

NEW PLAN

软考高级架构师

• — 段 新 征 程 •





大纲介绍





软件架构风格



软件架构风格的概念：是描述某一特定应用领域中系统组织方式的**惯用模式**。架构风格定义一个系统家族，即一个**架构定义**、一个**词汇表**和一组**约束**。架构定义描述了系统的整体结构和组织方式。词汇表中包含一些构件和连接件类型，而这组约束指出系统是如何将这些构件和连接件组合起来的。简单来说，软件体系结构风格就是一个模板，它规定了特定应用领域中软件系统应该如何构建。



软件架构风格的作用：反映了领域中众多系统所共有的**结构和语义特性**，并指导如何将各个模块和子系统有效地组织成一个完整的系统。对软件架构风格的研究和实践促进对设计的重用，一些经过实践证实的解决方案也可以可靠地用于解决新的问题。



架构设计的一个核心问题是能否达到架构级的软件复用，强调对架构设计的重用。



架构风格定义了用于描述系统的术语表和一组指导构建系统的规则。



软件架构风格



数据流风格：面向数据流，按照一定的顺序从前向后执行程序，代表的风格有批处理序列、管道-过滤器。



调用/返回风格：构件之间存在互相调用的关系，一般是显式的调用，代表的风格有主程序/子程序、面向对象、层次结构、客户端服务器。



独立构件风格：构件之间是互相独立的，不存在显式的调用关系，而是通过某个事件触发、异步的方式来执行，代表的风格有进程通信、事件驱动系统(隐式调用)。



虚拟机风格：自定义了一套规则供使用者使用，使用者基于这个规则来开发构件，能够跨平台适配，代表的风格有解释器、基于规则的系统。



以数据为中心风格(数据共享风格、仓库风格)：以数据为中心，所有的操作都是围绕建立的数据中心进行的，代表的风格有数据库系统、仓库系统、黑板系统。



软件架构风格-数据流风格



批处理序列：构件为一系列固定顺序的计算单元，多件事情同步顺序执行，**构件之间只通过数据传递交互。每个处理步骤是一个独立的程序，每一步必须在其前一步结束后才能开始，数据必须是完整的，以整体的方式传递。**比如批量图像处理：一次性处理大量图像，例如调整大小、添加水印或转换格式。



管道-过滤器：每个构件都有一组输入和输出，构件读取输入的数据流，经过内部处理，产生输出数据流。**前一个构件的输出作为后一个构件的输入**，前后数据流关联。过滤器就是构件，连接件就是管道。比如文本处理管道：在文本分析中，可以将文本处理划分为分词、词干提取、情感分析等多个阶段，每个阶段是一个过滤器。



早期编译器就是采用的这种架构，要一步一步处理的，均可考虑此架构风格。



二者区别：**批处理风格通常一次性处理大量数据，离线执行，适用于批量处理任务。管道过滤器风格将任务分解为多个阶段，每个阶段逐个处理数据，通常是实时或近实时执行，适用于可组合和可扩展的任务。**

软件架构风格-调用/返回风格



主程序/子程序：单线程控制，把问题划分为若干个处理步骤，构件即为主程序和子程序，子程序通常可合成为模块。过程调用作为交互机制，充当连接件的角色。



面向对象：构件是对象，对象是抽象数据类型的实例。连接件即使对象间交互的方式，对象是通过函数和过程的调用来交互的。



层次结构：构件组成一个层次结构，连接件通过决定层间如何交互的协议来定义。每层为上一层提供服务，使用下一层的服务，只能见到与自己邻接的层。修改某一层，最多影响其相邻的两层(通常只能影响上层)。优点是可以将一个复杂问题分解成一个增量步骤序列的实现。缺点是并不是每个系统都可以很容易的划分为分层的模式，并且因为进行层次调用，会影响效率。



客户端服务器(C/S)：早期是两层C/S架构模式，由三个部分组成：数据库服务器、客户端服务器和网络，其中服务器负责数据管理，客户端完成用户交互，称之为“胖客户机，瘦服务器”，后期又增加了一个应用服务器，称之为三层C/S架构,分别是表示层、功能层和数据层，表示层就是用户交互、功能层就是业务逻辑处理、数据层就是数据库处理，以上三层独立；



软件架构风格-独立构件风格



进程通信: 构件是独立的**进程**, 连接件是**消息传递**。构件通常是命名过程, 消息传递的方式可以是**点对点、异步或同步方式, 以及远程过程(方法)调用**等。进程通信涉及不同的进程或线程之间的通信和数据共享。这些进程可以运行在同一台计算机上, 也可以分布在不同的计算机上。

- 举例: 一个典型的示例是一个多线程的文件下载器, 其中一个线程负责下载文件, 另一个线程负责监视下载进度。这两个线程需要通过进程通信来共享下载状态信息, 以便监视线程可以显示下载进度。



事件驱动系统(隐式调用): 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**。构件中的过程在一个或多个事件中注册, **当某个事件被触发时, 系统自动调用在这个事件中注册的所有过程**。

- 举例: 一个典型的示例是图形用户界面(GUI)应用程序。用户的操作(如点击按钮、键盘输入)会生成事件, 应用程序的控件或组件会注册事件处理程序来响应这些事件。



主要**优点**是为软件复用提供了强大的支持, 为构件的维护和演化带来了方便; 缺点是构件放弃了对系统**计算的控制, 只能被动的控制**。



软件架构风格-虚拟机风格



解释器：通常包括一个完成解释工作的解释引擎、一个包含将被解释的代码的存储区、一个记录解释引擎当前工作状态的数据结构，以及一个记录源代码被解释执行的进度的数据结构。涉及解析和执行一系列指令或命令，通常通过解释器来实现。解释器将文本或代码解析成可执行的操作。缺点是执行效率低。

- 举例：编程语言的解释器是最常见的示例。例如，Python解释器会解释和执行Python编写的代码。用户输入一个数学表达式，解释器会将其解析并执行计算，然后返回结果。例如，用户输入"2 + 3"，解释器会计算并返回"5"。



基于规则的系统：包括规则集、规则解释器、规则/数据选择器和工作内存，一般用在人工智能领域和DSS(决策支持系统)中。适用于根据一组事先定义的规则或条件来控制系统的行为。这种风格通常用于实现灵活的业务规则和决策逻辑。

- 举例：银行的贷款审批系统。系统使用基于规则的风格，根据客户的信用评分、贷款金额和其他因素来应用一组贷款批准规则。如果满足规则的条件，系统将自动批准或拒绝贷款申请。



软件架构风格-数据为中心系统



仓库风格的架构将数据存储在一个中央仓库或数据库中，各个组件可以从仓库中读取和写入数据。组件之间通过共享数据仓库进行通信和协作。

- 示例说明：假设一个天气预报系统具有多个数据源，包括气象站、卫星数据和气象传感器。这些数据源将数据写入共享的中央仓库，然后天气预报应用程序可以定期查询仓库以获取最新的气象信息并生成天气预报。



黑板风格的架构类似于一个黑板或公告板，多个独立的组件称为"专家"共享一个公共存储区(黑板)，它们可以读取和写入数据。专家根据黑板上的信息进行推断和决策，适用于解决复杂的非结构化问题。

- 示例说明：假设一个医学诊断系统包括放射科医生、心脏专家和内科医生。每个专家可以根据患者的病历信息(例如X光片、心电图和实验室报告)向黑板提交自己的诊断。黑板负责集成各个专家的诊断，并生成最终的诊断结果。

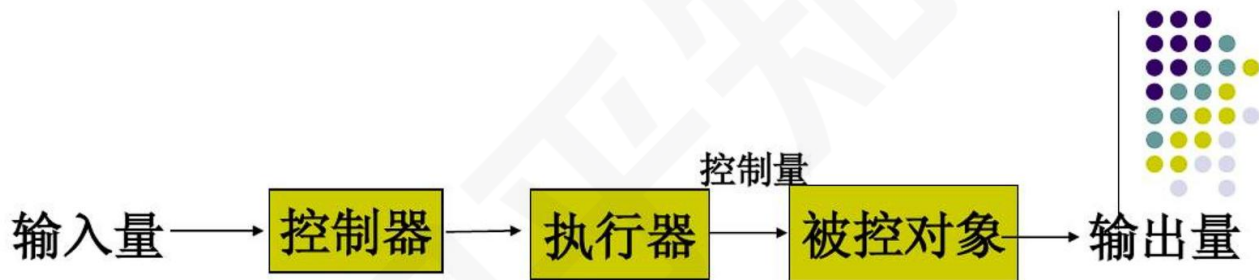


总之，仓库风格强调数据的集中存储和共享，组件通过访问共享的仓库来交互。而黑板风格侧重于多个独立的组件共享一个中央知识存储区，根据共享的信息进行推断和决策。这两种风格在处理复杂问题和协作方面都具有一定的优势。

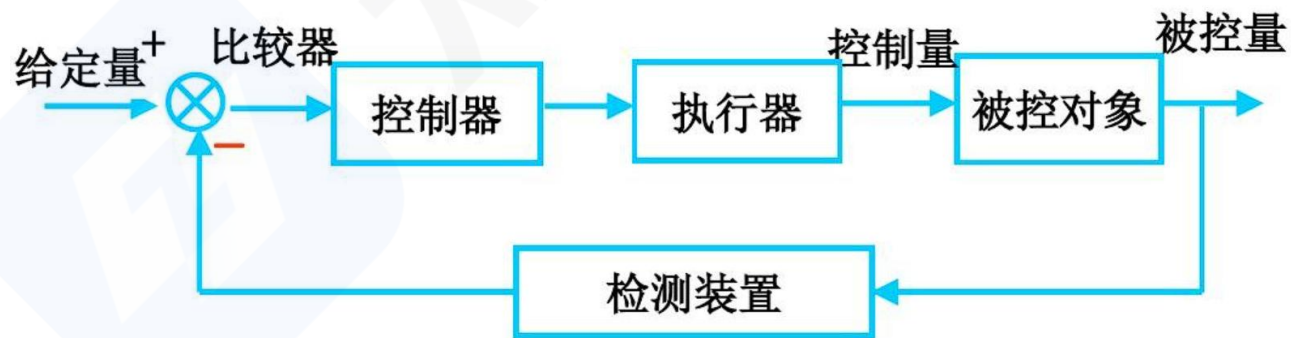
软件架构风格



闭环控制：当软件被用来操作一个物理系统时，软件与硬件之间可以粗略的表示为一个反馈循环，这个反馈循环通过接受一定的输入，确定一系列的输出，最终使环境达到一个新的状态，适合于嵌入式系统，涉及连续的动作与状态。比如开空调，不会一调到某个温度就马上能到该温度，是逐渐接收室内的温度来变化输出空调的冷气；



对于闭环控制系统，通常可以用以下的方框图表示：

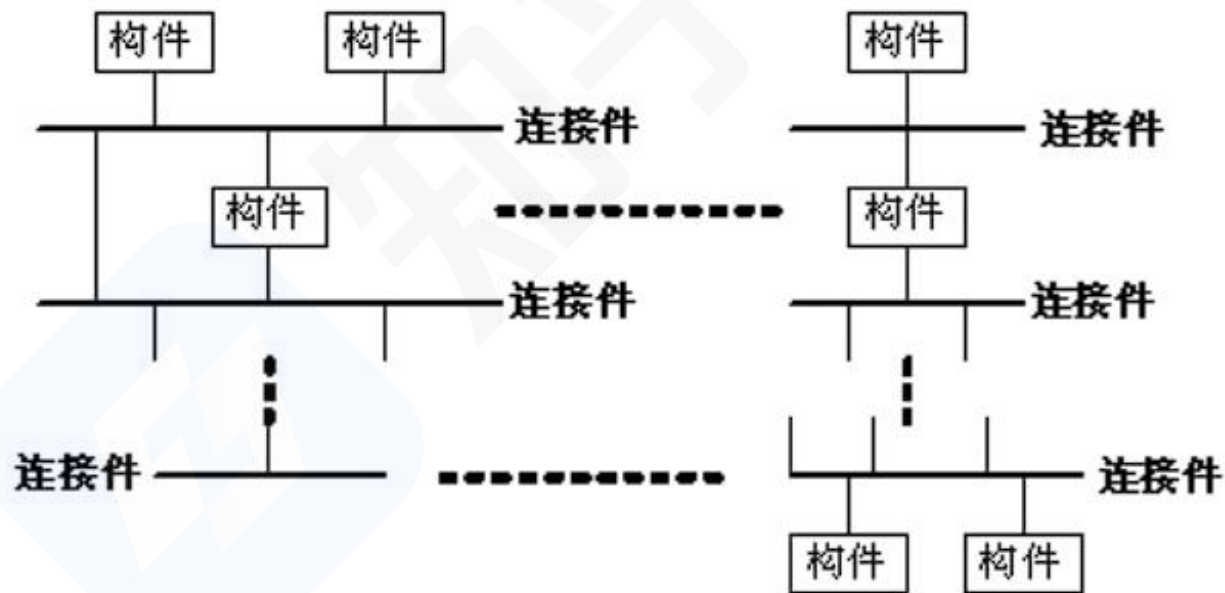


软件架构风格



构C2体系结构风格可以概括为：**通过连接件绑定在一起的按照一组规则运作的并行构件网络**。C2风格中的系统组织规则如下：

- 系统中的构件和连接件都有一个顶部和一个底部；
- 构件的顶部应连接到某连接件的底部，构件的底部则应连接到某连接件的顶部，而构件与构件之间的直接连接是不允许的；
- 一个连接件可以和任意数目的其它构件和连接件连接；
- 当两个连接件进行直接连接时，必须由其中一个的底部到另一个的顶部。





软件架构风格

序号	架构风格名	常考关键字及实例	简介
1	数据流-批处理	传统编译器，每个阶段产生的结果作为下一个阶段的输入，区别在于整体。	一个接一个，以整体为单位。
2	数据流-管道-过滤器		一个接一个，前一个输出是后一个输入。
3	调用/返回-主程序/子程序	无	显示调用，主程序直接调用子程序。
4	调用/返回-面向对象		对象是构件，通过对象调用封装的方法和属性。
5	调用/返回-层次结构		分层，每层最多影响其上下两层，有调用关系。
6	独立构件-进程通信		进程间独立的消息传递，同步异步。
7	独立构件-事件驱动(隐式调用)	事件触发推动动作，如程序语言的语法高亮、语法错误提示。	不直接调用，通过事件驱动。
8	虚拟机-解释器	自定义流程，按流程执行，规则随时改变，灵活定义，业务灵活组合。 机器人。	解释自定义的规则，解释引擎、存储区、数据结构。
9	虚拟机-规则系统		规则集、规则解释器、选择器和工作内存，用于DSS和人工智能、专家系统。
10	仓库数据库	现代编译器的集成开发环境IDE,以数据为中心。 又称为数据共享风格	中央共享数据源，独立处理单元。
11	仓库超文本		网状链接，多用于互联网。
12	仓库-黑板		语音识别、知识推理等问题复杂、解空间很大、求解过程不确定的这一类软件系统，黑板、知识源、控制。
13	闭环过程控制	汽车巡航定速，空调温度调节，设定参数，并不断调整。	发出控制命令并接受反馈，循环往复达到平衡。
14	C2风格	构件和连接件、顶部和底部。	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



考试真题



【2022年】48-49.软件体系结构风格是描述某一特定应用领域中系统组织方式的惯用模式，其中，在批处理风格软件体系结构中，每个处理步骤是一个独立的程序，每一步必须在前一步结束后才能开始，并且数据必须是完整的，以()的方式传递。基于规则的系统包括规则集、规则解释器、规则/数据选择器及()。

A.迭代

B.整体

C.统一格式

D.递增

A.解释引擎

B.虚拟机

C.数据

D.工作内存



【2021年】48.软件架构风格是描述某一特定应用领域中系统组织方式的惯用模式，按照软件架构风格，物联网系统属于(48)软件架构风格。

A.层次型

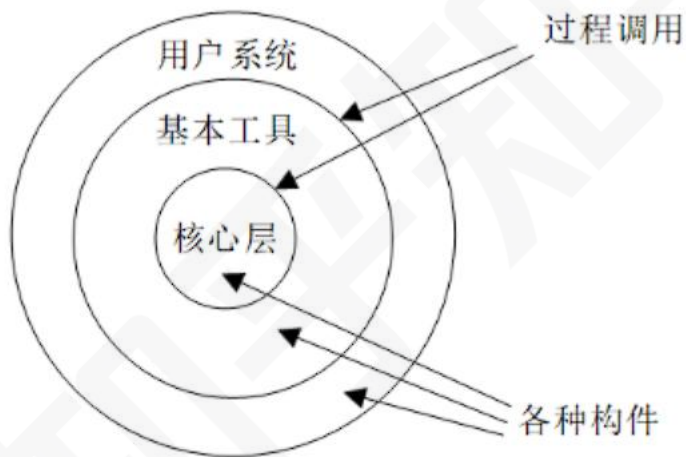
B.事件系统

C.数据流

D.C2



【2019】对软件体系结构风格的研究和实践促进了对设计的复用。Garlan和Shaw对经典体系结构风格进行了分类。其中，（ ）属于数据流体系结构风格；（ ）属于虚拟机体系结构风格；而下图描述的属于（ ）体系结构风格。



A.面向对象

B.事件系统

C.规则系统

D.批处理

A.面向对象

B.事件系统

C.规则系统

D.批处理

A.层次型

B.事件系统

C.规则系统

D.批处理



考试真题



【2017】某公司拟开发一个VIP管理系统，系统需要根据不同商场活动，不定期更新VIP会员的审核标准和VIP折扣系统。针对上述需求，采用()架构风格最为合适。

A.规则系统

B.过程控制

C.分层

D.管道-过滤器



【2016】()架构风格可以概括为通过连接件绑定在一起按照一组规则运作的并行构件。

A.C2

B.黑板系统

C.规则系统

D.虚拟机

T H E E N D

功不唐捐，玉汝于成！

• 开 启 新 征 程 •

