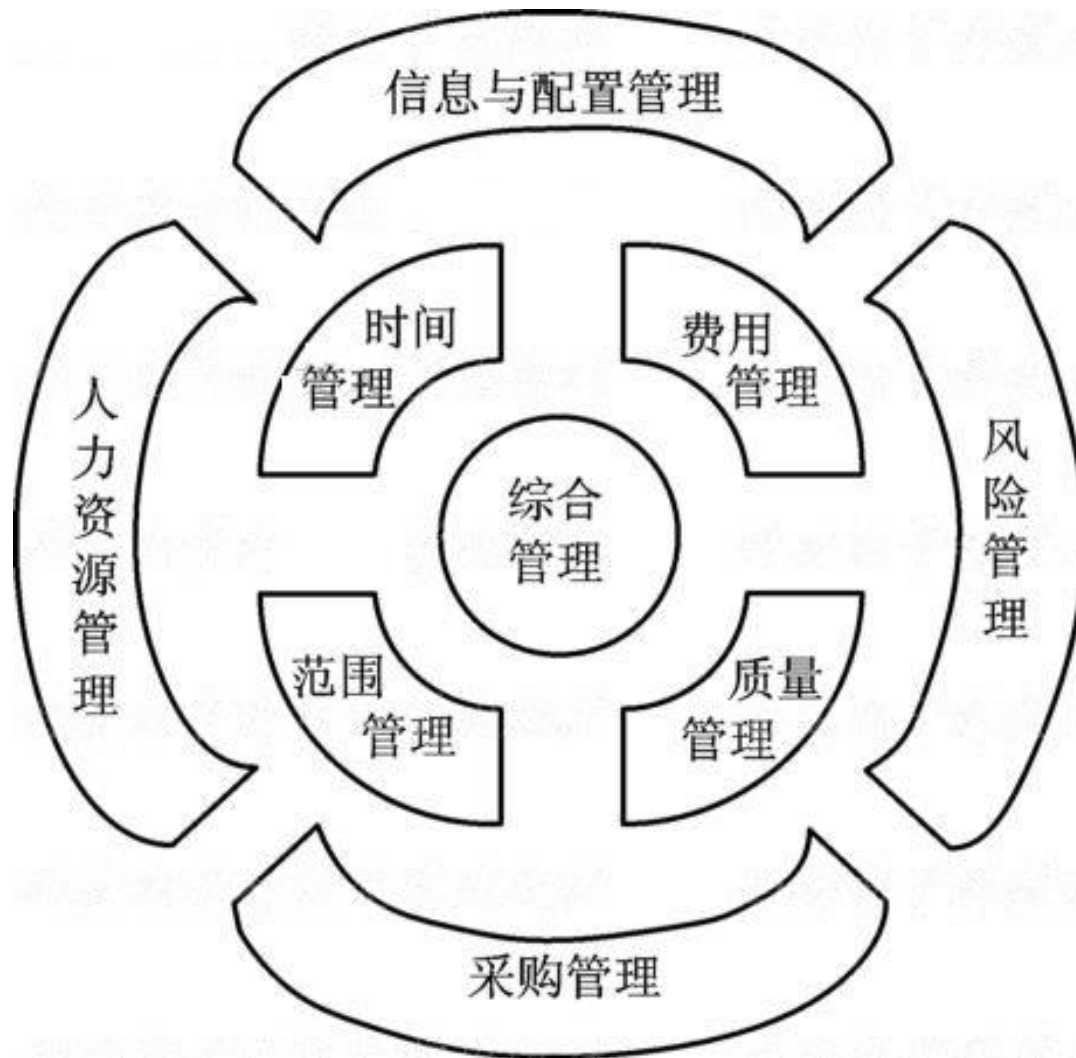


图1.2 项目管理关联图

通常将战略管理、生产运作管理、市场营销管理、财务管理、人力资源管理、采购管理、信息管理等统称为企业的“面上”管理，而将项目管理称为企业的“点线”管理。这两部分管理共同构成了企业管理的两大部分，它们之间的关联或层次结构见图1.3。

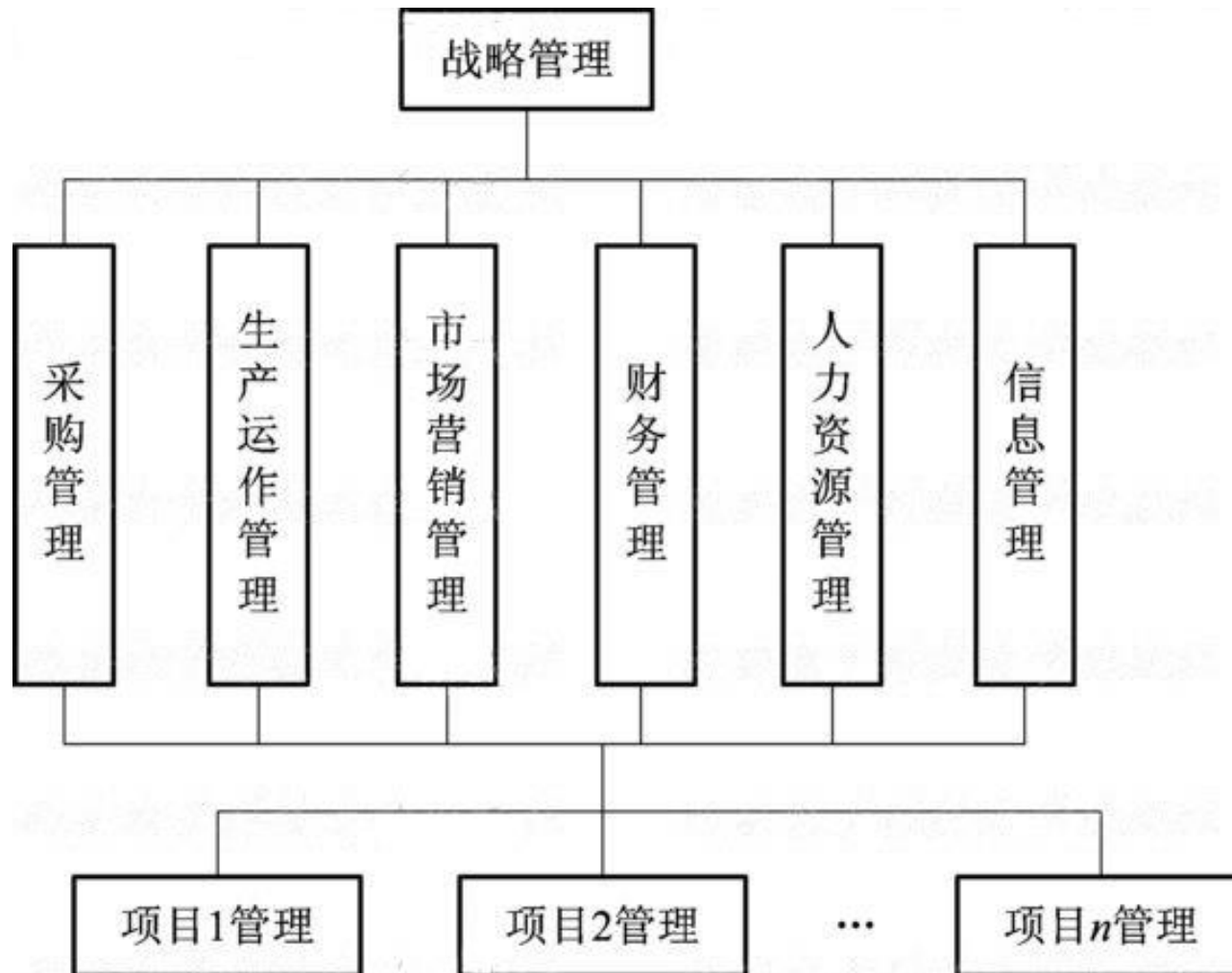


图1.3 企业管理关联图

1.2 软件工程

1.2.1 软件工程的概念与分类

软件工程是一门适用于软件开发全过程的系统工程方法论的学科，其目的是为满足人们对软件生产的成本、质量、时间(工期)和效率、效益和管理等的需求。

软件工程的四个核心理念：**复用、分治、折中、演化。**

1.2.1 软件工程的分类

软件工程是包含软件开发技术和软件工程管理两方面。软件工程经济学属于软件工程管理的内容。

工程与科学的区别

计算机科学	软件工程
发现和研究长期的、客观的真理	短期的实际结果（具体的软件会过时）
理想化的	对各种因素的折衷
确定性，完美，通用性	对不确定性和风险的管理，足够好，具体的应用
各个学科独立深入研究，做出成果	关注和应用各个相关学科的知识，解决问题
理论的统一	百花齐放的实践方法
强调原创性	最好的、成熟的实践方法
形式化，追求简明的公式	在实践中建立起来的灵感和直觉
正确性	可靠性

1.2.2 软件生存周期、开发模型与任务分解

软件作为一个特殊产品与其他产品一样有其自生到灭的生存过程。通常我们将软件以概念形成开始，经过开发、使用和维护，直到最后退役的全过程称为软件的生存周期。在此生存周期中，软件可根据其所处的状态、特征以及软件开发活动的目的、任务划分为若干阶段。图1.4给出了划分为七个阶段的软件生存周期阶段划分图。

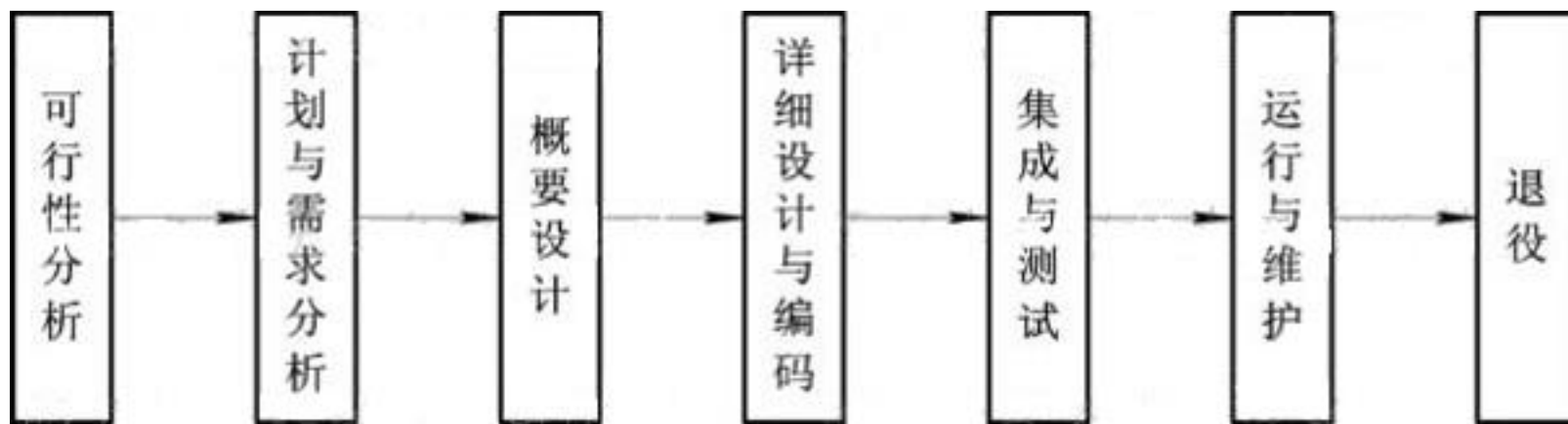


图1.4 软件生存周期阶段划分图

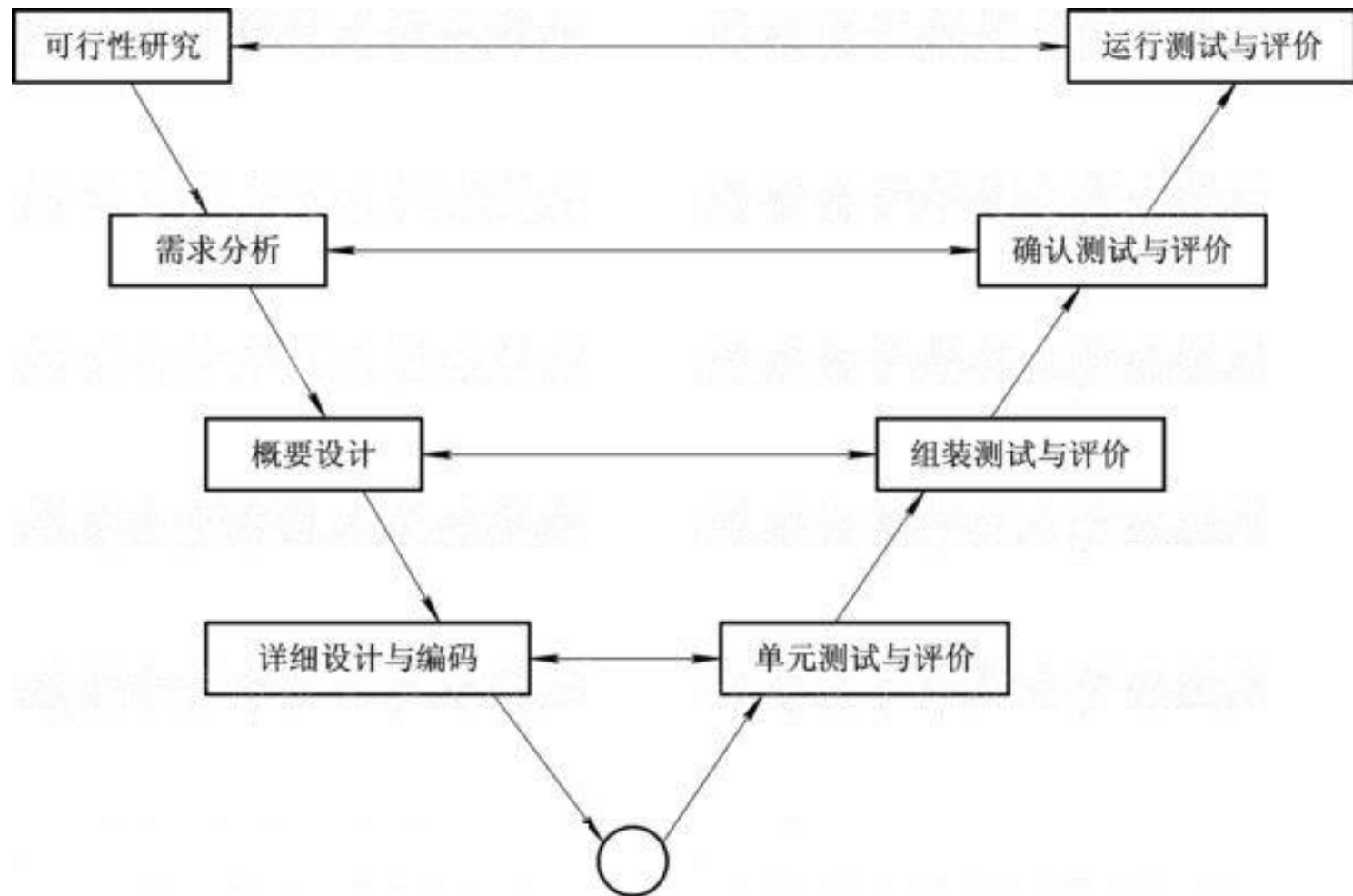


图1.5 软件开发阶段活动关联图

为了给软件开发过程提供原则和方法，以及为软件工程管理提供里程碑和进度表，人们设计了软件生存周期中各阶段活动的关联图示，这种关联图示称为软件的开发模型。目前软件开发模型有瀑布模型、原型模型、螺旋模型、基于第四代技术(4GL)的模型、变换模型和组合模型等。软件工工程经济学中目前常用的是瀑布模型和螺旋模型，图1.6和图1.7分别给出了带反馈的瀑布模型和螺旋模型，其他模型的图示可参见《软件工程》教材。

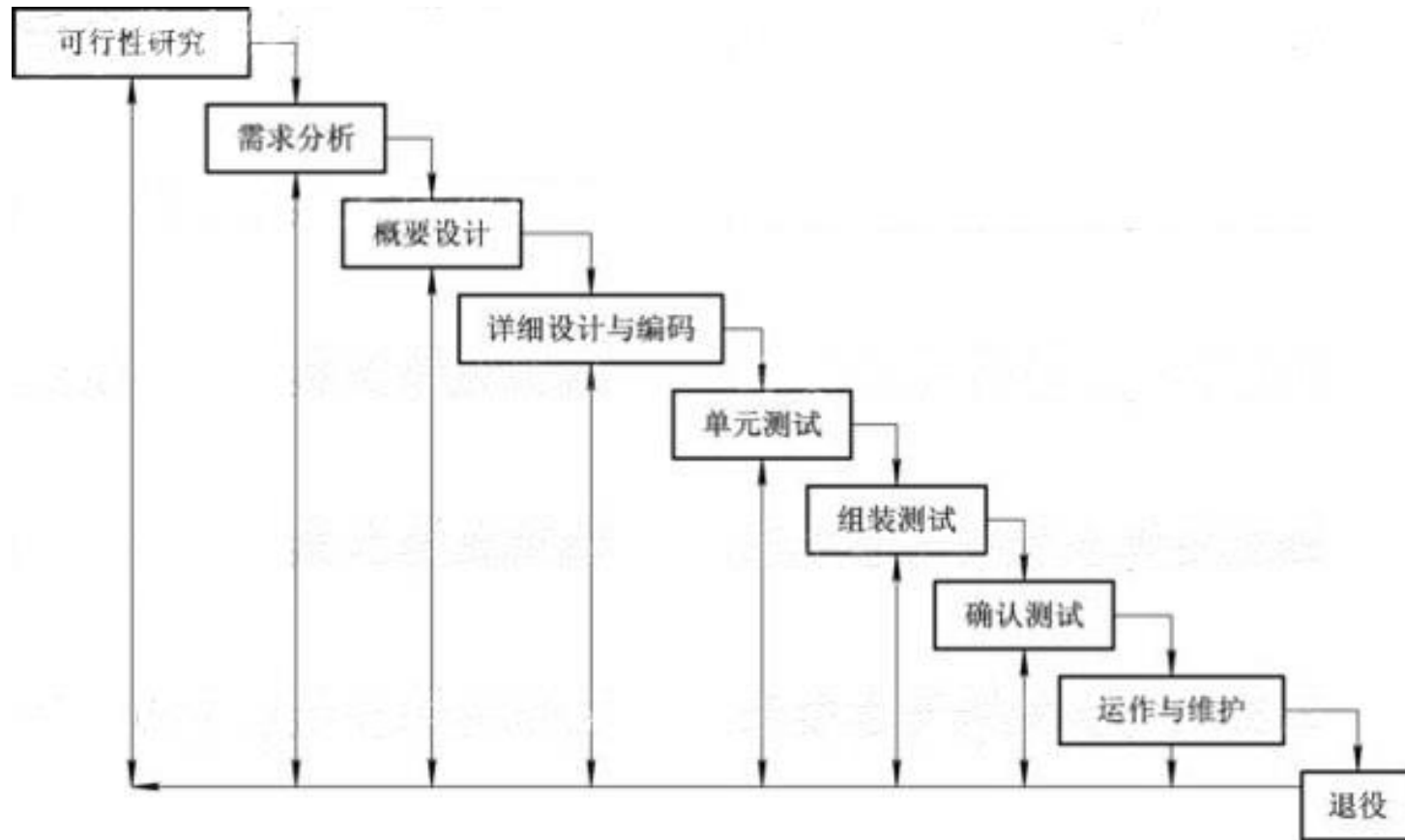


图1.6 带反馈的瀑布模型

第1章 软件工程与软件工程经济学

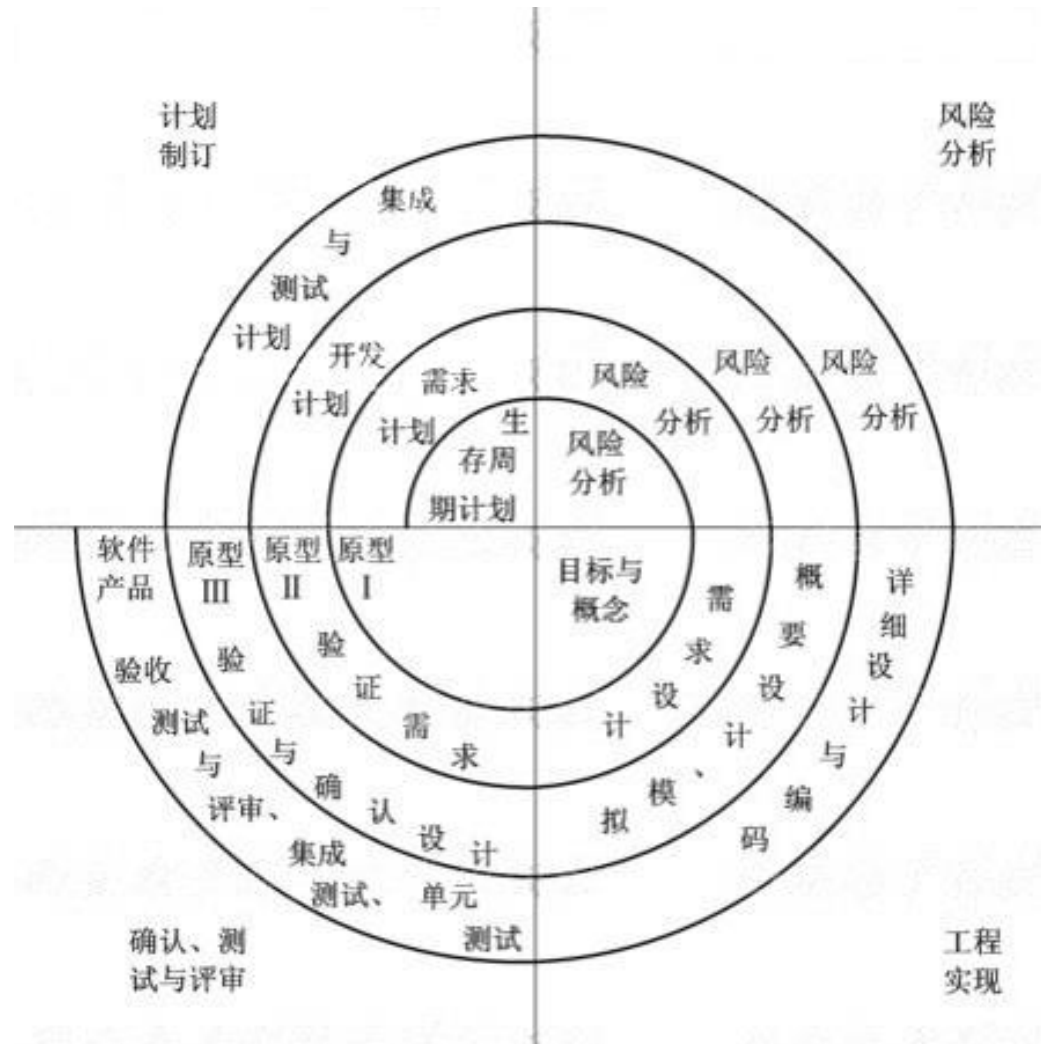


图1.7 螺旋模型

第1章 软件工程与软件工程经济学

软件项目与硬件产品生产一样，为了获得高效率的生产和质优、价低的产品，必须运用科学的理论与方法如系统工程、工程经济学、项目管理的理论与方法来指导软件开发的全过程，并进行必要的定量分析与评价。为此必须对产品生产的诸多要素与属性进行度量(Metrics)，考虑到软件度量涉及的范围较广，以下仅介绍在软件工程经济学中所涉及的软件基本度量，如软件规模、软件复杂性、软件可靠性、软件安全性与软件质量等的度量。

1.2.3 软件规模与复杂性度量

1. 软件规模度量

目前在软件工程界中影响较大的软件规模度量单位有**程序源代码行**(Lines Of Code., LOC)和**功能点**(Functional Point, FP)两种。

源代码行由于可用人工或软件工具直接测量，因而其估算简易可行。而功能点作为度量单位是Albrecht于1979年提出的，目前在欧共体使用十分普遍。与统计源代码行的直接度量方法不同，它是一种间接度量的方法。

功能点计算方法的基本思想为首先计算软件的五个基本信息量：外部输入数(External Input., EI)、外部输出数(External Output., EO)、外部查询数(External Query., EQ)、内部逻辑文件数(Internal Logical File., ILF)、外部接口文件数(External Interface File., EIF)的加权和CT，然后对其通过14个环境复杂性因子作如下修正，即有

$$\left\{ \begin{array}{l} \text{FP} = \text{CT} \cdot \text{PCA} \\ \text{CT} = \sum_{j=1}^5 w_j d_j \\ \text{PCA} = 0.65 + 0.01 \sum_{i=1}^{14} F_i \end{array} \right. \quad (1.1)$$

式中,

d_1 即为外部输入数(EI), 它包括了每个用户为软件提供的输入参数个数(不包括查询数), 体现了软件面向用户服务的数量特征;

d_2 即外部输出数(EO), 它指软件为用户提供的输出参数个数, 如报告数、屏幕帧数、错误信息个数等;

d_3 即外部查询数(EQ), 它规定一个联机输入确定一次查询, 软件以联机输出的形式实时地产生一个响应, 统计各种查询个数;

d_4 即内部逻辑文件数(ILF), 它要求统计内部逻辑主文件数;

d_5 即外部接口文件数, 通常指所有机器可读的界面(如磁盘或磁带上的数据文件), 利用此接口界面可以将信息从一个系统传送到另一个系统;

w_j 为第 j 个基本信息量 d_j 的加权系数或复杂程度系数，它由表1.10取值决定；

CT称为软件的功能数，PCA称为系统功能的复杂性调整因子。

表 1.10 基本信息加权系数

基本信息量权系数	简单	一般	复杂
用户输入权系数 w_1	3	4	6
用户输出权系数 w_2	4	5	7
用户查询权系数 w_3	3	4	6
内部逻辑文件权系数 w_4	7	10	15
外部接口文件权系数 w_5	5	7	10

(1.1)式中的14个环境复杂性因子 F_i 分别体现了数据通信、分布式数据处理、软件性能、硬件负荷、事务频度、联机数据输入、界面复杂度、内部处理复杂度、代码复用性要求、转换和安装、备份和恢复、多平台考虑、易用性等环境属性的复杂程度。 F_i 的取值可以通过表1.11中的问题来选定取值，其中要求 F_i 在0、1、2、3、4、5等六个量级中选择1个。

第1章 软件工程与软件工程经济学

表 1.11 环境复杂性参数取值

F_i	F_i 取值准则与问题	F_i	F_i 取值准则与问题
F_1	系统需要数据通信的程度	F_8	系统的输入输出文件查询复杂性
F_2	系统具有分布处理功能的程度	F_9	系统的内部处理复杂性
F_3	系统有否临界状况等性能要求	F_{10}	代码设计的可重用性
F_4	系统是否在一个现存的实用操作环境下运行	F_{11}	设计中包括转换和安装的程度
		F_{12}	系统中需要的备份和复原程度
F_5	系统需要联机 on-line 数据入口吗?	F_{13}	系统设计支持不同组织的多次安装状况
F_6	联机数据入口需要用输入信息建造复杂的屏幕界面和操作吗?		
F_7	系统需要联机更新主文件吗?	F_{14}	系统设计有利于用户的修改和使用状况

注： F_i 根据对应的准则与问题从 0、1、2、3、4、5 等六个值中选择其一，需求越复杂，取值越大。

〔例1.1〕 某软件根据需求分析，对照表1.11的各项要求，得到环境复杂性因子 $\sum_{i=1}^{14} F_i = 24$ ，五个信息量的数值 d_j 及其对应权系数 w_j 之取值见表1.12，于是由(1.1)式可得软件系统其需求功能点为

$$\begin{aligned} \text{FP} &= \text{CT}(0.65 + 0.01 \sum_{i=1}^{14} F_i) = \sum_{j=1}^5 w_j d_j (0.65 + 0.01 \sum_{i=1}^{14} F_i) \\ &= 615 \times (0.65 + 0.24) = 615 \times 0.89 = 547.35 \end{aligned}$$

若功能点与源代码行的转换率 μ 为

$$\mu=15 \text{ LOC/FP}$$

则该软件系统有规模

$$L_S=\text{FP} \cdot 15=8210\text{LOC}=8.21 \text{ kLOC}$$

表 1.12 d_j 和 w_j 值

j	1	2	3	4	5
d_j	32	60	24	7	3
w_j	4	5	4	10	7

上述功能点法称Albreach功能点法，除此之外，还有MarkII功能点法、COSMIC全功能点法等。详可参见有关文献。

2. 软件复杂性度量

复杂性是软件的重要属性之一，任何一个有经验的程序员都知道。对于同种规模而复杂性不同的软件，其花费的成本和工期会有很大的差异。**如何来描述与衡量软件的复杂性，目前尚无统一的定论。**K.Magel认为如下的六个方面可作为软件复杂性描述的依据：

- (1) 理解程序的难度；
- (2) 纠错、维护程序的难度；
- (3) 向他人解释程序的难度；
- (4) 按指定方法修改程序的难度；
- (5) 根据设计文件编写程序工作量的大小程度；
- (6) 执行程序时需要资源的多少程度。

20世纪70年代M.Halstead从统计学和心理学的角度来研究软件复杂性问题，提出用程序中可执行代码的词汇量(操作符与操作数)来计算和分析软件复杂性的方法，并在此基础上还可将其转换成软件规模的测算。

Halstead认为程序是一个符号序列，此序列由操作符、操作数交替出现组成。其中，**操作符是指由程序设计语言定义并在程序中出现的语法符号**，如FORTRAN、PASCAL等语言中的+、-、*、/、IF、THEN、DO、END等(不含注释性语句)；**操作数是指操作符所作用的对象**，它同样由程序定义并引用，可以是变量、常量、数组、记录、指针等。

通过数学推导，可以证明程序语言的符号长度(又称词汇总数) N 可近似地由下式确定：

$$N = n_1 \lg n_1 + n_2 \lg n_2 \quad (1.2)$$

式中， n_1 为程序中不同操作符的个数； n_2 为不同操作数的个数； $\lg n$ 即 $\log_2 n$ 。此外，利用转换公式：

$$L = \frac{N}{C} \quad (1.3)$$

还可将程序语言的符号长度 N 转换成源程序行数 L (不含注释性语句), 其中 C 为转换系数, 它与所使用的程序设计语言有关, 同时也与软件类型以及程序员的编程风格等因素有关, 可以通过对历史数据的统计分析来估计。在一般情况下, FORTRAN语言编程时有 $C=7.5$, 用PASCAL语言编程时有 $C=4.0$ 。

1.2.4 软件差错与可靠性度量

软件作为一种特殊产品，同样需要一系列技术性能指标来衡量其所具有的功能与性能的技术水平，如软件对数据的处理能力、存储空间、算法复杂性、预测成本、投资额的精度等；同样也需要一些可靠性指标来衡量软件保持良好的功能与性能水平的持久能力。

1. 软件差错与可靠性

与硬件相比，由于软件生产更多地依赖于人的劳动，开发人员的智力、精力与经验、时间的有限，于是程序编写发生错误或未按规范完成程序编写；测试人员的经验、技术与工具的不足而使设计差错未能检测出来；用户与开发机构以及开发人员间的沟通不足造成对目标需求的理解错误与不充分以及需求变更太过频繁、开发人员对软件质量的错误认识、不负责任乃至自高自大态度、进度上的压力、管理上的缺失等现象，不可避免地地每一个软件项目开发过程中或多或少地发生。

这就导致每一个交付给用户的软件产品都不可避免地会有差错(Soft Error), 从而导致软件产品本身具有功能与性能不完整或不正常或难以使用等方面的软件缺陷, 于是这样的软件产品使用时就会发生软件故障(Software Fault), 如数据丢失、死锁、操作系统失灵以及程序不能退出、输入 / 输出错误、计算错误等。由于中大型软件是复杂的逻辑产品, 人们不可能通过枚举法来对程序运行的所有路径来进行测试并排除差错, 这就导致软件特别是嵌入型软件的使用具有很大的危险性。

例如，由于火箭惯性制导系统(软件)的一个差错，导致在1996年6月发射的耗资67亿美元的阿丽亚娜(Ariane)501火箭在首次飞行实验中，点火升空37秒后即在空中爆炸，造成巨大的损失；1993年海湾战争中，美“爱国者”导弹的雷达跟踪系统由于一个软件差错而导致导弹发射发生了1/3秒的时间误差，从而不仅未能击中伊拉克的“飞毛腿”导弹，反而导致美军的重大伤亡；2003年，日本东京机场的航空调度系统的一个软件故障，造成全日本200多个航班停飞，2000多个航班误点，1324架次的航班晚点达30分钟以上，使日本航空业的形象大损.....

2. 软件可靠性度量指标

设程序按照规格说明从初始时刻 $t = 0$ 开始运行直到发生故障为止这一连续时间段称为软件的寿命。易知软件寿命是一个非负随机变量 ξ ，其分布函数 $F(x) = P(\xi \leq x)$ 称为软件产品的寿命分布函数，而软件产品在时刻 t 的生存概率

$$R(t) = P(\xi > t) = 1 - F(t)$$

称为该软件产品的可靠度函数或可靠度。

软件产品的可靠度也可以定义为：产品在规定的条件下，在规定的时间内，完成规定功能的概率。其中，规定的条件通常是指软件的运行环境如运行平台、操作系统、编程工具等，规定的时间是指软件使用的连续时间，而规定的功能则指软件产品已完成的目标功能。若软件寿命 ξ 的概率密度为 $f(t)$ ，则由概率论知有

$$f(t) = \frac{dF(t)}{dt}, F(t) = \int_0^t f(x) dx \quad (1.4)$$

若软件寿命 ξ 的概率密度存在，则在 t 时刻软件产品的失效率(或故障率) $\lambda(t)$ 被定义为

$$\lambda(t) = \frac{f(t)}{1 - F(t)} = \frac{f(t)}{R(t)} \quad (1.5)$$

注意到有如下概率近似等式：

$$P(\xi \leq t + \Delta t \mid \xi > t) = \frac{P(t < \xi \leq t + \Delta t)}{P(\xi > t)} \approx \frac{f(t)\Delta t}{R(t)} = \lambda(t)\Delta t$$

由此可知失效率 $\lambda(t)$ 可理解为：软件产品在正常工作一直到时刻 t 的条件下，它在 $(t, t + \Delta t]$ 区间内失效的概率。当 Δt 很小时， $\lambda(t)$ 还可理解为软件产品在 t 前正常工作的条件下，而在 $t + \Delta t$ 瞬间发生失效的概率密度。

由于有

$$\lambda(t) = \frac{f(t)}{R(t)} = -\frac{1}{R(t)} \frac{dR(t)}{dt}, R(0) = P(\xi > 0) = 1$$

容易求解上述微分方程可得

$$R(t) = \exp\left\{-\int_0^t \lambda(u) du\right\} \quad (1.6)$$

随着对软件产品的不断测试(单元测试、集成测试、验收测试、运行测试), 软件早期存在的差错将不断地被检测到而改正, 故软件产品的失效率或故障率函数呈单调下降趋势。而硬件产品在生存周期内其失效率曲线如同一个浴盆一样, 故称为浴盆曲线。它分成三个阶段, \overline{OA} 段被称为早期失效阶段, 在此阶段中由于电子产品常含有不合格的元器件或在设计与工艺中存在不完善的地方, 因而此阶段呈高失效率特征, 且伴随着元器件筛选及技术工艺的改进, 使失效率单调下降。

第1章 软件工程与软件工程经济学

在第二时间段 \overline{AB} ，由于产品性能趋于稳定，故失效率大致不变，产品失效往往是运行环境突变等偶然原因造成的，故称为偶发性失效阶段。在第三阶段 \overline{BC} ，由于产品运行时间较长，元器件因物理磨损而逐步老化，从而使失效率呈单调上升阶段，故此阶段称为损耗失效期。

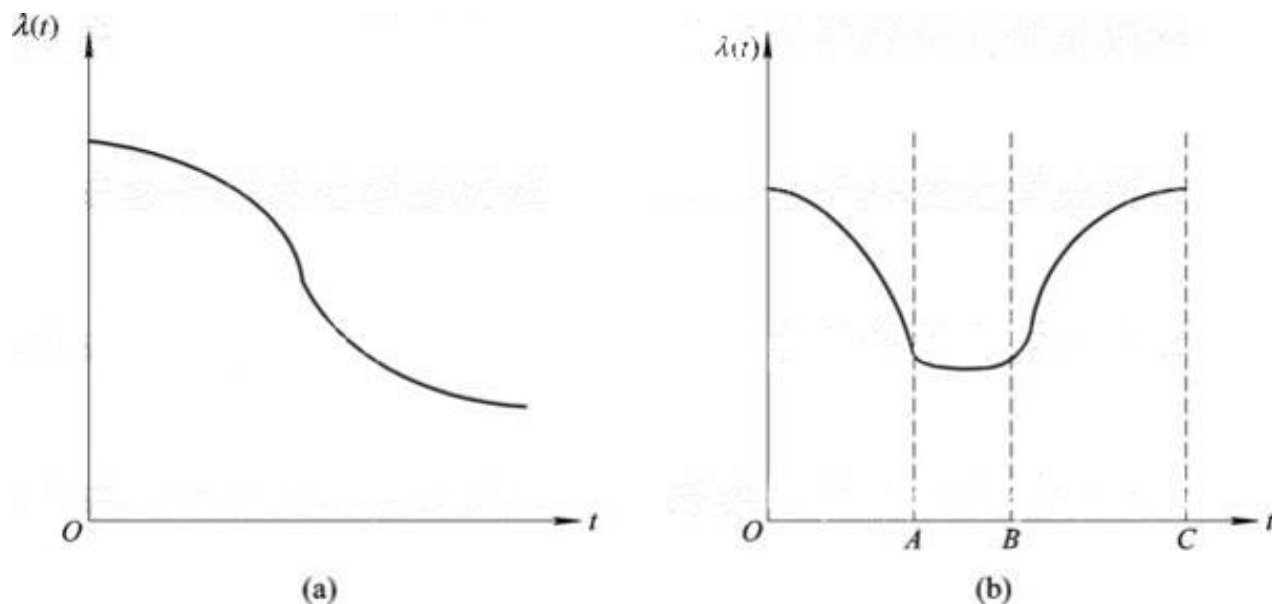
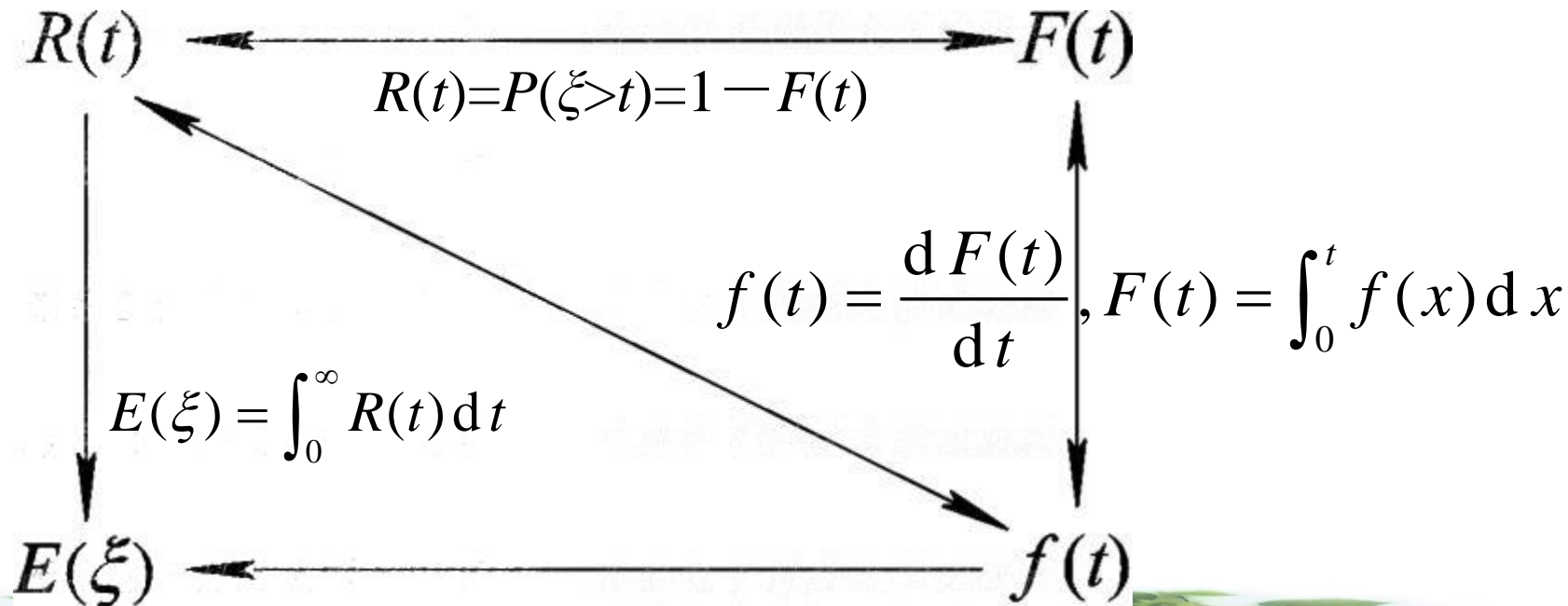


图1.8 软、硬件失效率曲线

此外，软件寿命 ξ 的期望值 $E(\xi)$ 称为软件的**平均寿命**。并容易证明 $E(\xi)$ 与 $R(t)$ 有如下关系式：

$$E(\xi) = \int_0^{\infty} R(t) dt \quad (1.7)$$

(1.4)、(1.5)、(1.6)、(1.7)式描述了 $F(t)$ 、 $f(t)$ 、 $R(t)$ 、 $E(\xi)$ 的相互关系，这种相互的关联关系可由下图来表示。



除了上述四个可靠性度量指标之外，还有软件的维修函数，维修率，平均维修时间，软件交付时的初始潜在固有差错数，交付后经测试又排除了 n 个差错后软件的残存差错数，将这些残存差错全部排除的期望时间和方差等可靠性指标。

1.2.5 软件质量

1. 软件质量评价

20世纪70~80年代, McCall等人相继发表了他们的研究报告。McCall将软件质量定义为“满足规定的和潜在需要能力的总和”, 并提出了一个由三个层次构成的软件质量度量模型。

其中, 最高层的元素称为**质量要素**(Quality Factor), 共11个; 中层的元素称为**评价准则**(Evaluation Criteria), 共30个; 底层元素称为**质量度量**(Quality Metric), 详见图1.10。该度量模型表示软件质量保证任何一个软件都可由使用单位人员通过对软件开发过程中软件本身所具有的原始属性(每一个原始属性称为一个评价准则)的逐一度量后, 即可得到对该软件各质量要素的评价, 并进而提出对被评软件的总体评价。

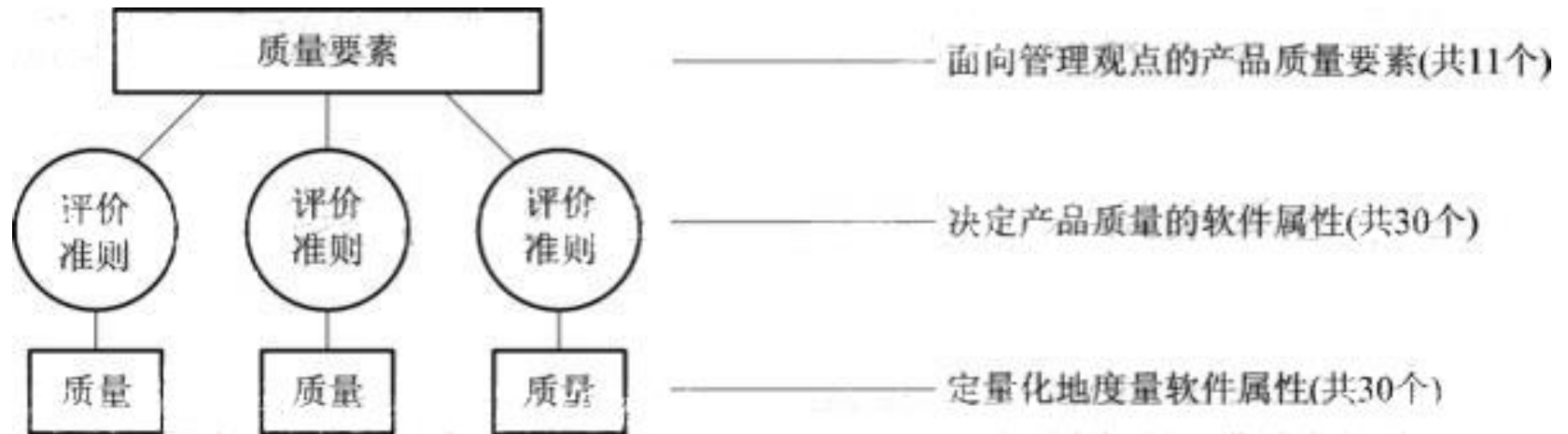


图1.10 McCall软件质量度量模型

随后，国际标准化组织ISO在McCall等人所提出的软件质量度量模型的基础上，于1991年发布了ISO/IEC9126质量特性国际标准，并建立了与图1.10类似的ISO软件质量度量模型，见图1.11。

在ISO软件质量度量模型中，最高层用软件质量需求准则(SQRC)代替McCall模型的质量要素，中层用软件质量设计评价准则(SQDC)代替McCall模型的评价准则，而底层则用软件质量度量评价准则(SQMC)代替McCall模型的度量概念，并规定如何对每一个SQDC作为度量(SQMC)可根据用户的实际需要自行给出。所不同的是，SQRC由11个元素改为8个元素，SQDC由30个元素改为23个元素。

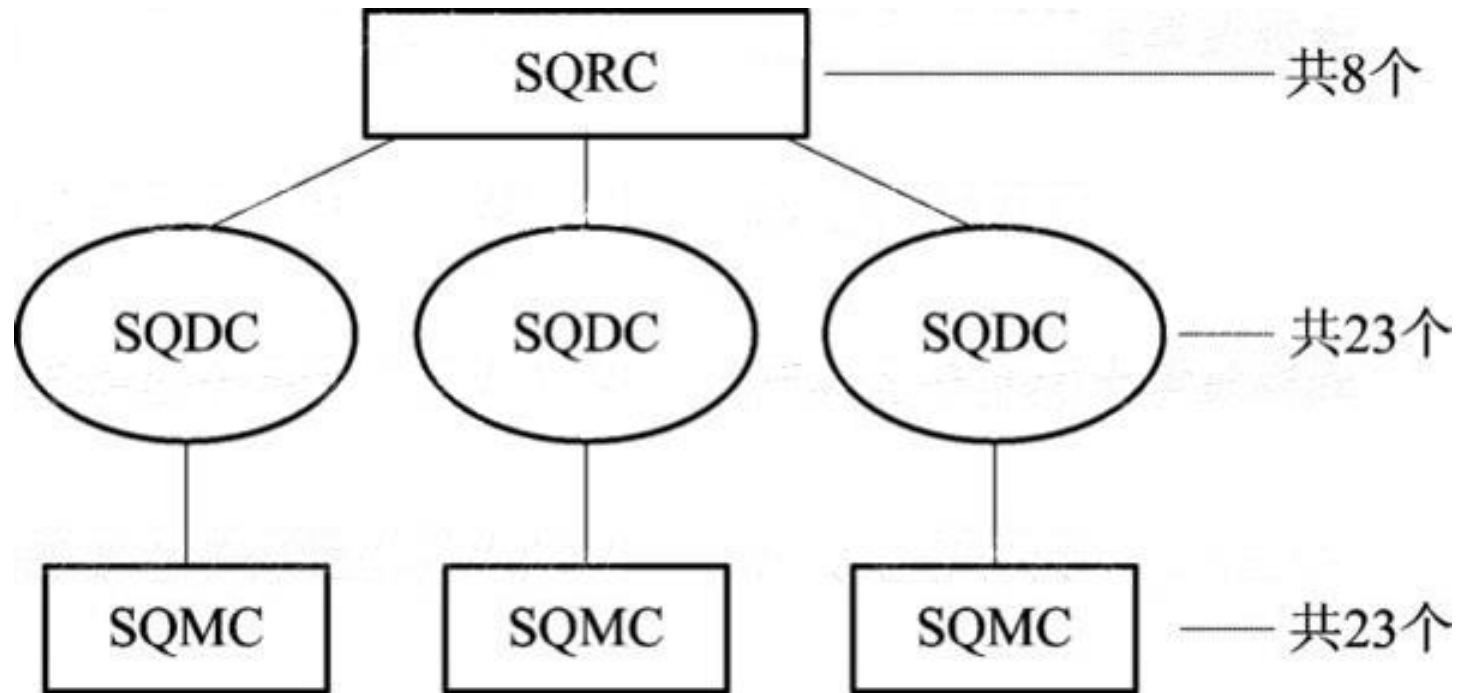


图1.11 ISO软件质量度量模型

以下对SQRC的8个元素及SQDC的23个元素逐一给出其基本含义或解释。有关SQRC中的8个元素与SQDC中的23个元素的隶属关系见表1.13。

第1章 软件工程与软件工程经济学

表 1.13 SQRC 与 SQDC 隶属关系表 (ISO 软件质量模型)

SQDC \ SQRC	SQRC							
	正确性	可靠性	效率	安全性	可使用性	可维护性	灵活性	连接性
(1) 可追踪性	○							
(2) 一致性	○	○				○		
(3) 完备性	○							
(4) 准确性		○						
(5) 容错性		○						
(6) 简单性		○				○		
(7) 模块性						○	○	○
(8) 通用性							○	
(9) 可扩充性							○	
(10) 工具性(自检性)						○		
(11) 自描述性						○	○	
(12) 执行效率			○					
(13) 存储效率			○					
(14) 存取控制				○				
(15) 存取审查				○				
(16) 可操作性					○			
(17) 可培训性					○			
(18) 通信性					○			
(19) 软件系统独立性							○	
(20) 机器独立性							○	
(21) 通信共享性								○
(22) 数据共享性								○
(23) 简明性(可理解性)						○		

注：记号“○”表示其所在行元素受所在列元素的支配(或隶属)。

1) 软件需求准则(SQRC)

(1) 正确性(Correctness): 指程序满足需求说明及用户目标的能力。

(2) 可靠性(Reliability): 指程序按要求的精度完成预期功能的能力。

(3) 效率(Efficiency): 指程序完成其功能所需的资源及代码的数量。

(4) 安全性(Security): 指对未经许可的人员接近软件或数据加以控制的能力。

(5) 可使用性(Usability): 指熟悉程序操作, 为程序准备输入数据和翻译程序输出所需付出的努力。

(6) 可维护性(Maintainability): 指确定可运行程序中的错误所需要付出的努力。

(7) 灵活性(Flexibility): 指修改可运行程序所需要付出的努力。

(8) 连接性(Interoperability): 指程序与其他系统耦合的能力。

2) 软件质量设计评价准则(SQDC)

(1) 可追踪性(Tractability): 指在规定的开发和运行环境中, 联结软件需求和软件实现的线索的清晰程度。

(2) 完备性(Completeness): 指软件需求功能全面实现的程度。

(3) 一致性(Consistency): 指软件设计技术、实现技术和标识符的协调和统一程度。

(4) 准确性(Accuracy): 指软件计算和输出的精确程度。

(5) 容错性(Error tolerance): 指软件在非正常条件下, 具有的继续运行的能力。

(6) 简单性(Simplicity): 指实现软件规定功能所采用方式的简单和容易理解的程度。

(7) 模块性(Modularity): 指软件具有独立模块结构的程度。

(8) 通用性(Generality): 指软件可履行功能的跨度。

(9) 可扩充性(Expandability): 指软件的功能和数据存储, 允许扩充的程度。

(10) 工具性(Instrumentation): 指软件为错误识别和应用测试所提供条件的程度。

(11) 自描述性(Self Descriptiveness): 指软件解释执行功能的清晰程度。

(12) 执行效率(Execution Efficiency): 指软件执行的快速程度。

(13) 存储效率(Storage Efficiency): 指软件运行所需的存储量的精简程度。

(14) 存取控制(Access Audit): 指对接近软件和存取数据的控制程度。

(15) 存取审查(Access Audit): 指对接近软件和存取数据所做的审查的严格程度。

(16) 可操作性(Operability): 指软件运行所需操作的复杂程度。

(17) 培训性(Training): 指熟悉软件操作的困难程度。

(18) 通信性(Communicativeness): 指吸收、利用程序输入和输出的困难程度。

(19) 软件系统独立性(Software System Independence): 指软件对环境的依赖程度。

(20) 机器独立性(Machine Independence): 指软件对硬件系统的依赖程度。

(21) 通信共享性(Communications Commonality): 指软件采用标准协议和常规接口的程度。

(22) 数据共享性(Data Commonality): 指软件采用标准数据结构的程度。

(23) 简明性(Conciseness): 指软件实现预期功能所需代码量的精简程度。

2. 软件质量保证

软件的质量度量模型反映了对一个给定目标软件的度量的质量需求准则、质量设计评价准则和质量要素评价的基本内容和从属层次关系，并为解决待开发(或正在开发)的软件能够达到用户的各种需求目标提供了求解的思路与规范，但与此同时，**人们在软件开发过程中还必须有专人从事软件质量保证工作。**

软件质量保证(Software Quality Assurance, SQA)包括如下多种工作：

- (1) 推行与确认软件工程质量标准；
- (2) 研究与采用各种技术手段来保证软件质量；

- (3) 对软件的各种变更进行控制;
- (4) 制订并执行软件测试策略测试计划;
- (5) 按照软件质量标准对软件的质量进行度量;
- (6) 组织各种技术评审会或评审活动;
- (7) 对软件质量的度量情况及时记录和生成SQA报告。

遵循软件质量标准来进行软件质量度量是软件质量分析(SQA)的另一重要工作。这需要建立一系列的度量指标(定性或定量指标)或度量指标体系, 并应有相关确定的评价方法来具体确定每一度量指标的度量值以及整个软件的综合度量值。

软件评审(Software Review)是软件质量标准的另一重要手段，通过对软件开发过程中任一阶段的评审，可以发现软件设计与开发中的隐藏错误并加以排除，同时也起到了对软件开发的检查与监督作用。软件评审工作最常见的是召开由第三方(专家)、开发人员、用户代表组成的评审会，首先评审组起草评审文件，确定评审要求及各评审人员的职责，待检查的软件项目测试清单和评审进度等；然后评审委员会根据开发人员提交的软件(含相关文档)按照测试清单进行逐项测试，并对软件开发过程的各种问题提出询问，要求开发人员作出解答；最后讨论与通过开发机构所提交的软件是否“通过评审”的决议。

1.3 软件工程经济学的概念与任务

经济学(Economics)是研究人类在从事生产、交换以及对产品和劳务消费过程中, 如何有效地利用和合理地配置可供选择的各种有限资源(又称稀缺资源), 以使人类的现在和将来的无限欲望得到最大满足的一门学科。

按照研究的范畴不同, 经济学可划分为宏观经济学(Macro Economics)和微观经济学(Micro Economics)。

按照所研究的对象与属性的不同, 经济学又可有工程经济学、管理经济学、区域经济学、发展经济学、制度经济学、信息经济学、经济统计学等分支学科。

经济学的十大基本原理

第一部分：人们如何做出选择

原理一：人们面临权衡取舍

原理二：凡事皆有成本

原理三：边际效应

原理四：激励机制

第二部分：人们如何相互影响

原理五：交易能使每个人状况更好

原理六：市场化

原理七：政府调控

第三部分：整体经济是如何运行的

原理八：国家经济水平取决于其生产物品与劳务的能力

原理九：当政府发行过多货币时，物价上升（通货膨胀）

原理十：通货膨胀与失业之间的短期交替

什么是工程经济学

工程经济学(Engineering Economics)工程经济学是对工程技术问题进行经济分析的系统理论与方法。工程经济学是在资源有限的条件下,运用工程经济学分析方法,对工程技术(项目)各种可行方案进行分析比较,选择并确定最佳方案的科学。它的核心任务是对工程项目技术方案的经济决策。

研究对象是工程项目技术经济分析的最一般方法,即研究采用何种方法、建立何种方法体系,才能正确估价工程项目的有效性,才能寻求到技术与经济的最佳结合点。

1.3.1 软件工程经济学的内涵与任务

软件工程经济学(Software Engineering Economics, SEE)

从名词上看可以理解为工程经济学与软件工程的交叉学科。

我们将其定义为以软件工程领域中的经济问题和经济规律为研究对象的一门经济学分支学科，具体地说，就是研究为实现特定功能需求的软件工程项目而提出的在技术方案、生产(开发)过程、产品或服务等方面所作的经济分析与论证，计算与比较的一门系统方法论学科。

作为一门有待发展的新兴学科，从系统工程的研究思路来看，软件工程经济学至少应该包括如下四个部分：

- (1) 学科研究的对象、任务、特征、研究范围和研究方法；
- (2) 软件系统的内部构成要素和经济活动及其关联分析，如投资、成本、利润、效益、工期、效率、质量及研制、开发、维护、管理活动及其关联分析；
- (3) 软件系统的组织结构、管理决策及其与经营活动的关系；
- (4) 软件系统的物流、资金流、信息流的输入与输出及其对系统外部——国家、地区经济、社会、国防、人民生活的影响。

其中(1)为软件系统的基础概念与理论部分，(2)、(3)为软件系统的微观经济分析部分，(4)为软件系统的宏观经济分析部分。作为(2)与(3)的细化，研究内容具体如下：

- 软件工程经济分析基本原理及应用，如价值工程原理、规模经济与生产函数原理、成本效益分析与边际分析原理，项目开发的时间、成本/效益、质量、效率的均衡原理、优化原理与敏感性分析等。
- 软件项目的成本估算、成本控制与融资。
- 软件项目开发的风险与不确定性分析和投资可行性分析。

- 软件产品的质量评价、经济效益评价、财务评价以及主要质量指标——软件可靠性、维护性等的经济评价方法。
- 软件生存周期中各种生产(开发)与管理活动的经济分析与决策，如软件工具与设备的采购决策、信息获取决策、开发技术方案的评价与决策、成本的阶段分配决策、软件发行决策、产品定价决策等。
- 软件项目的工作任务分解与计划制订、组织与协调及其经济分析与优化。
- 软件开发过程的动态规律描述及其各经济要素的关联分析。
- 软件开发效率(劳动生产率)的影响因素分析及改进策略研究。

1.3.2 软件工程经济学的研究特点与方法体系

1. 研究特点

软件工程经济学的研究具有如下特点：

(1) 注意到软件产品的“人工制作”的特点和经济学中产品的质量、成本/效益、时间/进度、效率等目标要素的重要性，因此软件工程经济学的研究重点始终环绕着软件产品的质量、成本/效益、时间/进度、效率等目标要素的关联分析及其人的组织与协调(管理)对上述各目标的影响分析进行，详见图1.12。

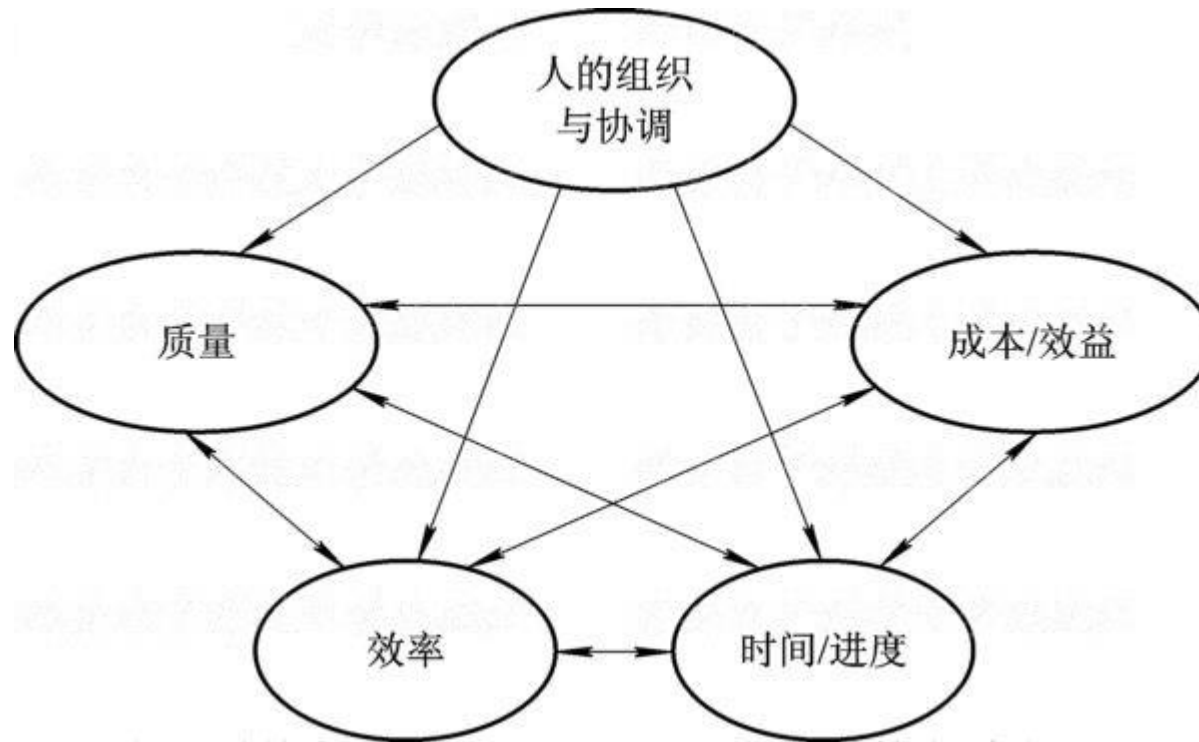


图1.12 SEE要素关联图

(2) 软件工程经济学的研究思想来自于系统工程，因此注意软件系统目标的整体性(总体性)、要素的层次性(有序性)和关联性、系统结构的合理性(协调性)、系统环境的适应性等始终是人们研究的指导准则。

(3) 注意到人的组织与协调度量的复杂性，因此软件工程经济学的研究方法采用了经济学中的传统思路，即采用定性分析与定量分析相结合、理论分析与实证验证相结合的思路，其中定量分析中由于目标的多样性，因而又为多目标决策的理论与方法提供了用武之地。

(4) 考虑到我国与西方发达国家在文化与价值观念、技术水平、经营机制、管理水平与生产效率以及软件工程环境上的差异，因此在大力学习与借鉴西方发达国家有关软件工程经济学的理论、方法与应用成果的同时要注意环境的差异性对数量分析的影响，从而可在数学分析的思路与方法的通用性之基础上来寻找适合于我国国情的研究成果。

2. 方法体系

根据上述分析，软件工程经济学作为一门交叉学科，其理论与方法应该与如下五类学科有着紧密的关系，它们是：

① 社会学、管理学等；② 经济学(宏观经济学、微观经济学、工程经济学、管理经济学、信息经济学等)；③ 软件工程(软件工程技术学、软件工程管理学)；④ 计算机通信网络与信息系统；⑤ 系统工程与运筹学、应用统计学、模糊数学、系统动力学等。它们之间的关联与方法体系详见图1.13。

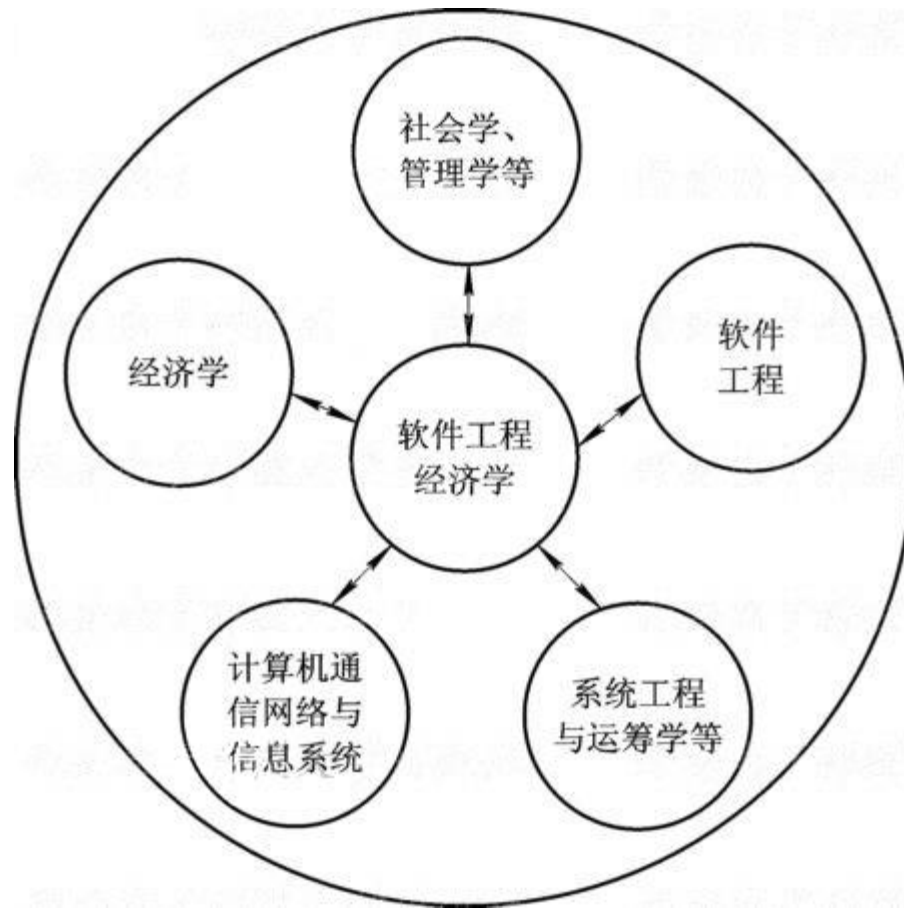


图1.13 SEE理论与方法关联图

1.3.3 软件工程经济学的研究与发展

软件工程的诞生源于“软件危机”。软件危机是指在计算机软件开发中的一系列问题。其中既有软件技术问题，如怎样开发软件？怎样维护现有的、容量又在不断增加的软件？我们怎样做才能满足人类对软件需求的不断增长等等；也有经济问题，如软件开发过程中成本和进度估计往往不精确，软件质量与可靠性的概念十分可疑，如何来处理一些相互对立的软件目标，如成本、工期、可靠性等，软件测试究竟需要多长时间才能投放市场等等。因此软件工程经济学的研究始终是伴随着软件工程的发展而前进的，而日渐成熟的应用统计学、运筹学、系统工程、工程经济学为其发展提供了科学而系统的方法论。

第1章 软件工程与软件工程经济学

早期(20世纪70年代)的软件工程经济学的研究对象均来自于计算机科学和软件工程中的范例，希望通过建造、使用工具原型来降低软件开发与维护成本，以后逐渐发展到对**软件成本、时间进度、可靠性、效率的建模、方法及其比较评价和均衡优化**，对软件开发过程的系统动力学研究以及软件企业管理中的**采购、计划、生产、销售(投放市场)、维护、报废等最优决策研究**。目前已发展到对软件工具的经济评价以及在软件开发与维护过程中的提高劳动生产率的研究。

软件工工程经济学的研究最早始于美、英等国，其中较有影响的有Boehm B.W、Putnam L.H和Banard.L等专家。

Boehm在研究成本测算的过程中提出了构造型成本模型(Constructive Cost Model, COCOMO)，给出了由软件规模计算工作量，进而确定成本与工期的经验统计模型，并于1981年出版了其专著《软件工工程经济学》(Software Engineering Economics)。在将该模型推向市场的同时，Boehm利用市场手段不断地收集用户的反馈意见，进而对模型作出不断的修正与提高，以适应软件工工程在生存周期、技术、组件、工具、表示法以及企业文化等方面的明显变化，

并提出了COCOMOII的模型与方法体系，且于2000年出版了他的第二本有重大影响的著作：《软件成本估算——COCOMOII模型方法》(Software Cost Estimation with COCOMOII)。

另一个较有影响的专家是Putnam L. H，他在Noder. P. V提出的诺顿—瑞利(Noder [CD*2]Rayleigh)曲线的基础上研究了软件开发与运行过程的系统动力学模型(1987年)，而Londeix. B则在该系统动力学模型的基础上作了进一步的扩展，并出版了他的专著：《软件开发的成本估计》(Cost Estimating of Software Develop)。软件质量与可靠性的研究是在硬件质量与可靠性研究上的继续与发展，较好的论著有Stephen H.Kan的《软件质量工程的度量和模型》


(Metrics and Model in Software Quality Engineering)(1995年), Misra.P.N的《软件可靠性分析》(Software Reliability Analysis); 在软件最优投放问题的研究中有Koch H. S and Kubat P的《计算机软件的最优投放时间》(Optimum Release Time of Computer Software)和Okumoto. K and Goel A. L的《基于可靠性和成本准则的软件系统最优释放时间》(Optimum Release Time for Software Systems Based on Reliability and Cost Criteria)。

人们注意到软件开发组织的结构与能力直接影响着软件产品的质量、工程进度和成本预算的执行和控制, 然而如何度量软件开发组织的能力始终是人们需要探讨的难题。采用质量度量国际标准制订的类似思路。

第1章 软件工程与软件工程经济学

1987年美国卡内基—梅隆(Carnegie [CD*2]Mellon)大学软件工程研究所(SEI)在Mitre公司的支持下,在美国国防部的指导下,经过广泛的调查,开发了“软件过程评估”和“软件成熟度评价”两个模型,经四年使用后,于1991年8月公布和发表了软件能力成熟度模型(Capability Maturity Model for Software, CMM)CMM v1.0。

在广泛征求了政府部门、企业界、学术界对CMM v1.0的意见后, SEI又于1993年2月发布了经修改后的CMM v1.1, 由于CMM模型的推广和应用在提高软件开发组织的自身建设, 进而达到提高产品质量, 降低产品成本, 按时履行交付承诺等目标, 因而取得了明显的经济效益和社会效益。1999年美国国防部规定, 承接美国国防部大型软件项目的承包商必须具备CMM成熟度3级的认证。



我国的软件工工程经济学的研究尚处于起步阶段。1990年和1991年由机械工业出版社相继组织出版了Boehm的著作：

《软件工工程经济学》和Londeix.B的著作：《软件开发成本估算》，对软件工工程经济学的概念、方法的宣传起到了一定的作用，在国内的一些学术刊物上陆续有一些有关软件成本测算和定价策略、软件最优投放时间、软件质量评估等方面的论文发表，个别软件企业也开始了CMM评估和认证工作。但从总体来看，软件工工程经济学的概念、理论与方法尚未为国内软件工程界所熟悉，从事软件工工程经济学专门研究的人员极少，有关软件开发的诸多工工程经济参数，如软件成本、工期、复杂性与项目难度、可靠性、劳动生产率等尚无专门机构收集、存储与分析。

因此，加大对软件工程经济学的宣传，吸引更多的人来从事软件工程经济学的研究，唤起企业界的更多需求，大力培养软件工程经济学的研究人才，成立专门机构从事软件工程经济学的研究等已成为当务之急。我们相信，在国家科技部门的领导下，在我国学术界与企业界的努力下，在不久的将来，我国软件工程经济学的理论与应用水平必将取得新的突破，以迎头赶上世界先进水平。

作业

1. 什么是软件生存周期？ 软件生存周期一般可划分为哪几个阶段？ 各阶段间有何关联？
2. 软件的规模与复杂性如何度量？ 什么是软件可靠性？ 软件可靠性有哪些度量指标？
3. 什么是软件工程经济学？ 软件工程经济学的研究内容有哪些？ 有何研究特点？ 软件工程经济学与哪些学科有较紧密的关联？
4. 你期望从本门课程中学到哪些知识？ 对上好本门课程有哪些建议？

习 题 一

1. 什么是软件？软件与其他产品相比较，具有哪些特点？软件按产品功能、规模、标准化程度以及其与其他硬/软件关联程度为准则划分，可分为哪些子类？

2. 软件企业如何分类？软件企业管理包括哪些类别管理？各子类管理的系统目标和主要管理活动有哪些？

3. 什么是软件项目？其分项管理及其相应的主要管理活动有哪些？

4. 什么是软件生存周期？软件生存周期一般可划分为哪几个阶段？各阶段间有何关联？

5. 什么是软件开发模型？你所熟知的软件开发模型有哪些？说出这些模型的主要内涵与功能。

6. 软件的规模与复杂性如何度量？什么是软件可靠性？软件可靠性有哪些度量指标？

7. 什么是软件质量？国际标准化组织(ISO)于1991年发布了软件质量度量模型，该模型的高层需求准则、中层设计准则、底层度量评价准则包括哪些内容？

8. 什么是软件质量保证？如何进行软件质量保证？

9. 什么是软件工程经济学？软件工程经济学的研究内容有哪些？有何研究特点？软件工程经济学与哪些学科有较紧密的关联？