Integrated Control and Real-Time Scheduling

Anton Cervin

Department of Automatic Control
Lund Institute of Technology

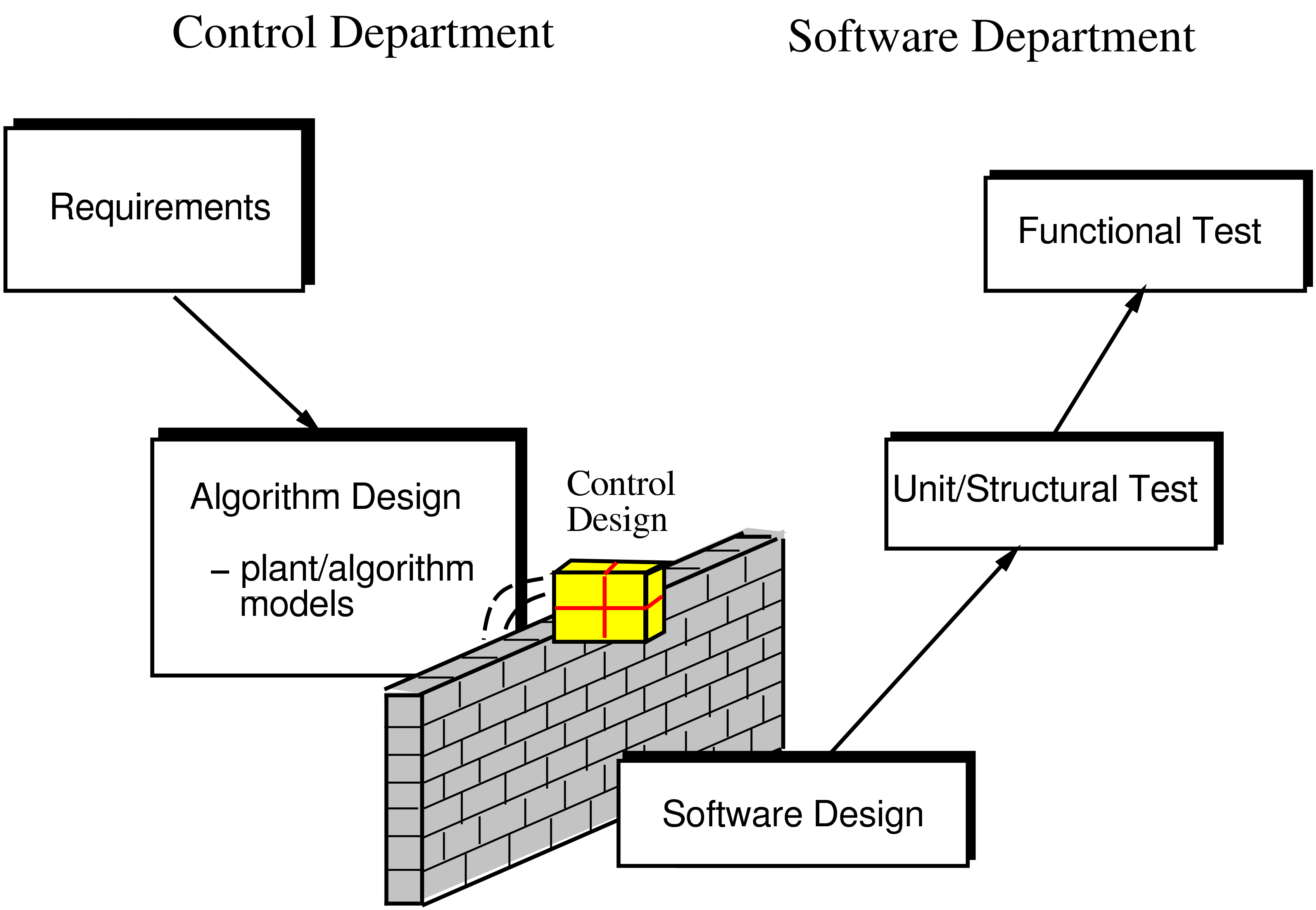
Supervisor: Karl-Erik Årzén

Research project sponsored by ARTES/SSF

Outline

1. Background
2. Overview of contributions
3. Subtask scheduling
4. Feedback scheduling
5. The Control Server
6. Analysis using Jitterbug
7. Simulation using TrueTime
8. Summary

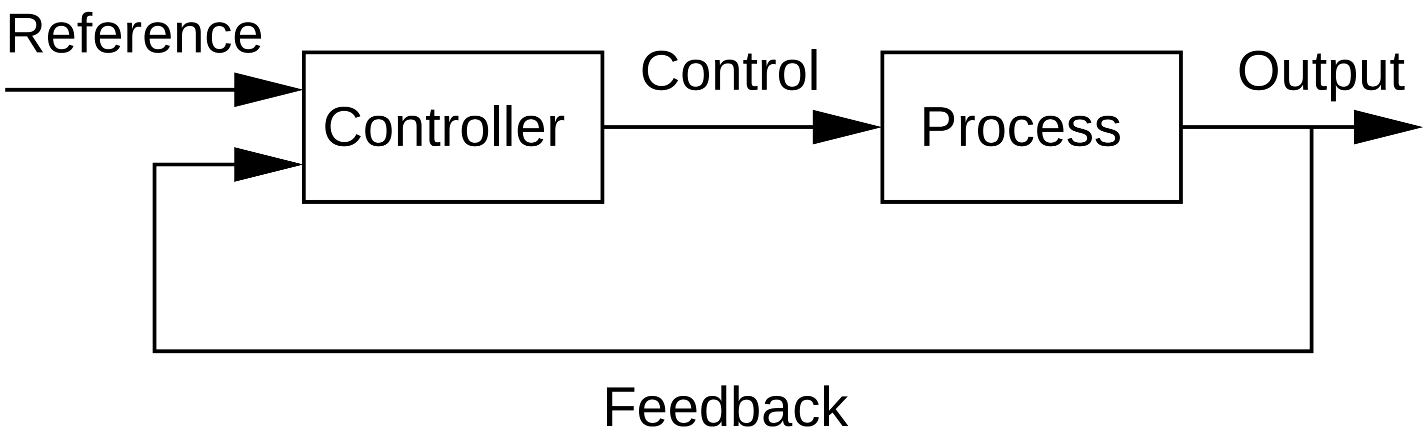
Control System Development Today



Problems

- The control engineer does not know what happens in the implementation
- The software engineer does not understand the timing requirements of the controller
- Control theory and real-time scheduling theory have evolved as separate subjects during the past 30 years

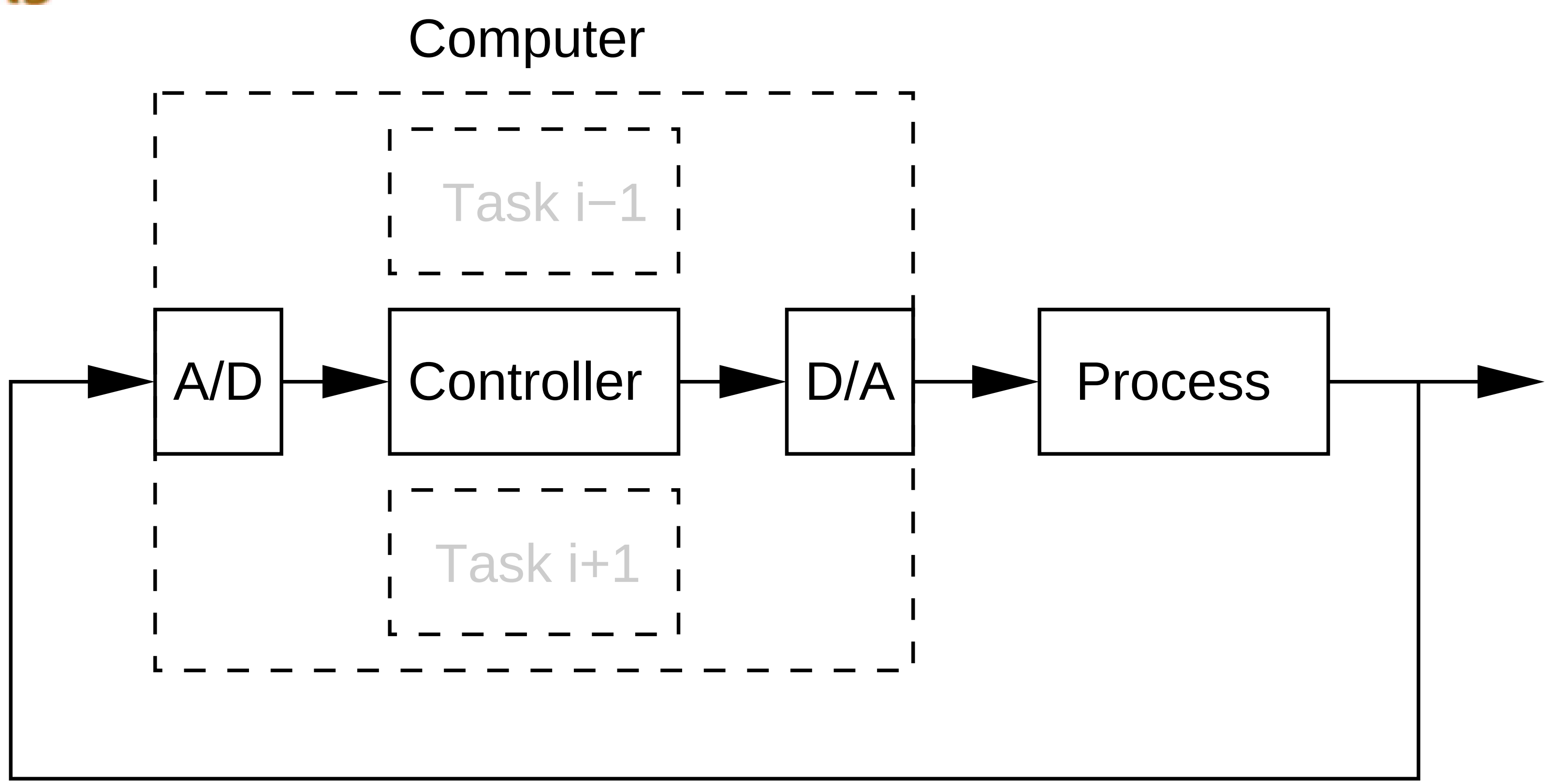
A Classical Control System



Process: Continuous dynamics

Controller: Continuous dynamics

A Computer-Controlled System



Controller:

- Discrete dynamics (control-theoretical view)
- Piece of code executing in an operating system, together with other tasks (computer science view)

Embedded Control Systems

Many controllers are embedded in mass-market products.

Characteristics:

- Cheap, slow CPUs
- Limited memory
- Limited network bandwidth

Problems:

- CPU and network are shared resources which must be scheduled
- Delay and jitter in the computer system degrade the control performance

2. Contributions of the Thesis

More detailed controller scheduling analysis:

- Subtask scheduling of Calculate and Update
- Delay reduction gives better control performance

Introduction of feedback in the computing system:

- Cope with varying workload using feedback
- Control the CPU utilization using period rescaling
- Simulation case studies

Contributions, Cont'd

A novel computational model:

- The Control Server creates the abstraction of a *real-time control component* with predictable performance
- Control components may be composed into more complex components
- Implemented in the STORK public domain RTOS

New analysis tools:

- Understanding of what happens when a controller is implemented and scheduled as a real-time task
- Jitterbug – performance analysis with varying delays
- TrueTime – co-simulation of real-time control systems

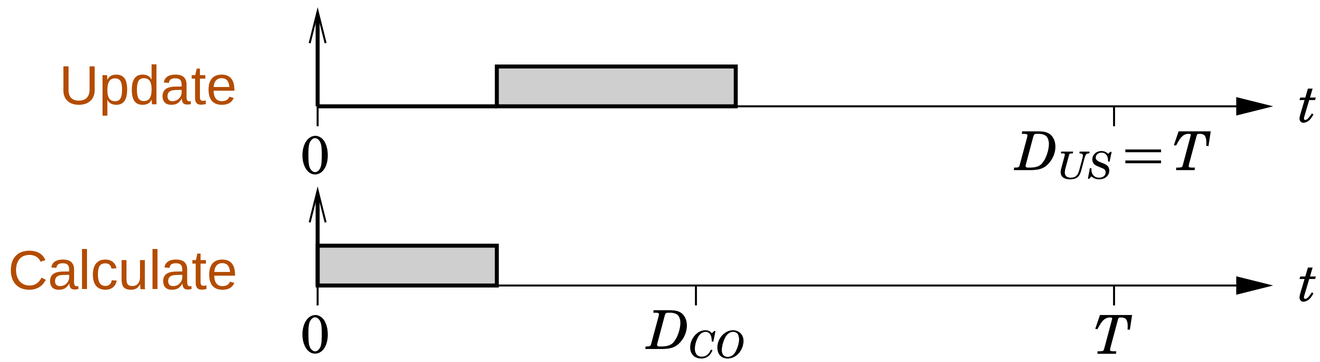
3. Subtask Scheduling

Typical implementation of a control task:

```
LOOP
  Read input;
  Calculate output;
  Write output;
  Update state;
  Wait until next period;
END
```

Basic idea: Schedule Calculate and Update as separate tasks to reduce the input-output latency.

Analysis



- The deadline for Update equals the period
- The deadline for Calculate should be minimized
- Analysis under FP and EDF scheduling given

Simple Implementation

The analysis results in different priorities for Calculate and Update:

```
SetPriority(P_CO);
LOOP
  Read input;
  Calculate output;
  Write output;
  SetPriority(P_US);  // lower the priority
  Update state;
  SetPriority(P_CO);  // raise the priority
  Wait until next period;
END
```

4. Feedback Scheduling

Idea: Perform the scheduling design on-line to cope with varying or unknown workloads

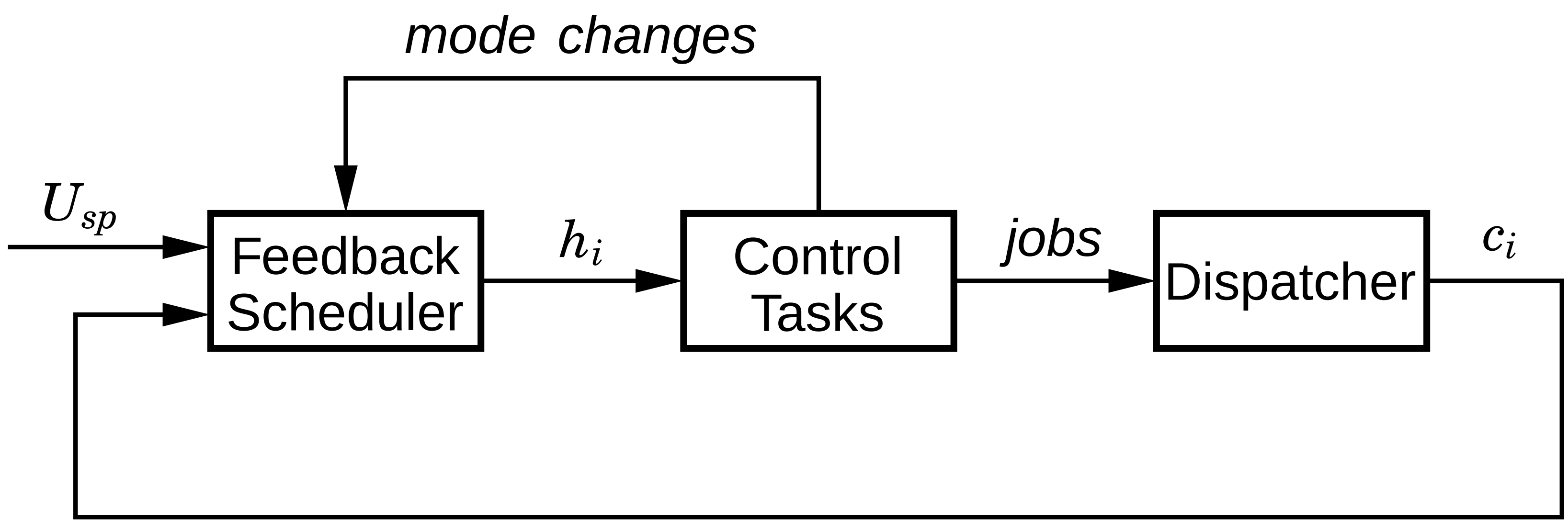
Control examples:

- Hybrid controllers
- Model-predictive controllers

Two problems:

- Control the CPU utilization
- Distribute the resources to optimize QoS

A Feedback Scheduling Architecture



Control system analogy:

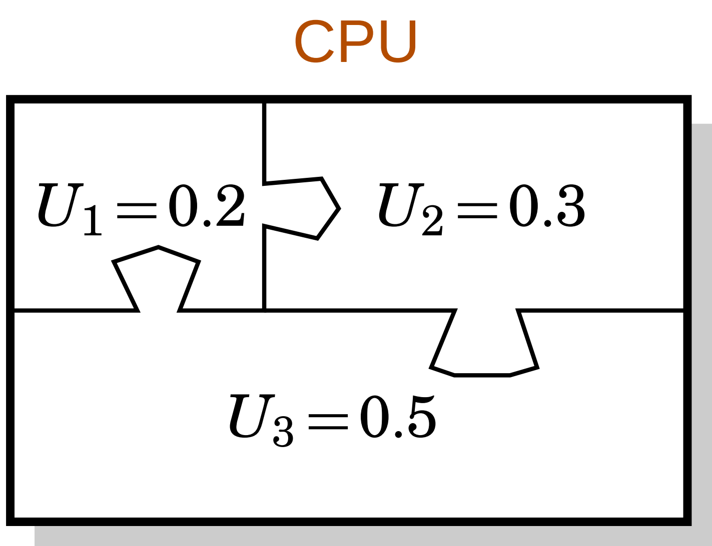
- Setpoint: desired CPU utilization
- Measurement signal: execution time of control tasks
- Control signal: sampling period of control tasks
- Feedforward from controller mode changes

5. The Control Server

Combination of two ideas:

- Reserve a given fraction of the CPU to each control task
- Let the kernel handle all I/O ($\Rightarrow$ no jitter)

CPU reservation can be performed by Constant Bandwidth Servers (CBSs) [Abeni and Buttazzo, 1998]:



Features

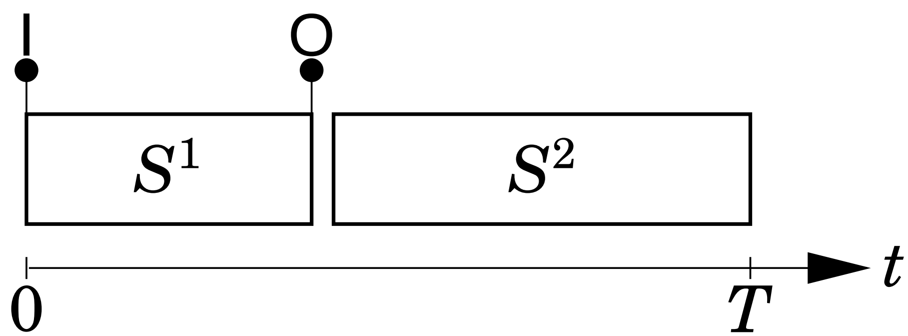
The Control Server provides

- isolation between unrelated tasks
- minimal jitter
- short and predictable input-output latency
- a simple interface between control design and real-time design – the task utilization factor U
- a possibility to combine several tasks (components) into a new task (component) with predictable control and real-time behavior

The Model

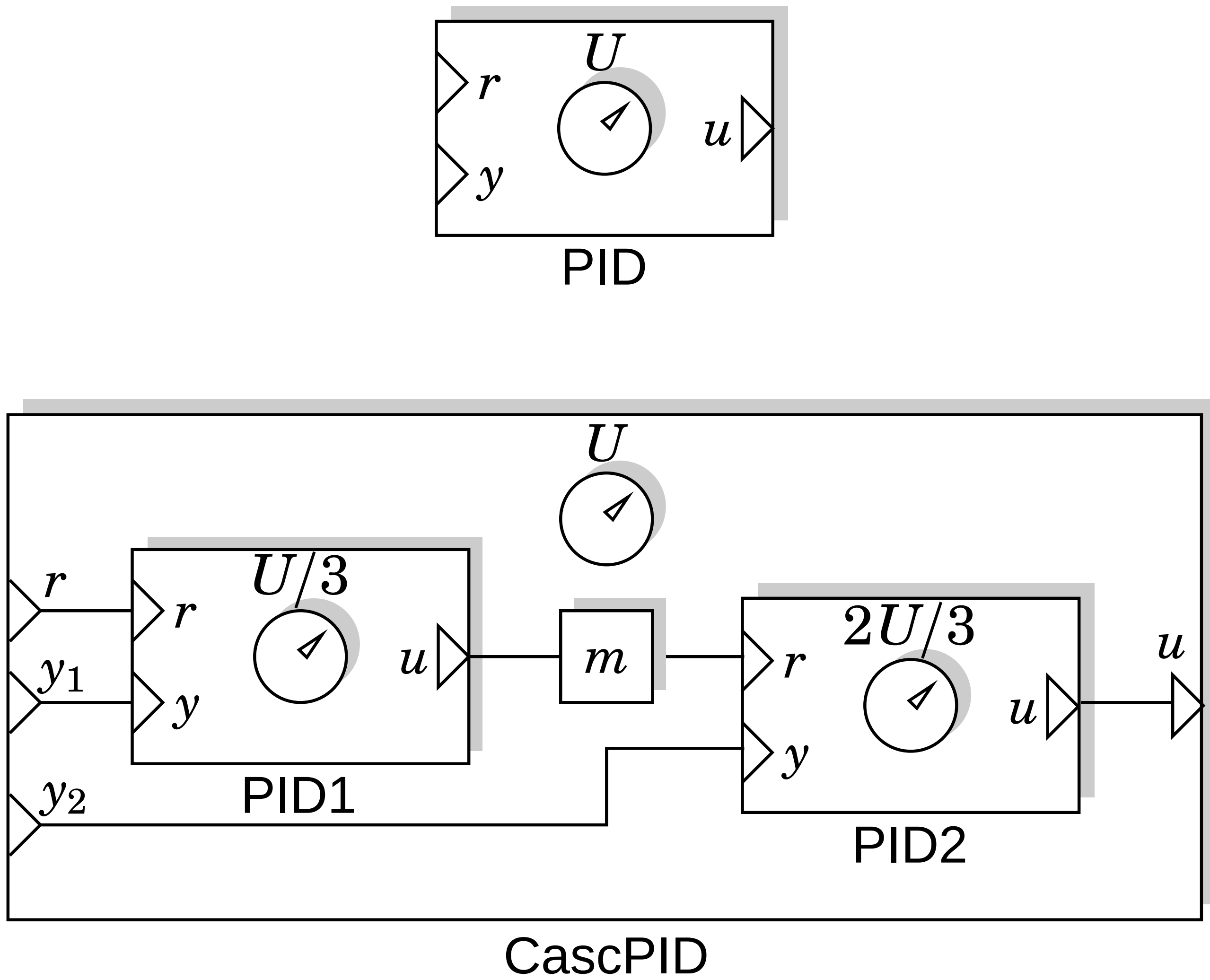
A Control Server task is described by

- a CPU utilization factor, U
- a period, T
- a number of code segments, S^i



- Static scheduling of inputs and outputs
- Dynamic scheduling of computations in-between

Real-Time Control Components



6. Analysis Using Jitterbug

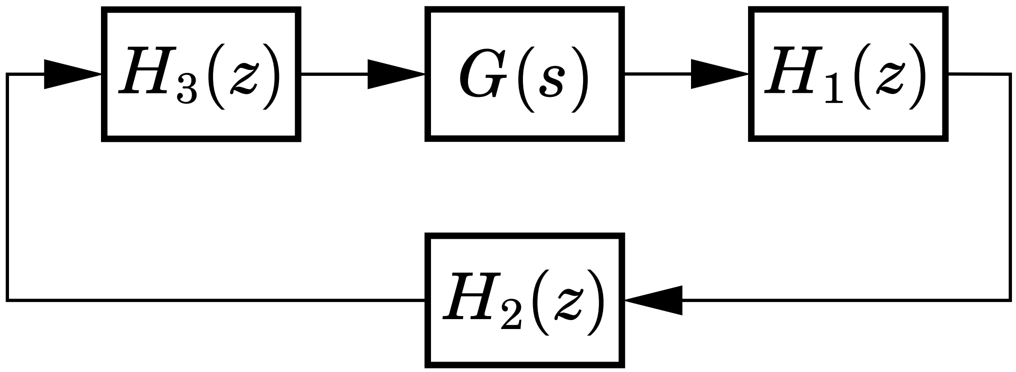
- MATLAB-based tool
- Analysis of mixed continuous/discrete-time linear systems with jitter
- Timing model with random delays describes the execution of the discrete systems
  - models scheduling/network delays, lost samples, etc.
- The systems are driven by white noise
- A quadratic cost function is computed, e.g.,

$$J = \lim_{T \to \infty} \frac{1}{T} \int_0^T x^T(t) Q x(t) \, dt$$

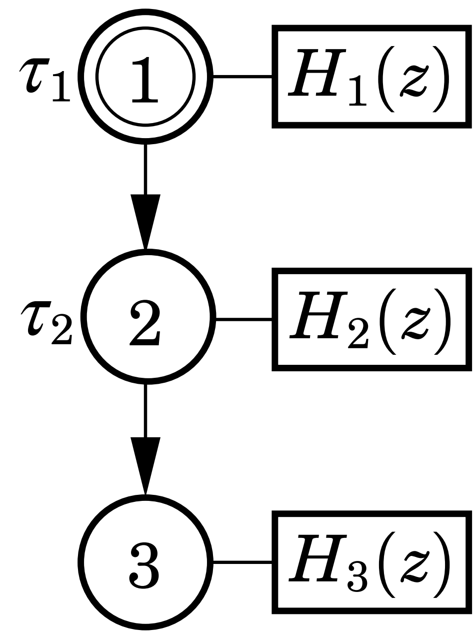
Example of a Jitterbug Model

Distributed control system:

Signal model:



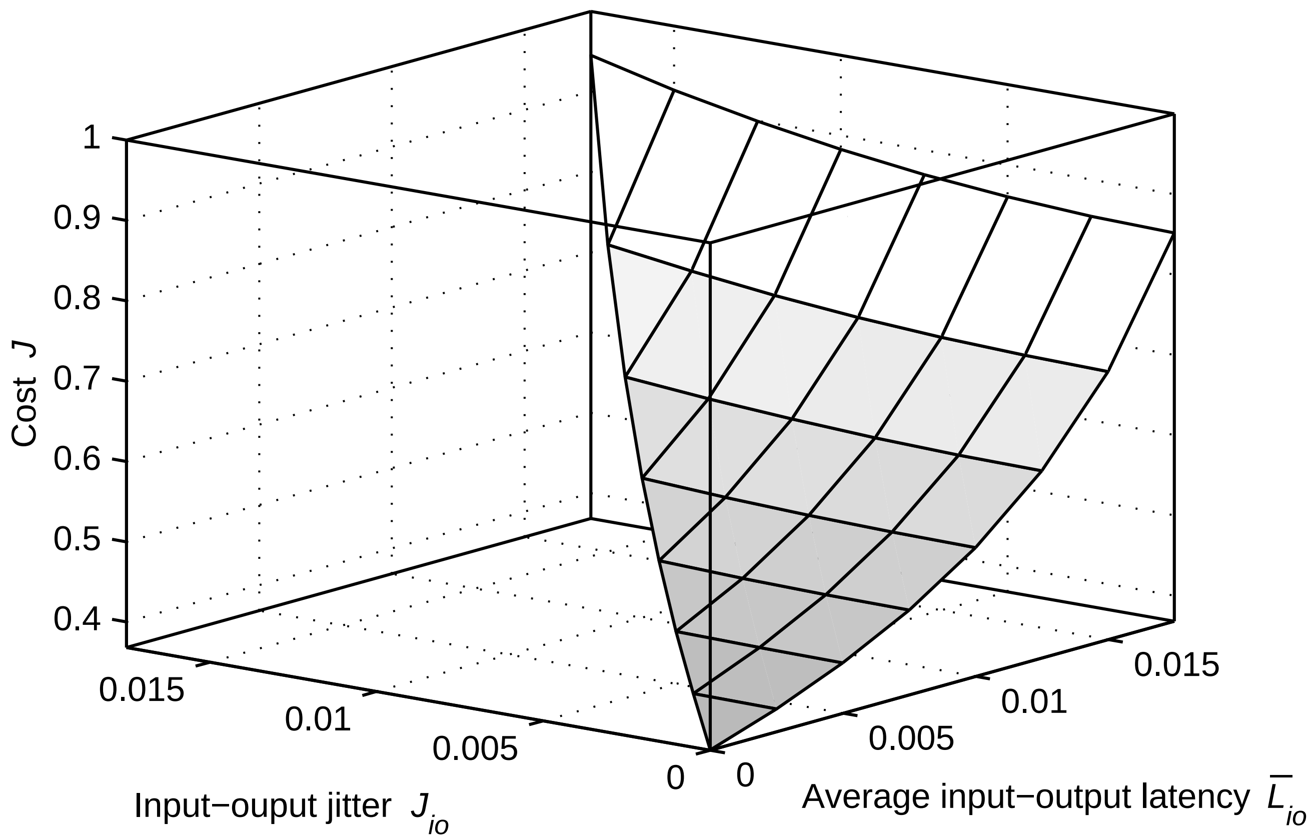
Execution model:



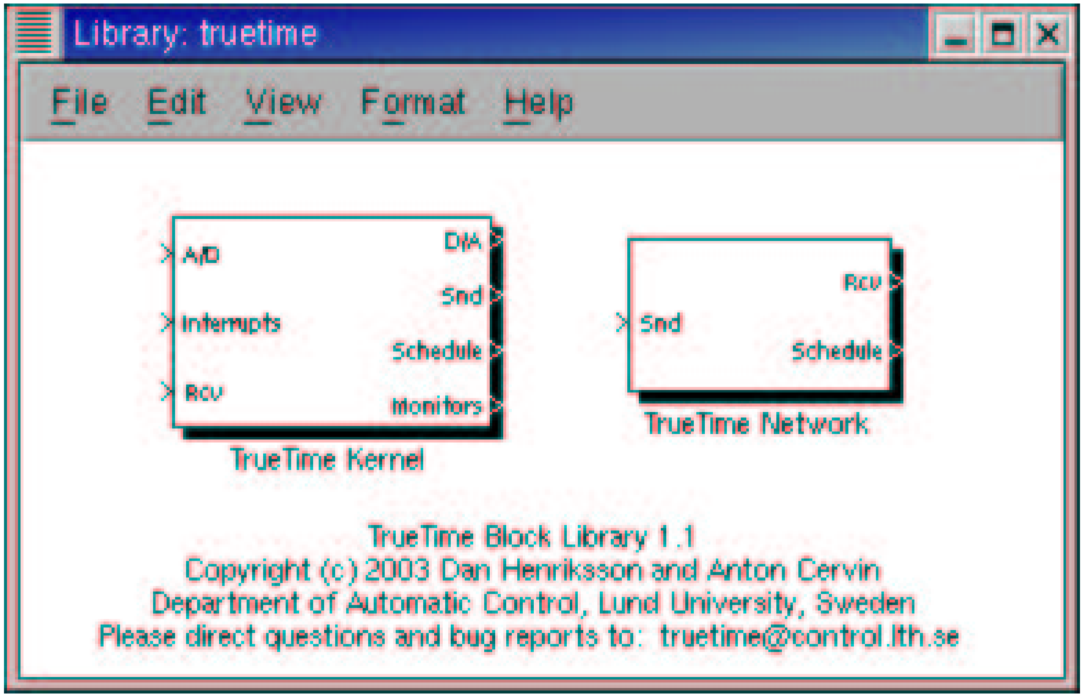
τ_1, τ_2 random delays with given probability density functions

Example of a Cost Function

Cost as a function of delay and jitter:



7. Simulation Using TrueTime

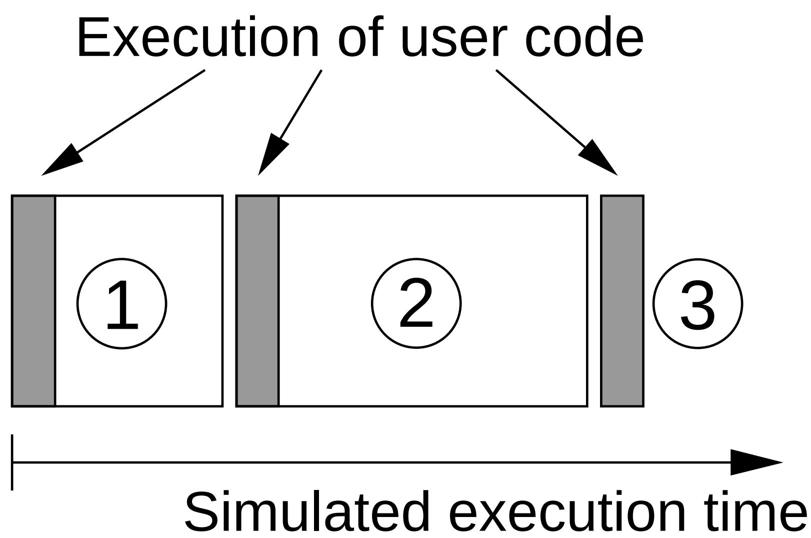


- MATLAB/Simulink-based tool
- Offers a Kernel and a Network block
  - Simulink S-functions written in C++

The Kernel Block

- Simulates a full, event-based real-time kernel
- Executes user-defined tasks and interrupt handlers
- Arbitrary user-defined scheduling policy
- Supports external interrupts
- Supports common real-time primitives (sleepUntil, wait/notify, setPriority, etc.)
- More features: context switches, overrun handlers

Task Execution Model

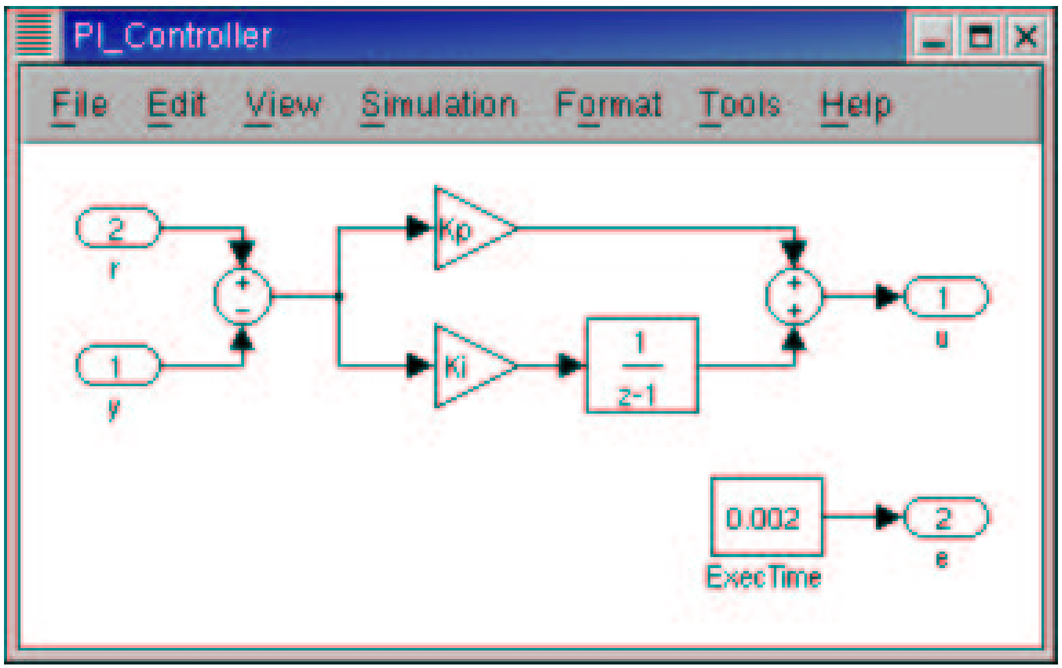


- Execution modeled by a sequence of segments
- The execution time of each segment is returned by the code function (may be data-dependent, random, etc.)

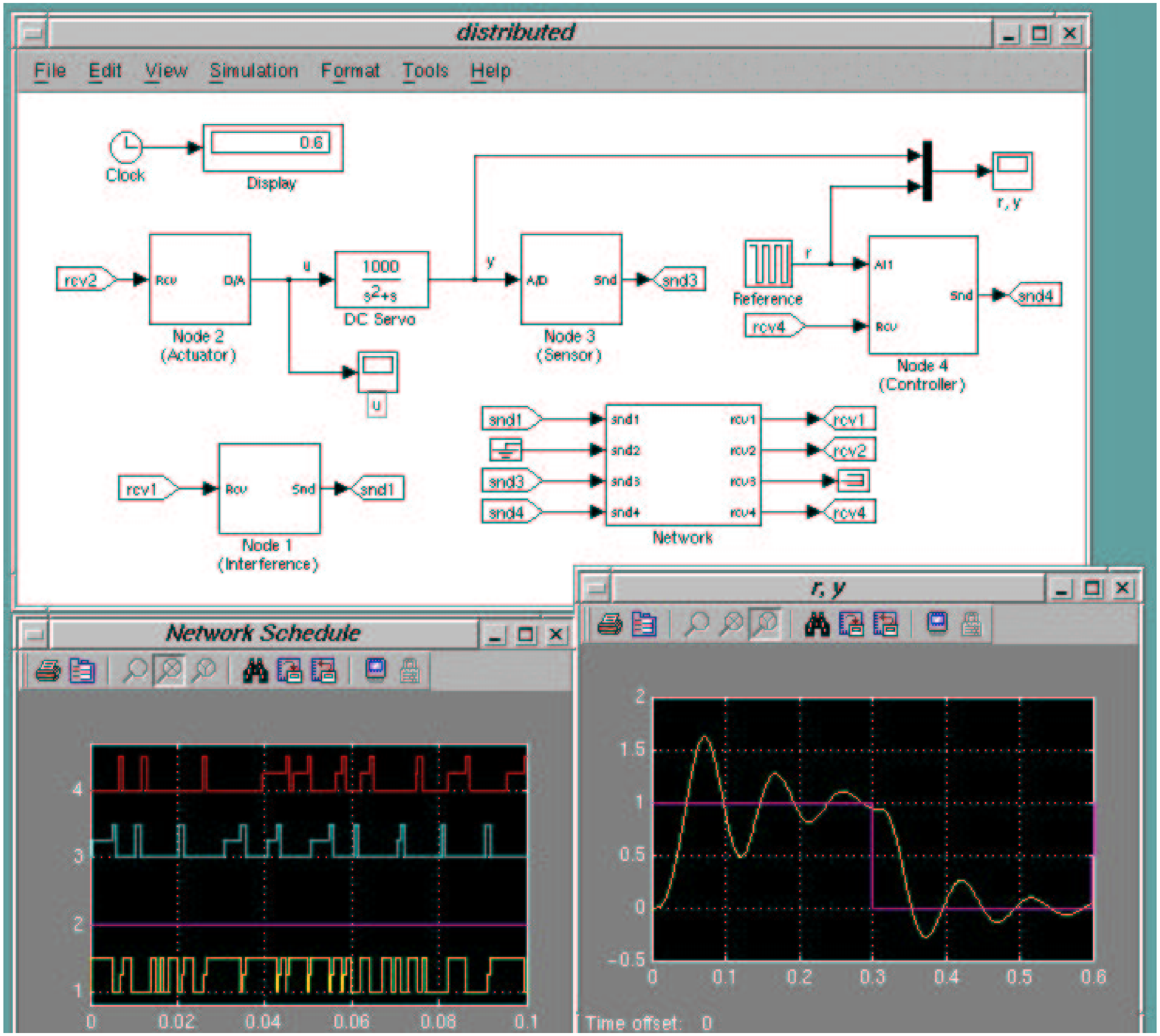
Controller Implementation

Choices:

- C++ function (fast)
- Matlab function (medium)
- Simulink block diagram (slow)



Screenshot


distributed
File Edit View Simulation Format Tools Help
Clock
Display
0.6
Node 2 (Actuator)
DC Servo
Node 3 (Sensor)
Reference
Node 4 (Controller)
r, y
Node 1 (Interference)
Network
Network Schedule
Time offset: 0

8. Summary

Scheduling techniques tailored to control tasks:

- Subtask scheduling – reduce latency
- Feedback scheduling – handle CPU load variations
- The Control Server – real-time control components

Tools for analysis of control performance:

- Analysis using Jitterbug – linear systems
- Simulation using TrueTime – general systems

Download

Jitterbug:

http://www.control.lth.se/˜lincoln

TrueTime:

http://www.control.lth.se/˜dan