Design Space Exploration of Stream-based Dataflow Architectures

Dr. ir. Bart Kienhuis,

Delft University of Technology

In cooperation with
Philips Research Laboratories

E-mail: kienhuis@cas.et.tudelft.nl

29 January, 1999

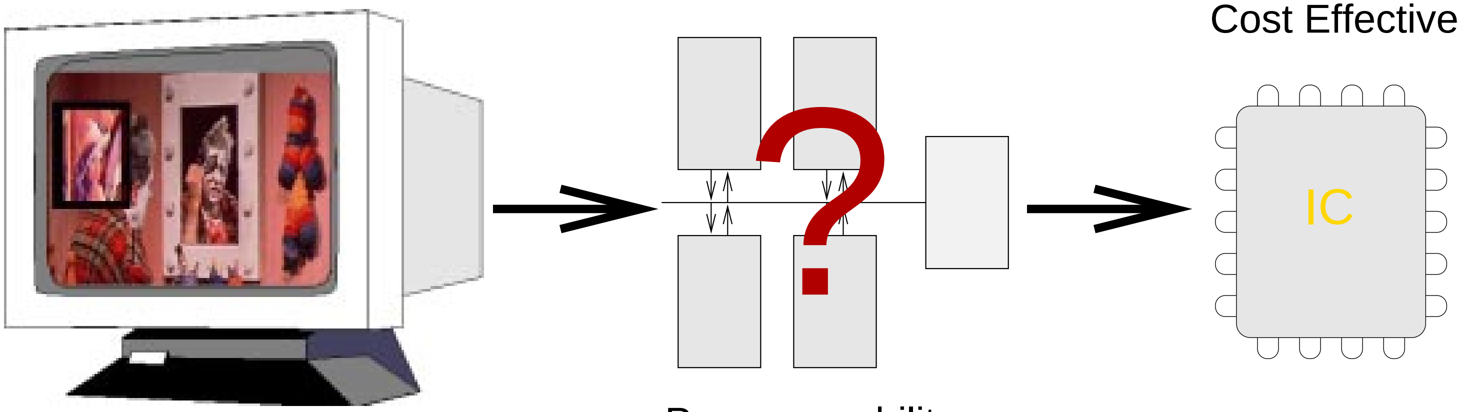
System Level Design

Application

Architecture

Integrated Circuit

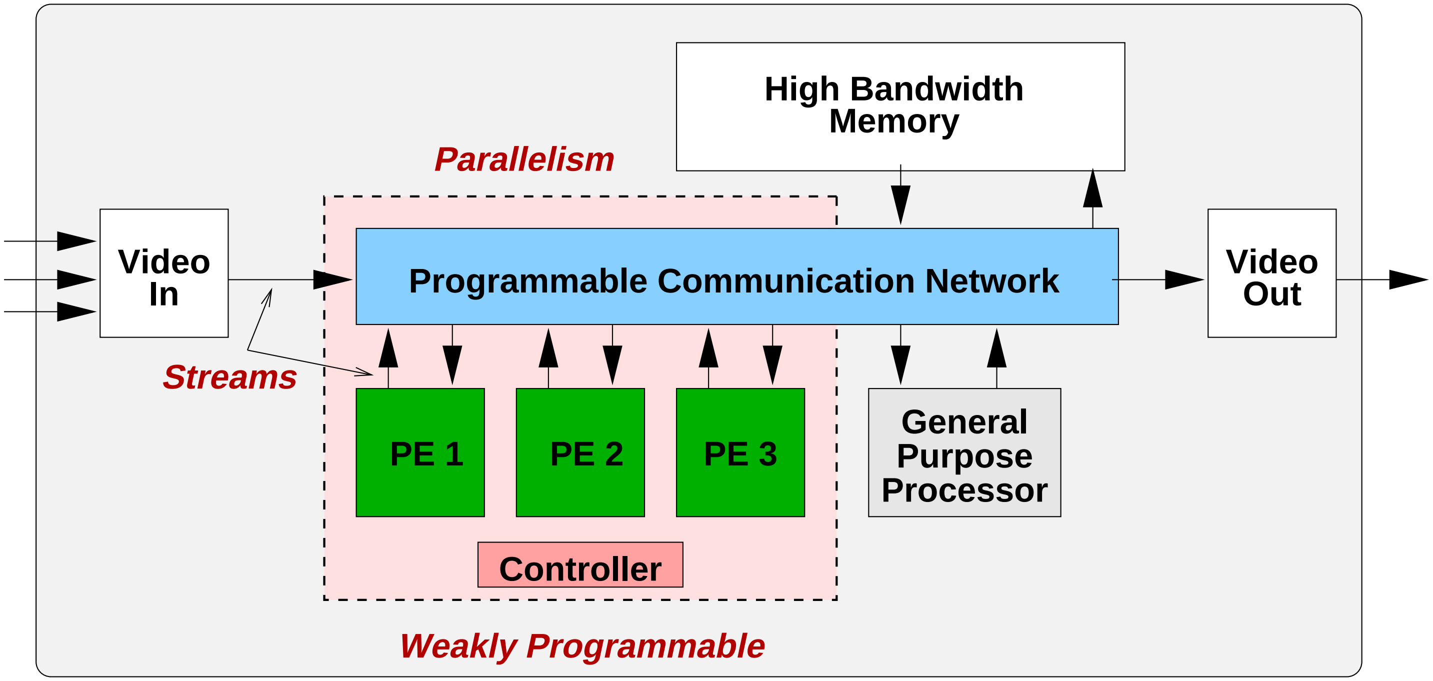
Cost Effective



(Real-time)

Programmability
Multi-functional
Multi-standard

New High-performance DSP Architectures



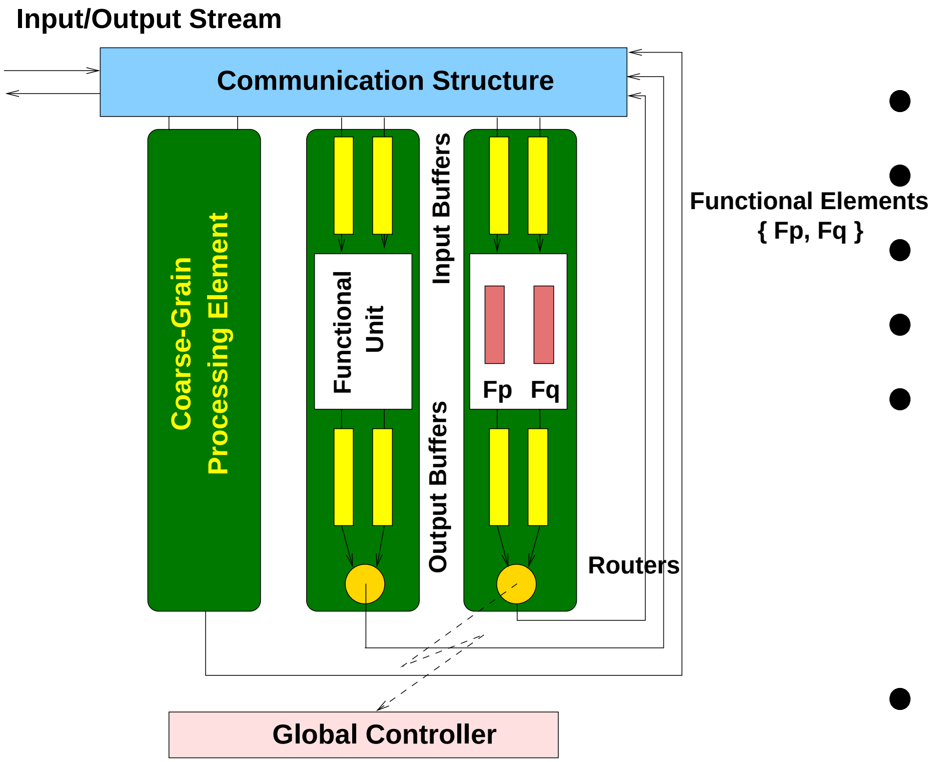
- Operations: 1 Gops - 10 Gops
- Bandwidth: 1000 Mbytes - 10.000 Mbytes per second

Focus is on **Stream-based Dataflow Architectures**

Outline

- Architecture Template & Video Applications
- The Y-chart Approach
- The Y-chart Environment for Stream-based Dataflow Architectures
  - Retargetable Simulator
  - Mapping
- Design Space Exploration
- Example
- Conclusions

Architecture Template



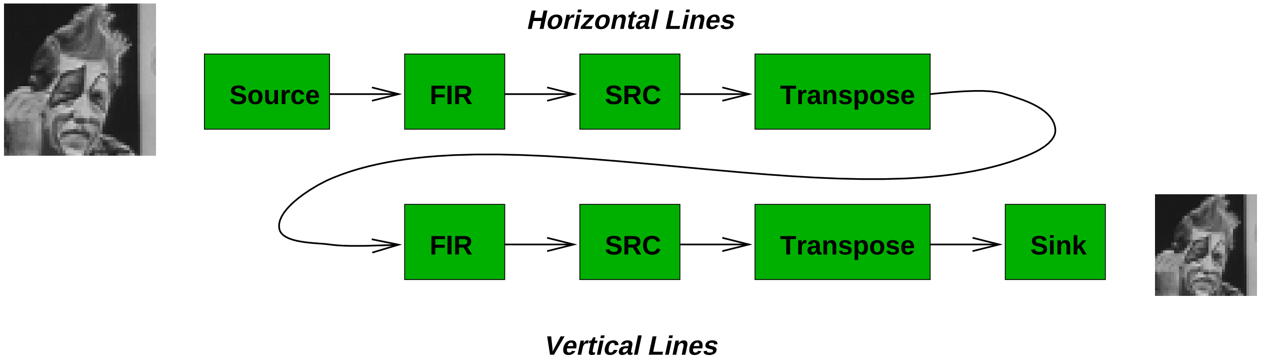
Design Choices

- Function Repertoire
- Grain Size
- Controller Protocol
- Buffer Capacity
- Packet Length

Metrics Constraints

- Throughput (real-time)
- Utilization

Video Algorithms



Specified as Kahn Process Networks (Lee&Parks'95)

- Dynamic Dataflow
- Deterministic execution trace (Kahn'74)

Assumptions

- Coarse-grained Functions
- Sample Based

Problem Statement

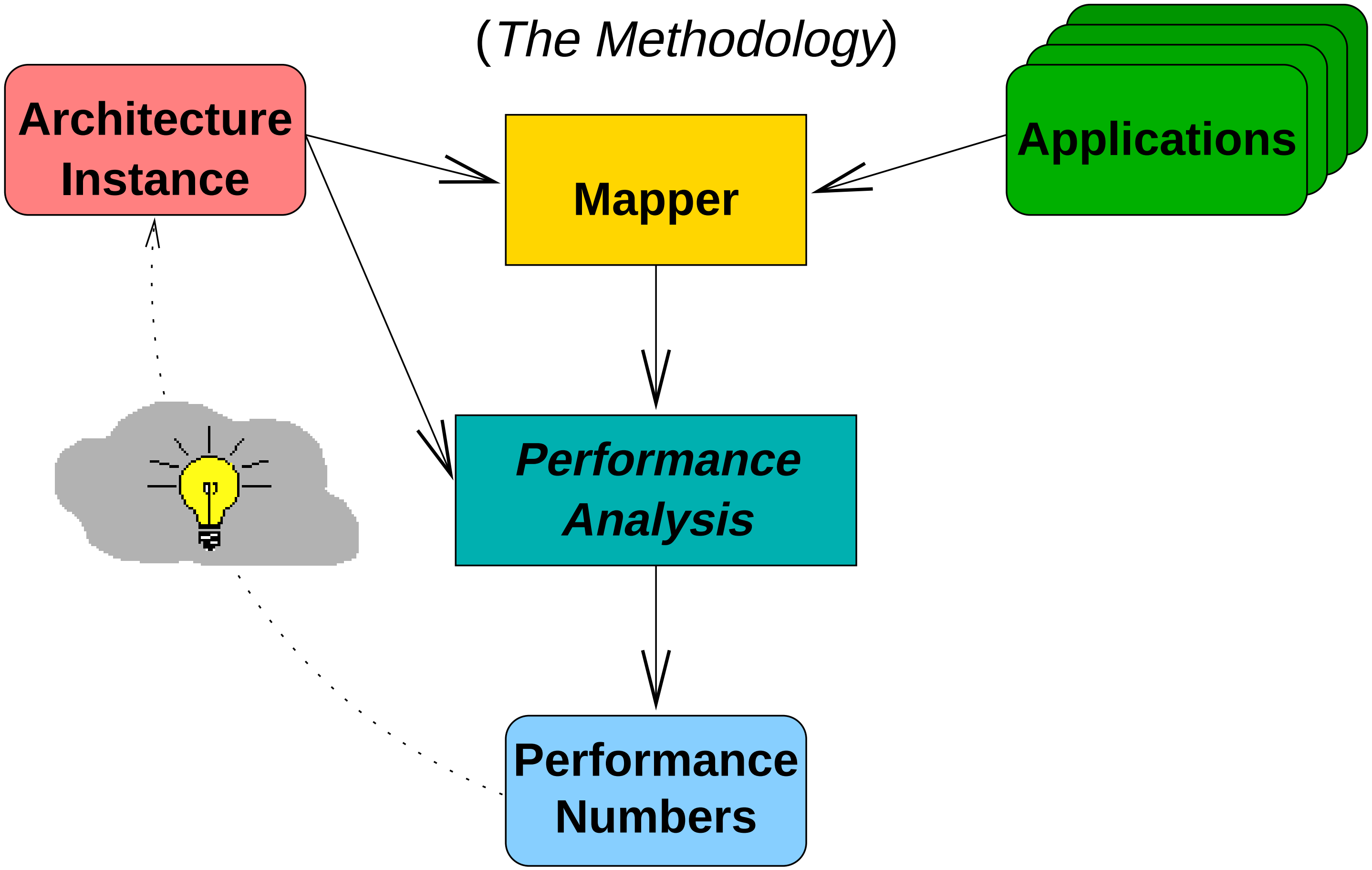
The Designer’s Problem

- Many design choices
- Need to evaluate different design alternatives

☞ General and structured design approaches are lacking

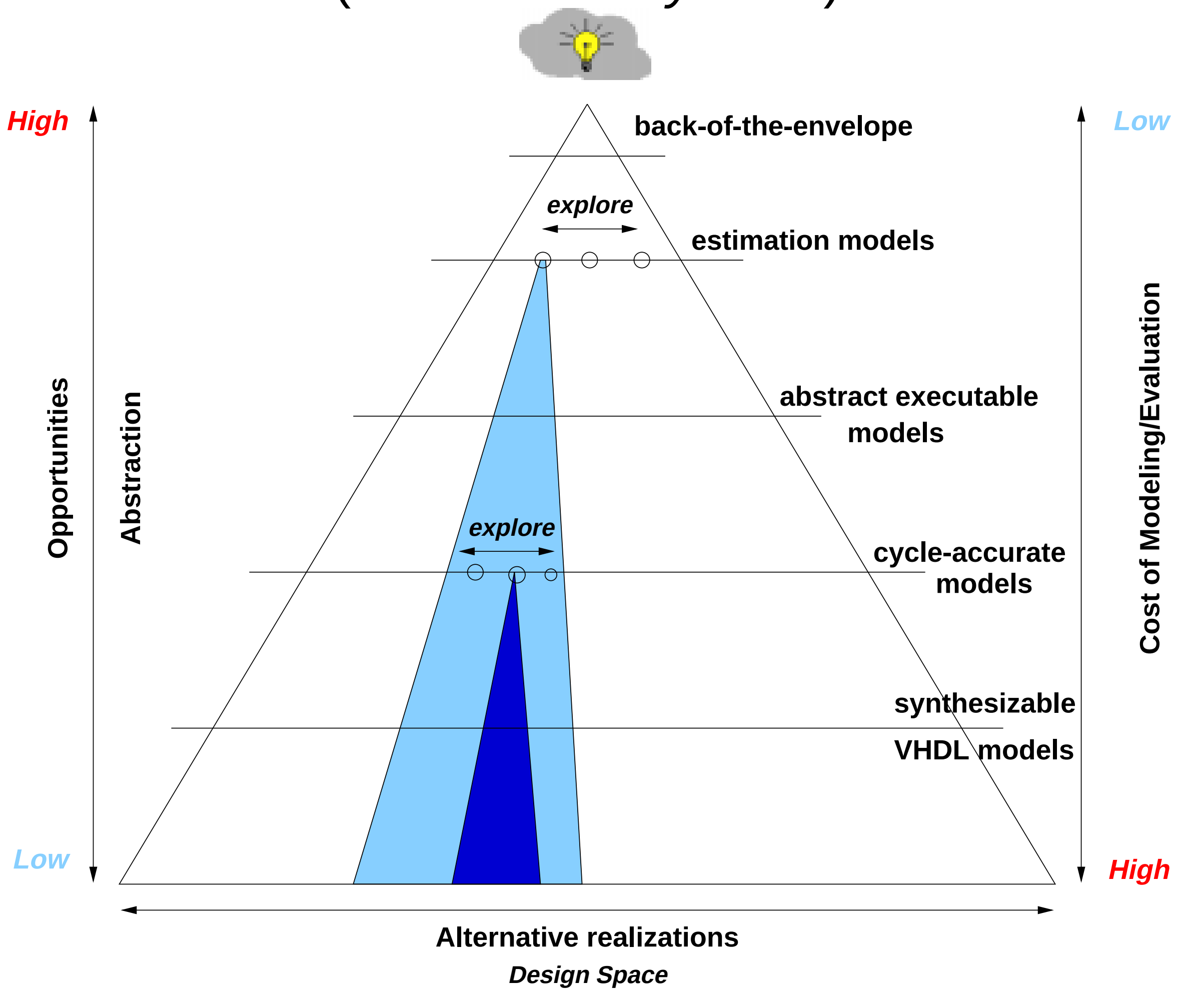
The Y-chart Approach

(*The Methodology*)



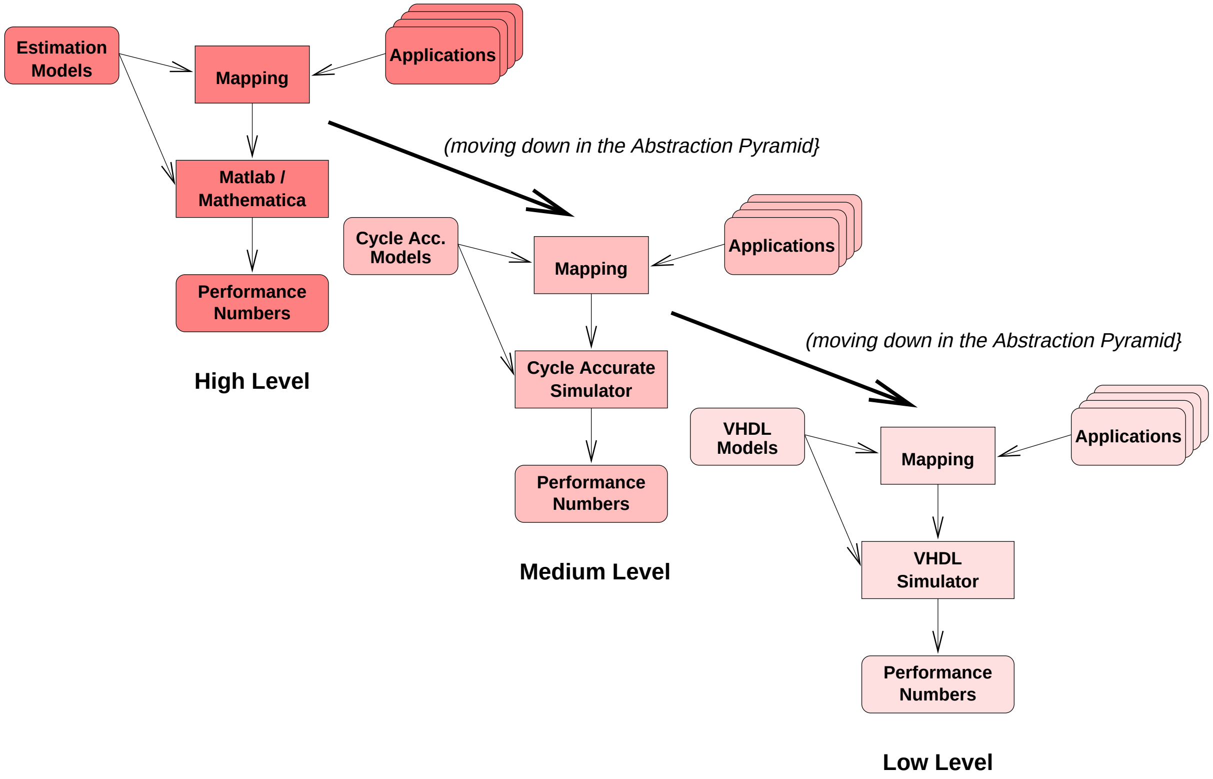
The Y-chart Approach (cont'd)

(*Abstraction Pyramid*)



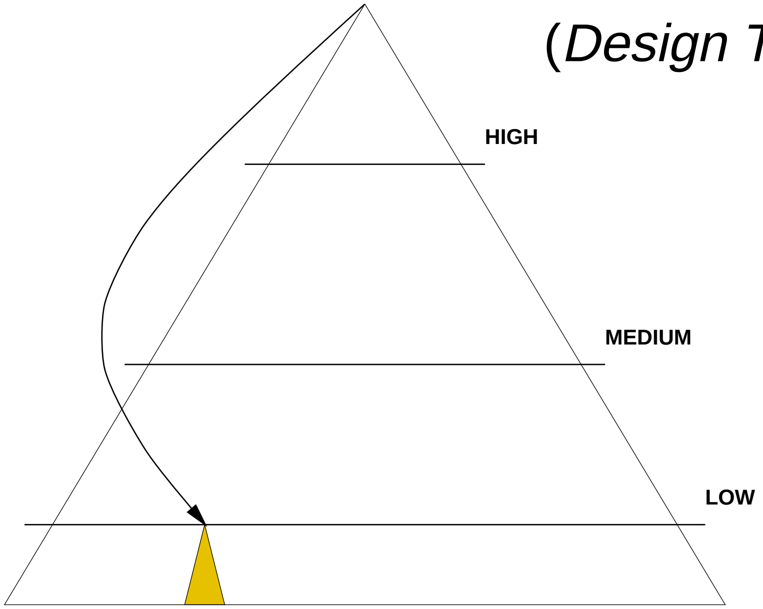
The Y-chart Approach (cont'd)

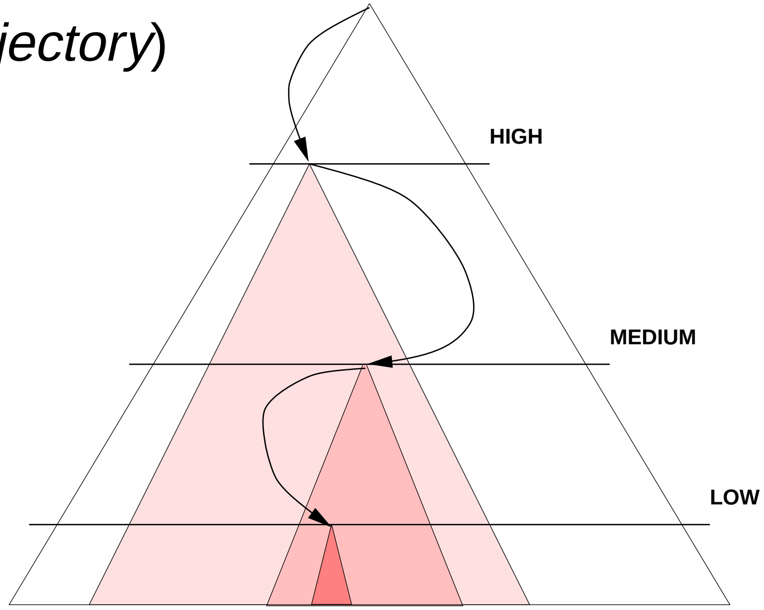
(*Stack of Y-Charts*)



The Y-chart Approach (cont’d)

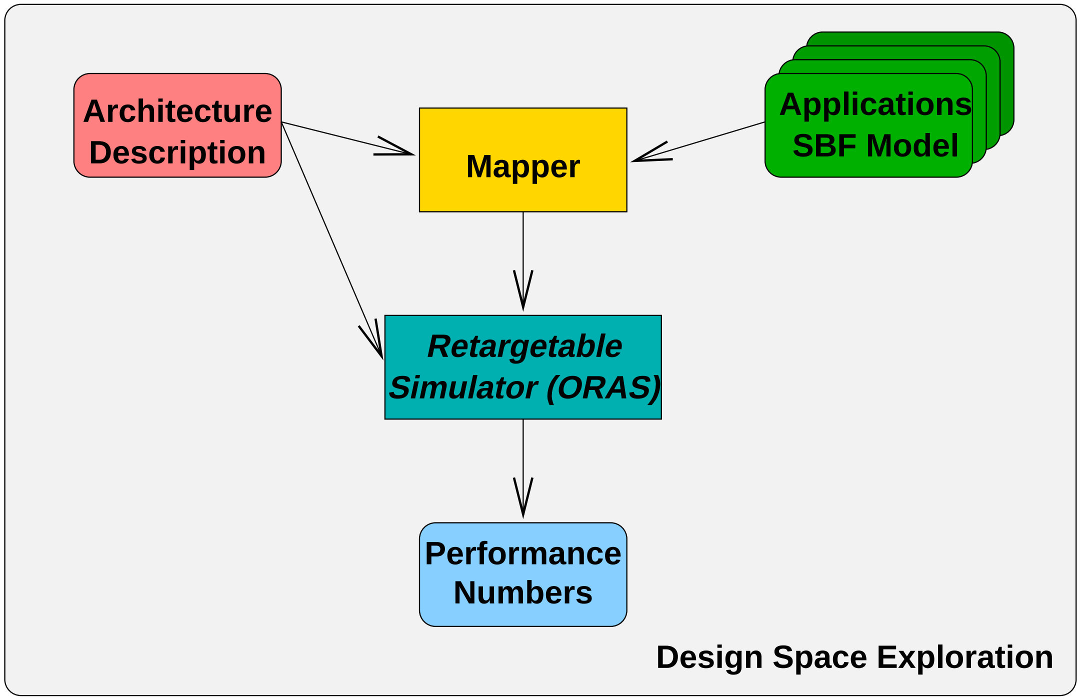
(*Design Trajectory*)





Y-Chart Environment

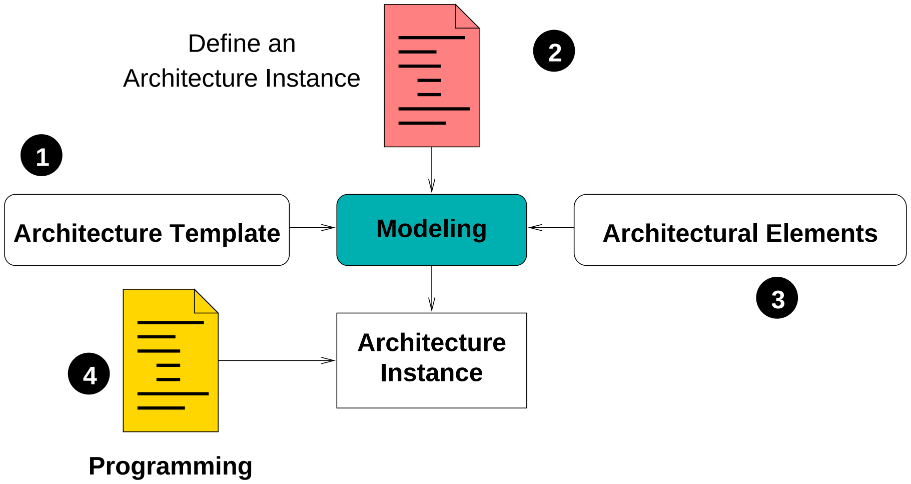
(*For Stream-based Dataflow Architectures*)



Retargetable Architecture Simulator

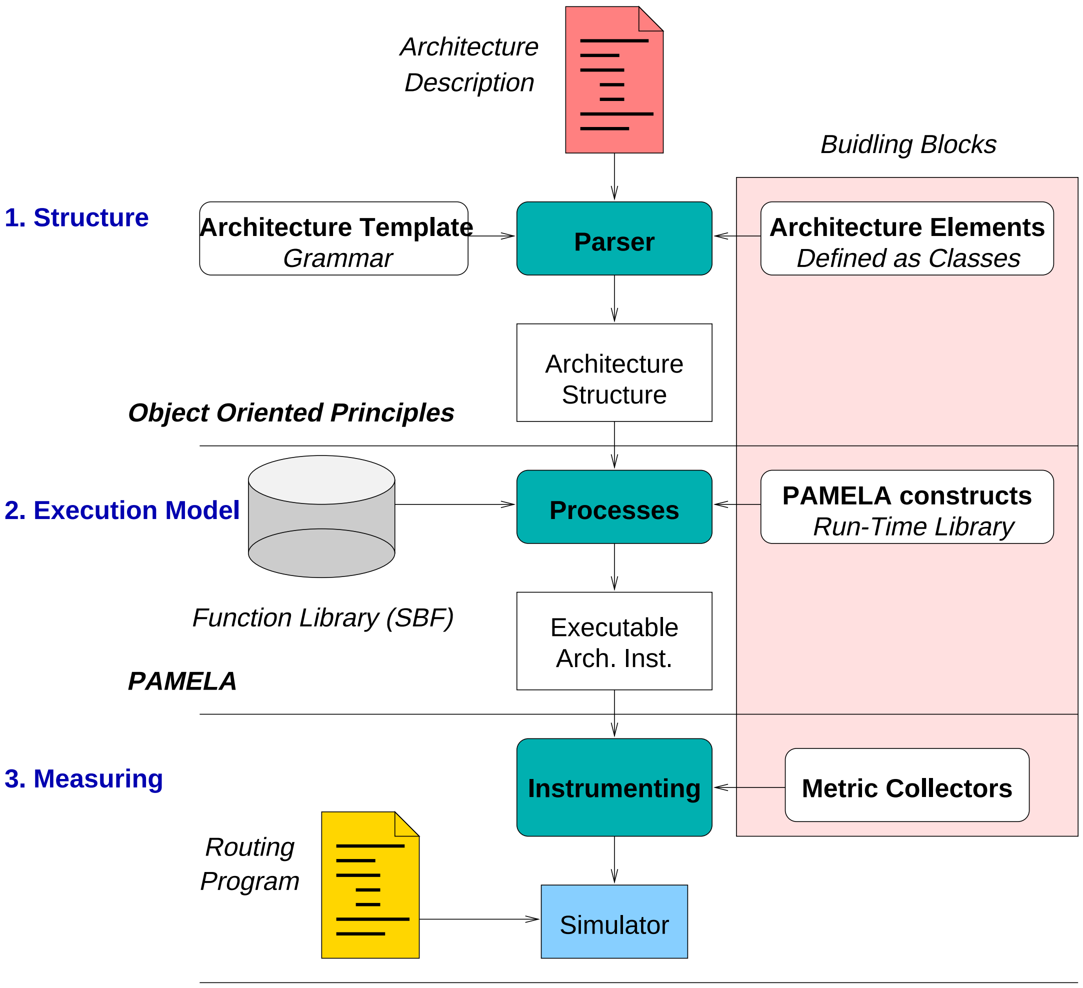
Required:

- Retargetable simulator
- Cycle accurate
- Fast Simulator
- Functional Correct



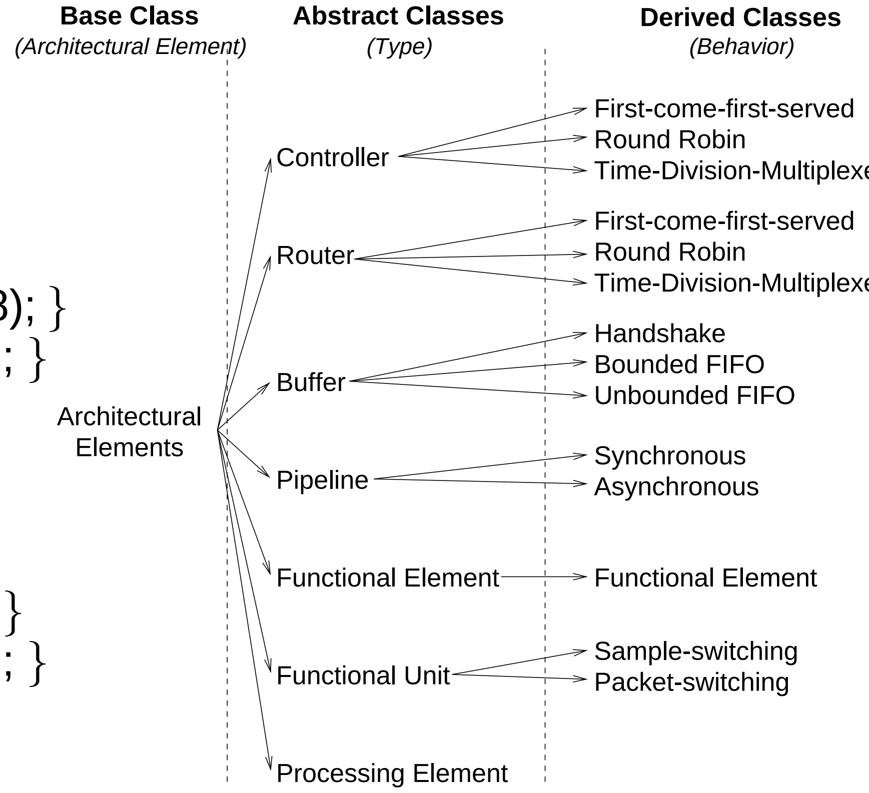
Simulator (Architecture)	Lang.	Accuracy	Sim. Speed Instr./sec	1 Video Frame
SPIM (MIPS 3000) tmsim (TriMedia) DLX (DLX)	C C VHDL	instruction clock-cycle RTL	200.000 40.000 500	10 min. 54 min. 1.2 day.
ORAS	C++	cycle	10.000	3.6 hours

Object Oriented Retargetable Architecture Simulator



Step1: Architecture Description

```
FunctionalUnit {
    Type: Packet;
    FunctionalElement FilterA(1,1) {
        Type: Synchrone;
        Function { Type:
             HighPass(throughput=1,latency=18); }
        Binding { Input ( 0->0 ); Output ( 0->0 ); }
    }
    FunctionalElement FilterB(1,1) {
        Type: Synchrone;
        Function { Type:
           LowPass(throughput=1,latency=15); }
        Binding { Input ( 0->1 ); Output ( 0->1 ); }
    }
}
```



Base Class
(Architectural Element)

Abstract Classes
(Type)

Derived Classes
(Behavior)

Architectural Elements

Controller
First-come-first-served
Round Robin
Time-Division-Multiplexe

Router
First-come-first-served
Round Robin
Time-Division-Multiplexe

Buffer
Handshake
Bounded FIFO
Unbounded FIFO

Pipeline
Synchronous
Asynchronous

Functional Element
Functional Element

Functional Unit
Sample-switching
Packet-switching

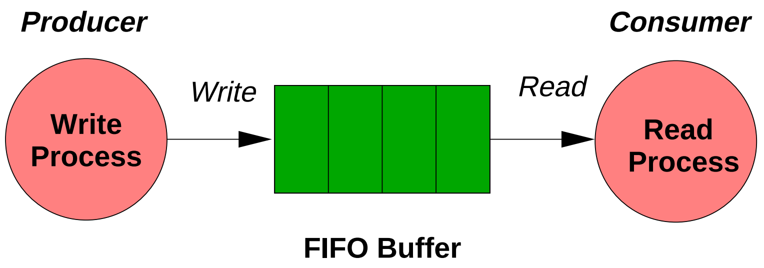
Processing Element

Step2: Execution Model

```
FIFO::read
{
        pam_P (data);
        aSample = queue[readfifo];
        readfifo = (++readfifo)%cap;
        pam_delay (1);
        pam_V (room);
}
FIFO::write( aSample )
{
        pam_P (room);
        queue[writefifo] = aSample;
        writefifo = (++writefifo)%cap;
        pam_delay (1);
        pam_V (data);
}
```

Performance Modeling PAMELA

- Mutual Exclusivity
- Condition Synchronization

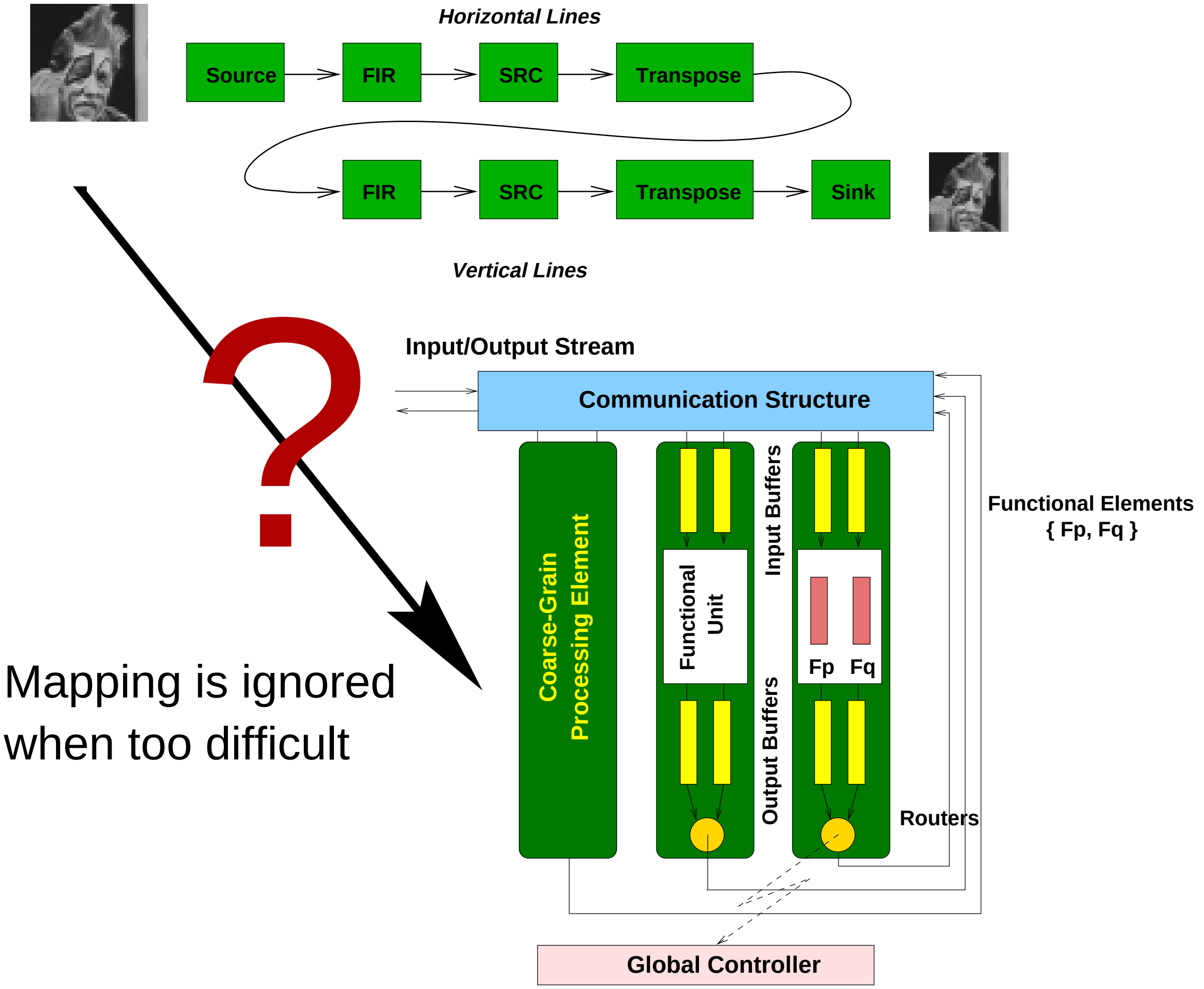


- Processes
- Synchronization Primitives
- Time

Step3: Metric Collectors

Element Type	*Performance Metric*
Comm. Structure	Utilization
Controller	Utilization
Buffer	Filling distribution
Routers	Response Time Controller
Functional Unit	Utilization, Number of Context Switches
Functional Element	Utilization, Pipeline Stalls Throughput, Number of Operations
Architecture	Number of Operations, Total execution time

Mapping

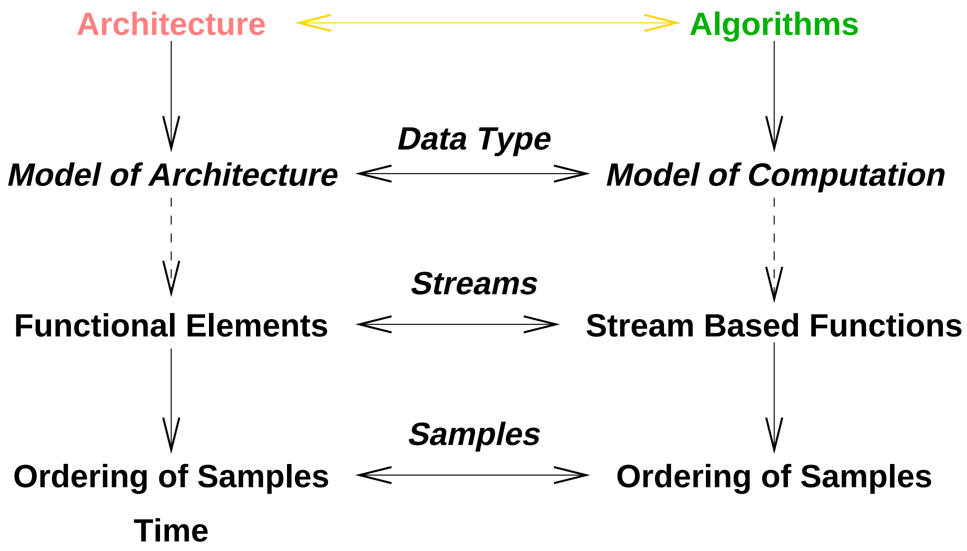


Mapping is ignored when too difficult

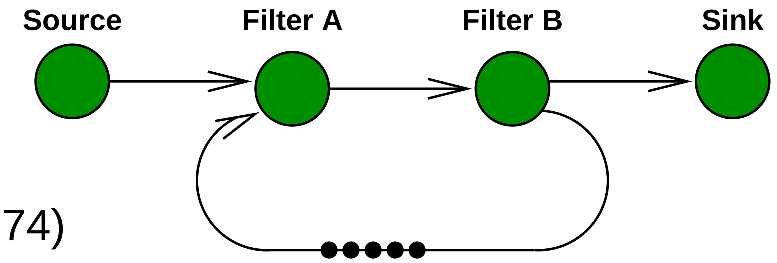
Mapping (cont'd)

Mapping Approach:

- Explicit description of both Architecture and Applications
- Description Formalisms and Data Types should correspond



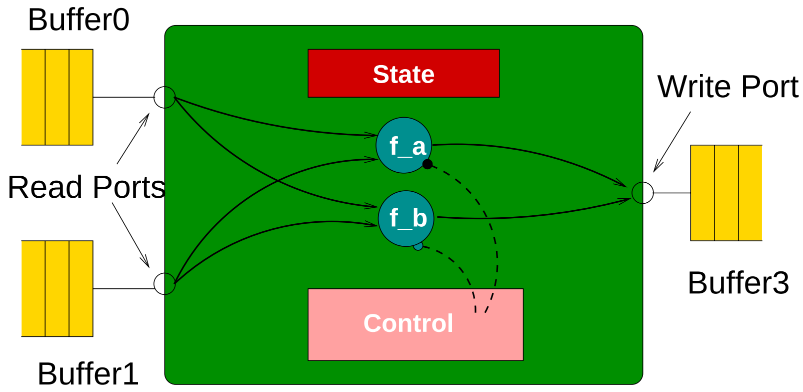
Stream-based Functions



Basis:

1. Describe a network as a Kahn Process Network (Kahn '74)

2. Structure the nodes based on the AST model of Backus (Backus '78)
   - Controller
   - State
   - Set of Functions



SBF Object

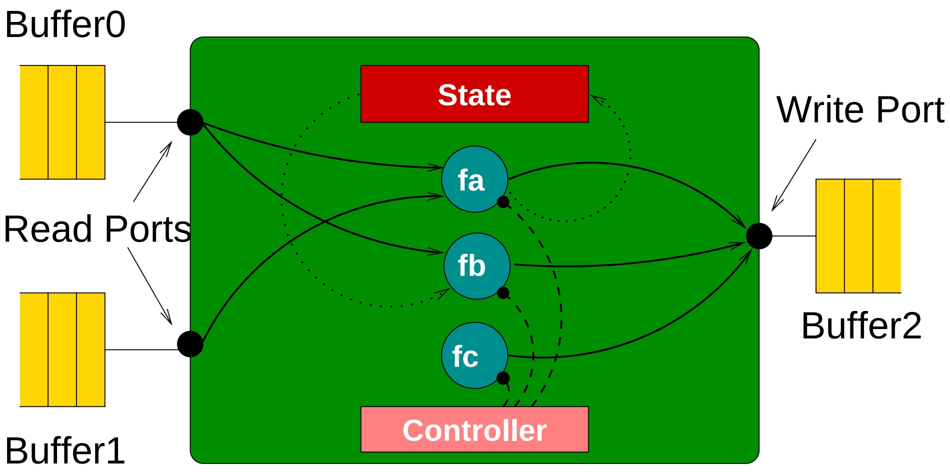
Example of an SBF Object

Binding Function

$$\mu(s) = \begin{cases} f_a, & \text{if } s = 0 \\ f_a, & \text{if } s = 1 \\ f_b, & \text{if } s = 2 \\ f_c, & \text{if } s = 3, \end{cases}$$

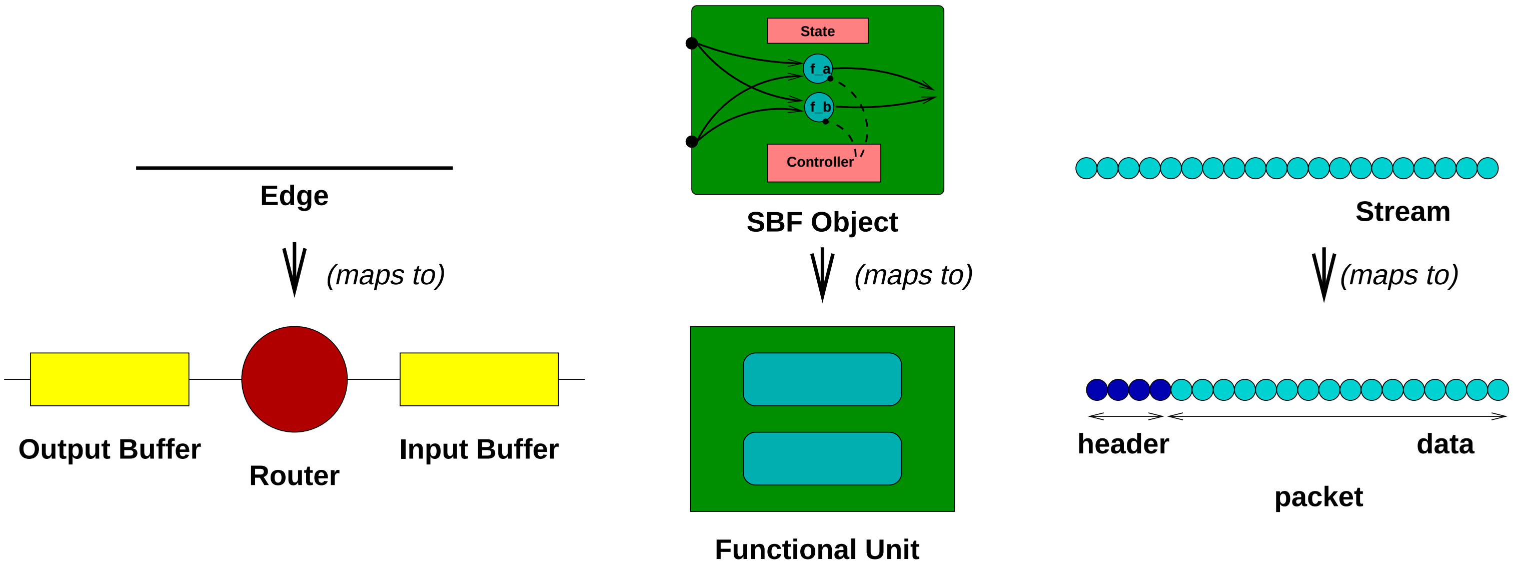
Transition Function

$$\omega(s) = s + 1 \pmod 3.$$



Current State	Function	Buffer0	Buffer1	Buffer2
s_0	f_a	R	R	W
s_1	f_a	R	R	W
s_2	f_b	R		W
s_3	f_c			W

Mapping of an SBF Object



Mapping (cont'd)

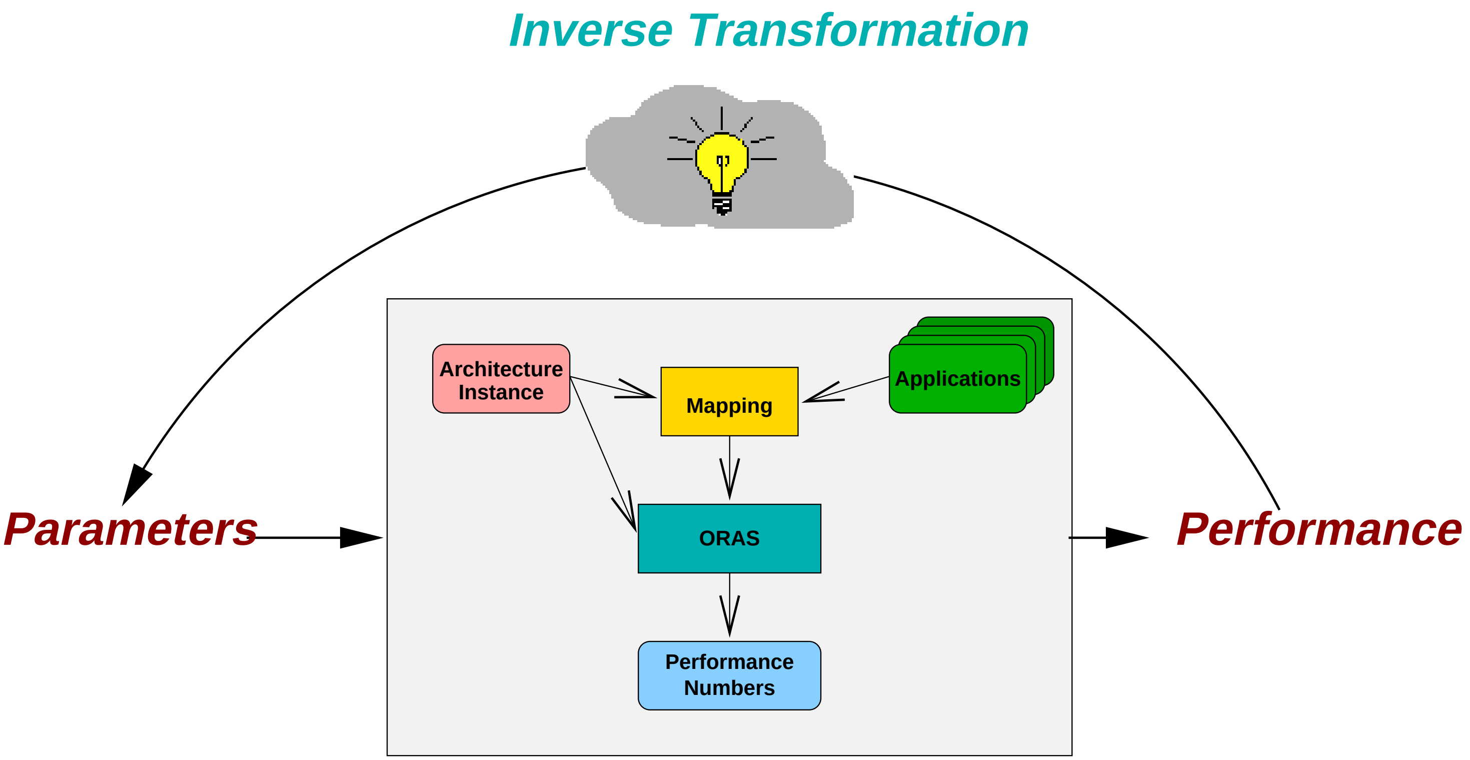
The Mapping approach results in an Application/Architecture interface

- No need to rewrite applications
- Capture the correct timing behavior of applications on arbitrary architecture instances
  - Pipeline and Throughput

Example:

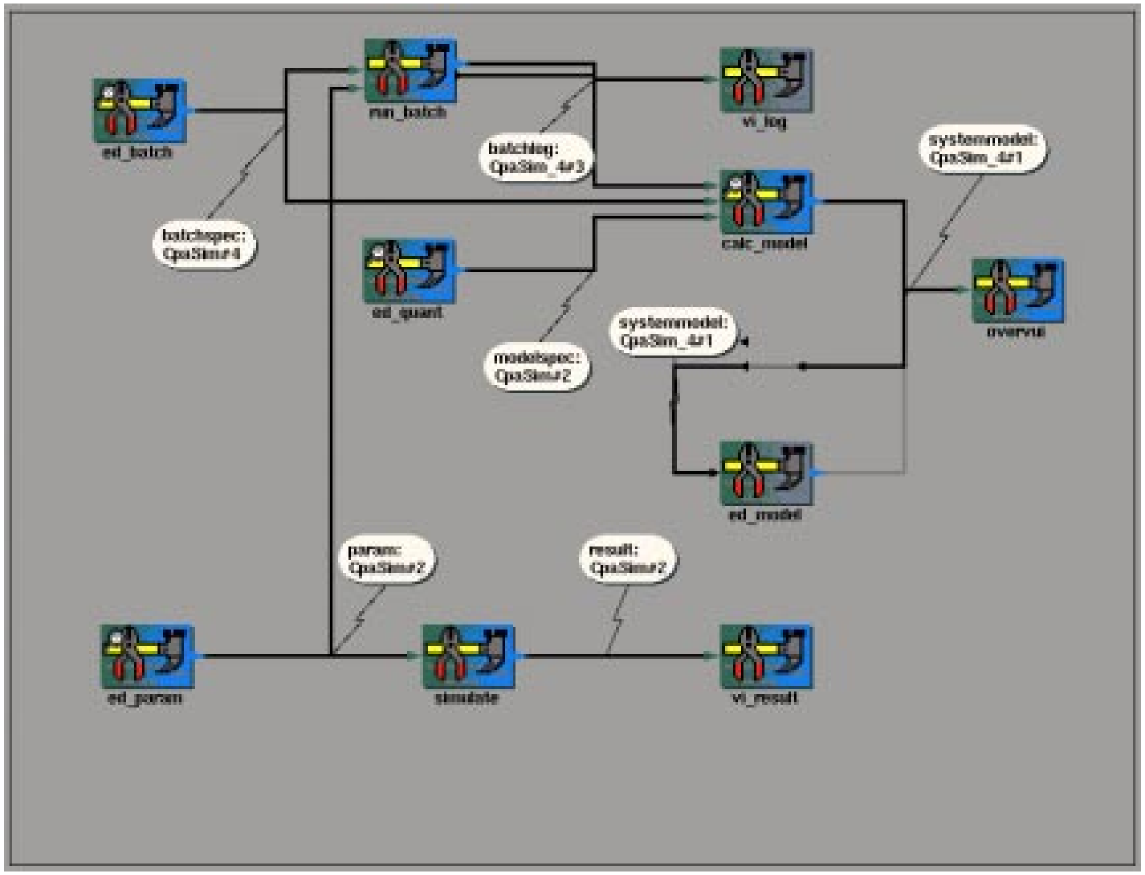
Function { Type: LowPass(throughput=1,latency=15); }

Design Space Exploration

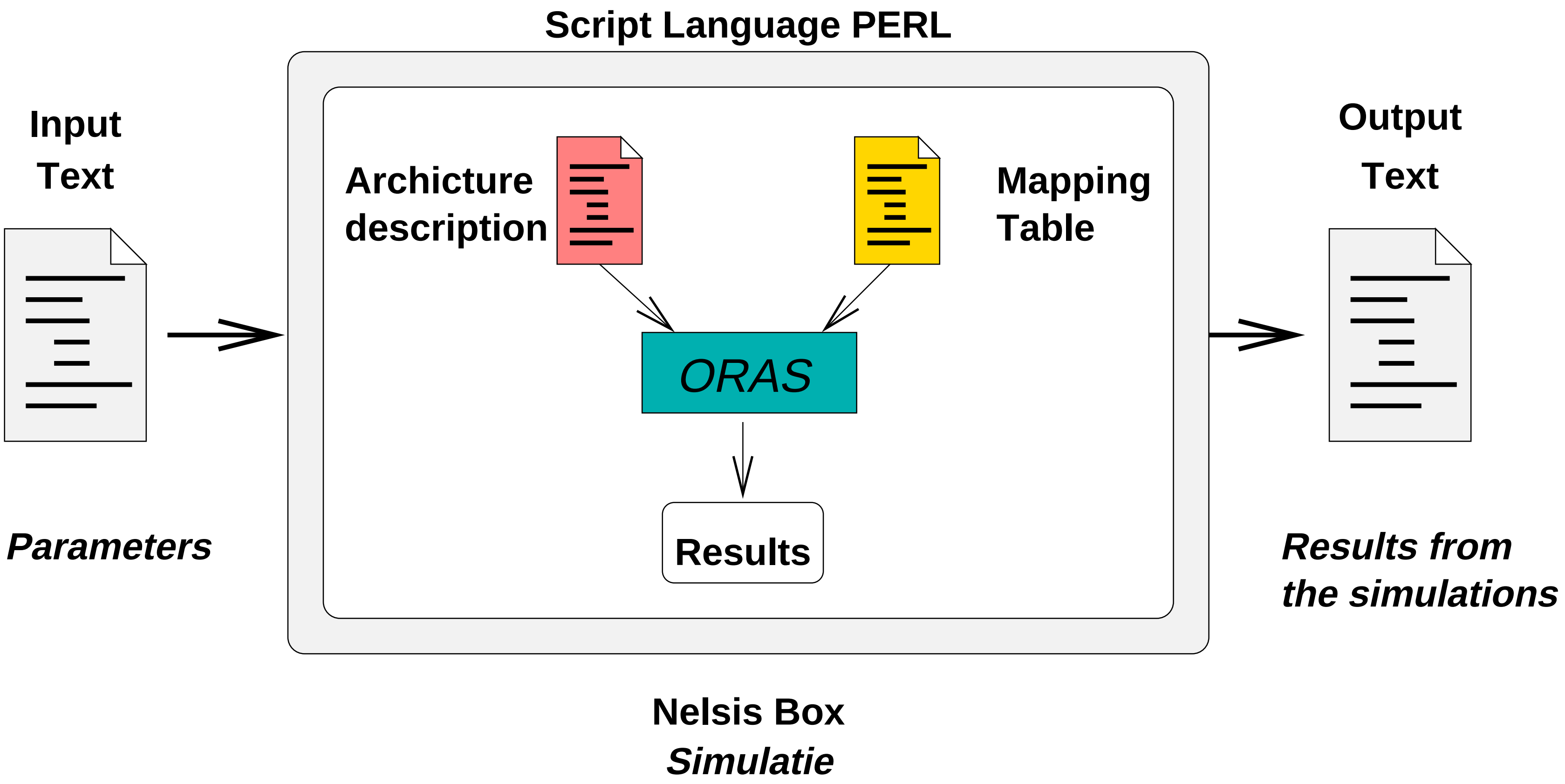


The Acquisition of Insight

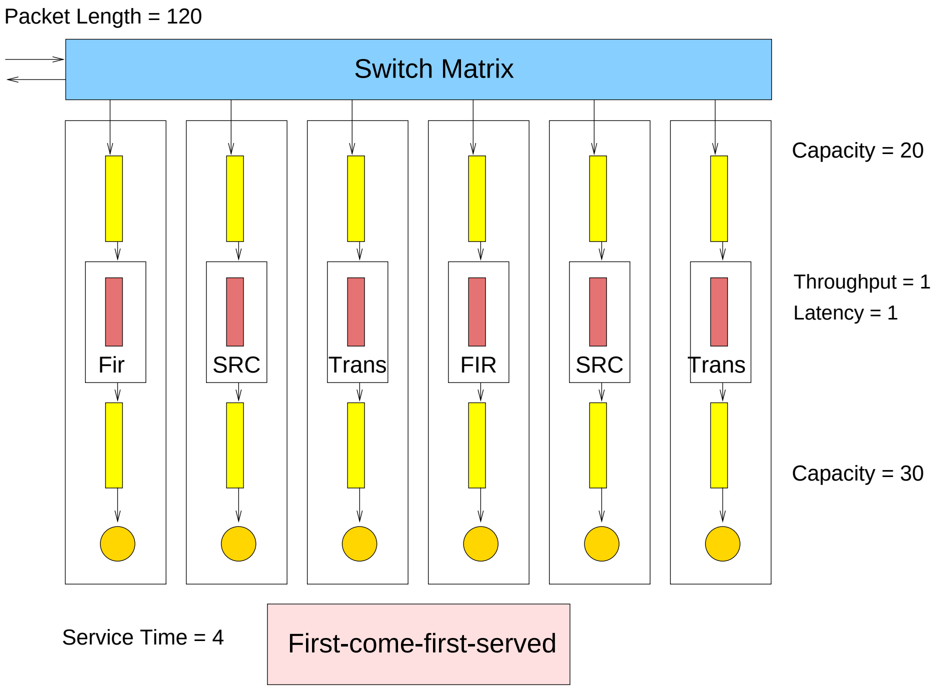
Design Space Exploration (cont’d)



Design Space Exploration (cont’d)

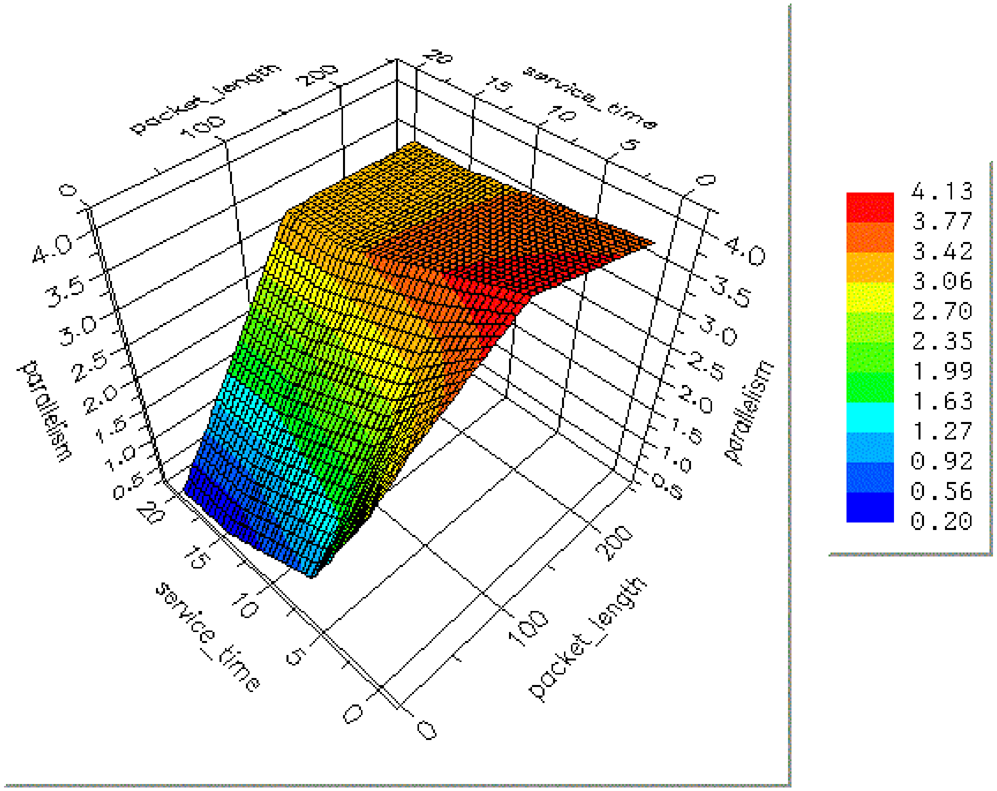


Experiment



- Packet Length: { 4 ... 256 } Samples per Packet
- Service Time: { 1 ... 20 } Cycles per Request

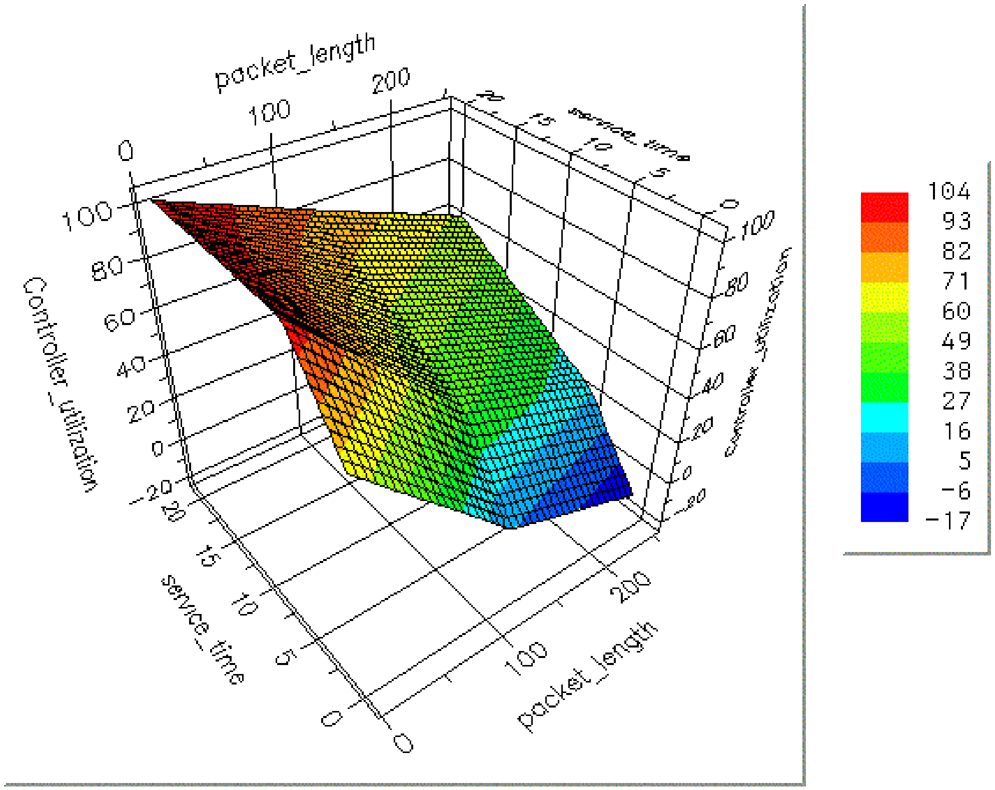
Results



Parallelism

Result of simulating 25 different Architecture Instances

Results (cont’d)



Utilization

Conclusions

Presented a Method and Tools for exploring Stream-based Dataflow Architectures.

☞Better Engineered Architectures in less Time

Method

- Y-chart approach to quantify design choices
- Y-chart environment for Stream-based Dataflow Architectures
- Systematic exploration of the design space

Tools

- Object Oriented Retargetable Architecture Simulator (ORAS)
- The SBF Model
- Design Space Exploration environment based on Nelsis

Future Work

- Generalize presented methods and tools for heterogeneous system design at different levels of abstraction (Lieverse et al.)
- Compiling Matlab applications into descriptions in terms of the SBF Model (Rypkema et al.)

For more information look at:

➥`http://cas.et.tudelft.nl/research/hse.html`