分析比较KWIC系统实现四种不同体系结构风格

目录

[分析比较KWIC系统实现四种不同体系结构风格 1](#_Toc70884638)

[1.管道-过滤器风格 1](#_Toc70884639)

[2. 主程序/子程序风格（Main Program/Subroutine Style） 2](#_Toc70884640)

[3. 面向对象式风格 2](#_Toc70884641)

[4. 分层的方法 2](#_Toc70884642)

### 1.管道-过滤器风格

管道-过滤器风格将系统的功能逻辑建立为部件集合。每个部件实例完成一个对数据流的独立功能处理，它接收数据流输入，进行转换和增量后进行数据流输出。连接件是管道机制，它将前一个过滤器的数据流输出传递给后一个过滤器作为数据流输入。连接件也可能会进行数据流的功能处理，进行转换或增量，但连接件进行功能处理的目的是为了适配前一个过滤器的输出和后一个过滤器的输入，而不是为了直接承载软件系统的需求。

各个过滤器可以并发执行。每个过滤器都可以在数据输入不完备的情况下就开始进行处理，每次接到一部分数据流输入就处理和产生一部分输出。这样，整个的过滤器网络就形成了一条流水线

优点：易于理解并支持变换的复用。工作流风格与很多业务处理系统体系结构很匹配。通过添加变换的方式进行进化是很显然的。可以实现为顺序的系统，也可以实现为并发的系统。

（1）使得软构件具有良好的隐蔽性和高内聚、低耦合的特点；  
（2）允许设计者将整个系统的输入/输出行为看成是多个过滤器的行为的简单合成；  
（3）支持软件重用。重要提供适合在两个过滤器之间传送的数据，任何两个过滤器都可被连接起来；

（4）系统维护和增强系统性能简单。新的过滤器可以添加到现有系统中来；旧的可以被改进的过滤器替换掉；

（5）允许对一些如吞吐量、死锁等属性的分析；

（6）支持并行执行。每个过滤器是作为一个单独的任务完成，因此可与其它任务并行执行。 缺点：在通信变换间所传输的数据格式必须协商好。每个变换必须解析它的输入并写成约定的格式输出。这增加了系统的负荷，意味着不可能复用实用不兼容数据结构的函数变换。

### **2. 主程序/子程序风格（Main Program/Subroutine Style）**

将系统组织成层次结构，包括一个主程序和一系列子程序。主程序是系统的控制器，负责调度各子程  序的执行。各子程序又是一个局部的控制器，调度其子程序的执行。 优缺点：

优点：易于理解并支持变换的复用。工作流风格与很多业务处理系统体系结构很匹配。通过添加变换的方式进行进化是很显然的。可以实现为顺序的系统，也可以实现为并发的系统。

 缺点：在通信变换间所传输的数据格式必须协商好。每个变换必须解析它的输入并写成约定的格式输出。这增加了系统的负荷，意味着不可能复用实用不兼容数据结构的函数变换。 运行结果：

    面向对象式风格，需要强调说明的是它的“对象”是部件，属于高层结构的元素，虽然名称相同，但它并不是面向对象方法中所述的“对象”实体。“面向对象式”风格的命名是因为它借鉴了面向对象方法的思想，而不是因为它使用面向对象方法实现体系结构，这也是在该风格名称中有一个“式”字的原因。

### 3. 面向对象式风格

主要实现机制是模块实现，它将每个对象部件实例都实现为一个模块。存在连接的对象部件实例之间会存在模块的导入/导出关系。

每个模块内部可以是基于面向对象方法的实现，也可以是基于结构化方法的实现。 优缺点：

优点：1内部实现的可修改性。因为面向对象式风格要求封装内部数据，隐藏内部实现，所以可以在不影响外界的情况下，变更其内部实现。

      2易开发、易理解、易复用的结构组织。面向对象式风格将系统组织为一系列平等、自治的单位，每个单位负责自身的正确性，不同单位之间仅仅是通过方法调用相连接，这非常契合模块化的思想，能够建立一个易开发、易理解、易复用的实现结构。

 缺点：当一个对象和其他对象交互，它必须知道其他对象的标识。当一个对象的标识改变时，必须修改那些显式调用它的对象。

分离性和独立性的概念是体系结构设计的基础，因为分离性和独立性使得变更得到局部化。分层体系结构模式是实现分离性和独立性的一个方式。这里，一个分层系统（Layered Systems）按照层次结构组织，系统的功能被划分成几个独立的层次，每一层只依赖紧接的下一层所提供的服务和设施。定义的一系列不同层次各自完成其自身的操作，这些操作逐渐接近机器的指令集。在外层，构件完成建立用户界面的操作；在内层，构件完成建立操作系统接口的操作；中间层提供各种实用工具服务和应用软件功能。

### 4. 分层的方法

支持系统的增量式开发。如一个层被开发完，该层提供的服务就可以被用户使用了。这个体系结构还是可改变的和可移植的。如果一层的接口被保留下来，这个层就能被另外的一个对等层替换。当一层的接口改变或增加了新设施的时候，只有毗邻的层受影响。因为分层系统的抽象机依赖的是内层中的抽象机，因此，转换到其他机器上实现是比较容易的，此时只有内部与具体机器相关的层需要重新实现以适应不同的操作系统或数据库。 优缺点：

优点：允许在接口保持不变的条件下更换整个一层。在每一层中可以提供冗余服务（例如身份验证）以增加系统的可靠性。

缺点：在具体实践中，在各层之间提供一个干净的分离通常是困难的，高层可能不得不直接与低层进行直接交互而不是间接通过紧邻的下一层进行交互。性能可能是个问题因为服务请求会在每一层中被处理所以会需要多