

管道过滤器模式

管道过滤器模式由一系列处理单元组成，每个单元的**输出**是下一个单元的**输入**。在连续的单元之间通常提供一定的缓冲。

```
graph LR; In(( )) --> U1[ ]; U1 --> B1(( )); B1 --> U2[ ]; U2 --> B2(( )); B2 --> U3[ ]; U3 --> B3(( )); B3 --> U4[ ]; U4 --> Out(( ))
```

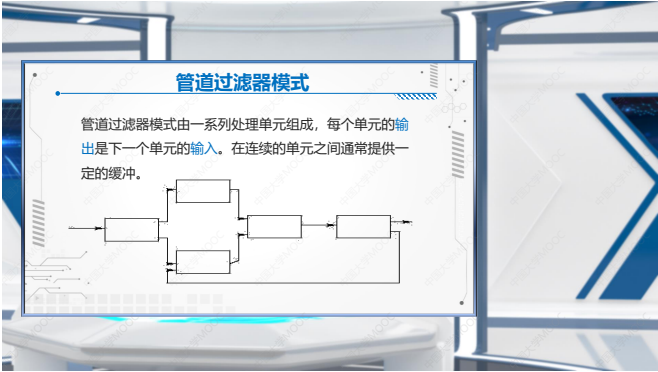
The diagram illustrates the Pipeline Filter Pattern. It shows a sequence of four processing units (represented by rectangles) connected by buffers (represented by circles). The input enters the first unit, and the output of each unit flows into a buffer, which then feeds into the next unit. This structure allows for asynchronous processing and backpressure management.

管道过滤器模式

管道过滤器模式由一系列处理单元组成，每个单元的**输出**是下一个单元的**输入**。在连续的单元之间通常提供一定的缓冲。

```
graph LR; Input(( )) --> U1[ ]; U1 --> B1(( )); B1 --> U2[ ]; U2 --> B2(( )); B2 --> U3[ ]; U3 --> B3(( )); B3 --> U4[ ]; U4 --> Output(( ))
```

The diagram illustrates the Pipeline Filter Pattern. It shows a sequence of four processing units (represented by rectangles) connected by buffers (represented by circles). The flow starts from an input on the left, passes through the first unit, then through a buffer, then through the second unit, then through another buffer, then through the third unit, then through a third buffer, and finally through the fourth unit to an output on the right. This structure allows for asynchronous processing and backpressure management.



管道过滤器模式

- 模式名称: 管道/过滤器模式
- 组件: 过滤器 - 处理数据

从输入读取数据流并输出数据流

- 连接器: 管道 - 数据翻译与传输

将过滤器的输出数据传送给其他过滤器

```
graph LR; Input[Data Stream] --> F1[F1]; F1 --> P1[P1]; P1 --> F2[F2]; F2 --> P2[P2]; P2 --> F3[F3]; F3 --> P3[P3]; P3 --> F4[F4]; F4 --> P4[P4]; P4 --> Output[Data Stream];
```

管道过滤器模式

- 模式名称: 管道/过滤器模式
- 组件: 过滤器 - 处理数据

从输入读取数据流并输出数据流

- 连接器: 管道 - 数据翻译与传输

将过滤器的输出数据传送给其他过滤器

```
graph LR; Input[Data Stream] --> F1[F1]; F1 --> P1[P1]; P1 --> F2[F2]; F2 --> P2[P2]; P2 --> F3[F3]; F3 --> P3[P3]; P3 --> F4[F4]; F4 --> P4[P4]; P4 --> Output[Data Stream];
```

- ### 管道过滤器模式

 - 模式名称: 管道/过滤器模式
 - 组件: 过滤器 - 处理数据

从输入读取数据流并输出数据流

 - 连接器: 管道 - 数据翻译与传输

将过滤器的输出数据传送给其他过滤器

```
graph LR; Input[Data Stream] --> F1[F1]; F1 --> P1[P1]; P1 --> F2[F2]; F2 --> P2[P2]; P2 --> F3[F3]; F3 --> P3[P3]; P3 --> F4[F4]; F4 --> P4[P4]; P4 --> Output[Data Stream];
```

- ### 管道过滤器模式

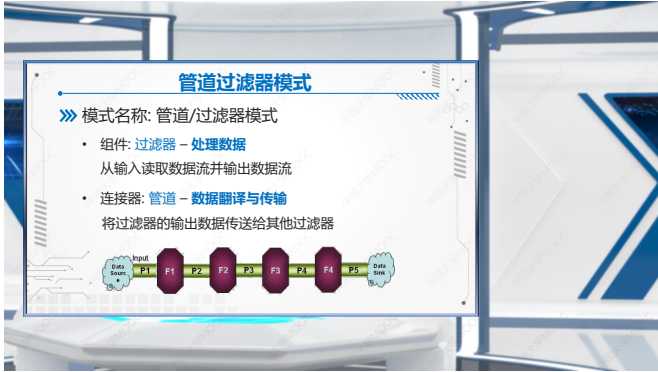
 - 模式名称: 管道/过滤器模式
 - 组件: 过滤器 - 处理数据

从输入读取数据流并输出数据流

 - 连接器: 管道 - 数据翻译与传输


将过滤器的输出数据传送给其他过滤器

```
graph LR; Input[Data Stream] --> F1[F1]; F1 --> P1[P1]; P1 --> F2[F2]; F2 --> P2[P2]; P2 --> F3[F3]; F3 --> P3[P3]; P3 --> F4[F4]; F4 --> P4[P4]; P4 --> Output[Data Stream];
```





管道过滤器模式

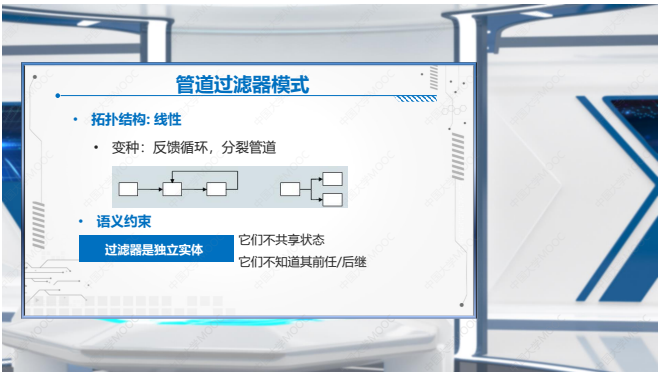
- 拓扑结构: 线性
 - 变种: 反馈循环, 分裂管道




- 语义约束
 - 过滤器是独立实体
 - 它们不共享状态
 - 它们不知道其前任/后继

- ## 管道过滤器模式
- 拓扑结构: 线性
 - 变种: 反馈循环, 分裂管道
- 
- 语义约束
 - 过滤器是独立实体
 - 它们不共享状态
 - 它们不知道其前任/后继


- ## 管道过滤器模式
- 拓扑结构: 线性
 - 变种: 反馈循环, 分裂管道
- 
- 语义约束
 - 过滤器是独立实体
 - 它们不共享状态
 - 它们不知道其前任/后继



- ## 管道过滤器模式
- 拓扑结构: 线性
 - 变种: 反馈循环, 分裂管道
- 
- 语义约束
 - 过滤器是独立实体
 - 它们不共享状态
 - 它们不知道其前任/后继

管道过滤器模式

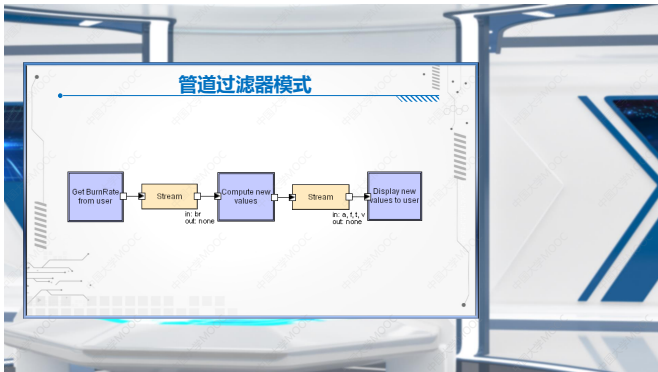
- 拓扑结构: 线性
 - 变种: 反馈循环, 分裂管道



- 语义约束
 - 过滤器是独立实体
 - 它们不共享状态
 - 它们不知道其前任/后继

管道过滤器模式

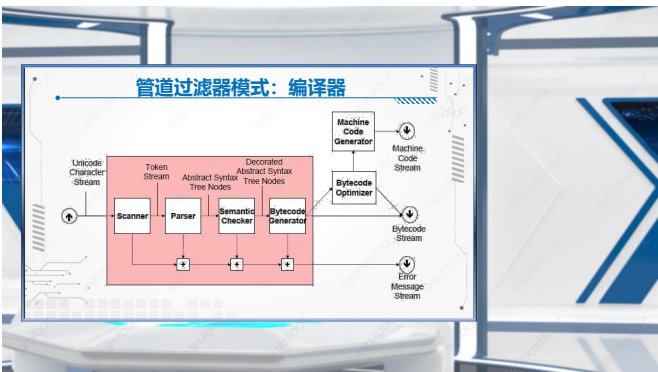
```
graph LR; A[Get BurnRate from user] --> B[Stream]; B --> C[Compute new values]; C --> D[Stream]; D --> E[Display new values to user]; B -- "in, out: none" --> C; D -- "in: 0, 1, 2, out: none" --> E;
```



管道过滤器模式：编译器

The diagram illustrates the Pipeline Filter Mode for a compiler, showing the flow of data through various components and streams:

- Unicode Character Stream** (Input Stream) feeds into the **Scanner**.
- The **Scanner** outputs a **Token Stream** to the **Parser**.
- The **Parser** outputs **Abstract Syntax Tree Nodes** to the **Semantic Checker**.
- The **Semantic Checker** outputs **Decorated Abstract Syntax Tree Nodes** to the **Bytecode Generator**.
- The **Bytecode Generator** outputs a **Bytecode Stream** to the **Bytecode Optimizer**.
- The **Bytecode Optimizer** outputs a **Machine Code Stream** to the **Machine Code Generator**.
- The **Machine Code Generator** outputs the final **Machine Code Stream**.
- Each component (Scanner, Parser, Semantic Checker, Bytecode Generator) also outputs to an **Error Message Stream** via a plus sign (+) in a box.



管道过滤器模式的优缺点

» 优点:

- 01 高内聚：过滤器是执行特定功能的自包含处理服务，具有较强的内聚性
- 02 低耦合：过滤器间仅通过管道通信
- 03 可重用：支持过滤器的重用
- 04 可简单地实现为并发或顺序系统

管道过滤器模式的优缺点

» 优点:

- 01 高内聚：过滤器是执行特定功能的自包含处理服务，具有较强的内聚性
- 02 低耦合：过滤器间仅通过管道通信
- 03 可重用：支持过滤器的重用
- 04 可简单地实现为并发或顺序系统

- ## 管道过滤器模式的优缺点
- » 优点:
- 01 高内聚：过滤器是执行特定功能的自包含处理服务，具有较强的内聚性
 - 02 低耦合：过滤器间仅通过管道通信
 - 03 可重用：支持过滤器的重用
 - 04 可简单地实现为并发或顺序系统

