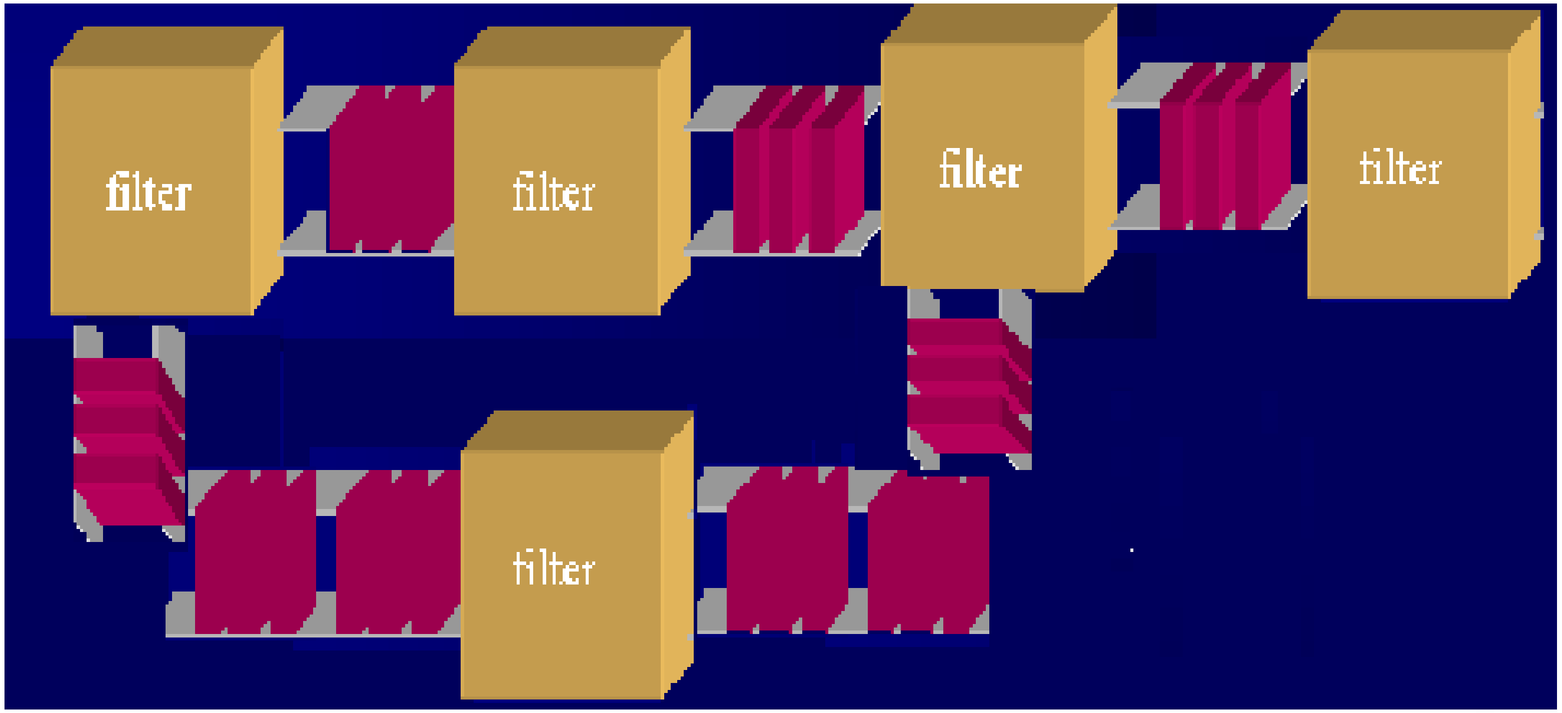
Software Architecture –
Pipe and Filter Architecture

Pipe and Filter Style



Pipe and Filter Style

- Type of Data Flow Architecture
- Filter is a component and pipe is a connector
- Filter has interfaces from which a set of inputs can flow in and a set of outputs can flow out.
- Incremental transformation of data by successive components.
- All data does not need to be processed for next filter to start working.
- Any set of filters may be combined in any order, although reasonable semantics are not guaranteed by this style.

Pipe and Filter Style

Filter

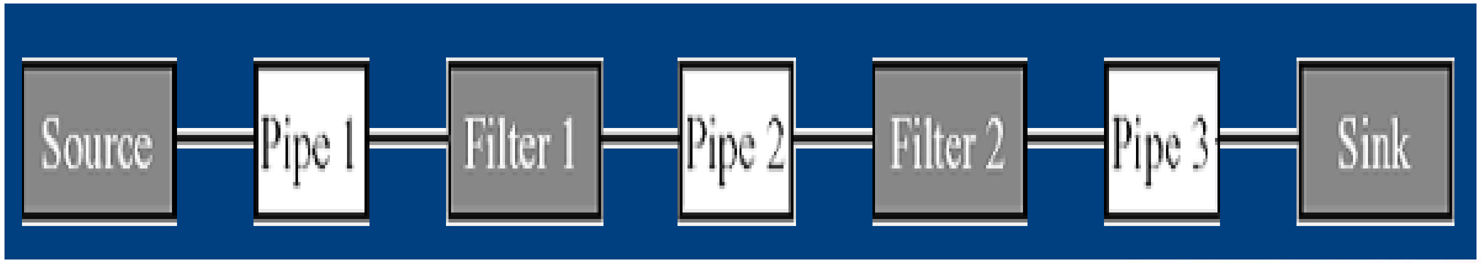
- Independent entities
- Does not share state with other filters.
- No do not know the identity to upstream and downstream filters.

Pipes

- Stateless data stream
- Source end feeds filter input and sink receives output.

Pipeline Architecture

- Common specialization of pipe and filter style is pipeline architecture
- This architecture restricts the topologies to linear sequences of filters.



Pipe and Filter Style:

Advantages and Disadvantages

Advantages:

- Simplicity – Allows designer to understand overall input/output behavior of a system in terms of individual filters.
- Maintenance and reuse
- Concurrent Execution –Each filter can be implemented as a separate task and be executed in parallel with other filters.

Pipe and Filter Style:

Advantages and Disadvantages

Disadvantages:

- Interactive transformations are difficult – Filters being independent entities designer has to think of each filter as providing a complete transformation of input data to output data.
- No filter cooperation.
- Performance – may force a lowest common denominator on data transmission

  -parse and unparse

  -latency

Example

- Compiler *(Example of pipeline architecture)*

  Stages: Lexical analysis,parsing,semantic analysis, code generation

- Programs written in Unix shell *(Example of pipeline architecture)*

  ls –l *.java | grep “foobar” | lpr –P gaston

- Functional programming

  Kahn’s example. 3 models –each goes through 3 kinds of algebraic operations

- Distributed systems.

  CORBA components : Push and pull model.

References

- http://www-2.cs.cmu.edu/afs/cs/project/able/ftp/intro_softarch/intro_softarch.pdf
- www.ics.uci.edu/~edashofy/classes/ics228/ 02-architecture-driven-development.ppt
- http://www.ics.uci.edu/~dsr/old-home-page/edo99-middleware.pdf
- G. Kahn. The semantics of a simple language for parallel programming.