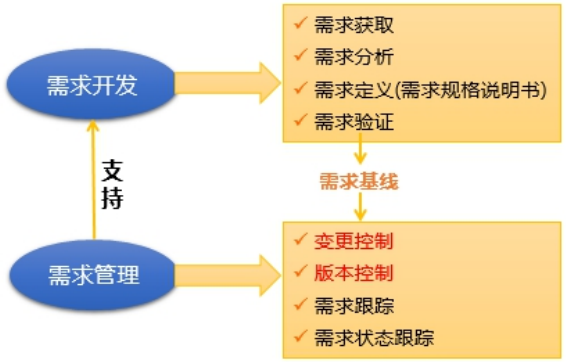
系统分析师考试背记精要

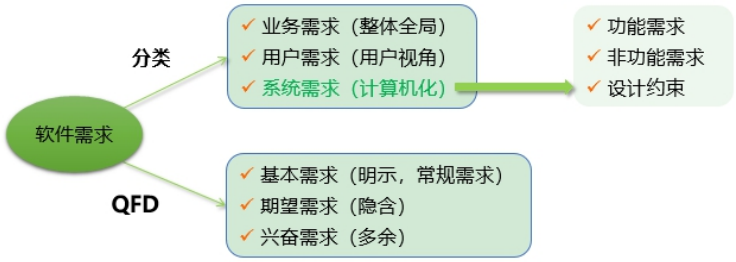
1、什么是软件需求

软件需求是指用户对系统在功能、行为、性能、设计约束等方面的期望。

软件需求是指用户解决问题或达到目标所需的条件或能力，是系统或系统部件要满足合同、标准、规范或其他正式规定文档所需具有的条件或能力，以及反映这些条件或能力的文档说明。



2、需求分类



（1）业务需求：是指反应企业或客户对系统高层次的目标要求，通常来自项目投资人、购买产品的客户、客户单位的管理人员、市场营销部门或产品策划部门等。通过业务需求可以确定项目视图和范围，为以后的开发工作奠定了基础。

（2）用户需求：描述的是用户的具体目标，或用户要求系统必须能完成的任务。也就是说，用户需求描述了用户能使用系统来做些什么。

（3）系统需求：是从系统的角度来说明软件的需求，包括功能需求、非功能需求和设计约束等。**功能需求**也称为行为需求，它规定了开发人员必须在系统种实现的软件功能，用户利用这些功能来完成任务，满足业务要求。**非功能需求**是指系统必须具备的属性或品质，又可细分为软件质量属性和其他非功能需求。**设计约束**也称为限制条件或补充规约，通常是对系统的一些约束说明。

3、PIECES 框架是系统非功能性需求分类的技术。

性能（Preformance）：性能用于描述企业当前的运行效率，可以分析当前业务的处理速度。

信息（Information）：信息和数据指标用于描述业务数据的输入、输出以及处理方面存在的各种问题。

经济（Economics）：经济指标主要是从成本和收益的角度分析企业当前存在的问题。

控制（Control）：提高信息系统的安全和控制水平。

效率（Efficiency）：提高企业的人、财、物等的使用效率。

服务（Service）：提高企业对客户、供应商、合作伙伴、顾客等的服务质量。

4、需求获取方法

（1）用户访谈：1 对 1-3，有代表性的用户。用户访谈是最基本的一种需求获取手段，其形式包括结构化和非结构化两种。用户访谈是通过 1 对 1（或 1 对 2，1 对 3）的形式与用户面对面进行沟通，以获取用户需求。用户访谈具有良好的灵活性，有较宽广的应用范围。但是，也存在着许多困难，例如，用户经常较忙，难以安排时间；面谈时信息量大，记录较为困难；沟通需要很多技巧，同时需要系统分析师具有足够的领域知识等。另外，在访谈时，还可能会遇到一些对于企业来说比较机

密和敏感的话题。因此，这看似简单的技术，也需要系统分析师具有丰富的经验和较强的沟通能力。

（2）问卷调查：用户多，无法一一访谈。与用户访谈相比，问卷调查可以在短时间内，以低廉的代价从大量的回答中收集数据；问卷调查允许回答者匿名填写，大多数用户可能会提供真实信息；问卷调查的结果比较好整理和统计。问卷调查最大的不足就是缺乏灵活性。

（3）现场观摩：针对较为复杂的流程和操作。想获取系统某些较为复杂的流程和操作过程，则现场观摩方法比较合适。对于一些较为复杂的流程和操作而言，是比较难以用语言和文字进行表达的，对于这种情况，可以采用到客户的工作现场，一边观察，一边听客户讲解，从而更直观的了解客户需求。

（4）联合需求计划（JRP）：高度组织的群体会议，各方参与，成本较高。为了提高需求获取的效率，越来越多的企业倾向于使用小组工作会议来代替大量独立的访谈。联合需求计划（Joint Requirement Planning，JRP）是一个通过高度组织的群体会议来分析企业内的问题并获取需求的过程，它是联合应用开发（Joint Application Development，JAD）的一部分。

（5）情节串联板：一系列图片，通过这些图片来讲故事。

（6）收集资料：把与系统有关的、对系统开发有益的信息收集起来。

（7）参加业务实践：有效地发现问题的本质和寻找解决问题的办法。

（8）阅读历史文档：对收集数据性的信息较为有用。

（9）抽样调查：降低成本。采样是指从种群中系统地选出有代表性的样本集的过程，通过认真研究所选出的样本集，可以从整体上揭示种群的有用信息。对于信息系统的开发而言，现有系统的文档（文件）就是采样种群。当开始对一个系统做需求分析时，查看现有系统的文档是对系统有初步了解的最好方法。但是，系统分析师应该查看哪些类型的文档，当文档的数据庞大，无法一一研究时，就需要使用采样技术选出有代表性的数据。抽样能够提高需求获取效率，但抽样往往是由系统分析师来抽的，所以会受到他的主观因素影响。

5、可行性分类

经济可行性：成本收益分析，包括建设成本、运行成本和项目建设后可能的经济收益。

技术可行性：技术风险分析，现有的技术能否支持系统目标的实现，现有资源（员工，技术积累，构件库，软硬件条件）是否足以支持项目的实施。

法律可行性(社会可行性)：不能与国家法律或政策相抵触。

用户使用可行性：执行可行性，从信息系统用户的角度评估系统的可行性。

管理可行性：系统与现有管理机制的一致性，改革的可能性。

运行可行性：用户方便使用的程度。

6、成本分类

固定成本：不随产量变化。管理人员的工资、办公费、固定资产折旧费、员工培训费、广告费、技术开发经费等。

变动成本：随产量变化。直接材料费、产品包装费、外包费用、开发奖金等。

混合成本：水电费、电话费、质量保证人员的工资、设备动力费等。

直接成本：直接投入在项目上。项目组人员工资，材料费用。

间接成本：分摊到项目上。水电费，员工培训费。

7、REST 概念：REST（Representational State Transfer，表述性状态转移）是一种只使用 HTTP 和 XML 进行基于 Web 通信的技术，可以降低开发的复杂性，提高系统的可伸缩性。

8、REST 的五个原则：网络上的所有事物都被抽象为资源；每个资源对应一个唯一的资源标识；通过通用的连接件接口对资源进行操作；对资源的各种操作不会改变资源标识；所有的操作都是无状态的。

9、、负载均衡技术：（1）应用层负载均衡：http 重定向、反向代理服务器；（2）传输层负载均衡：DNS 域名解析负载均衡、基于 NAT 的负载均衡；（3）硬件负载均衡：F5；（6）软件负载均衡：LVS、Nginx、HAproxy。

10、有状态和无状态：

（1）无状态服务（stateless service）对单次请求的处理，不依赖其他请求，也就是说，处理一次请求所需的全部信息，要么都包含在这个请求里，要么可以从外部获取到（比如说数据库），服务器本身不存储任何信息。

（2）有状态服务（stateful service）则相反，它会在自身保存一些数据，先后的请求是有关联的。

11、业务流程建模方法

（1）标杆瞄准；（2）IDEF（一系列建模、分析和仿真方法的统称）；（3）DEMO（组织动态本质建模法）；（4）Petri 网；（5）业务流程建模语言；（6）基于服务的 BPM。

12、面向对象基本概念

对象：属性（数据）+方法（操作）+对象 ID。

类（实体类/控制类/边界类）。

继承与泛化：复用机制。

封装：隐藏对象的属性和实现细节,仅对外公开接口。

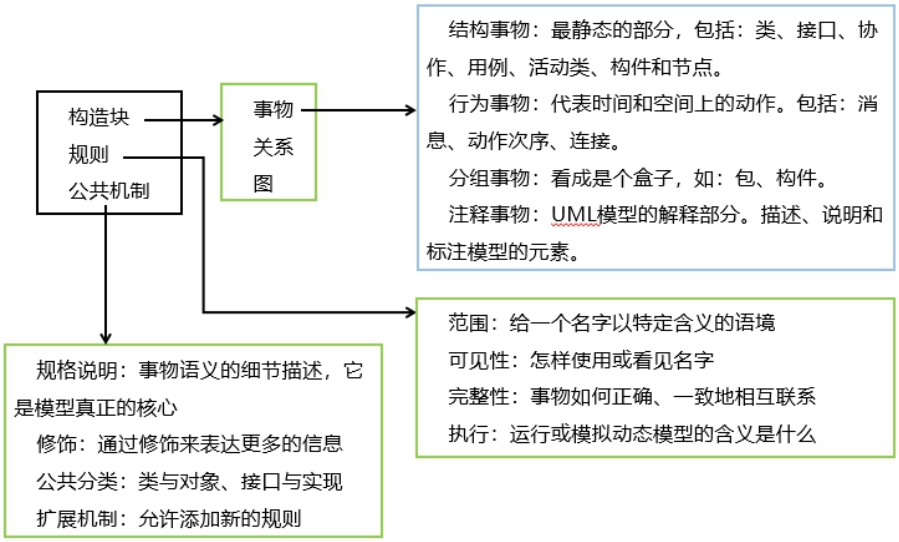
多态：不同对象收到同样的消息产生不同的结果。

接口：一种特殊的类，他只有方法定义没有实现。

重载：一个类可以有多个同名而参数类型不同的方法。

消息和消息通信：消息是异步通信的。

13、UML 概念



UML 包括两组公共分类，分别是类与对象（类表示概念，而对象表示具体的实体）、接口与实现（接口用来定义契约，而实现就是具体的内容）。

（1）结构事物。

结构事物在模型中属于最静态的部分，代表概念上或物理上的元素。UML 有七种结构事物，分别是类、接口、协作、用例、活动类、构件和节点。**类**是描述具有相同属性、方法、关系和语义的对象的集合，一个类实现一个或多个接口；**接口**是指类或构件提供特定服务的一组操作的集合，接口描述了类或构件的对外的可见的动作；**协作**定义了交互的操作，是一些角色和其它事物一起工作，提供一些合作的动作，这些动作比事物的总和要大；**用例**是描述一系列的动作，产生有价值的结果。在模型中用例通常用来组织行为事物。用例是通过协作来实现的；活动类的对象有一个或多个进程或线程。**活动类**和类很相似，只是它的对象代表的事物的行为和其他事物是同时存在的；**构件**是物理上或可替换的系统部分，它实现了一个接口集合；**节点**是一个物理元素，它在运行时存在，代表一个可计算的资源，通常占用一些内存和具有处理能力。一个构件集合一般来说位于一个节点，但有可能从一个节点转到另一个节点。

（2）行为事物：行为事物是 UML 模型中的动态部分，代表时间和空间上的动作。UML 有两种主要的行为事物。第一种是交互（内部活动），**交互**是由一组对象之间在特定上下文中，为达到特定目的而进行的一系列消息交换而组成的动作。交互中组成动作的对象的每个操作都要详细列出，包括消息、动作次序（消息产生的动作）、连接（对象之间的连接）；第二种是**状态机**，状态机由一系列对象的状态组成。

（3）分组事物。分组事物是 UML 模型中组织的部分，可以把它们看成是个盒子，模型可以在其中进行分解。UML 只有一种分组事物，称为包。**包**是一种将有组织的元素分组的机制。与构件不同的是，包纯粹是一种概念上的事物，只存在于开发阶段，而构件可以存在于系统运行阶段。

（4）注释事物。注释事物是 UML 模型的解释部分。

14、UML 图的分类

15、UML 图的概念

（1）类图（class diagram）:类图描述一组类、接口、协作和它们之间的关系。

（2）对象图（object diagram）:对象图描述一组对象及它们之间的关系。对象图描述了在类图中所建立的事物实例的静态快照。

（3）构件图（component diagram）。构件图描述一个封装的类和它的接口、端口，以及由内嵌的构件和连接件构成的内部结构。构件图用于表示系统的静态设计实现视图。对于由小的部件构建大的系统来说，构件图是很重要的。构件图是类图的变体。一个封装的类和它的接口

（4）部署图（deployment diagram）。部署图描述对运行时的处理节点及在其中生存的构件的配置。部署图给出了架构的静态部署视图，通常一个节点包含一个或多个部署图。软硬件之间映射

（5）制品图：系统的物理结构

（6）包图：由模型本身分解而成的组织单元，以及他们之间的依赖关系

（7）组合结构图

（8）用例图：系统与外部参与者的交互

（9）顺序图（sequence diagram，序列图）：顺序图是一种交互图（interaction diagram），它强调对象之间消息发送的顺序，同时显示对象之间的交互。强调按时间顺序。

（10）通信图（communication diagram）。协作图。通信图也是一种交互图，它强调收发消息的对象或参与者的结构组织。顺序图和通信图表达了类似的基本概念，但它们所强调的概念不同，顺序图强调的是时序，通信图强调的是对象之间的组织结构（关系）。

（11）定时图：强调实际时间

（12）交互概览图

（13）状态图（state diagram）。状态图描述一个状态机，它由状态、转移、事件和活动组成。状态图给出了对象的动态视图。它对于接口、类或协作的行为建模尤为重要，而且它强调事件导致的对象行为，这非常有助于对反应式系统建模。状态转换变迁

（14）活动图（activity diagram）。活动图将进程或其他计算结构展示为计算内部一步步的控制流和数据流。活动图专注于系统的动态视图。它对系统的功能建模和业务流程建模特别重要，并强调对象间的控制流程。类似程序流程图，并行行为