Taming Heterogeneity the Ptolemy Approach

Presented by Atisha Ribeiro

University of Antwerp

Authored by

- Johan Eker
- Jörn W. Janneck
- Edward A. Lee
- Jie Liu
- Xiaojun Liu
- Stephen Neuendorffer
- Sonia Sachs
- Yuhong Xiong
- Jozsef Ludvig

The Ptolemy Project

- Published in 2003
- Based at UC Berkeley
- Heterogeneous models
- Ptolemy II software

Outline

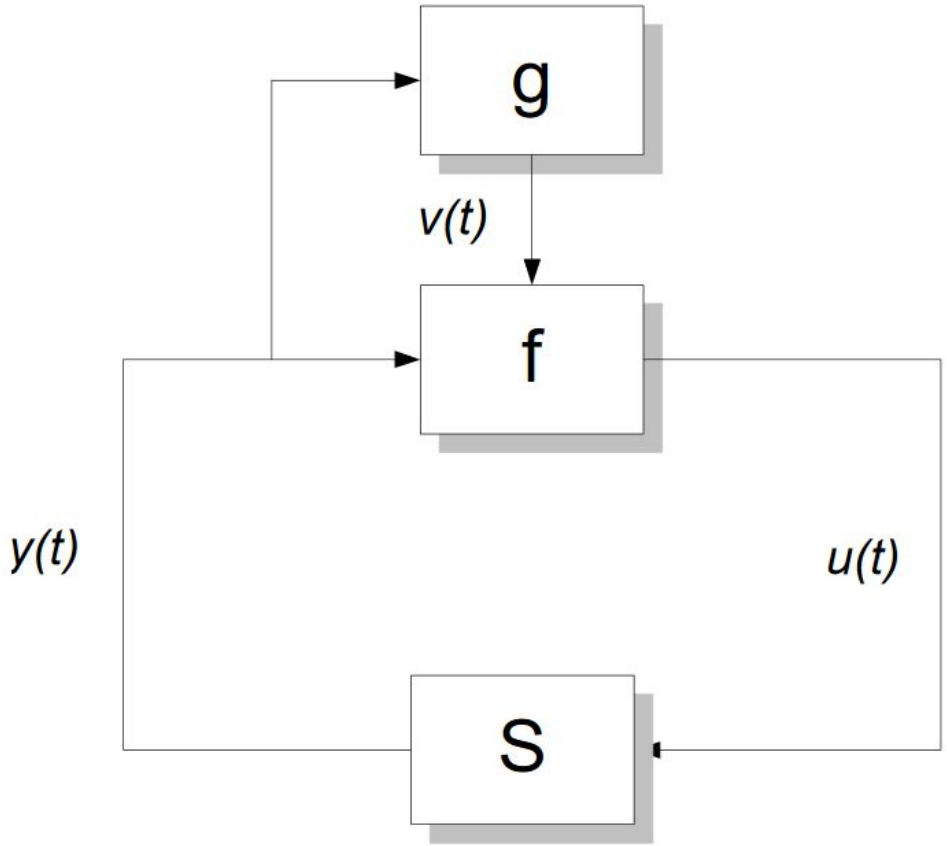
1. Heterogeneity
2. Example
3. Formalism
4. Domain Polymorphism

What is heterogeneity?

Sub-systems of different “types”

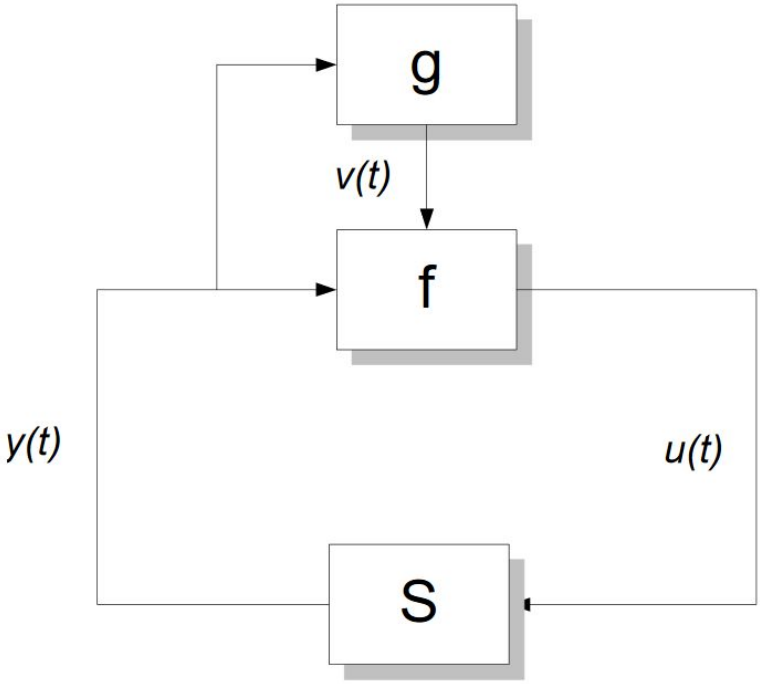
↔ Homogeneity

Homogeneous System

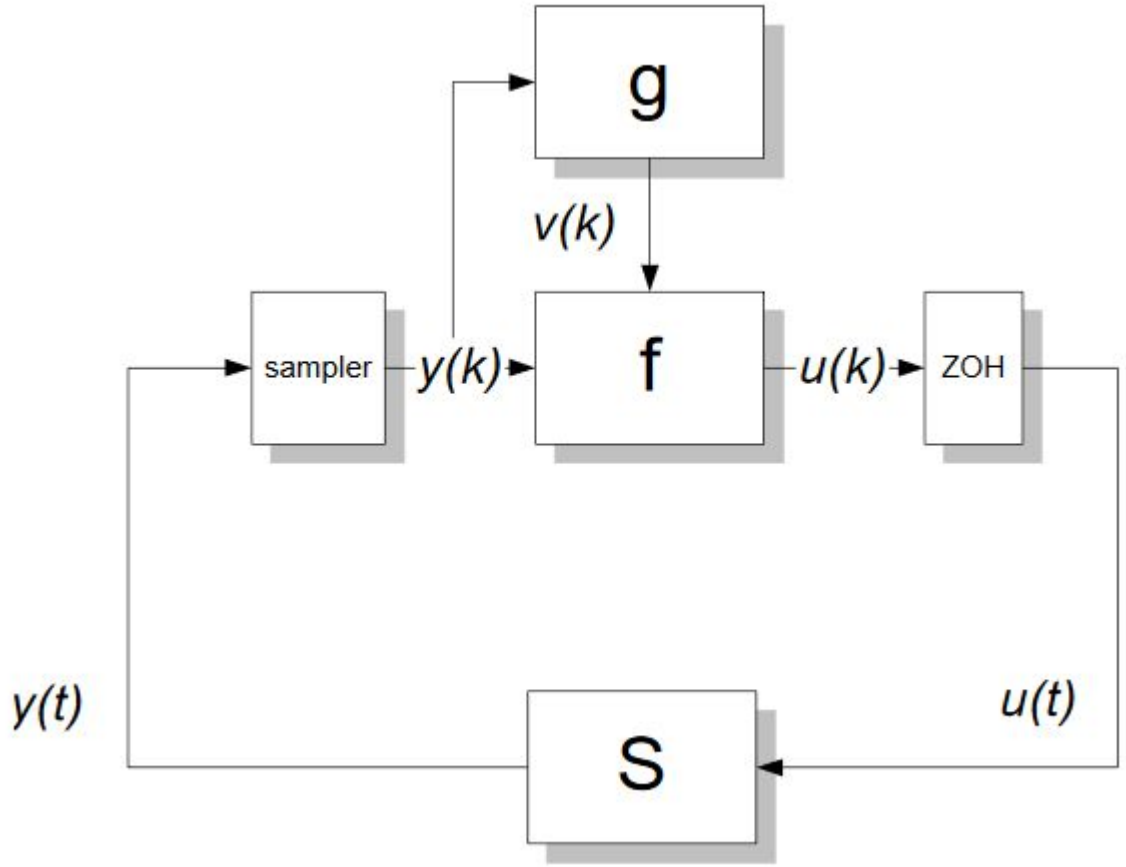


Homogeneous System

1. y(t) = S(u(t))
2. v(t) = g(y(t))
3. u(t) = f(g(y(t)), y(t))



Heterogeneous System

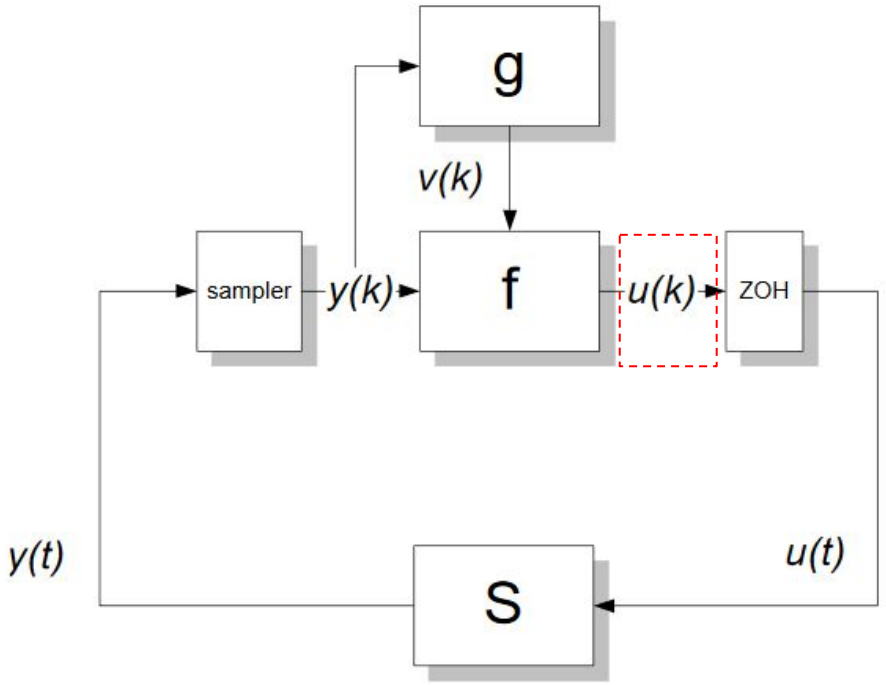


Heterogeneous System

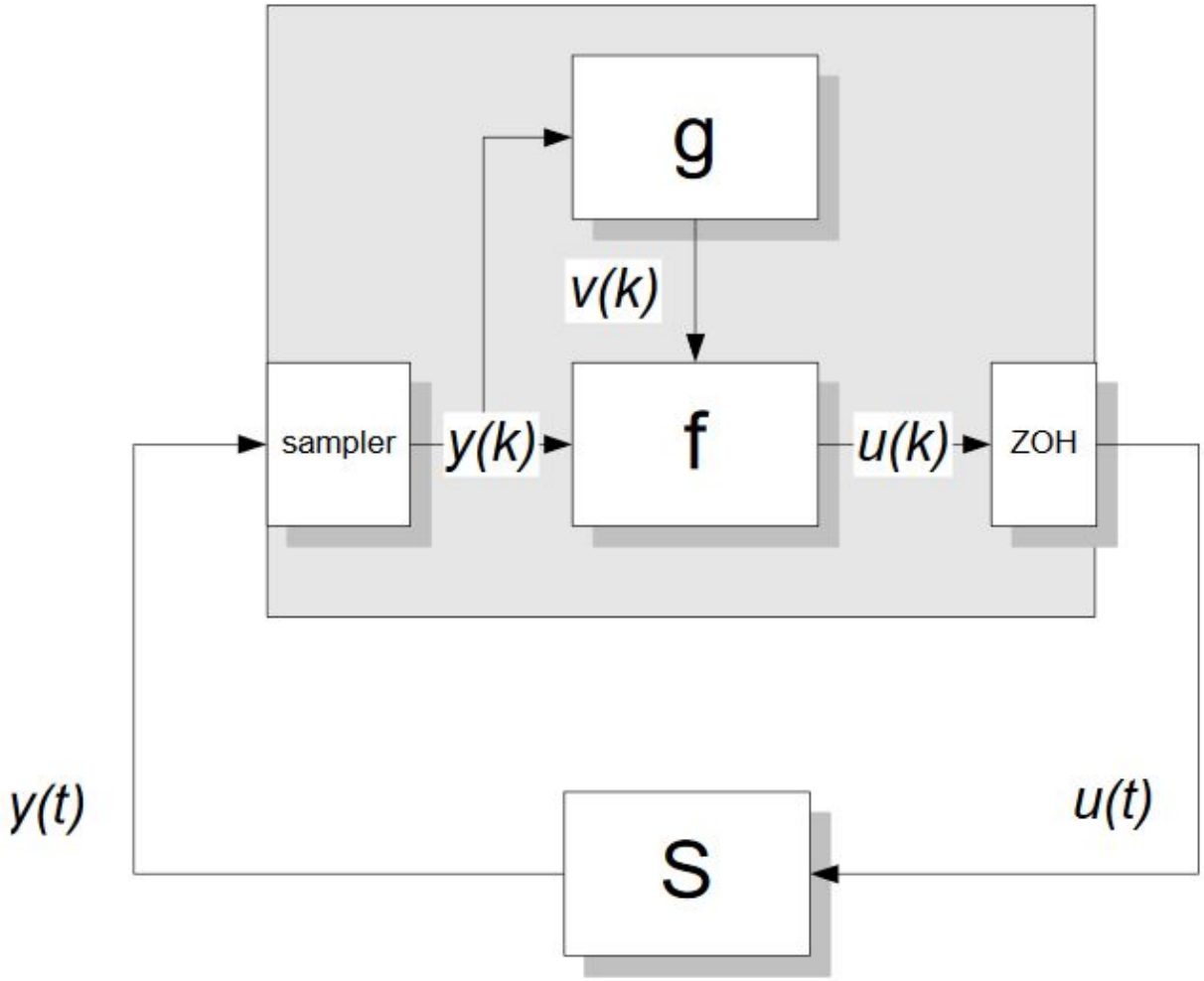
u(k) = f(v(k), y(k))

or

u(k) = f(v(k-1), y(k))



Hierarchical Heterogeneity



Formalism

Actors

- Basic building block
- Uses ports for communication

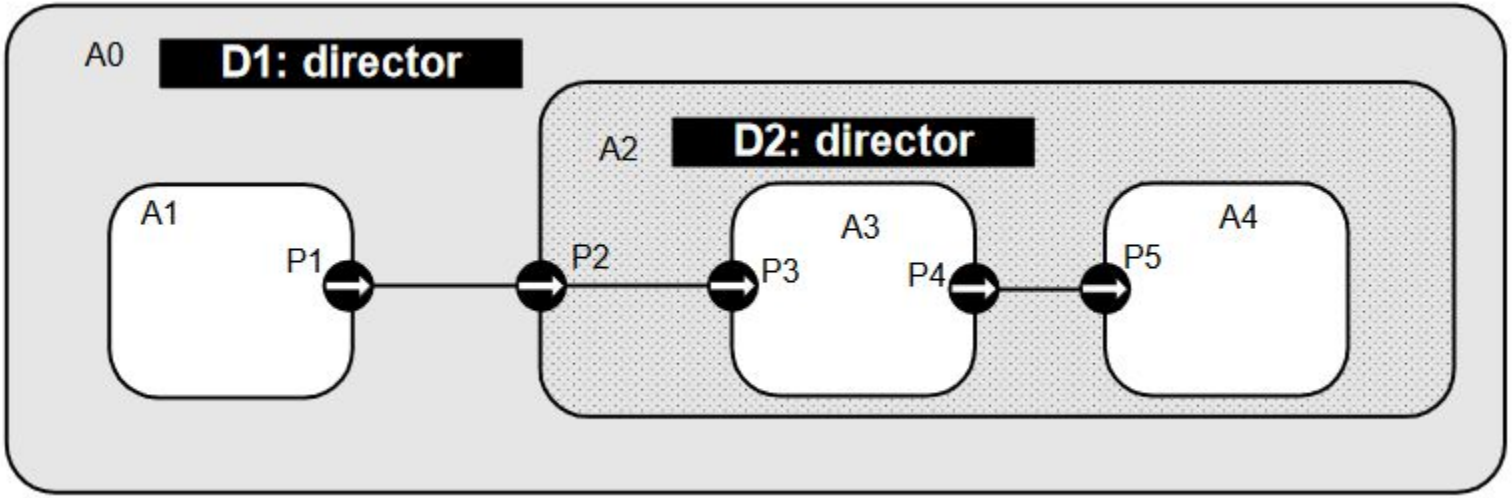
Actors

Communication & Execution is decoupled

1. Makes actors reusable
2. Easier to analyse and understand

Directors

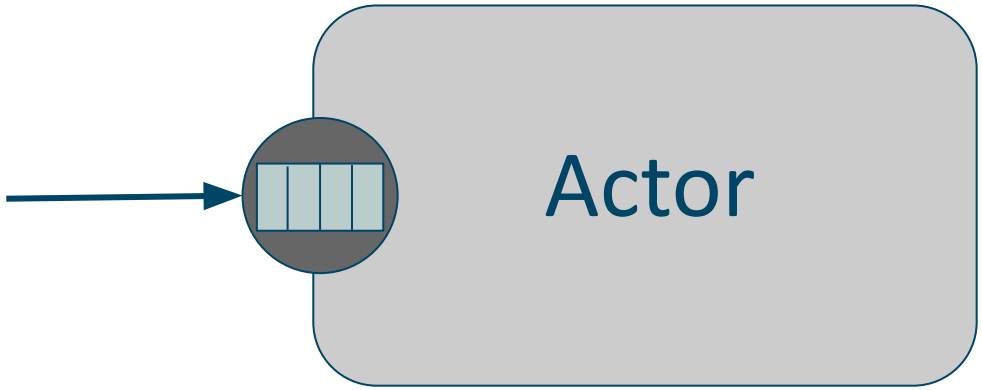
- Decides when actors are executed
- Controls the flow of information



Domains

A director acts according to a domain

Communication done with receivers



Domains

Some examples:

- Continuous Time (CT)
  - Receiver: Buffer of size 1
- Discrete Event (DE)
  - Receiver: Global event queue
- Synchronous Data Flow (SDF)
  - Receiver: Fixed length FIFO

Domain Polymorphism

Incompatibilities

Synchronous Data Flow:

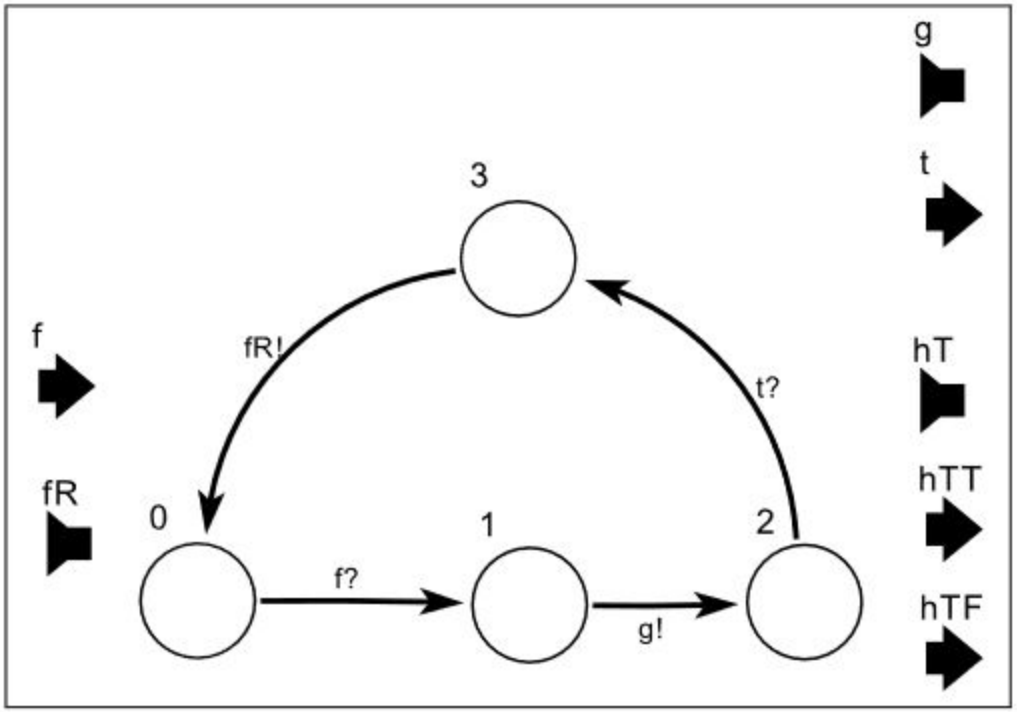
- No need to check for token

Discrete Event:

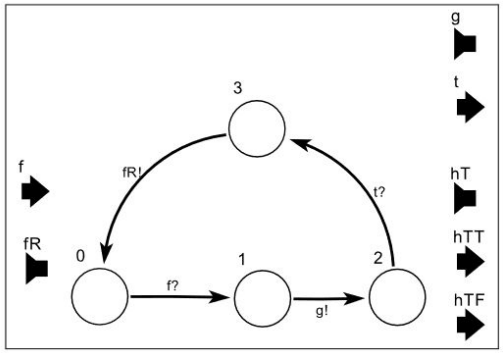
- Need to check for token

Modelling Behaviour

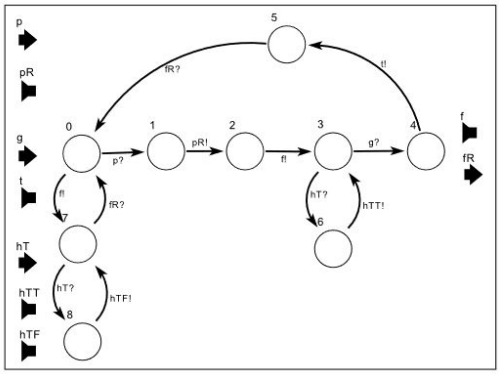
-



Modelling Behaviour



x



= empty

Conclusion