Pipes and Filters

Context

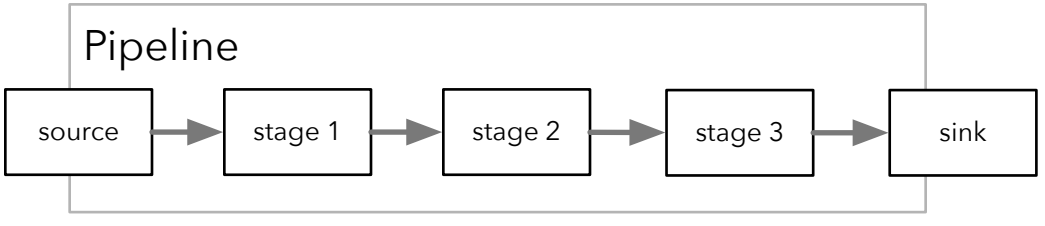
Processing data streams

Solution

Divide task into separate processing stages. Connect them sequentially: the output of one stage is the input to the next. A filter consumes and delivers data **incrementally**. A *data source* supplies the initial input, while a *data sink* consumes the final output. The data source, filters, and data sink are connected via pipes. The assembly is a processing pipeline. Some external entity constructs the pipeline.

Forces

- Data stream processing which naturally subdivides into stages
- May want to recombine stages
- Non-adjacent stages don't share information
- May desire different stages to be on different processors



Example: Unix pipes, Apache Spark

Question: What weaknesses do you see in pipes&filters?
How do you decide between *push* and *pull* data flow?