



# A Multi-model Framework to Implement Self-managing Control Systems for QoS Management

Tharindu Patikirikorala

Alan Colman

Jun Han

Liuping Wang

# QoS control of Software systems



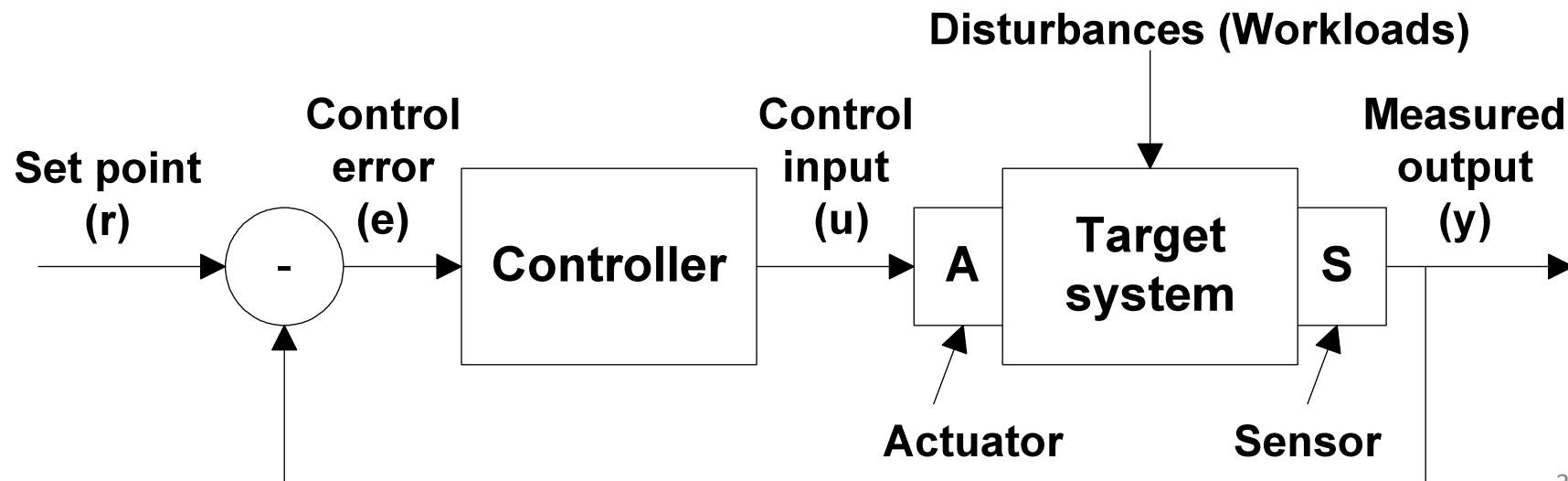
Effective self-managing capabilities are required by the software systems due to

- Increasing complexity
- The dynamic unpredictable operating conditions
  - Unpredictable workload conditions (e.g. slashdots)
  - Operating conditions of the system (e.g. caching)
  - Evolution of the software (e.g. new feature additions, bug fixes)
  - Component failures or replacements

The **feedback control loop** is recognized as one of the key concepts. (e.g. MAPE-K loop)



- Control engineering uses feedback control loop as the first class design concept.
- Consequently, techniques are borrowed to realize self-management in software systems.





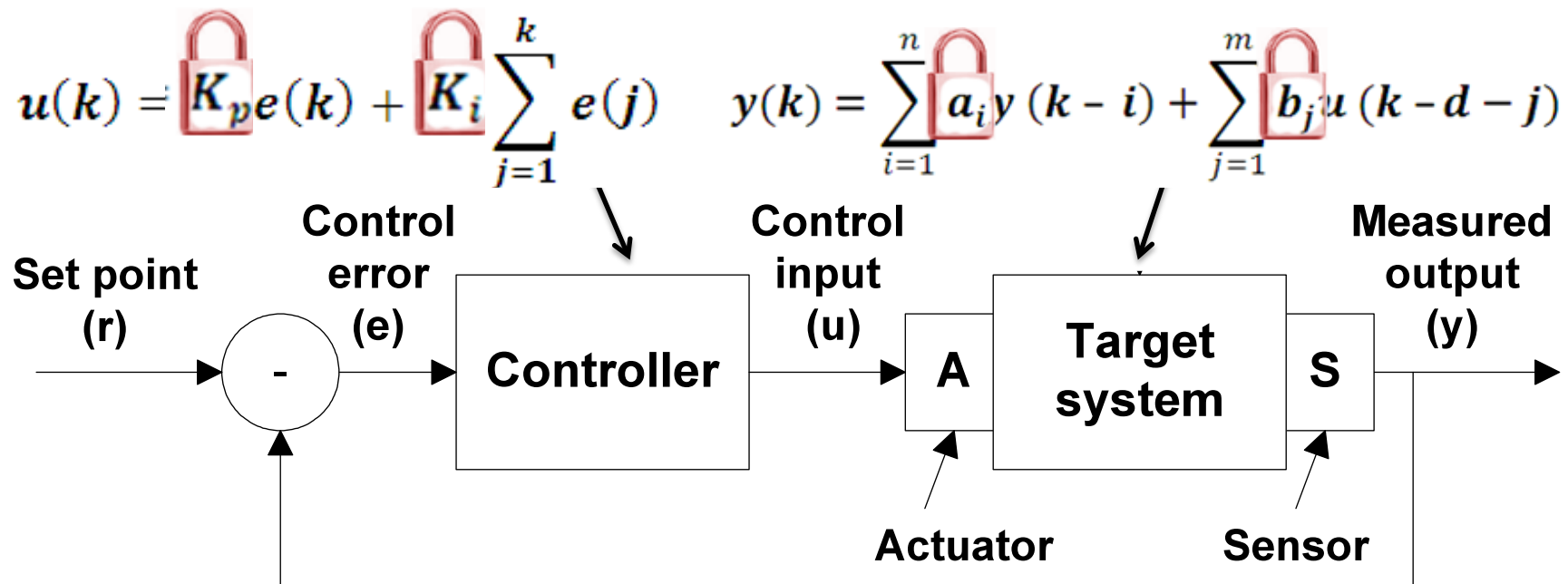
# Background

- ❑ In order to build a control system, behavior of the system has to be described with a formal model.  
(u-y relationship)
- ❑ Typically, a **single linear model** is used to describe the *behavior on a particular operating region and condition*.
- ❑ Then this model is used to design a controller.

## Fixed gain control



- Controller tuning parameters are decided at design time and remain constant at runtime.



# Fixed gain control



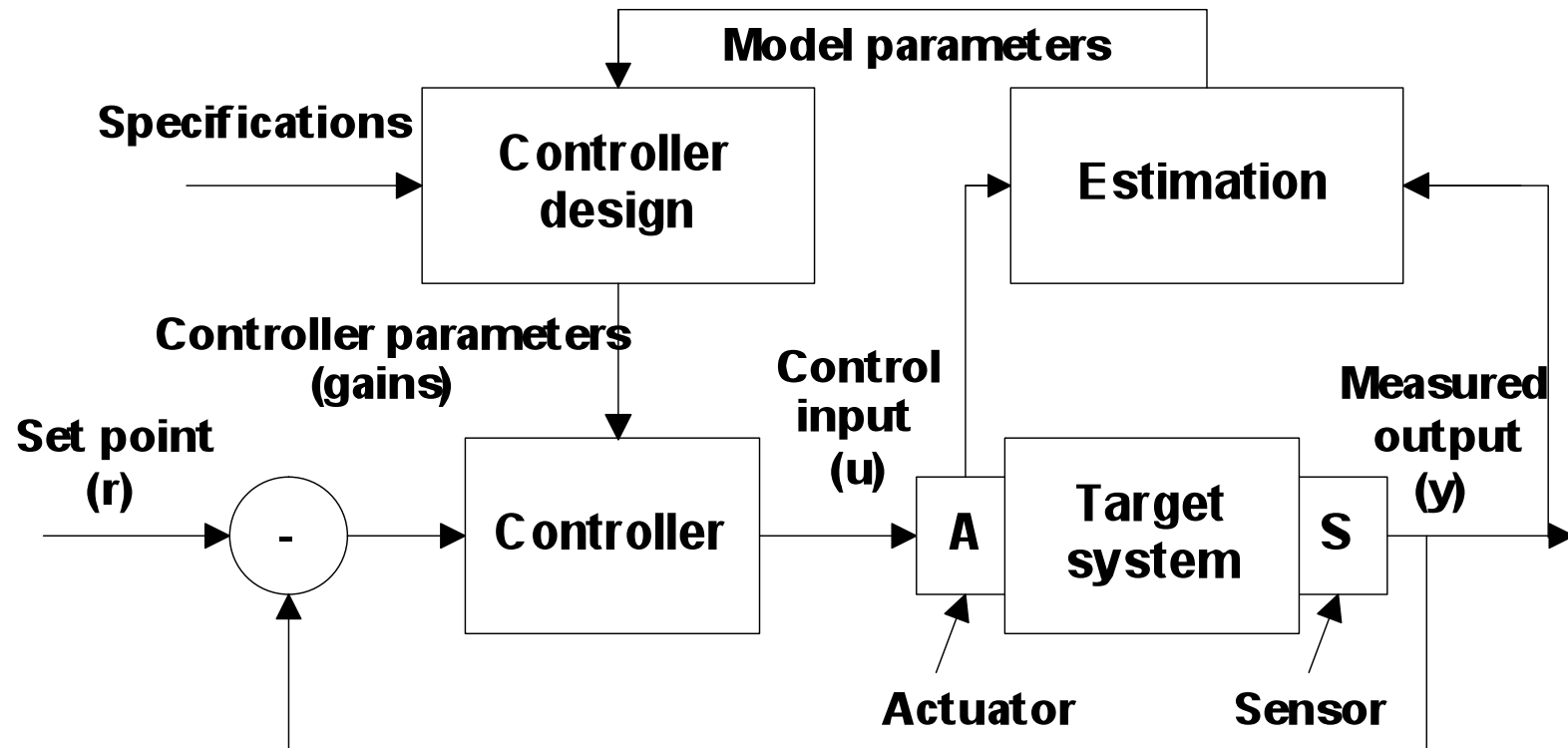
## □ Limitations ☹️

- Control objectives are hard to achieve under changing conditions.
- Limited design flexibility.

# Adaptive control



- A solution for changing conditions



# Adaptive control



## □ Limitations ☹️

- Instabilities under fast changing conditions
- Assumptions and requirements

# A multi-model solution



- The complex behavior of software systems cannot be described by a single linear model.

Consequently,

- **Multiple models and controllers** maybe required to represent the behavior and provide effective runtime control.



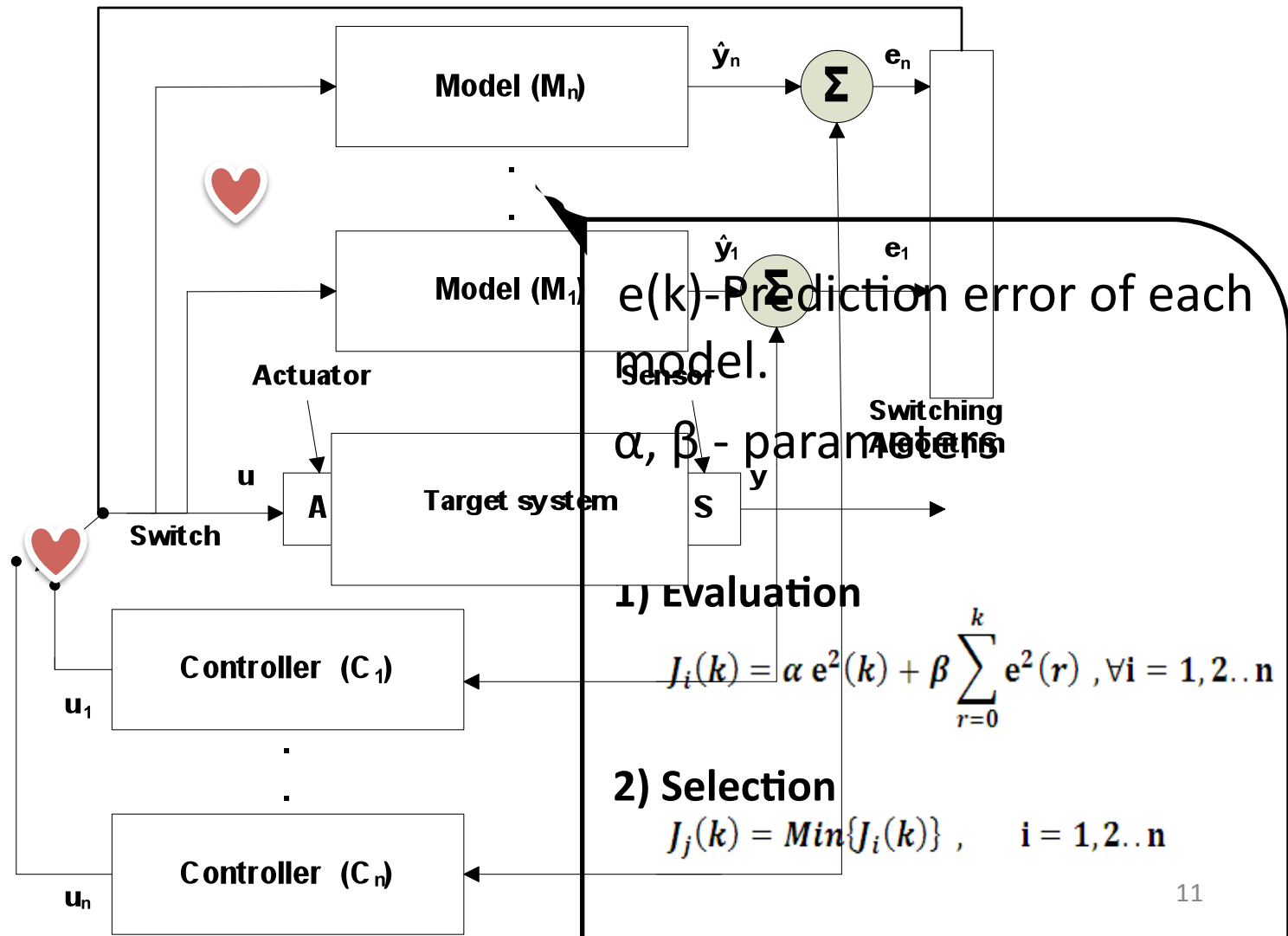
- The solution requires the ability to
  - Integrate multiple models
  - Integrate multiple controllers
  - Autonomously detect the change of conditions
  - Select the most suitable controller to provide runtime control.

- We call them

**Multi-model self-managing control systems.**

10

# We propose MMST adaptive control





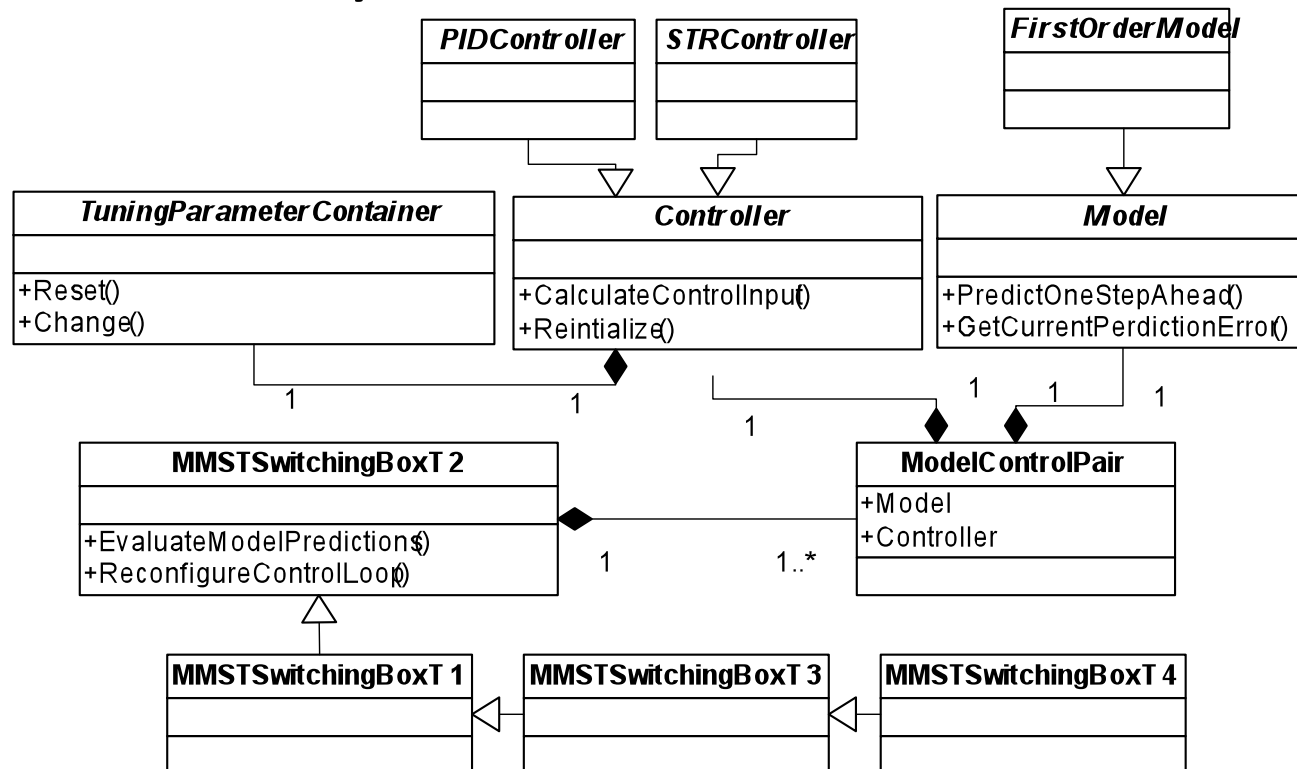
## Four MMST schemes

- Type 1:** *All adaptive models*
- Type 2:** *All Fixed models* ✓
- Type 3:** *One Adaptive model and one Fixed model*
- Type 4:** *Adaptive models and Fixed models* ✓

✓ most suitable for software systems



- Using MMST as a basis we propose a reference model and class library to build self-managing control systems.





An extendable class library with

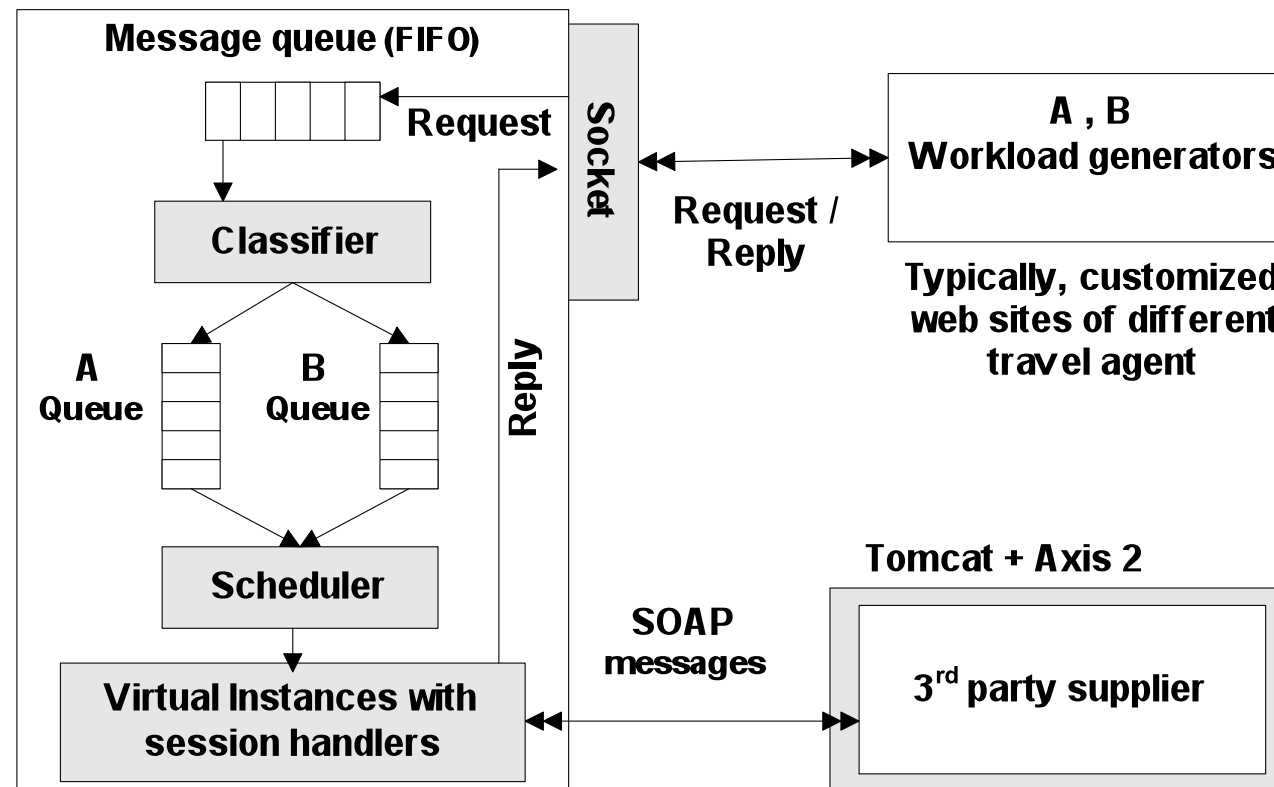
- PID controller**
- Model predictive controller (MPC)**
- Indirect self-tuning regulator**
- Self-tuning PID controller**
- Fixed and adaptive models**
- All four MMST adaptive control schemes.**

URL-<http://www.ict.swin.edu.au/personal/tpatikirikorala/Downloads>

# Case study



Architecture of a flight reservation system with two travel agent A and B.





# Relative Control

- The objective is to maintain the response time of the workloads of both agents while allocating the limited amount of sessions.
- For such system relative guarantee control scheme is a useful design.

Let say response time of A and B are  $R_a$  and  $R_b$  and the session allocations are  $S_a$  and  $S_b$ .

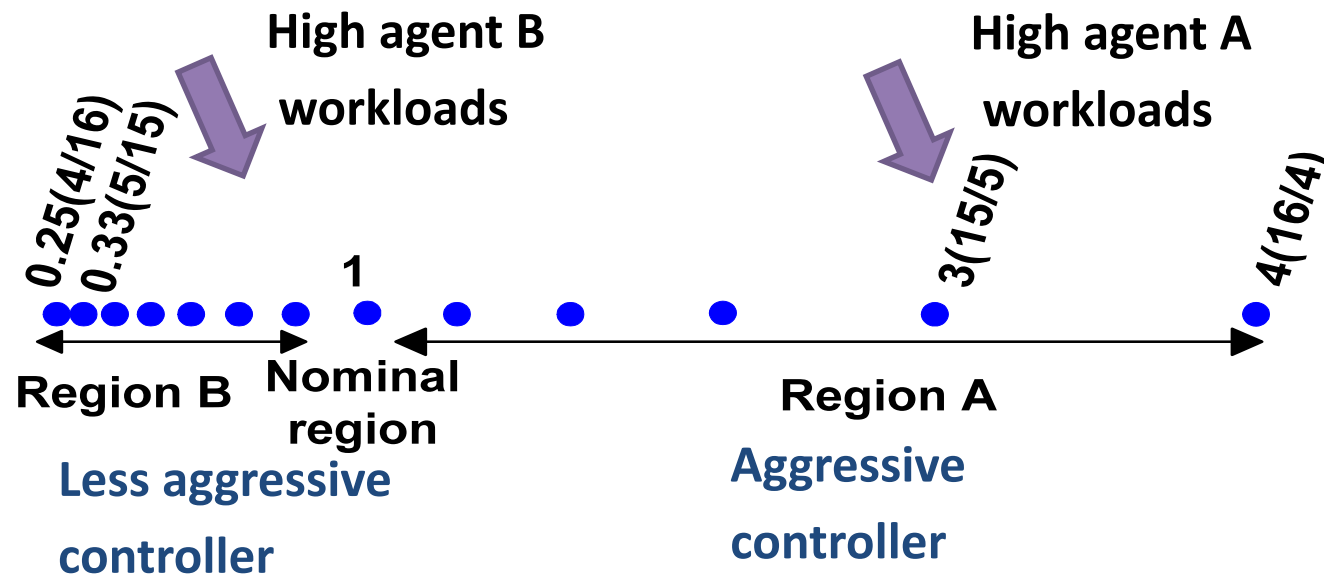
The control system variables are

- Measured output =  $y = R_a/R_b$
- Control input =  $u = S_a/S_b$

# Why multiple models?

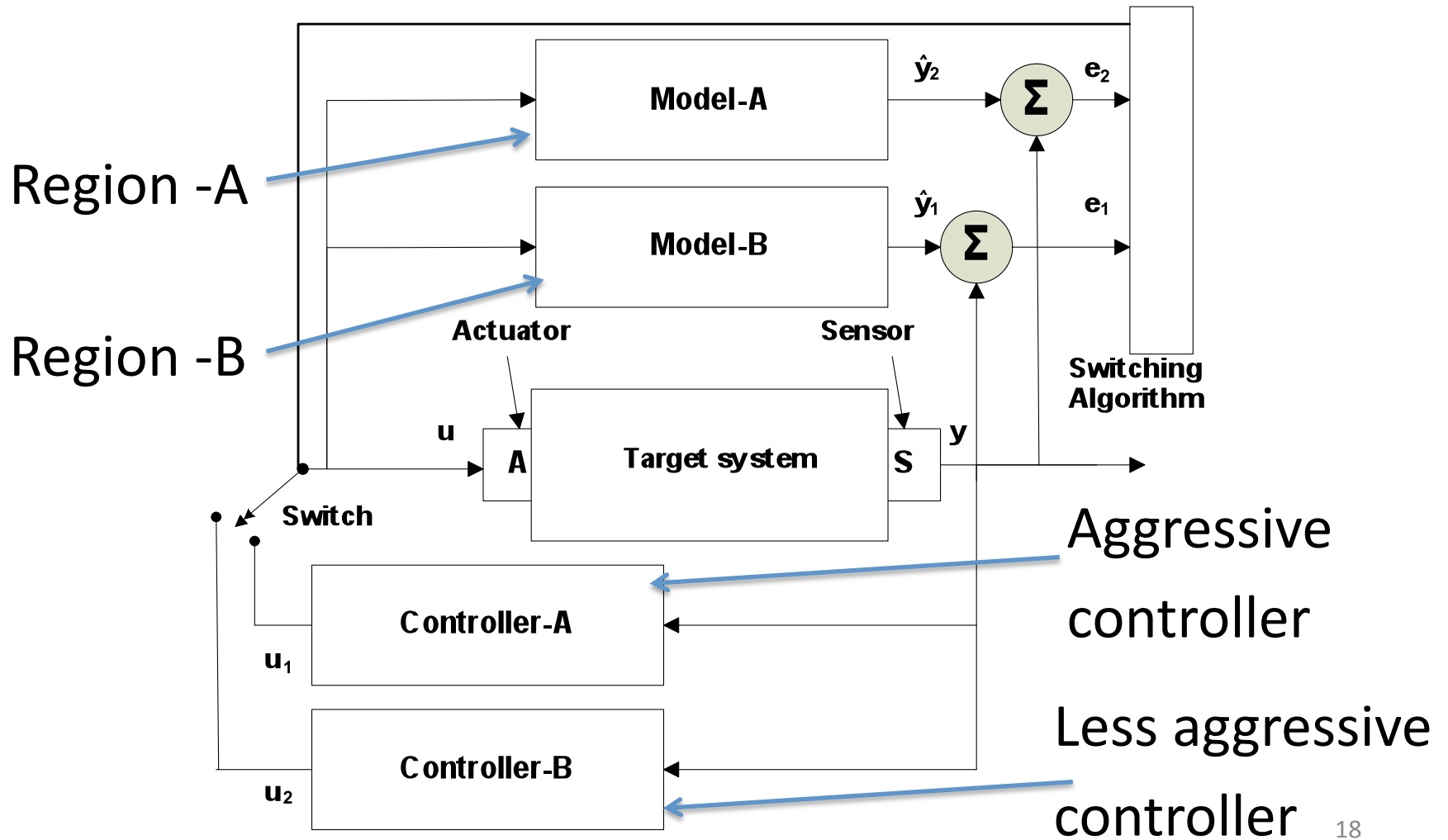


- The total number of sessions are 20.  $S_a, S_b \geq 4$ .
- $S_a/S_b$  (4/16, 5/15, ..1...15/5, 16/4)

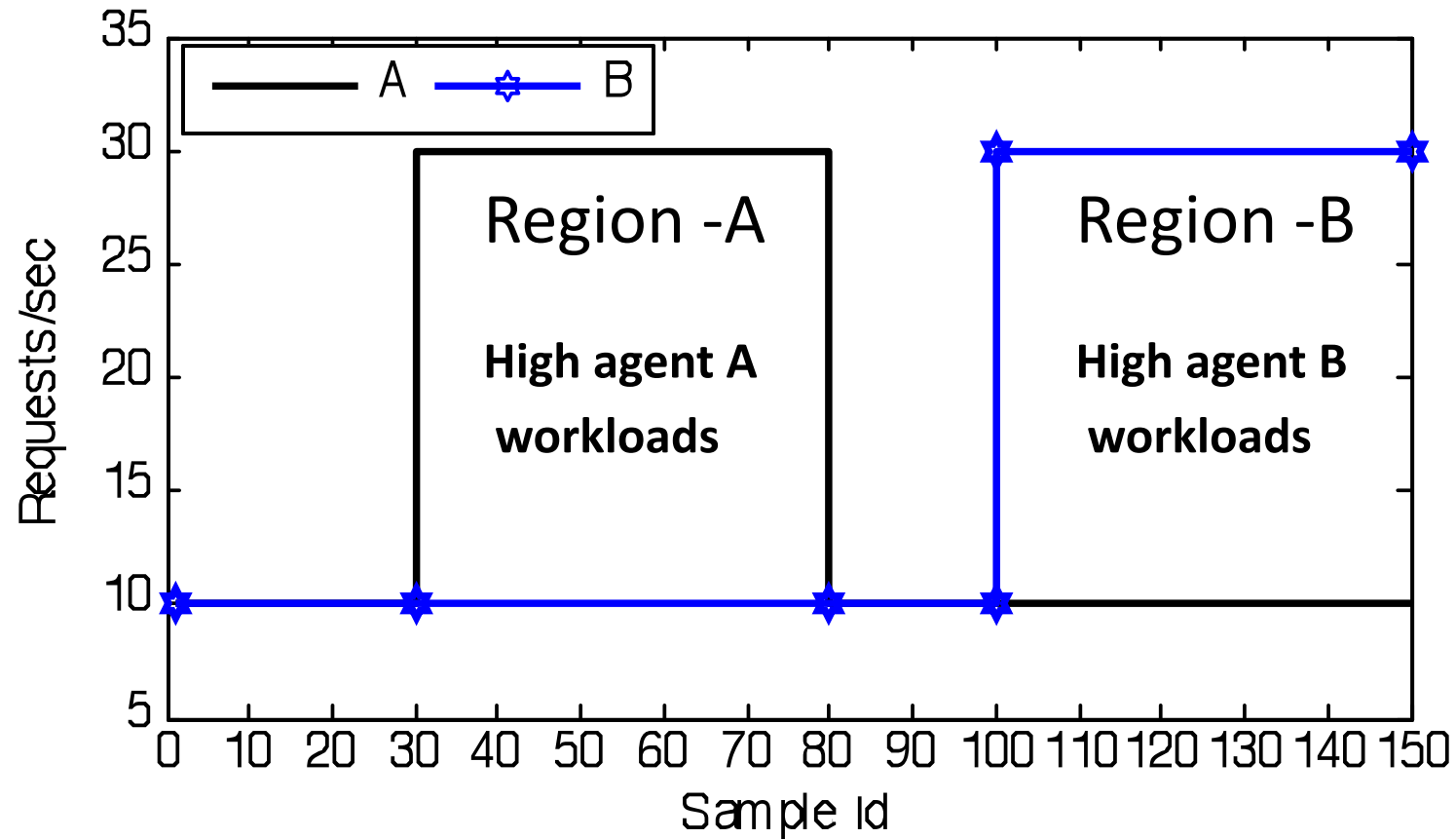


- *Two models and two controllers were designed simulating conditions of region A and B.*

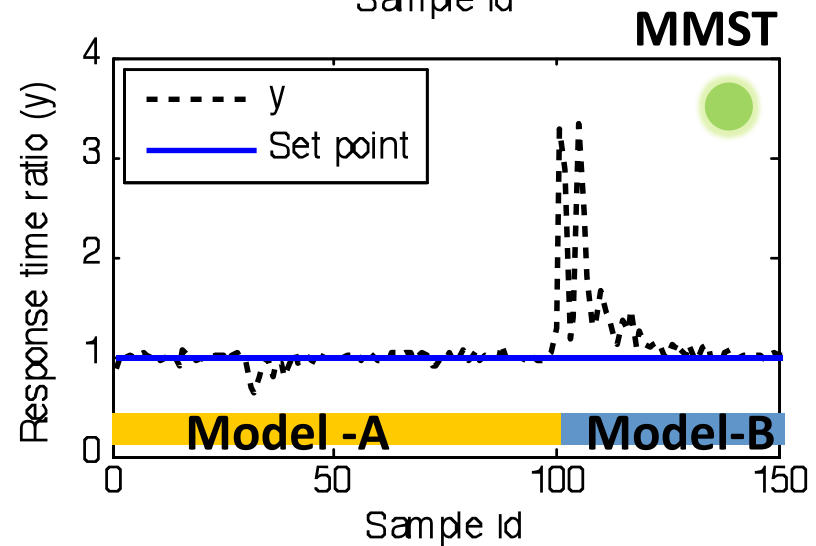
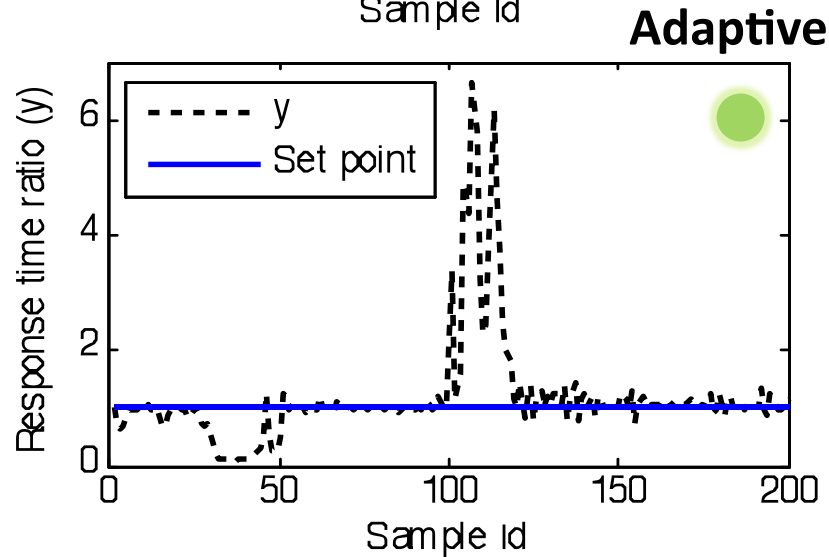
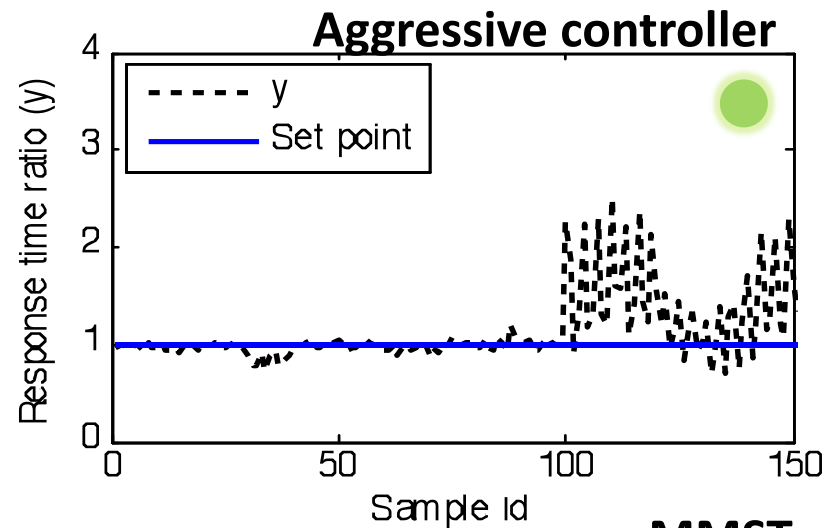
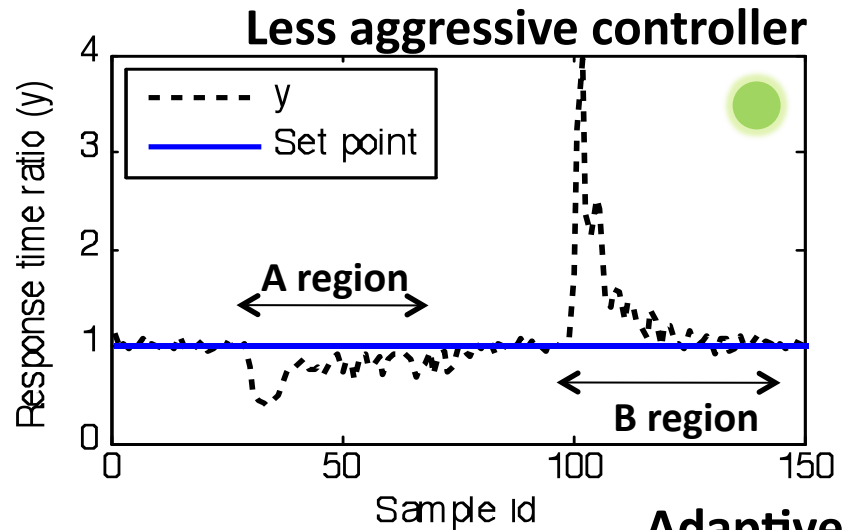
# MMST Type 2 design



# Experimental settings



# Experiment results



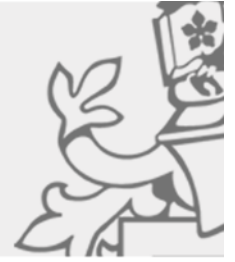


- ❑ Chattering
- ❑ Design cost
- ❑ Computational overhead
  
- ❑ As future work, to reduce the design cost a online model persistence mechanism will be investigated.



# Conclusions

- We propose a technique to design multiple model based self-managing control systems.
  
- Self-managing control provides
  - better performance under changing operating conditions
  - Design flexibilityCompared to a single model based approaches.



# Thank you!



# Issue of fixed gain control

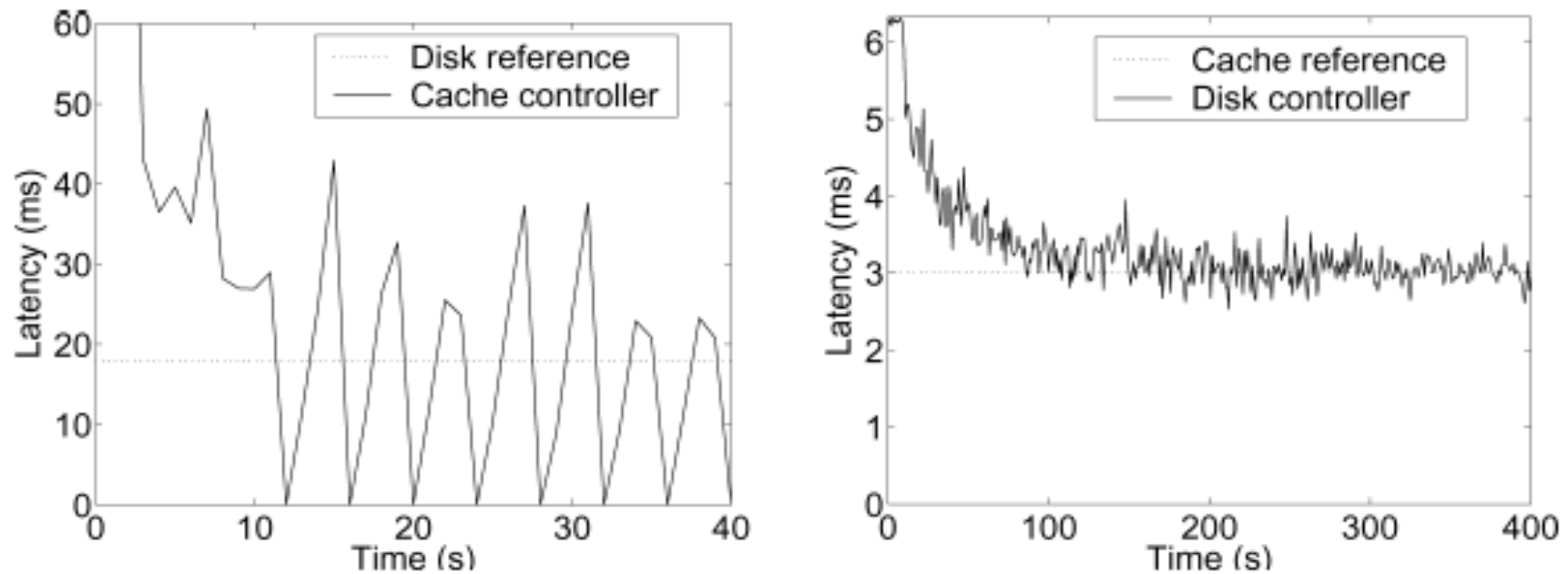


Fig. 5. The performance of the nonadaptive controller. A controller designed for the in-cache model is applied to a mostly on-disk workloads (left). A controller for the on-disk model is applied to a mostly in-cache workload (right). A latency of 0 means that there were no requests during that sample period.

Extract from *–Triage: Performance differentiation for storage systems using adaptive control’* by Karlsson et al. 2005

# Model Switching

