软件架构设计基础理论（有些过时）

由于一些知识性的特殊需要，要求掌握比较过时的软件架构设计理论，因而作此文案用于记忆和查询。该部分内容与现实中软件开发相去甚远，也可以理解一些东西之间确实存在很大的鸿沟，不多说，开始码字咯。

Bass、Clements和Kazman的定义：系统的一个或多个结构，结构中包含软件的构件，构件的外部可见属性以及它们之间的关系。体系结构并非是一个可以运行的软件，而是一种表达，使软件工程师能够分析设计针对需求的有效性、尽早的考虑多种方案、降低风险。

* 软件架构设计与生命周期

需求分析阶段：需求是问题空间，软件架构是解空间，在软件需求模型向SA模型转化时主要关注，如何根据需求模型构建SA模型和如何保证模型转化的可追踪性。

设计阶段：SA(software architecture)模型的研究关注，基本概念、描述语言、多视图支持。

实现阶段：研究SA的开发过程支持，如项目组织结构、配置管理；寻求从SA向实现过渡的途径，如将程序设计语言引入到SA阶段、模型映射、构件组装、复用中间件平台等；研究SA的测试技术。

构件组装阶段：支持可复用构建的互联、在组装时消除体系结构的失配问题，例如将中间件作为连接子。

部署阶段：提供高层体系结构视图描述部署阶段的软硬件模型、基于模型分析部署方案的质量属性。

后开发阶段：围绕维护、演化、复用等方面进行，研究方向包括动态软件体系结构、体系结构恢复和重建。

* 软件架构的重要性，包括：能够满足系统的品质要求；使受益人达成一致的目标；支持计划设定；指导开发；管理软件固有的复杂性；为复用奠定基础；降低维护费用；支持冲突分析。

ABSD(Architecture-Based Software Design)，基于体系结构的软件设计的基础包括：功能的分解、选择体系结构风格来实现质量和商业需求、软件模板的使用。概念包括：设计元素、视角与视图、用例和质量场景。

* 体系结构开发模型

需求：需求的获取来自三个方面，包括系统质量目标、商业目标和开发人员商业目标。标识构件过程包括生成类图、对类进行分组、把类打包成构件，之后进行架构需求的评审。

设计：首先选择软件架构风格，之后将以表示的构件映射到软件体系结构并分析其相互作用，最后产生架构文档并评审。

文档化：产出体系结构规格说明和测试体系结构需求的质量设计说明书两个文档。

复审：有外部人员（用户代表和领域专家）参加的评审，目的在于标识潜在的封校和发现设计的缺陷和错误。

实现：按照设计决策，分割规定构件，按照指定方式交互并通过测试。

演化：需求变化归类，指定演化计划，增加、修改、删除构件、更新相互作用、组装和测试。

1. 经典软件体系架构风格（内容灰常的陈旧）

* 管道和过滤器：每个构件都有一组输入和输出，数据输入构件，经过内部处理，然后产生输出。因此，这里的构件被称为过滤器，这种风格的连接件是数据流传输的管道，将一个过滤器的输出传到另一过滤器的输入。
* 数据抽象和面向对象组织：抽象数据类型概念对软件系统有着重要作用，目前软件界已普遍转向面向对象系统。这种风格建立在数据抽象和面向对象基础上，数据的表示方法和他们的相应操作封装在一个抽象数据类型或者对象中。这种风格的构件就是对象，或者说是抽象数据类型的实例。
* 事件驱动系统：事件驱动系统风格是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。系统中的其他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。
* 分层系统：层次系统组成一个层次结构，每一层为上层服务，并作为下层的客户。在一些层次系统中，除了一些精心挑选的输出函数外，内部的层接口只对相邻的层可见。这样的系统中构件在层上实现了虚拟机。连接件通过决定层次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。由于每一层只于上下两层交互，为软件重用提供了强大的支持。
* 仓库系统和知识库：在仓库风格中，有两种不同的构件：中央数据结构说明当前状态，独立构件在中央数据存储上执行。一方面，若构件控制共享数据，则仓库是传统数据库；另一方面，若中央数据结构的当前状态触发进程执行的选择，则仓库是一黑板系统。Tip：主要是看中心仓库是被控方—数据库，还是控制方—黑板系统。
* C2风格：通过连接件绑定在一起按照一组规则运作的并行构件网络。其规则包括：系统中构件和连接件都有一个顶部和底部；构件的顶部应连接到某一连接件的底部，构件的底部应该连接到一个连接件的顶部，而构件和构件之间的直接连接是不被允许的；一个连接件可以和任意数目的其他构件和连接件相连；当两个连接件进行直接连接时，必须由其中一个的底部连接到另外一个的顶部。简而言之，就是一个层级结构，但构件间必须通过连接件相连。

1. 常见架构风格

* C/S风格，三层C/S结构风格：表示层、功能层、数据层。
* B/S风格，浏览器/Web服务器/数据库服务器

DSSA(Domain Specific Software Architecture)就是在一个特定领域中为一组应用提供组织结构参考的标准软件体系结构。其特征为：一个严格定义的问题域和问题解域；具有普遍性；对整个领域的构件组织模型有很恰当的抽象；具备该领域固定的、典型的在开发过程中可重用元素。其基本活动包括：领域分析，获得领域模型；领域设计，获得DSSA；领域实现，依据领域模型和DSSA开发和组织可重用信息。

DSSA的建立过程包括5个阶段：

* 定义领域范围，输出为满足用户需要的一系列需求
* 定义领域特定的元素，编译领域字典和领域术语的同义词词典
* 定义领域特定的设计和实现需求约束，描述解空间中有差别的特性
* 定义领域模型和体系结构，产生一般的体系结构，并说明构成它们的语法和语义
* 产生、搜索可重用的产品单元，为DSSA增加构件，并说明其语法和语义

1. 体系结构评估的目标可以是一个体系结构，也可以是一组体系结构。在体系结构评估过程中，关注其质量属性，其相关质量属性包括：

* 性能，系统的响应能力，如一个页面请求的响应速度、一段时间内系统能处理的事件总数，经常使用基准测试程序完成单元测试。
* 可靠性，表示在意外和出错时，系统的恢复能力。通常用平均失效等待事件(mean time to failure, MTTF)和平均失效间隔事件(mean time between failure, MTBF)来衡量。可靠性可以分为：容错能力—即出错时的处理能力，健壮性—在出错时按照指定方式终止执行。
* 可用性，系统能够正常运行的时间比例。
* 安全性，系统在向合法用户提供服务的同事能够阻止非授权用户使用的企图或拒绝服务的能力。
* 可修改性，能够快速的以较高的性价比对系统进行变更的能力，包括：可维护性、可扩展性、结构重构、可移植性。
* 功能性，系统所能完成所期望的工作的能力。
* 可变性，系统结构经过扩充或变更而成为新体系结构的能力，尤其是作为软件产品线时，该特性是很重要的。
* 互操作性，需要暴露出边界的UI和精心设计的接口。

1. 评估中的重要概念

敏感点(sensitivity point)，是一个或者多个构件的特性，研究敏感点可使得设计人员或分析员明确实现资料目标时需要注意什么。

权衡点(tradeoff point)，影响多个质量属性的特性，是多个质量属性的敏感点，例如修改加密级别将同时影响安全性和性能。

风险承担者(stakeholder)，也称为利益相关人，由于体系结构涉及很多人的利益，每个人都会都该结构施加不同的影响，已达到自己的目标。

场景(scenarios)，通过设定场景，找出质量目标，并作为判定该体系结构优劣的标准。一般使用刺激（stimulate, environment, response）三方面对场景进行描述。

1. 主要评估方法

* SAAM(Scenarios-based Architecture Analysis Method)

卡耐基梅隆大学Kazman于1983提出的最早的一种形成文档并得到广泛应用的软件质量属性的分析评价体系。其主要概念包括：特定目标，SAAM的目标时对描述应用程序属性的文档进行假设和原则的验证；评估技术，场景技术；质量属性，把任何形式的质量属性都具体化场景，可修改性是该分析的主要属性；风险承担者；体系结构描述；方法活动，输入问题是问题描述、需求生命、体系结构描述，其包含步骤，场景开发、体系结构描述、单一场景评估、场景交互和总体评估；可重用性；方法验证，医用到空中交通管制系统、嵌入式音频系统等。

* ATAM(Architecture Tradeoff Analysis Method)

基于SAAM，主要针对性能、实用性、安全性和可修改性，在系统开发前，对质量属性进行评价和折中。其主要概念包括：特定目标，多个相互影响的质量属性；质量属性；风险承担者；体系结构描述，受历史遗留系统、互操作性和以前失败的项目约束，基于Kruchten的4+1视图；评估技术，提供三种不同的类型的场景，分别是用例（典型场景）、增长场景（修改场景）、探测场景（受压时的极端场景），并且使用定性的启发式分析方法对一个质量属性构造一个精确分析模型进行分析；方法的活动包括**4个主要的活动领域**（阶段），在之后详细介绍；领域知识的可重用性，领域知识库通过基于属性的体系结构风格(ABAS, Attribute-Based Architecture Style)维护；方法验证。

4个主要的活动领域

场景和需求收集：收集场景、收集需求/约束/环境

体系结构视图和场景实现：描述体系结构视图、实现场景

属性模型的构造和分析：特定属性分析，选用优秀的单一理论

折中：标志折中、标志的敏感性