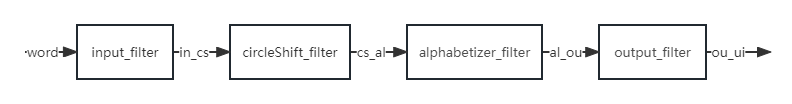
**pipeline风格设计：**

流程图：



图中矩形代表过滤器：

Input\_filter：输入过滤器；

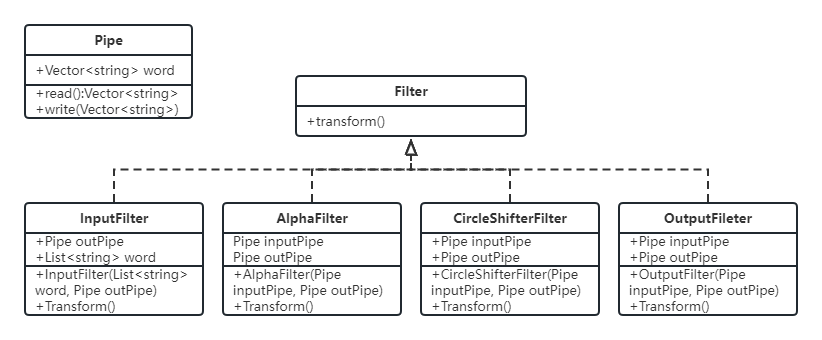
Circleshift\_filter：循环移位过滤器；

Alphabetizer\_filter：排序过滤器；

Output\_filter：输出过滤器。

箭头代表管道。过滤器之间只能通过管道实现数据的传递。

类图：



Pipe类中word用于存储管道中的数据，read函数实现了从管道中读出的功能，write函数实现了将数据读入管道的功能。

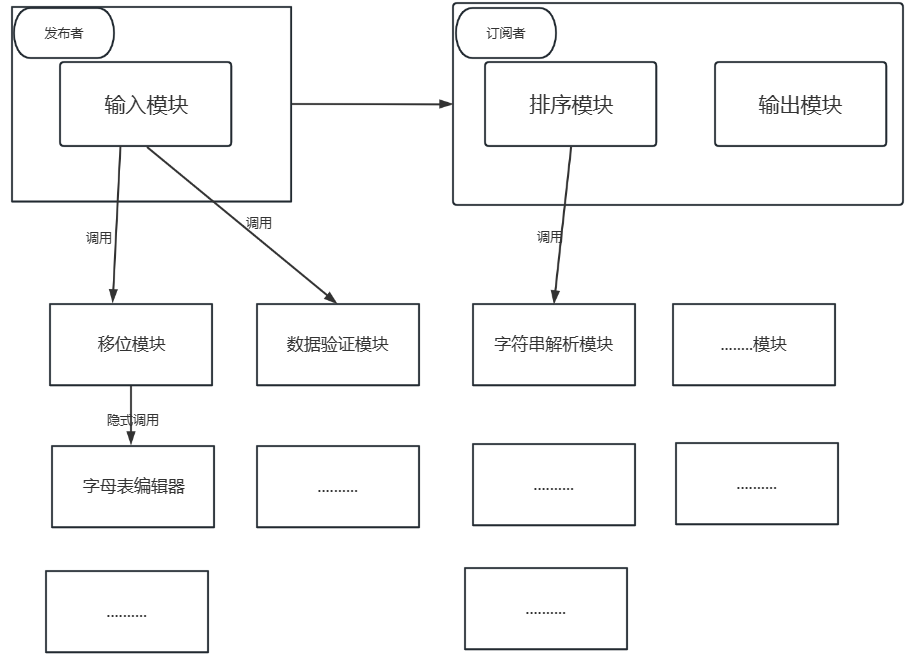
虚基类Filter中声明了transform函数，InputFilter等类中继承实现了Filter类中的transform函数。

InputFilter类中的构造函数有两个参数，分别代表输入的数据和输出的数据。Transform函数实现了读入数据的转化。

AlphaFilter类、CircleShifterFilter类和OutputFilter类的构造函数的两个参数也分别用户实现读入和写出数据。Transform函数完成排序、移位和输出功能。

在每个函数的transform函数中，会首先通过inputPipe的read函数接收从上一级传来的数据（inputFilter类直接由传入的word参数获得数据），经过自己这一阶段的处理后通过outputPipe的write函数将数据写入下一级，以此完成数据的处理。

**隐含调用风格设计：**



1.发布者：

KWIC系统中的发布者是输入模块，负责从外部系统或文件读取数据，并将其转换成KWIC系统可处理的格式，将数据发送给下游模块进行处理。

2.订阅者：

KWIC系统中的订阅者包括两个模块，分别是排序模块和输出模块。

排序模块：订阅输入模块发送的数据流，对其中的关键字进行排序，并将排序结果传递给下游模块。

输出模块：订阅排序模块发送的数据流，并将排序结果输出到指定位置或应用程序。

3.隐含调用：

KWIC系统中存在一些隐含调用，例如：

①输入模块会隐含调用数据验证模块，对输入数据进行格式和内容的检查。

②排序模块会隐含调用字符串解析模块，将文本按照给定的规则进行分词和解析。

通过以上发布/订阅风格的设计，可以将问题进行分解，同时进行并行处理调用，系统能够实现高效的数据处理功能，每个模块之间互相独立并具有良好的扩展性和重用性。由于数据被抽象地访问，它也使计算与数据表示的变化隔绝。而隐含调用则使得系统更加完整和可靠，提高了软件质量。

但是缺点是很难控制隐式调用模块的处理顺序。此外，由于调用是数据驱动的，这种分解的最自然实现往往比之前考虑的分解使用更多空间。而且，构件放弃了自身对系统计算的控制，无法保证事件被处理的先后顺序，进而造成数据管理也是一个较大的考验。同时，模块之间的间接通信，也会降低一部分性能。