-软件危机的原因。

软件规模加大，复杂性变高。

软件是逻辑产品，尚未完全认识其本质和特点。

缺乏有效的开发，维护大型项目的技术和方法。

用户与开发人员对于需求不统一，经常更改。

软件开发的技术人员和管理人员缺乏软件工程化的素质和要求，对工程化的开销认识不足。

-软件危机包含的问题

**如何开发软件，以满足不断增长，日趋复杂的要求，**

**如何维护数量不断膨胀的软件产品**

软件= 程序+数据+文档

程序= 算法+数据结构

-SE和CE有什么区别？

一个偏工程，一个偏理论

-怎么认识质量？

**超越的观点：可以被认识不能定义，没有最好只有更好；**

**制造的观点：质量是与需求说明的一致**

**产品的观点：质量是与产品的内在特性相联系的。**

基于价值的观点：用户愿意花多少钱买（意义不大对于软件工程，因为是在工程完毕后才产生的）

-影响软件工程的7大因素。

**交付时间的重要性**

**计算行业经济的变化趋势**

**功能强大的桌面运算**

**互联网络**

**面向对象的技术**

**图形用户界面**

**瀑布模型的不可预知性**

-什么是过程

过程是指为了达到给定目的而执行的实践的集合。

-软件生命周期：

三个时期：软件定义，软件开发，软件运行。

七个阶段：**问题定义，可行性分析，需求分析，系统设计，编码，测试，维护**。

-软件生命周期各阶段产品。

可行性报告，软件需求规格说明书SRS，用户需求定义文档，确认测试计划

概要设计规格说明书，数据结构设计说明书，集成测试计划

详细设计规格说明书，单元测试计划，源代码

单元测试报告，集成测试报告，满足概要设计要求的程序

可供用户使用的软件产品（文档和源程序）

-单元测试，集成测试，确认测试各自的任务。

单元测试：对每个功能模块进行测试

集成测试：测试各模块连接是否正确以及系统功能和性能是否满足要求

确认测试：用户参与，以SRS为依据，检查软件产品是否满足了用户需求。

-过程模型

**沟通，策划，建模，构建，部署**

-瀑布模型的优缺点。

优点：**以软件生命周期为模型，使得开发过程具备系统性，可控性克服了随意性。**

缺点：**人的认知是不断加深的，用户和开发人员会不断加深对于项目的理解**，需要进行更改迭代，后期如果进行修改，代价太大，而且线性开发模式胡造成**阻塞状态**。

-形式化开发方法

**用严格的、数学的符号体系来规约、开发和验证基于计算机的系统。**

解决软件开发过程使用其它范型难以克服的二义性、不完整性和不一致性。

可以产生无缺陷软件的承诺。

**费时、昂贵、难沟通，需要培训**

**是建造重要的、安全的软件的开发者可以考虑开发范型。比如：航空电子、医疗设备软件的开发。**

-RUP的核心思想

**用例驱动、以体系结构为核心、迭代及增量。**

-RUP模型的四个阶段。

**先启** **精化** **构建** **产品化**

PS：这四个阶段和软件生命周期是二维的，因为有多个增量和多次迭代，所以进行的每次迭代，增加的每个增量都是一次软件生命周期。（不太知道对不对）

-RUP模型和瀑布模型的区别。

瀑布模型被包含在RUP模型中，每个增量都可以使用瀑布模型。（不太知道对不对）

-先启阶段的主要工作。

**确定项目开发的目标和范围**

**定义主要的需求**：用例以及主要的用例场景

根据一些主要的用例场景来**构建一个基本架构**

**估算开发周期和成本估计潜在的风险**

主要实践活动－用例建模

-精化阶段的主要工作

精化阶段不是一个需求或设计阶段，**而是一个迭代实现核心架构并降低高风险的阶段**

在细化前不需要定义大多数需求，**10％的详细用例书写出来即可**

**先处理具有风险的元素**，开始实际产品代码的编写，产生可执行架构

除了极少数理解良好的**问题**外**需要多次迭代**

-构建阶段的主要工作。

**尽快完成软件产品的开发**

**迭代实现遗留下来的风险较低和比较容易的元素**，准备部署

在保证开发进度的同时达到足够的软件质量

-转化阶段

获得涉众的认同：**产品部署已经完成并且满足预定的质量标准**

**尽快达到最终稳定的产品基线**

-拿到项目先做什么？

**项目分解**

-WBS是什么？

Work breakdown structure，将项目分解为足够小而且清晰，可交付的工作任务。

-简述WBS叶子节点

可执行，可落地的工作任务，并且有责任人，工作时长，和产品。

-叶子节点多长时间合适？

4个小时

-WBS两种分解方式。

面向工作过程，面向可交付成果。

-对效率影响的因素：工具、方法、技术or 人员？

**人员。**

-最好的程序员有何特征，即经验重要还是基础重要？

**经验重要。**

-项目人员包括？

项目管理人员，高级管理人员就是领域专家，开发人员（码农），客户，最终用户。

-如何选择人员？

**开展工作的能力**

**工作兴趣**

**经验**（类似应用，类似工具，类似技术，类似开发环境）

**培训**（不太懂，经历过培训？）

**与人交流能力**

**共同承担责任能力**

**管理能力**

-LOC和FP各自的优缺点。

LOC 优点：直观，简单；

缺点：要在代码写完后才能知道，不能事先估算，与程序设计语言相关，对于设计

精巧的代码不友好。

FP 优点：可以通过**用户的观点**事先估算，与程序设计语言无关。

缺点：不够直观，涉及主观因素比较多，不一定准确，每个人估算的可能不一样，

没有具体的物理意义，信息领域中的某些数据有时不容易采集。

-FP功能点分类

**数据类型功能点**（ILF 内部逻辑文件，EIF 外部接口文件）

**人机交互类型功能点**（EI 外部输入 EO外部输出 EQ外部查询）

-EO和EQ的区别

EO是经过逻辑处理的数据或信息，EQ是直接提取

-DET和RET的定义。

数据元素类型(Data Element Types DET)是一个**用户可识别的、唯一性的、非递归**的域。（**属性**）

记录元素类型(Record Element Types RET)是ILF或者EIF中用户能够识别的数据元素小组（**集合**）

-怎么计算功能点

**识别功能点类型**

**识别应用程序的边界范围**

**计算数据类型功能点提供的未调整的功能点数量（查表）**

**计算人机交互功能所提供的未调整的功能点数量（查表）**

**确定调整因子（根据系统特性表进行打分）**

**计算调整后的功能点**。

-成本估算

专家判定，算法模型，机器学习。

-可靠性估算

错误植入法

分别测试法

软件平均故障间隔时间估算

-进度估算的目标

确保软件项目在规定的时间内按期完成。

-进度编制的技术

关键路径法（CPM） 计划评审技术（PERT）图形评审技术（GERT）

CPM使用最大可能估计，PERT使用期望值估计

-什么是风险？

**可能发生的结果与预期结果之间的差异**，差异越大，风险越大。

-风险后果是什么？风险成本是什么？

**风险后果：与该事件有关的损失**。

风险概率：风险发生的可能性

**风险成本=风险后果x风险概率**

-风险管理包含的四个部分

**风险标识，风险预测，风险评估，风险管理与监控**

风险标识：列出可能存在的风险

风险预测：建立一个尺度，评价每种风险发生的可能性，以及导致的后果。

风险评估：进一步审查风险，定义一个风险参考水准，发现风险太高，就终止项目

风险管理与监控：辅助项目建立处理风险的策略（风险避免，风险监控，风险管理及监控计划）

-用户的观点为什么不用在质量管理上

因为用户的观点是在产品交付后，参考价值不大。

-审计是什么?

审计是查看是否按照规范。

-QC是什么

建立一个反馈循环，通过反馈进行不断地调整

所有产品都定义好可度量的规约。

-QA是什么

软件质量保证（SQA）是建立一套有计划，有系统的方法，来向管理层保证拟定出的标准、步骤、实践和方法能够正确地被所有项目所采用。

-QA和QC的区别

QC会对产品进行调整，而QA只负责审计整个过程是否符合规范，并向管理层报告。（不保证正确）

-软件质量评价特性举例

正确性，可靠性，易用性，健壮性，可移植性。

-软件配置是什么

软件工程过程中产生的所有信息项的某一时刻的瞬间影响。

-软件配置管理是什么

协调软件开发，使得混乱减小的技术。

-敏捷是什么，敏捷开发为什么迅速发展

过软件工程的出现使得敏捷开发有了必要性，过度的规范使得软件开发延期甚至失败。

软件开发和工业制造不一样，人要有创造性和能动性，这和机器是不一样，**所以要把人放在第一位**。

软件工程的最初目的是为了解决软件危机，不能为了软件工程而软件工程，本身最初是为了提高软件质量和开发效率。**站在客户角度，尽快交付产品**。

**敏捷就是持续的，快速的开发**。

**个体和交互** 胜过 过程和工具

**可以工作的软件** 胜过 面面俱到的文档

**客户合作** 胜过 合同谈判

**响应变化** 胜过 遵循计划

-XP模型的特点是什么

一系列相互依赖的实践组成。实践是行之有效，经过证明的。

-time box不止一个，迭代完也要有。开会

-敏捷开发中，如果增量不通过，则将问题作为新的素材进行开发，**不能反馈，回退**。

-XP和scrum模型的异同点？

相同：**迭代**

不同：**人的管理，有无master**。

-**软件工程硕士的能力应该主要体现在分析和设计。**

-需求工程是什么

捕获客户的**真实需求**，和用**没有歧义**的语言进行描述。

-需求是什么

需求是系统的特征，或者**为了实现系统目标，系统能做什么**的一个描述。

-获取需求的四个阶段

**需求导出，分析建模，规格说明，需求确认和校验。**

-需求协同收集-准备哪些列表+

对象列表：构成系统的对象。

服务列表：服务操作或与对象交互的服务（各种服务功能）

约束列表：约束条件（成本，时间，业务规则等）

性能列表：性能指标（响应速度，存储能力）

-确定利益共同者

**获取不同的观点。**

-识别多种观点

尽可能记录下来对于系统的描述。

查看相似软件以及客户已有软件（方便客户的旧有习惯）

学习领域和业务知识。

-需求准备工作的首次会议是由什么驱动的

**问题**

-结构化的核心

**数据字典，ER图，DFD图，STD图**

-需求确认和校验是什么

**确认是看用户需求是否srs都有了**

**校验是需求定义文档和srs是否一致**

-为什么要分析建模

可以从不同的角度， 不同的抽象级别**精确**地说明对问题的理解，对目标软件的需求。

模型可以帮助用户和分析人员排除用户不一致，不合理的部分，挖掘潜在的用户要求。

模型是形成srs和进行软件设计的基础。

-过程的观点是什么

**数据和逻辑是独立分离的**，世界的演变是数据在逻辑的作用下发生改变的过程。

过程有着明确的输入，输出，开始和结束，每个过程都有严格的因果关系

过程是相对稳定，明确的，小过程组成大过程。

-结构化方法建模有哪些模型

**数据字典（核心），ER图，DFD图，STD图**，注意还有**说明**。

-为什么要抽象，为什么要分解

因为一开始无论用户还是开发人员对于需求都是很模糊的，所以要先抽象。

只有通过不断地分解，使得抽象变得具体，最终可以被执行。

-数据流组成？

-并不是所有的的系统都要画std图，只有系统很复杂，有多个状态时，需要std