软件工程：一系列系统的（systematic）规范的（disciplined）可量化的（quantifiable）方法，来开发运行维护software

层次化技术：quality focus、process model（流程）、method、tools

Process framework：框架活动和普适性活动（贯穿）

Communication：user（使用者）/customer（出钱方），customer和user需求不同；组员 [功能需求与非功能需求]

Planning

Modeling

Construction：编码/测试

Deployment：反馈 - 改进

Process model

The essence of practice精髓

Understand the problem（stakeholder利益相关方）

Plan a solution

Carry out the plan

Examine the result for accuracy

General principles：价值、简单、灵活、关注使用者、面向未来、复用（reuse）、Think

Process flow

Linear process flow 五个流程顺序执行

Iterative process flow

Evolutionary process flow

Parallel process flow

Process pattern（模式）：设计的体系风格

Process assessment：流程评价

CMMI Level 0 - level 6

初始级

受管理级，有计划有评估

已定义级，有标准

定量管理级，过程可度量，预测产品

持续优化级，评估提升标准与产品

模型

瀑布模型：顺序完成软件（有问题累计的隐患）

V-model

Incremental process model（增量模型）--- 堆砌

核心需求优先，不断迭代产品

Rapid application development（RAD）模块化推进，core与功能分离

Evolutionary process model（演化模型）--- 雕琢

每一轮迭代都有完整的产品（功能是完整的，效用不一定正常）

螺旋式

Concurrent process model

Specialized process model

组件重用

Project management（软件管理）

People

Stakeholder 利益相关方

senior managers业务规则

Project manager 项目管理者

Practitioner 实施者/执行者

Customer 需求（实现）确定者

End-user 使用者

Product 产品/文档-代码（范畴明晰）--- 善于分解问题

Context 边界

Information objectives input/output

Function and performance 功能及性能

Reliability interface security

Process 框架活动及任务执行

任务阶段性及安排

Project 项目本身

变更管理

正确的基础上开始

保持动力

提供足够的激励机制

跟踪进展

明智决策

事后分析

Team leader（MOI model）

Motivation 激发潜力（鼓动人心）

Organization 组织能力（工作与人能力匹配），布局 --- 备份机制

Ideas or innovation 创造力/创新

Organizational paradigms

Closed paradigm 层次化结构

Random paradigm 扁平化结构

Open paradigm 开放式结构

Synchronous paradigm 并行化结构

Select team structure

问题难易程度

项目规模

待在一起的时长

系统可靠程度

项目周期

Critical practices

风险管理

基于指标的工程管理

项目周期与估算

Effective software team attributes

团队参与度

PM 项目经理

QA 质量保障工程师

RE/TTL/AE ...

敏捷方法

团队及过程是可适应性的

XP （结对编程）

确保质量

云形式办公

软件工程管理原则（多方面）

Agile， quality， adapt， effective， risk

分而治之，抽象，设计一致，现成参考（look for pattern）

Listen，prepare，face to face

了解问题/约束/资源，融入客户，风险与预案

设计可回溯，体系结构，界面，数据设计

需求分析（业务需求，用户需求，系统需求）

起始（inception）

系统domain（对问题的基本了解）

面向people（要求特点，不同视角观察）---- 确认利益相关方

共同需求与需求冲突的折中

解决方案的本质（软件的核心环节）

交流合作的意识与机制

NFR（non functional requirement）非功能性需求

操作要求，环境要求，系统需求

获取（elicitation）阶段，获取需求（所有的沟通过程）

通过会议获取需求

确认问题

功能大概组成

需求中问题的协商

Quality function deployment（QFD）质量功能部署

用户价值方面考虑

三大功能

Normal requirement

Expected requirement

Exciting requirement（期望需求）

细化（elaboration），需求不断细化，分析模型

用例图（use-case），表达交互关系 --- 功能

角色与参与者，交互中的信息与输入输出

基于场景 / 基于类（面向对象） / 行为元素（状态- -函数）

借鉴别人的需求分析模型

协商（negotiation），版本与功能实现的对映

规格说明（specification），数学模型，用例与场景（系统性雏形）

确认（validation），错误，功能冲突

需求与目标一致

Requirement是否已经完成抽象

需求边界清晰

需求管理（requirement management），需求变更管理

三类元素

场景元素

Domain analysis（重用成熟的需求分析）

Use-cases 场景中用户与系统的交互

Review机制，考虑到所有的actor交互，交互是否正确等

Activity diagram活动图（类似于流程图）-- 输入/判断/并发/分叉/起止

泳道图，活动根据角色进行分组（既有过程图，又分角色显示）

基于类的对象（包括类之间的关系）

结构化

面向对象，特别注意协同关系

CRC模型，通过描述，构建类与类之间的关系

行为元素

数据流模型

数据流图

控制流图

经验法则

抽象，简化，解耦

Behavior modeling

外部交互响应

交互顺序，有哪些事件，通过输入刺激系统

状态图：State、state transition event action

Sequence diagram（类似于泳道图）

结构化设计系统

Data flow diagram（DFD）



Process（data 本身操作过程）

External entity 谁来与data交互

Pattern 已有的借鉴与沿用

Web app的需求分析

Content，内容的具体信息形式

内容建模，模块化与对象选择（属性 / 关系），data tree表现

Interaction，交互分析（功能与操作）| 界面原型

Functional，核心部分（提供什么功能与价值）

Configuration，配置分析，硬件软件要求 / 内容配置（部署）

Client side和server side（异构性）

Navigation，导航分析（核心功能的简便寻找）

质量

价值角度，也有使用支付意愿来评价

Quality of design 软件设计角度

Quality of conformance 功能实现符合性角度

软件质量：可由有效软件过程来为用户开发一个有用的产品，且这个产品可以为用户提供可度量的价值

Effective software process

Management aspects of process 关键点检测

Software engineering practice 文档、代码等过程质量

Useful product

可期待的内容、价值，关键：可信赖，可靠的系统

满足用户明确提出的需求 -> 额外需求

Value

用户、组织等多维价值

软件质量维度

Performance quality 性能质量，功能层面

Feature quality 特征特性

Reliability 可靠性

Conformance 一致性，符号标准，习惯层面

Durability 持久性（维护性）

Serviceability 适用性（可用性）

Aesthetics 审美

Perception 感知

Good enough即可上线，只要可以覆盖核心功能即可上线

Cost of quality

Prevention cost 预防成本

Evaluation cost 评估成本

Failure cost 失效成本

决策也对quality 产生影响

Estimation decision 估计决策

Scheduling decision 规划决策

Risk-oriented decision 风险应对决策

提升quality

Good project management

Solid practice

了解问题本身

消除缺陷 / bug

做好项目管理与版本控制

Design concepts 设计

好的设计

Firmness 可靠

Commodity 有用，合适

Delight 令人愉悦的

Principles 设计原则，concepts了解概念，practices 设计方法（具体化）

将分析类转化为实现类（数据结构）

Analysis model -> design model（类似于金字塔结构）

体系结构，定义系统中主要元素间关系

接口设计

具体的类设计

设计要求：

满足所有需求

可理解可读

完整视图

相关概念：

抽象

整体结构

模式

SOC 关注点分离

模块化

信息隐藏

功能解耦

精细化

面向方面的变成

重构

Architectural design

体系结构设计

Architectural：系统的结构，包括软件组件、组件的关系（依赖或调用）、组件的属性

体系结构优势：

设计是其中的一种实例

减小风险

评估设计过程

重要性：

提供表达方式，沟通载体

影响重大

整体表示

体系结构描述（AD）：

建立概念性框架和词汇表

准确描述

体系结构类型（architectural genres）：整个软件领域中（含有子类别）的特定类别

体系结构风格：

执行系统功能所需的组件

组件间联系合作的连接器

构成的约束条件

语义模型

具体风格：

Data centered结构（数据中心结构）

Data flow结构（数据流体系结构）--- 数据在系统中向一个方向流动

Pipes + fliters（数据单方向流动，过滤器之间独立）

退化为仅有顺序过滤器 batch形式

Call and return架构 主程序/子程序体系结构；远程过程调用体系结构

Object-oriented 结构，将数据/模块/系统看作对象，数据与处理模型合二为一，对象间通过消息传递来介导彼此通信

Layered 结构，分层结构；从core逐渐向外扩展，tool层，应用层，用户接口层

Service-oriented结构，定义多个服务接口供软件组件调用

Web service三要素：SOAP描述传递信息的格式、WSDL基于XML描述Web service、UDDI管理分发查询web service

体系结构模式（软件具有的特有属性）：

Concurrency 并发性

Persistence 驻留（数据驻留、应用层驻留）

Distribution 分布式

架构的上下文

ACD

上级系统，调用

下级系统，被调用

同级系统，相互调用等

参与者

原型，定义改进组件

体系结构决策的参考原则：economy（简洁），visibility（清晰），spacing（分离关注点），symmetry（一致性与平衡性），emergence（紧急可处置的）

敏感点：变更会对系统带来较大影响的属性

体系结构的复杂度：

共享依赖性

流依赖性

受约束依赖性

ADL：体系结构描述语言

体系结构评审

基于经验的推理：新的体系结构与过往体系结构对比

原型评估

基于模式的体系结构评审