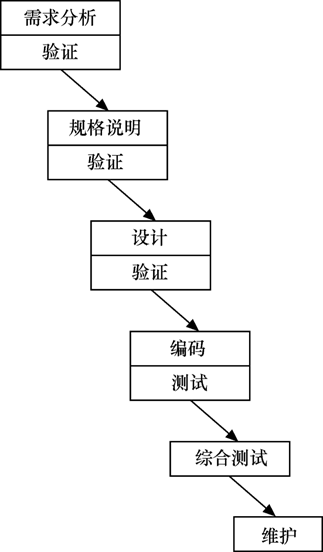
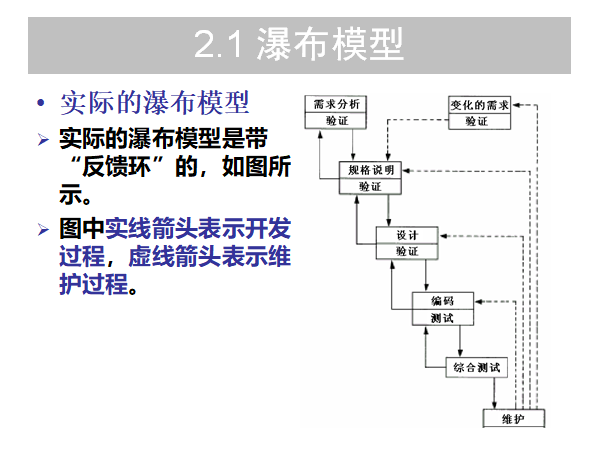
# 软件工程概论1-2

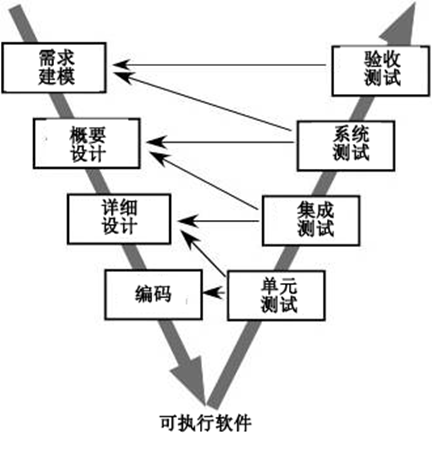
1. 一、软件与软件工程的概念
   1. 1.软件的概念、特点和分类
      1. 软件的概念
         1. 软件是计算机系统中与硬件相互依存的另一部分，它是包括程序，数据及其相关文档的完整集合。
         2. 程序是按事先设计的功能和性能要求执行的指令序列。
         3. 数据是使程序能正常操纵信息的数据结构。
         4. 文档是与程序开发，维护和使用有关的图文材料。
      2. 软件的特点
         1. (1)形态特性: 软件是无形的、不可见的逻辑实体。度量常规产品的几何尺寸、物理性质和化学成分对它却是毫无意义的。
         2. (2)智能特性：软件是复杂的智力产品，它的开发凝聚了人们的大量脑力劳动，它本身也体现了知识实践经验和人类的智慧，具有一定的智能。它可以帮助我们解决复杂的计算、分析、判断和决策问题。
         3. (3)开发特性: 尽管已经有了一些工具（也是软件）来辅助软件开发工作，但到目前为止尚未实现自动化。软件开发中仍然包含了相当份量的个体劳动，使得这一大规模知识型工作充满了个人行为和个人因素。
         4. (4)质量特性：目前还无法得到完全没有缺陷的软件产品 。
         5. (5)生产特性:与硬件或传统的制造业产品的生产完全不同，软件一旦设计开发出来，如果需要提供多个用户，它的复制十分简单，其成本也极为有限 。
         6. (6)管理特性：由于上述的几个特点，使得软件的开发管理显得更为重要，也更为独特 。
         7. (7)环境特性:软件的开发和运行都离不开相关的计算机系统环境，包括支持它的开发和运行的相关硬件和软件。软件对于计算机系统的环境有着不可摆脱的依赖性。
         8. (8)维护特性：软件投入使用以后需要进行维护，但这种维护与传统产业产品的维护概念有着很大差别。
         9. (9)废弃特性: 与硬件不同，软件并不是由于被“用坏”而被废弃的 。
         10. (10)应用特性：软件的应用极为广泛，如今它已渗入国民经济和国防的各个领域，现已成为信息产业、先进制造业和现代服务业的核心，占据了无可取代的地位。
      3. 软件的分类
         1. （1） 系统软件
            1. 操作系统
            2. 数据库管理系统
            3. 设备驱动程序
            4. 通信和网络处理程序等
         2. （2）支撑软件（工具软件）
            1. 纵向支撑软件：分析、设计、编码、测试工具等
            2. 横向支撑软件：项目管理工具，配置管理工具等
         3. （3）应用软件
            1. 工程与科学计算软件
            2. 商业数据处理软件
            3. ERP软件
            4. 计算机辅助设计／制造软件
            5. 系统仿真软件
            6. 智能产品嵌入软件
            7. 事务管理、办公自动化软件
         4. （4）可复用软件
            1. 标准函数库、类库、构件库等
   2. 2.软件危机与软件工程
      1. 软件危机主要有以下一些典型表现:
         1. 对软件开发成本和进度的估计常常很不准确。
         2. 用户对“已完成的”软件系统不满意的现象经常发生。
         3. 软件产品的质量往往靠不住。
         4. 软件常常是不可维护的。
         5. 软件通常没有适当的文档资料。
         6. 软件成本在计算机系统总成本中所占的比例逐年上升。
         7. 软件开发生产率提高的速度，既跟不上硬件的发展速度，也远远跟不上计算机应用迅速普及深入的趋势。
      2. 除了软件本身的特点，软件危机发生的主要原因有：
         1. 缺乏软件开发的经验和有关软件开发数据的积累，使得开发工作的计划很难制定。
         2. 软件人员与用户的交流存在障碍，使得获取的需求不充分或存在错误。
         3. 软件开发过程不规范。如，没有真正了解用户的需求就开始编程序。
         4. 随着软件规模的增大，其复杂性往往会呈指数级升高。需要很多人分工协作，不仅涉及技术问题，更重要的是必须有科学严格的管理。
         5. 缺少有效的软件评测手段，提交用户的软件质量不能完全保证。
      3. 如何摆脱软件危机?
         1. 彻底消除“软件就是程序”的错误观念。
         2. 充分认识到软件开发应该是一种组织良好、管理严密、各类人员协同配合、共同完成的工程项目。
         3. 推广和使用在实践中总结出来的开发软件的成功技术、方法和工具。
         4. 按工程化的原则和方法组织软件开发工作。
   3. 3.软件工程的目标
      1. 基本目标：
         1. 付出较低的开发成本
         2. 达到要求的软件功能
         3. 取得较好的软件性能
         4. 开发的软件易于移植
         5. 需要较低的维护费用
         6. 能按时完成开发工作，及时交付使用
      2. 软件的质量特性：功能性、可靠性、可使用性、效率、可维护性和可移植性。
         1. 功能性是指软件所实现的功能达到它的设计规范和满足用户需求的程度；
         2. 可靠性是指在规定的时间和条件下，软件能够正常维持其工作的能力；
         3. 可使用性是指为了使用该软件所需要的能力；
         4. 效率是指在规定的条件下用软件实现某种功能所需要的计算机资源的有效性；
         5. 可维护性是指当环境改变或软件运行发生故障时，为了使其恢复正常运行所做努力的程度；
         6. 可移植性是指软件从某一环境转移到另一环境时所做努力的程度。
   4. 4.软件生存期
      1. 软件生存期分为三个时期
         1. 软件定义时期
            1. 问题定义
            2. 可行性研究
            3. 需求分析
         2. 软件开发时期
            1. 概要设计
            2. 详细设计
            3. 编码（程序编码和单元测试）
            4. 测试（集成测试和系统测试）
         3. 运行维护时期
            1. 改正性维护，也就是诊断和改正在使用过程中发现的软件错误；
            2. 适应性维护，即修改软件以适应环境的变化；
            3. 完善性维护，即根据用户的要求改进或扩充软件，使它更完善；
            4. 预防性维护，即修改软件为将来的维护活动预先做准备
   5. 5.软件工程方法三要素及作用
      1. 概念
         1. 软件工程包含技术和管理两方面的内容，是技术和管理紧密结合所形成的工程学科。
         2. 通常将软件开发全过程中使用的一整套技术方法的集合称为方法学(methodology)，也称为范型(paradigm)。
         3. 目前使用最广泛的软件工程方法学：传统方法学（结构化方法学），面向对象方法学。
      2. 三要素：方法、工具和过程。
         1. 软件工程方法为软件开发提供了 “如何做” 的技术;
         2. 软件工具为软件工程方法提供了自动的或半自动的软件支撑环境;
         3. 过程是为了获得高质量的软件所需要完成的一系列任务框架，它规定了完成各项任务的工作步骤。
      3. 结构化方法学
         1. 也称为生命周期方法学或结构化范型。将软件生命周期的全过程依次划分为若干个阶段，采用结构化技术来完成每个阶段的任务。
         2. 特点：
            1. (1) 强调自顶向下顺序地完成软件开发的各阶段任务;
            2. (2) 结构化方法要么面向行为，要么面向数据，缺乏使两者有机结合的机制。
      4. 面向对象方法学
         1. 是将数据和对数据的操作紧密地结合起来的方法。
         2. 软件开发过程是多次反复迭代的演化过程。
         3. 面向对象方法在概念和表示方法上的一致性，保证了各项开发活动之间的平滑过渡。
         4. 对于大型、复杂及交互性比较强的系统，使用面向对象方法学更有优势。
2. 二、软件生存期模型
   1. **1.瀑布模型**
      1. 特点：
         1. （1）阶段间具有顺序性和依赖性。其中包含两重含义：
            1. ① 必须等前一阶段的工作完成之后，才能开始后一阶段的工作；
            2. ② 前一阶段的输出文档就是后一阶段的输入文档。
         2. （2）推迟实现的观点
            1. ① 瀑布模型在编码之前设置了系统分析和系统设计的各个阶段，分析与设计阶段的基本任务规定，在这两个阶段主要考虑目标系统的逻辑模型，不涉及软件的物理实现。
            2. ② 清楚地区分逻辑设计与物理设计，尽可能推迟程序的物理实现，是按照瀑布模型开发软件的一条重要的指导思想。
         3. （3）质量保证的观点
            1. ① 每个阶段都必须完成规定的文档，没有交出合格的文档就是没有完成该阶段的任务。
            2. ② 每个阶段结束前都要对所完成的文档进行评审，以便尽早发现问题，改正错误。
      2. 图
         1. 瀑布模型



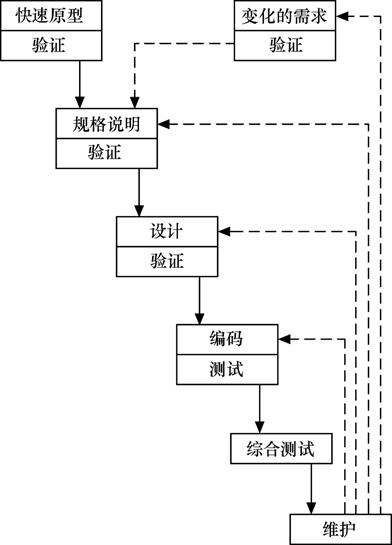
* + - 1. 实际的瀑布模型



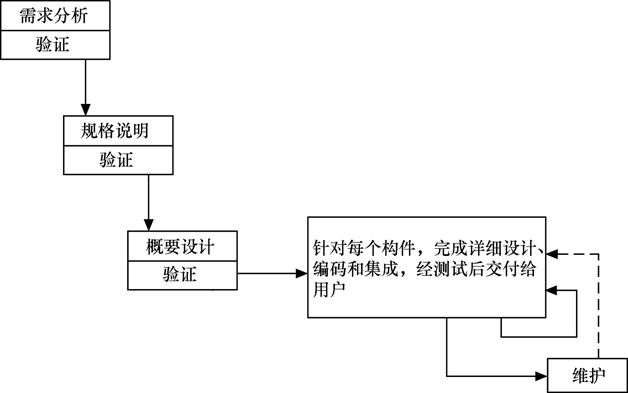
* + 1. V模型：瀑布模型的一个变体
       1. V模型描述了测试阶段的活动与开发阶段相关活动（包括需求建模、概要设计、详细设计、编码）之间的关系。



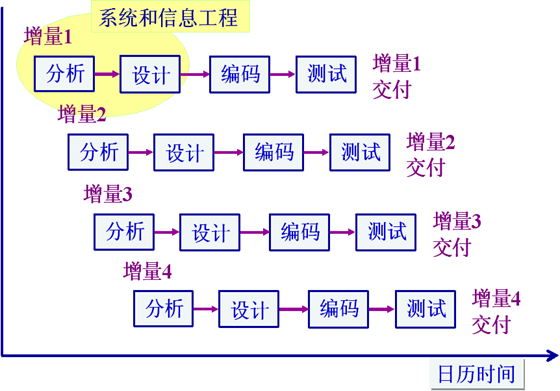
* + 1. 优点：
       1. 可强迫开发人员采用规范化的方法。
       2. 严格地规定了每个阶段必须提交的文档。
       3. 要求每个阶段交出的所有产品都必须是经过验证的。
    2. 缺点：
       1. 瀑布模型的缺点
       2. 由于瀑布模型几乎完全依赖于书面的规格说明，很可能导致最终开发出的软件产品不能真正满足用户的需要。如果需求规格说明与用户需求之间有差异，就会发生这种情况。
       3. 瀑布模型只适用于项目开始时需求已确定的情况。
  1. **2.快速原型模型**
     1. 特点：
        1. 快速原型是快速建立起来的可以在计算机上运行的程序，它所能完成的功能往往是最终产品能完成的功能的一个子集。
     2. 图：



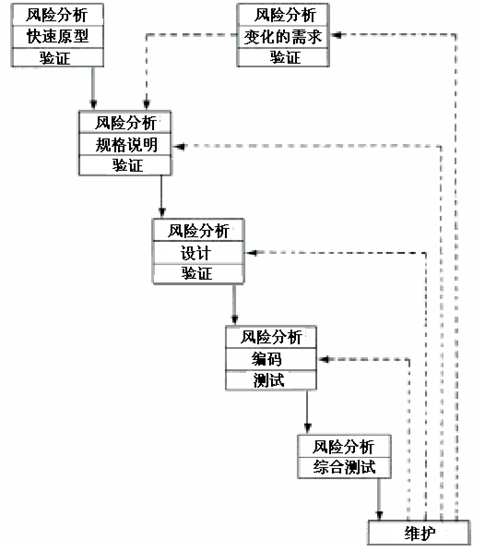
* + 1. 优点：
       1. (1)有助于满足用户的真实需求。
       2. (2)原型系统已经通过与用户的交互而得到验证，据此产生的规格说明文档能够正确地描述用户需求。
       3. (3)软件产品的开发基本上是按线性顺序进行。
       4. (4)因为规格说明文档正确地描述了用户需求，因此，在开发过程的后续阶段不会因为发现规格说明文档的错误而进行较大的返工。
       5. (5)开发人员通过建立原型系统已经学到了许多东西，因此，在设计和编码阶段发生错误的可能性也比较小，这自然减少了在后续阶段需要改正前面阶段所犯错误的可能性。
       6. (6) 快速原型的突出特点是“快速”。开发人员应该尽可能快地建造出原型系统，以加速软件开发过程，节约软件开发成本。
       7. 原型的用途是获知用户的真正需求，一旦需求确定了，原型可以抛弃，当然也可以在原型的基础上进行开发。
  1. **3.增量模型**
     1. 特点：
        1. 增量模型也称为渐增模型，是Mills等于1980年提出来的。
        2. 使用增量模型开发软件时，把软件产品作为一系列的增量构件来设计、编码、集成和测试。
        3. 每个构件由多个相互作用的模块构成，并且能够完成特定的功能。
     2. 图：



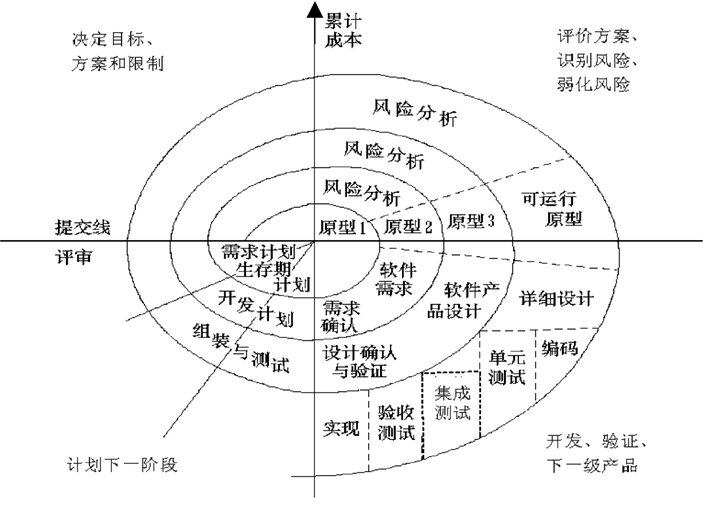
* + 1. 优点：
       1. (1)能在较短时间内向用户提交可完成一些有用的工作产品，即从第1个构件交付之日起，用户就能做一些有用的工作。
       2. (2)逐步增加产品的功能可以使用户有较充裕的时间学习和适应新产品，从而减少一个全新的软件可能给用户组织带来的冲击。
       3. (3)项目失败的风险较低，虽然在某些增量构件中可能遇到一些问题，但其他增量构件将能够成功地交付给客户。
       4. (4)优先级最高的服务首先交付，然后再将其他增量构件逐次集成进来。因此，最重要的系统服务将接受最多的测试。
    2. 问题：
       1. (1)在把每个新的增量构件集成到现有软件体系结构中时，必须不破坏原来已经开发出的产品。
       2. (2)软件体系结构必须是开放的，即向现有产品中加入新构件的过程必须简单、方便。
       3. 因此，采用增量模型比采用瀑布模型和快速原型模型更需要精心的设计。
    3. 增量构件开发
       1. 每个增量构件应当实现某种系统功能，因此增量构件的开发可以采用瀑布模型的方式，如图所示。



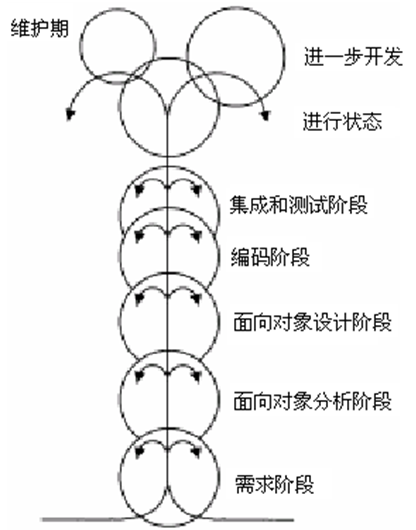
* 1. **4.螺旋模型**
     1. 特点：
        1. 螺旋模型最初是Boehm于1988年提出来的。
        2. 该模型将瀑布模型与快速原型模型结合起来，并且加入两种模型均忽略了的风险分析。
        3. 螺旋模型的基本思想是，使用原型及其他方法来尽量降低风险。
        4. 理解这种模型的一个简便方法，是把它看做在每个阶段之前都增加了风险分析过程的快速原型模型。
     2. 图
        1. 图



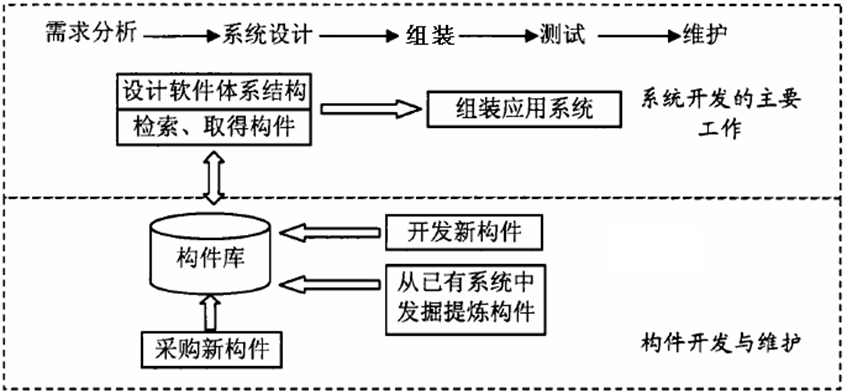
* + - 1. 完整的螺旋模型



* + 1. 完整的螺旋模型
       1. 在螺旋模型中，软件过程表示成一个螺线，而不是像以往的模型那样表示为一个具有回溯的活动序列。
       2. 在螺线上的每一个循环表示过程的一个阶段。
       3. 每个阶段开始时的任务是确定该阶段的目标、为完成这些目标选择方案及设定这些方案的约束条件。接下来的任务是，从风险角度分析上一步的工作结果，努力排除各种潜在的风险，通常用建造原型的方法来排除风险。如果成功地排除了所有风险，则启动下一步开发步骤，在这个步骤的工作过程相当于纯粹的瀑布模型。最后是评价该阶段的工作成果并计划下一个阶段的工作。
    2. 螺旋模型的4项活动 ：
       1. 螺线上的每一个循环可划分为4个象限，分别表达了4个方面的活动。
       2. (1)目标设定——定义在该阶段的目标，弄清对过程和产品的限制条件，制订详细的管理计划，识别项目风险，可能还要计划与这些风险有关的对策。
       3. (2)风险估计与弱化——针对每一个风险进行详细分析，设想弱化风险的步骤。
       4. (3)开发与验证——评价风险之后选择系统开发模型。
       5. (4)计划——评价开发工作，确定是否继续进行螺线的下一个循环。如果确定要继续，则计划项目的下一个阶段的工作。
    3. 优点：
       1. 对可选方案和约束条件的强调有利于已有软件的重用，也有助于把软件质量作为软件开发的一个重要目标。
       2. 减少了过多测试或测试不足所带来的风险。
       3. 在螺旋模型中维护只是模型的另一个周期，因而在维护和开发之间并没有本质区别。
    4. 缺点：
       1. 螺旋模型是风险驱动的，因此要求软件开发人员必须具有丰富的风险评估经验和这方面的专门知识，否则将出现真正的风险：当项目实际上正在走向灾难时，开发人员可能还以为一切正常。
  1. **5.喷泉模型**
     1. 特点：
        1. 喷泉模型是典型的面向对象生命周期模型。
        2. “喷泉”一词体现了迭代和无间隙特性。图中代表不同阶段的圆圈相互重叠，这明确表示两个活动之间存在重叠。
     2. 图：



* 1. 6.统一过程
     1. 统一过程的工作流
        1. ① 业务建模工作流。用商业用例为商业过程建立文档。
        2. ② 需求工作流。目标是描述系统应该做什么，确保开发人员构建正确的系统。为此，需明确系统的功能需求和非功能需求（约束）。
        3. ③ 分析和设计工作流。其目标是说明如何做。结果是分析模型和设计模型。
        4. ④ 实现工作流。用分层的方式组织代码的结构，用构件的形式来实现类，对构件进行单元测试，将构件集成到可执行的系统中。
        5. ⑤ 测试工作流。验证对象之间的交互、是否所有的构件都集成了、是否正确实现了所有需求、查错并改正。
        6. ⑥ 部署工作流。制作软件的外部版本、软件打包、分发、为用户提供帮助和支持。
     2. 统一过程的阶段：统一过程有4个阶段，分别是初始阶段、细化阶段、构造阶段和移交阶段。
        1. ① 初始阶段。初始阶段主要关注项目计划和风险评估，其目的是确定是否值得开发目标信息系统。
        2. ② 细化阶段。细化阶段关心定义系统的总体框架，其目标是：细化初始需求（用况）、细化体系结构、监控风险并细化它们的优先级、细化业务案例以及制订项目管理计划。
        3. ③ 构造阶段。构造阶段是建立系统，构造信息系统的第1个具有操作质量的版本，以能够交付给客户进行b测试的版本结束，有时称为测试版本。
        4. ④ 移交阶段。移交阶段包含测试时期，以发布完整的系统而终止，其目标是确保信息系统真正满足客户的需求。
  2. 7.基于构件的开发模型
     1. 基于构件的开发模型如下图



* + 1. 开发步骤：
       1. 不考虑构件的开发技术，基于构件的开发模型由以下步骤组成：
       2. (1)对于该问题领域的基于构件的可用产品进行研究和评估。
       3. (2)考虑构件集成的问题。
       4. (3)设计软件架构以容纳这些构件。
       5. (4)将构件集成到架构中。
       6. (5)进行充分的测试以保证功能正常。
    2. 典型的构件模型
       1. (1)OMG/CORBA。对象管理组织发布了公共对象请求代理体系结构（OMG/CORBA），一个对象请求代理提供了多种服务，使得可复用构件（对象）可以与其他构件通信。
       2. (2)Microsoft COM/DCOM/.NET。微软公司开发了构件对象模型（COM），此模型提供了构件的规格说明,在Windows操作系统，一个应用系统中可以使用不同厂商生产的构件。
       3. (3)Sun JavaBean构件。JavaBean构件系统是一个可移植的、平台独立的、使用Java程序设计语言开发的CBSE基础设施。
  1. 8.敏捷过程模型
     1. 敏捷原则 ：
        1. (1)我们最优先要做的是通过尽早、持续交付有价值的软件来使客户满意。
        2. (2)即使在开发的后期，也欢迎需求变更。敏捷过程利用变更为客户创造竞争优势。
        3. (3)经常交付可运行软件，交付的间隔可以从几个星期到几个月，交付的时间间隔越短越好。
        4. (4)在整个项目开发期间，业务人员和开发人员必须天天都在一起工作。
        5. (5)围绕有积极性的个人构建项目。给他们提供所需的环境和支持，并且信任他们能够完成工作。
        6. (6)在团队内部，最富有效果和效率的信息传递方法是面对面交谈。
        7. (7)可运行软件是进度的首要度量标准。
        8. (8)敏捷过程提倡可持续的开发速度。责任人、开发者和用户应该能够保持一种长期、稳定的开发速度。
        9. (9)不断地关注优秀的技能和好的设计会增强敏捷能力。
        10. (10)简单是必要的。
        11. (11)好的架构、需求和设计出自于自组织团队。
        12. (12)每隔一定时间，团队会反省如何才能更有效地工作，并相应调整自己的行为。

1. 五、面型对象方法和UML
2. 六、面向对象分析
3. 七、软件体系结构与设计模式
4. 八、面向对象设计
5. 十、软件测试方法
6. 十一、软件维护