**第一章：软件体系结构概论**

## 1、就项目管理方面而言，软件重用项目与非重用项目有哪些不同之处。

答：使用软件重用技术可减少重复工作,提高软件生产率, 缩短开发周期。同时，由于软构建大多经过严格的质量认证，因此有助于改善软件质量， 大量使用构建， 软件的灵活性和标准化程度可得到提高。

或：软件重用项目需要考虑生产可重用软部件的机制。同时，软件重用项目可以减少开发活动中大量的重复性工作，能提高软件生产率，降低开发成本，缩短开发周期。同时由于大量使用软构件，软件的灵活性和标准化程度也可望得到提高。因此，软件重用项目在管理是比非重用项目更节约管理资源，同时管理起来更方便。

## 2、实际参与/组织一个软件重用项目的开发，然后总结你是如何组织该项目的开发的？

答：参加了一个网页管理系统的开发，该项目重复使用已有的软件产品用于开发新的软件系统，以达到提高软件系统的开发质量与效率，降低开发成本的目的。在过程中使用了代码的复用、设计结果的复用、分析结果的复用、测试信息的复用等。

## 组织或参与一个采用B/S和C/S混合体系结构的软件项目的开发，总结开发经验。

首先，开发者根据一定的原则，将系统的所有子功能分类，决定哪些子功能适合采用C/S，哪些适合采用B/S。适合采用C/S的子功能应具备以下特点：1安全性要求高；2要求具有较强的交互性；3使用范围小，地点固定；4要求处理大量数据。例如，仓库管理系统中的入库单、领料单的输入功能，财务系统中的凭证输入功能等等。而适合采用B/S的子功能应具备以下特点：1使用范围广，地点灵活；2功能变动频繁；3安全性、交互性要求不同。例如：企业内部信息发布功能，意见箱输入功能，公司财务分析表的查询功能，总裁决策支持系统中的查询功能等等。

相对于单独采用C/S或B/S，这种方案的优点在于：1保证敏感数据的安全性，特别是对数据库的修改和新增记录加强了控制；2经济有效地利用企业内部计算机的资源，简化了一部分可以简化的客户端；3既保证了复杂功能的交互性，又保证了一般功能的易用与统一；4系统维护简便，布局合理；5网络效率最高。

如果系统开发者在系统设计阶段决定采用这种C/S与B/S相结合的模式，那么在系统开发生命周期的如下各个阶段相对这种新模式都应有所响应。 在系统设计阶段主要考虑的是MIS系统平台选择问题。在详细设计阶段，系统开发者需要根据企业自身的业务特点，以及一定的选择原则，来决定各个子功能采用哪一种模式并在系统说明书上分别注明。在编码设计阶段，系统开发者需要针对采用不同模式的子功能，选用不同的编码方式(例如：C/S可以采用VB编程环境，而B/S采用ASP方法)，然后编译生成不同的客户应用及Web服务程序。在安装调试阶段，其特点主要体现在系统的物理结构上，即特定的客户应用程序将被安装在特定的使用者的客户端上，Web服务程序需要被安装在Web服务器上，而每个客户端上都将被安装上浏览器，同时，客户应用的使用者必须接受一定的培训。在软件维护阶段，针对不同模式的子功能应采取不同维护方式。

## 4、组织或参与一个采用三层体系结构的软件项目的开发，总结开发经验。

三层体系结构包括：用户界面表示层(USL) 、业务逻辑层(BLL) 、数据访问层(DAL) 。各层的作用 ：1：数据访问层:主要是对原始数据(数据库或者文本文件等存放数据的形式)的操作层，而不是指原始数据，也就是说，是对数据的操作，而不是数据库,具体为业务逻辑层或表示层提供数据服务。2：业务逻辑层:主要是针对具体的问题的操作，也可以理解成对数据层的操作,对数据业务逻辑处理，如果说数据层是积木，那逻辑层就是对这些积木的搭建。3：表示层:主要表示WEB方式,也可以表示成WINFORM方式,WEB方式也可以表现成:aspx, 如果逻辑层相当强大和完善,无论表现层如何定义和更改,逻辑层都能完善地提供服务。

三层是指逻辑上的三层，即使这三个层放置到一台机器上。 三层体系的应用程序将业务规则、数据访问、合法性校验等工作放到了中间层进行处理。在保证客户端功能的前提下，为用户提供一个简洁的界面。这意味着如果需要修改应用程序代码，只需要对中间层应用服务器进行修改，而不用修改成千上万的客户端应用程序。“中间业务层”的用途有很多，例如：验证用户输入数据、缓存从数据库中读取的数据等等....但是，“中间业务层”的实际目的是将“数据访问层”的最基础的存储逻辑组合起来，形成一种业务规则。要保证“数据访问层”的中的函数功能的原子性！即最小性和不可再分。“数据访问层”只管负责存储或读取数据就可以了。

我们用三层结构主要是使项目结构更清楚，分工更明确，有利于后期的维护和升级。它未必会提升性能，因为当子程序模块未执行结束时，主程序模块只能处于等待状态。这说明将应用程序划分层次，会带来其执行速度上的一些损失。但从团队开发效率角度上来讲却可以感受到大不相同的效果。需要说明一下，三层结构不是.NET的专利，也不是专门用在数据库上的技术。它是一种更加普适的架构设计理念。此种架构要在数据库设计上注意表之间的关系，尽力满足主与子的关系。在功能上对用户要有一定的限制，不要表现在对于子表的删除操作一定要慎重，以免造成主表与子表的数据在逻辑上出现的主表的外键在子表中没有相对应的值。

“三层结构”开发模式，入门难度够高，难于理解和学习。这是对于初学程序设计的人来说的。以这种模式开发出来的软件，代码量通常要稍稍多一些。这往往会令初学者淹没在茫茫的代码之中。望之生畏，对其产生反感，也是可以理解的。

## 3、选择一个规模适中的系统，使用UML为其建模。

为一个网上购物系统利用UML为其建模，如下：

统一开发过程RUP把整个软件开发过程分为初始、细化、构造、交付四个阶段，具有用例驱动、以架构为中心、迭代和增量的特点。同其它软件开发方法相比较，RUP具有自身独特的优势，为软件开发提供了重要的方法论指导根据对网上购物系统的体系结构及建模分析，采用UML作为建模语言，结合RUP的基本开发过程，提出适合网上购物系统开发的建模过程。该过程遵循了RUP四个阶段的理论，主要是对初始和细化两个阶段进行了详细的分析。整个过程包括业务建模、需求建模、对象建模、数据库建模和物理建模等五个步骤，每个步骤都会生成一定的系统模型，并用相应的UML图来描述这些模型。在建模过程中，论文采用了RUP中迭代增量式的开发思想，把系统的建模进一步分解为迭代，一个迭代是一个从系统的业务建模到物理建模的完整过程，每一个迭代都会产生一个模型版本，是最终模型的一个子集，它增量式地发展，从一个迭代过程到另一个迭代过程，直到成为系统的最终模型。

1 业务建模

　　业务建模用于对网上购物系统环境的业务过程进行建模。系统分析人员通过网上购物系统的业务建模能够了解系统所处的环境和业务过程，业务建模能够将这些信息进行体现，并表现环境中存在或可以觉察到的过程，从而详细说明网上购物系统所要支持的业务过程。业务建模既确定了业务过程涉及到的业务对象和领域对象，又确定了每个业务过程所需要的资源和能力，包括人员、每个人员的职责和执行的操作、过程的执行方式和协作等。这些信息对于下一步的需求建模是非常重要和有用的。

　　业务建模一般通过业务过程图来进行描述。业务过程图是对事件逻辑的归类，这些事件被认为是业务的基本元素。其目的是将整个业务领域作为一个过程集进行描述，而不关心过程的次序或单个过程之间的交互作用。业务过程图不必严格精密，它应该全面而不是精确。重要的是，通过查看业务过程图，系统分析人员、设计人员、开发人员和用户能够迅速获得关于业务范围和活动的总体印象。

2 需求建模

　　需求建模的主要工作是获得系统的需求，建立待开发系统的模型。而用例则有助于更好地了解系统需求并以规范化的格式进行描述。需求建模就是要以用例的方式来描述系统的功能，其主要工作成果是用例模型。采用用例模型来描述进行需求建模的主要过程如下：（1）确定所要开发系统的参与者，参与者可以是人，也可以是与系统交互的外部系统。网上购物系统的参与者主要有管理员、工作人员、顾客、支付系统等。（2）从执行者的角度出发，分析他和系统需要进行的交互作用，并从这些交互过程中抽象出用例。从顾客的角度出发，网上购物系统一般有以下用例：用户登录、用户注册、浏览商品、搜索商品、购买商品、下订单、支付等；从系统管理员的角度出发，网上购物系统一般有以下用例：用户登录、用户管理、商品管理、订单管理等。（3）对每一用例确定其主要的业务过程。例如“用户登录”用例的业务过程为顾客、系统管理员、工作人员等通过用户登录可以获得相应的服务；“支付”用例的业务过程为顾客为所够买物品选择付款方式进行付款。（4）以信息流为中心逐步形成完整的用例模型。网上购物系统的完整用例模型包括很多的用例图，其中既包括系统的顶层用例图，也包括各种细化的用例图。

3 对象建模

　　确定了系统的需求分析、得出系统的用例模型以后，需要进行的主要任务就是对系统进行对象建模。对象建模的主要工作是把需求建模阶段产生的用例模型转化为系统的静态结构模型和动态行为模型，使建立的系统在特定的环境下完成需求分析中的任务和功能，有利于系统的实现和迭代。这其中主要包括静态结构建模和动态对象建模两部分。

4 数据库建模

　　数据库建模是从计算机系统的角度对系统所要处理的数据进行建模。数据库系统是整个网上购物系统的基础，数据库建模的好坏直接影响到整个系统的结构、实现的复杂程度、性能、安全性和可维护性等。传统的逻辑数据库建模工具“实体一联系(E．R)""图只针对数据建模，不能对行为建模。而UML的类图能够更好的用于数据库建模。UML的类图不但对数据建模，而且能对行为建模，这些行为在物理数据库中被设计成触发器和存储过程。即使是关系数据库，也可以在类图设计落实后，再采取标准方法把类图映射到具体的关系模型。从类图到关系模型的转换，按照一个类映射为一个关系的原则进行，而类的属性即为关系的属性，标识的标识符即为关系的主键。

5 物理建模

　　物理建模用于网上购物系统建模过程的最后阶段，是对网上购物系统的物理方面进行建模。它使用UML中的组件图描述网上购物系统中代码组件的物理结构及各个组件之间的依赖关系，使用配置图定义网上购物系统的软硬件结构及通信机制，表示软硬件系统之间的协作关系。

　　以上五个步骤是根据RUP的四个阶段细化的结果，分别对应了RUP中相应的核心工作流。其中业务建模对应RUP的业务建模工作流，需求建模对应RUP的需求建模工作流，对象建模和数据库建模对应RUP的分析和设计工作流，物理建模对应RUP的实施工作流，对开发完成的系统进行测试、部署和管理分别对应RUP的测试工作流、部署工作流等。

## SOA和微服务

**SOA**：SOA的提出是在企业计算领域，就是要将紧耦合的系统，划分为面向业务的，粗粒度，松耦合，无状态的服务。服务发布出来供其他服务调用，一组互相依赖的服务就构成了SOA架构下的系统。基于这些基础的服务，可以将业务过程用类似BPEL流程的方式编排起来，而BPEL反映的是业务处理的过程，这些过程对于业务人员更为直观，调整也比hardcode的代码更容易。当然企业还需要对服务治理，比如服务注册库，监控管理等。我们知道企业计算领域，如果不是交易系统的话，并发量都不是很大的，所以大多数情况下，一台服务器就容纳将许许多多的服务，这些服务采用统一的基础设施，可能都运行在一个应用服务器的进程中。虽然说是面向服务了，但还是单一的系统。

**微服务**：而微服务架构大体是从互联网企业兴起的，由于大规模用户，对分布式系统的要求很高，如果像企业计算那样的系统，伸缩就需要多个容纳续续多多的服务的系统实例，前面通过负载均衡使得多个系统成为一个集群。但这是很不方便的，互联网企业迭代的周期很短，一周可能发布一个版本，甚至可能每天一个版本，而不同的子系统的发布周期是不一样的。而且，不同的子系统也不像原来企业计算那样采用集中式的存储，使用昂贵的Oracle存储整个系统的数据，二是使用MongoDB，HBase，Cassandra等NOSQL数据库和Redis，memcache等分布式缓存。那么就倾向采用以子系统为分割，不同的子系统采用自己的架构，那么各个服务运行自己的Web容器中，当需要增加计算能力的时候，只需要增加这个子系统或服务的实例就好了，当升级的时候，可以不影响别的子系统。这种组织方式大体上就被称作微服务架构。微服务与SOA相比，更强调分布式系统的特性，比如横向伸缩性，服务发现，负载均衡，故障转移，高可用。互联网开发对服务治理提出了更多的要求，比如多版本，比如灰度升级，比如服务降级，比如分布式跟踪，这些都是在SOA实践中重视不够的。Docker容器技术的出现，为微服务提供了更便利的条件，比如更小的部署单元，每个服务可以通过类似Node.js或Spring Boot的技术跑在自己的进程中。可能在几十台计算机中运行成千上万个Docker容器，每个容器都运行着服务的一个实例。随时可以增加某个服务的实例数，或者某个实例崩溃后，在其他的计算机上再创建该服务的新的实例。

## 3、为什么要研究软件体系结构？

答：1.软件体系结构是系统开发中不同参与者进行交流和信息传播的媒介。

2．软件体系结构代表了早期的设计决策成果。

3．软件体系结构可以作为一种可变换的模型。

## 4、根据软件体系结构的定义，你认为软件体系结构的模型应该由哪些部分组成？

答： **构件**(component)可以是一组代码，如程序的模块;也可以是一个独立的程序(如数据库的SQL服务器)；

**连接件**(connector)是关系的抽象，用以表示构件之间的相互作用。如过程调用、管道、远程过程调用等；

**限制**(constrain)：用于对构件和连接件的语义说明。

## 5、在软件体系结构的研究和应用中，你认为还有哪些不足之处？

答：（1）缺乏同一的软件体系结构的概念，导致体系结构的研究范畴模糊。

（2）ADL繁多，缺乏统一的ADL的支持。

（3）软件体系结构研究缺乏统一的理论模型支持。

（4）在体系结构描述方便，尽管出现了多种标准规范或建议标准，但仍很难操作。

（5）有关软件体系结构性质的研究尚不充分，不能明确给出一个良体系结构的属性或判定标准，没有给出良体系结构的设计指导原则，因而对于软件开发实践缺乏有力的促进作用。

（6）缺乏有效的支持环境软件体系结构理论研究与环境支持不同步，缺乏有效的体系结构分析、设计、方针和验证工具支持，导致体系结构应用上的困难。

（7）缺乏有效的体系结构复用方案。

（8）体系结构发现方法研究相对欠缺。

## 7、研究软件体系结构的意义是什么？

(1) 体系结构是风险承担者进行交流的手段

软件体系结构代表了系统公共的高层次的抽象。这样，系统的大部分有关人员（即使不是全部）能把它作为建立一个互相理解的基础，形成统一认识，互相交流。

 (2) 体系结构是早期设计决策的体现

* 明确了对系统实现的约束条件
* 决定了开发和维护组织的组织结构
* 制约着系统的质量属性
* 通过研究软件体系结构可能预测软件的质量
* 使推理和控制更改更简单
* 有助于循序渐进的原型设计
* 可以作为培训的基础

(3) 软件体系结构是可传递和可重用的模型

软件体系结构级的重用意味着体系结构的决策能在具有相似需求的多个系统中发生影响，这比代码级重用的作用更大。

## 8、从哪些方面评估软件体系结构?

答：（1）性能是指系统的影响能力，即要经过多长时间才能对某个事件作出响应，或者在某段事件内系统所能处理的事件的个数。

（2）可靠性是软件系统在应用或系统错误面前，在意外或错误使用的情况下维持软件系统的功能特性的基本能力。

（3）可用性是系统能够正常运行的时间比例。经常用两次故障间的时间长度或在出现故障时系统能够恢复正常的速度来表示。

（4）安全性是指系统在向合法用户提供服务的同时能够阻止非授权用户使用的企图或拒绝服务的能力。安全性又可划分为机密性、完整性、不可否认性及可控性等特性。

（5）可修改性是指能够快速的以较高的性能代价比对系统进行变更的能力。

（6）功能性是系统所能完成所期望的工作的能力。

（7）可变性是指体系结构经扩充或变更而成为新体系结构的能力。

（8）可集成性是指系统能与其他系统协作的程度

(9)互操作性是指与其他环境或者系统本身相互作用的能力。

# 第二章：软件体系结构建模

## 2、引入了软件体系结构以后，传统软件过程发生了哪些变化？这种变化有什么好处？

答：软件体系结构的引入使软件设计开发更加具体和形象，它的模型更使得软件过程更加方便和多样化。其好处在于：包括程序员在内的绝大多数系统的利益相关人员都借助软件体系结构来进行彼此理解、协商、达成共识或者相互沟通的基础，软件体系机构的模型可以应用到具有相似质量属性和功能需求的系统中，并能够促进大规模软件的系统级复用，在很多方面使得软件开发更加人性化。

## 3、软件体系结构的生命周期模型与软件生命周期模型有什么关系？

答：软件体系结构是贯穿于软件研发的整个生命周期的系统开发、运行、维护所实施的全部工作和任务的结构框架，给出了软件开发活动各阶段之间的关系，软件体系结构的生命周期模型为软件生命周期模型提供了很好的结构依据和参考，也为其构建了很好的开发方式。

# 第三章：软件体系结构风格

## 1、层次系统结构和基于消息的层次系统结构有什么区别？

答： 层次系统组织成一个层次结构，每一层为上层服务，并作为下层客户。在一些层次系统中，除了一些精心挑选的输出函数外，内部的层只对相邻的层可见。这样的系统中构件在一些层实现了虚拟机（在另一些层次系统中层是部分不透明的）。连接件通过决定层间如何交互的协议来定义，拓扑约束包括对相邻层间交互的约束。这种风格支持基于可增加抽象层的设计。允许将一个复杂问题分解成一个增量步骤序列的实现。由于每一层最多只影响两层，同时只要给相邻层提供相同的接口，允许每层用不同的方法实现，同样为软件重用提供了强大的支持。

而在基于消息的层次系统结构中构件不直接调用一个过程，而是触发或广播一个或多个事件。系统中的其它构件中的过程在一个或多个事件中注册，当一个事件被触发，系统自动调用在这个事件中注册的所有过程，这样，一个事件的触发就导致了另一模块中的过程的调用。这种风格的构件是一些模块，模块既可以是一些过程，又可以是一些事件的集合。过程可以用通用的方式调用，也可以在系统事件中注册一些过程，当发生这些事件时，过程被调用。这种风格的主要特点是事件的触发者并不知道哪些构件会被这些事件影响。这样不能假定构件的处理顺序，甚至不知道哪些过程会被调用，因此，许多隐式调用的系统也包含显式调用作为构件交互的补充形式。

## 3、SIS和DSSA分别用在哪些场合？

答：1.DSSA只对某一个领域进行设计专家知识的提取，存储和组织，但可以同时使用多种体系结构风格；而在某个体系结构风格中进行体系结构设计专家知识的组织时，可以将提取的公共结构和设计方法扩展到多个领域。

2.DSSA的特定领域参考体系结构通常选用一个或多个适合所研究领域的体系结构风格，并设计一个该领域专用的体系结构分析设计工具。

3．体系结构风格的定义和该风格应用的领域是直交的，提取的设计知识比用DSSA提取的设计专家知识的应用范围要广。

4．DSSA和体系结构风格是互为互补的两项技术。

SIS：交互式网络互连服务 DSSA：特定领域的软件体系结构

## 4、在软件开发中，采用异构结构有什么好处，其负面影响有哪些？

答：好处：采用异构结构可以更好的处理当实际问题需要用不同的结构去解决这种情况。关于软件包，框架、通信以及其他一些体系结构上的问题，目前存在多种标准。即使在某段时间内某一种标准占统治地位，但变动最终是绝对的。采用异构结构可以很好的处理这些变动。

负面影响：实际工作中，我们总会遇到一些遗留下来的代码它们仍有效用但是却与新系统有某种程度上的不协调。然在许多场合，将技术与经济综合进行考虑时，总是决定不再写它们。即使在某一单位中，规定了共享共同的软件包或相互关系的一些标准，仍会存在解释或表示习惯上的不同。

# 第七章：基于体系结构的软件开发

## 1、请把基于体系结构的软件开发模型与其他软件开发模型进行比较。

答：软件开发模型有演化模型、螺旋模型、喷泉模型、智能模型等。传统软件开发模型存在开发效率不高,不能很好地支持软件重用等缺点。在多个大中型软件项目的实践基础上,提出了基于体系结构的软件开发模(ABSD)。ABSD模型把整个基于体系结构的软件过程划分为体系结构需求、设计、文档化、复审、实现、演化等6个子过程,讨论了各个子过程所要完成的工作,给出了ABSD模型在劳动和社会保险领域的一个应用实例。实践表明,采用ABSD模型进行软件项目开发,具有结构清晰、易于理解、可移植性强、重用粒度大等优点。

答：一、基于体系结构为基础的基于构件组装的软件开发模型，如基于构件的开发模型和基于体系结构的开发模型等。基于体系结构的开发模型是以软件体系结构为核心，以基于构件的开发方法为基础。然后采用迭代增量方式进行分析和设计，将功能设计空间映射到结构设计空间，再由结构设计空间映射到系统设计空间的过程。

特点：是利用需求分析结果设计出软件的总体结构，通过基于构件的组装方法来构造软件系统。

优点：软件复用，提高了开发的效率。软件的结构框架更清晰，有利于系统的设计、开发和维护。

缺点：缺乏通用的组装结构标准，因而引入了较大的风险。可重用性和软件高效性不易协调。客户的满意度低，过分依赖于影响产品质量。

二、以软件需求完全确定为前提软件开发模型，如瀑布模型等。

特点：软件需求在开发阶段已经被完全确定，将生命周期的各项活动依顺序固定，强调开发的阶段性；

优点：开发流程简单。

缺点：是开发后期要改正早期存在的问题需要付出很高的代价，用户需要等待较长时间才能够看到软件产品，增加了风险系数。

三、在开始阶段只能提供基本需求的渐进式开发模型，如螺旋模型和原型实现软件开发模型等。

特点：软件开发开始阶段只有基本的需求，软件开发过程的各个活动是迭代的。通过迭代过程实现软件的逐步演化，最终得到软件产品。在此引入了风险管理，增加项目成功几率

优点：强调了其他模型均忽略了的风险分析。

缺点：由于需求的不完全性，从而为软件的总体设计带来了困难并要求对风险技能管理水平的高要求。

## 2、请把基于体系结构的软件设计方法与其他软件设计方法进行比较。

基于体系结构的开发方法出现了基于构件、基于框架、基于模板的开发技术，项目利用基于体系结构开发的思想，从需求到高层框架设计、架构设计、平台实现，充分体现了基于体系结构开发中从高层设计入手，复用体系结构框架、设计和实现可复用的构件的原则。

此外，与其他软件设计方法的区别还有：

1、 单个用户界面和业务逻辑的改动不会影响到系统整体

2、 开发人员的组织用户界面和业务逻辑的定制可由领域人员来完成；专业

计算机人员制作不同粒度的构件，完成模型的开发。

3、 不同业务对象有相似的业务逻辑，可直接复用其构件。 4、 在系统维护过程更新某一模块，不会影响系统的其它部分。

## 3、如何才能提高软件系统的可演化性。

答：构造性和演化性是软件的两个基本特性。软件进行渐变并达到所希望的形态就是软件演化，软件演化是由一系列复杂的变化活动组成。对软件变化的控制是软件开发者历来追求的目标。引起软件变化的原因是多方面的，如基本设施的改变，功能需求的增加，高性能算法的发现，技术环境因素的变化等。所以对软件变化甚至演化进行理解和控制显得比较复杂和困难。

(1)必须频繁地变化以适应要求。

(2)软件的复杂度不断地增长。

(3)通过自我调节以符合产品需求和过程特性。

 (4)在软件的生命周期中保持一定的组织稳定性。

(5)不同的版本之间保持一定的连贯性。  (6)功能持续地增加。

 (7)在没有严格的维护和适应性修改的情况下会出现质量衰退。  (8)是一个反馈系统。

# 第八章：软件体系结构的分析与测试

## 1、什么是软件体系结构的可靠性？为什么要研究软件体系结构的可靠性？

答：通过系统的详细说明书，确定系统所采用的体系结构风格。

 把每一种体系结构风格转换成状态视图，并计算状态视图中每一个状态的可靠性及其相应的迁移概率。

通过整个系统的体系结构视图，把所有的状态视图集成为一个整体状态视图。

通过整体状态视图构造系统的迁移矩阵，并计算系统的可靠性。

## 2、如何模型化系统的可靠性？

答：软件体系结构风险分析有哪些基本步骤

采用体系结构描述语言ADL对体系结构进行建模 通过模拟方法执行复杂性分析

通过FMEA和模拟运行执行严重性分析  为构件和连接件开发其启发式风险因子  建立用于风险评估的CDG

## 3、软件体系结构风险分析有哪些基本步骤？

答： 采用体系结构描述语言ADL对体系结构进行建模 通过模拟方法执行复杂性分析

通过FMEA和模拟运行执行严重性分析  为构件和连接件开发其启发式风险因子  建立用于风险评估的CDG

## 4、软件体系结构测试与程序测试的主要区别是什么？

答：软件体系结构测试要研究的对象时软件体系结构设计，并以此为基础产生高层次的测试用例集，以指导代码层的测试活动。他的目的是找出体系结构设计的错误和缺陷，产生指导代码测试的测试计划和测试用例， 软件体系结构测试与传统的软件测试既有联系又有区别，而软件体系结构的测试包含两个方面，一个是软件体系结构的结构分析，一个侍女软件体系结构的测试。

程序测试是指对一个完成了全部或部分功能、模块的计算机程序在正式使用前的检测，以确保该程序能按预定的方式正确地运行。

目前，软件的正确性尚未得到根本的解决，软件测试仍是发现软件错误和缺陷的主要手段。

为了发现系统中的错误，应竭力设计能暴露错误的测试用例。测试用例是有测试数据和预期结果构成的。一个好的测试用例是极有可能发现至今为止尚未发现的错误的测试用例。

高效的测试是指用少量的测试用例，发现被测软件尽可能多的错误。软件测试所追求的是以尽可能少的时间和人力发现软件产品尽可能多的错误。