Using Organic Computing to Control Bunching Effects

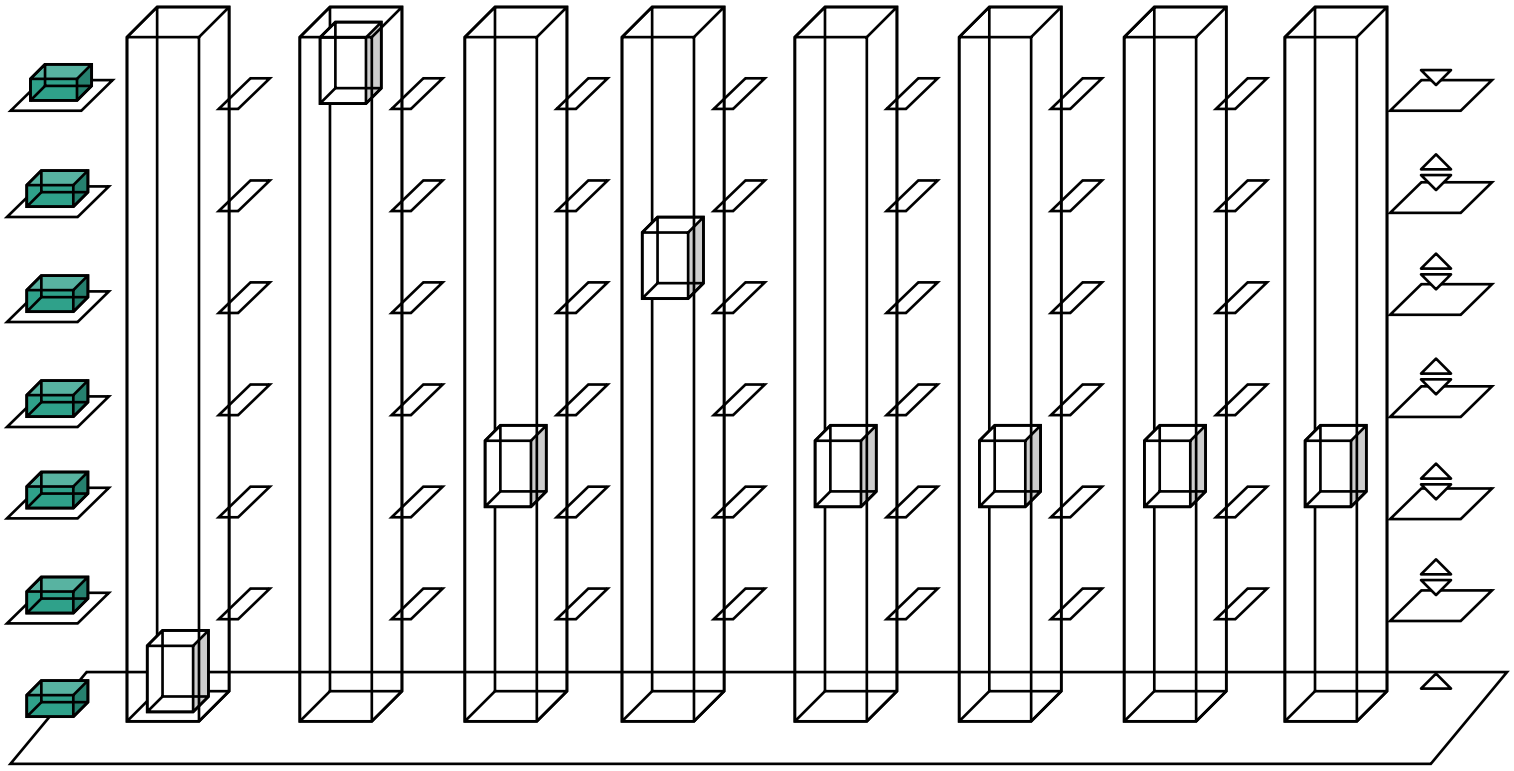
ARCS, Dresden 2008

Oliver Ribock, Urban Richter, and Hartmut Schmeck

Karlsruhe Institute of Technology (KIT)

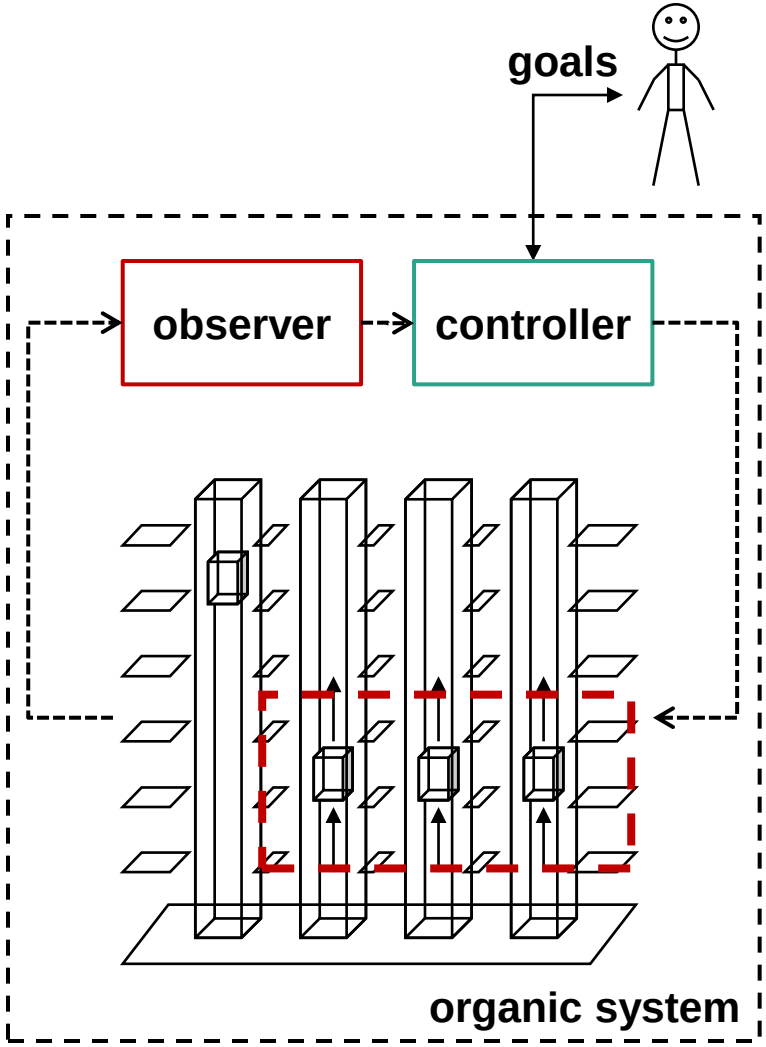
AIFB Universität Karlsruhe (TH)
Forschungsuniversität • gegründet 1825

Motivation



Outline

1. Organic Computing
2. Scenario of self-organising lifts
3. Measuring and controlling bunching effects
4. Observer/controller architecture
5. Experimental results
6. Summary and outlook



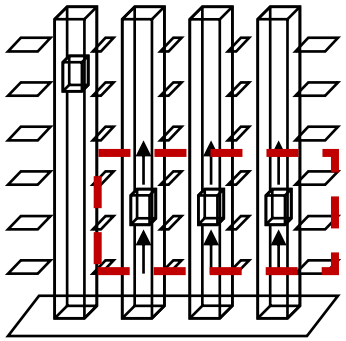
Organic Computing

- Technical systems show increasing complexity.
- Life-like characteristics (learning, adaptation,…)
- Maintain themselves to survive attacks and breakdowns (self-x-properties).

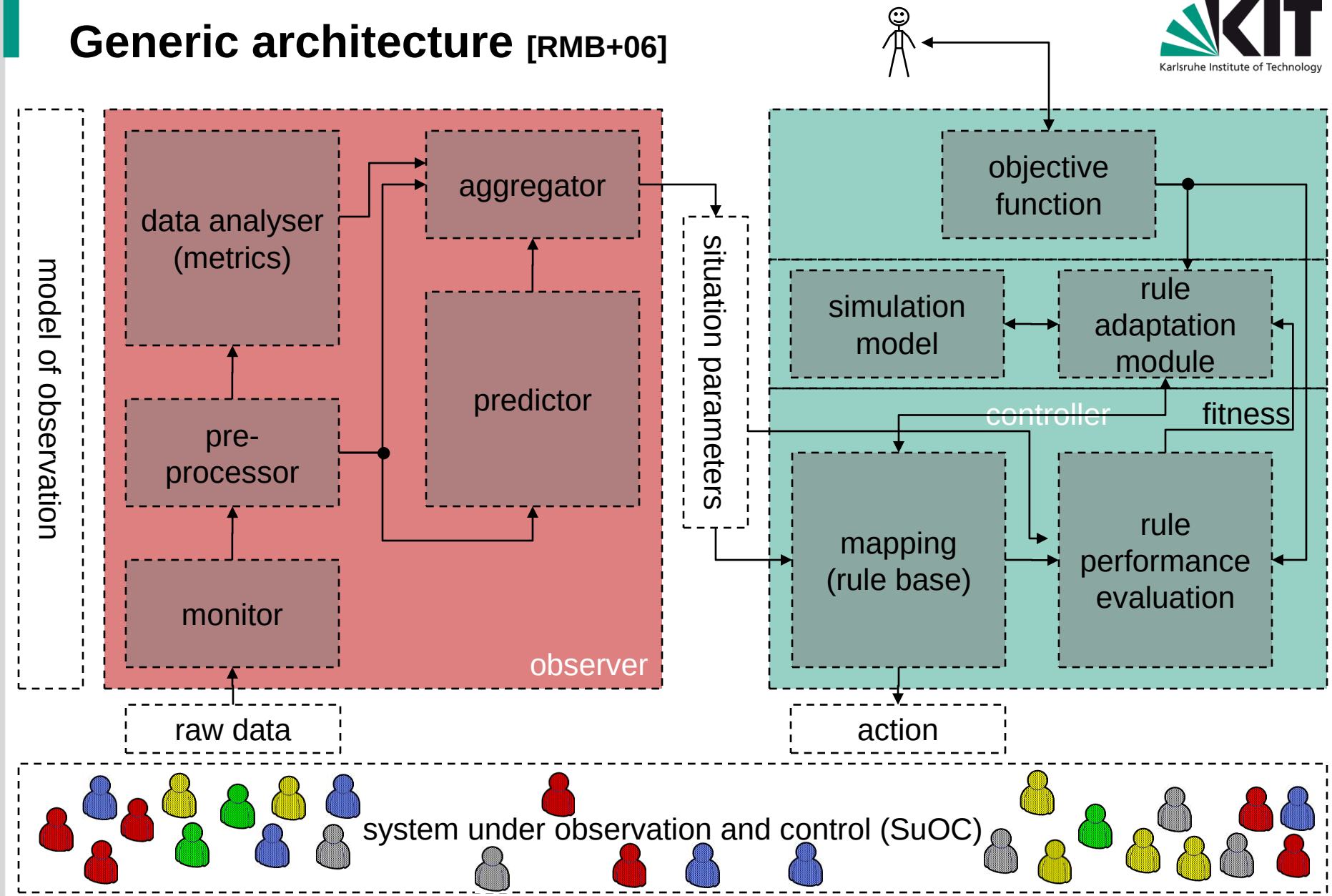
- Move from a centralised system to a decentralised one.
- Large number of interacting and self-organising sub-systems

- When collections of intelligent, autonomous devices cooperate in a self-organised way, unexpected things may happen: (positive/negative) emergence.
- Emergence = self-organised order

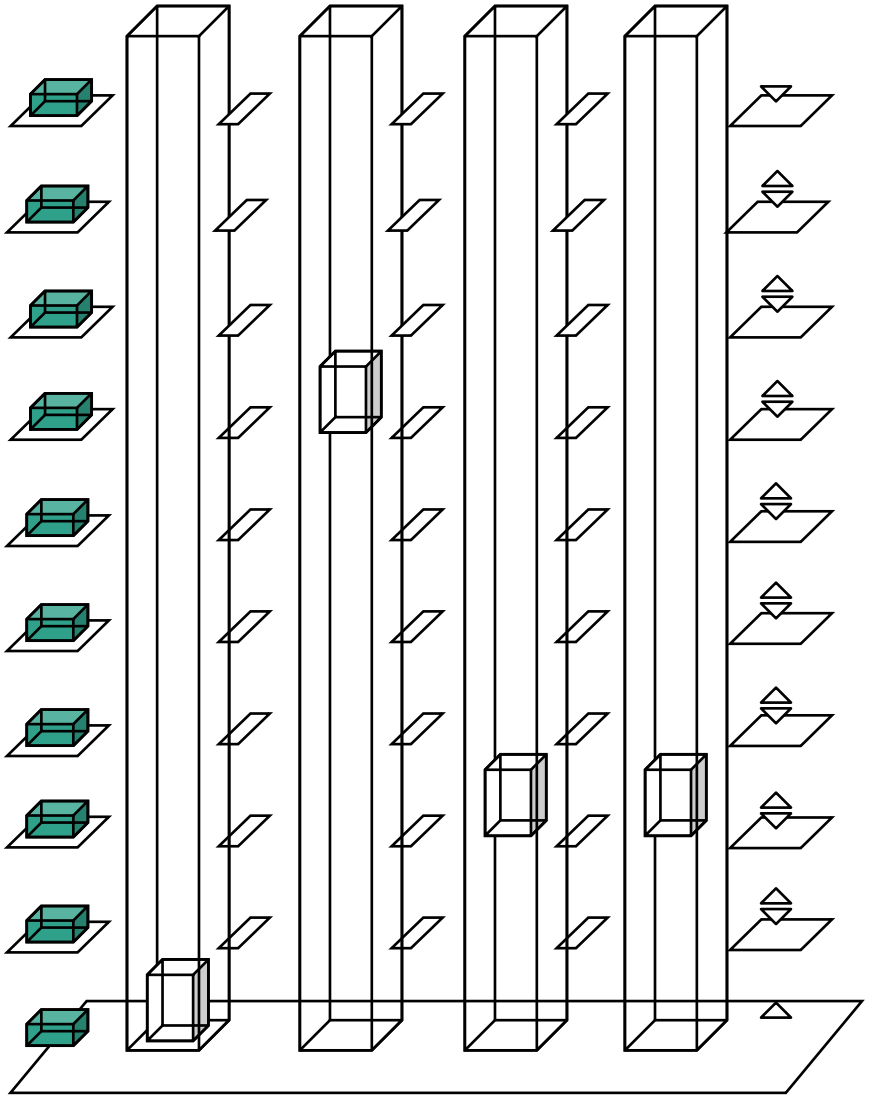
- Regulatory feedback mechanism



Generic architecture [RMB+06]

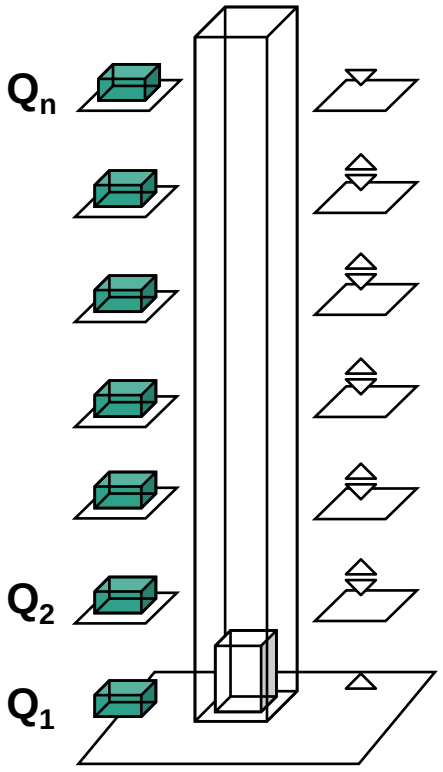


The scenario of self-organising lifts



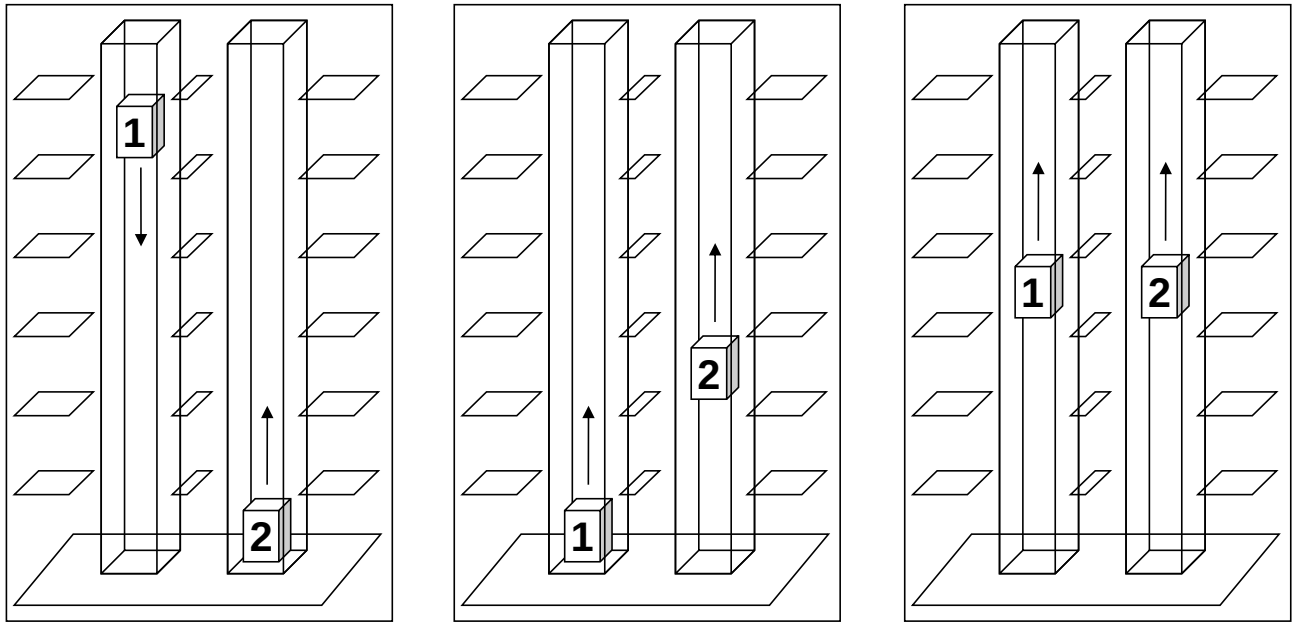
- Fixed capacity of 20 passengers
- 10 floors
- Arrival of new passengers on every floor follows a Poisson distribution.
- No central information
- No central control
- No communication between the lifts
- No information about the number of passengers waiting on the floors
- No priorisation of the hall calls
- Every lift has the same simple strategy to serve the hall calls.

Static strategy of a single lift



- Queues Q_1, …, Q_n of passengers on every floor (id, place of departure, destination)
- Queue in the lift depends on passengers.
- Lift strategy:
  - Serve every hall call in its running direction if free capacity exists.
  - Change the direction at the end of a building or if the cabin is empty.
  - Wherever a passenger wants to leave the cabin, the elevator stops.

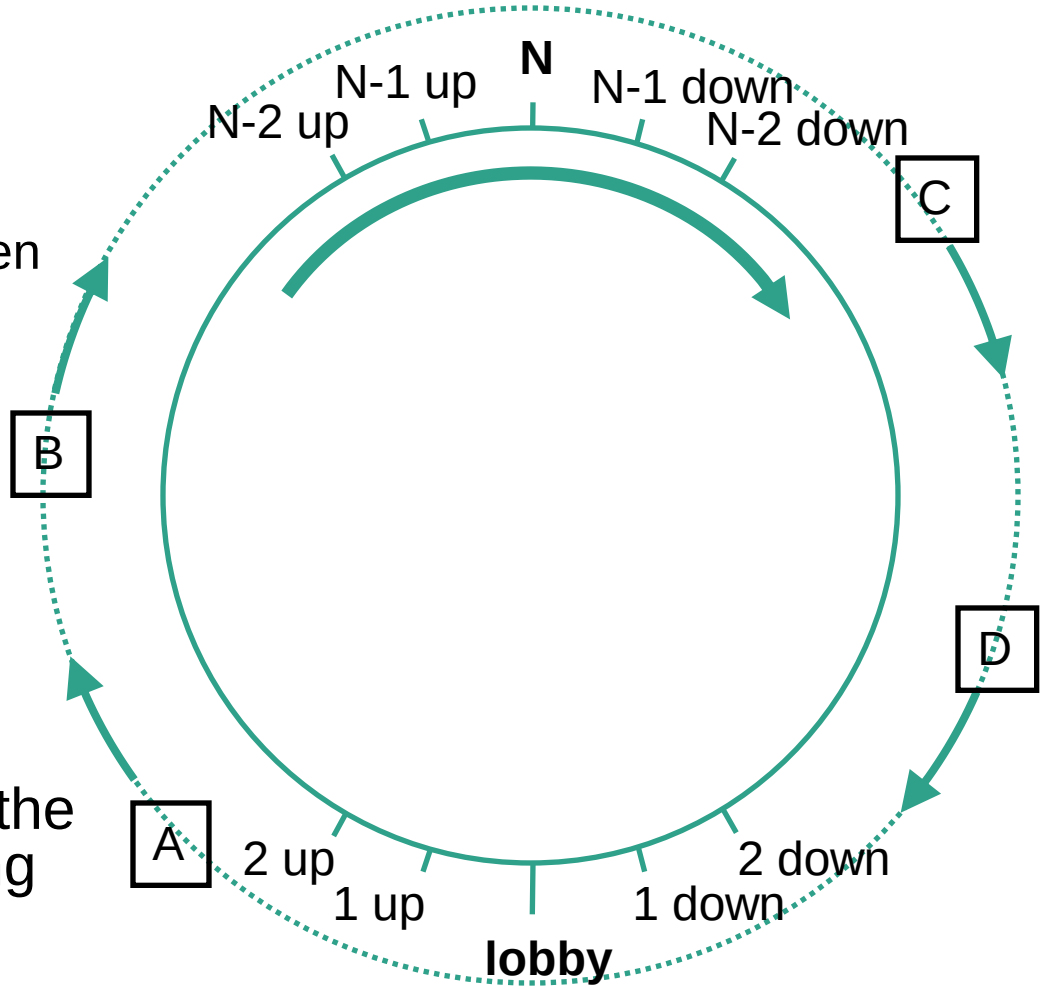
What is bunching?



- We call a situation *bunching*, whenever the lifts are not distributed uniformly over the building.
- *Bunching value*: Deviation of measured situation from ideal lift system behaviour

How to measure bunching?

- Optimal group behaviour
  - Lifts travel clockwise around the circle.
  - Measure the gaps between the lifts.
  - Gaps between A and B, B and C, C and D, and D and A should be equal.
  - Sum of deviation serves as a characterisation of bunching.
- Bunching effects waiting time.
- Bunching increases with the increase in system loading and the number of lifts.



According to [Pow95]

How to control bunching?

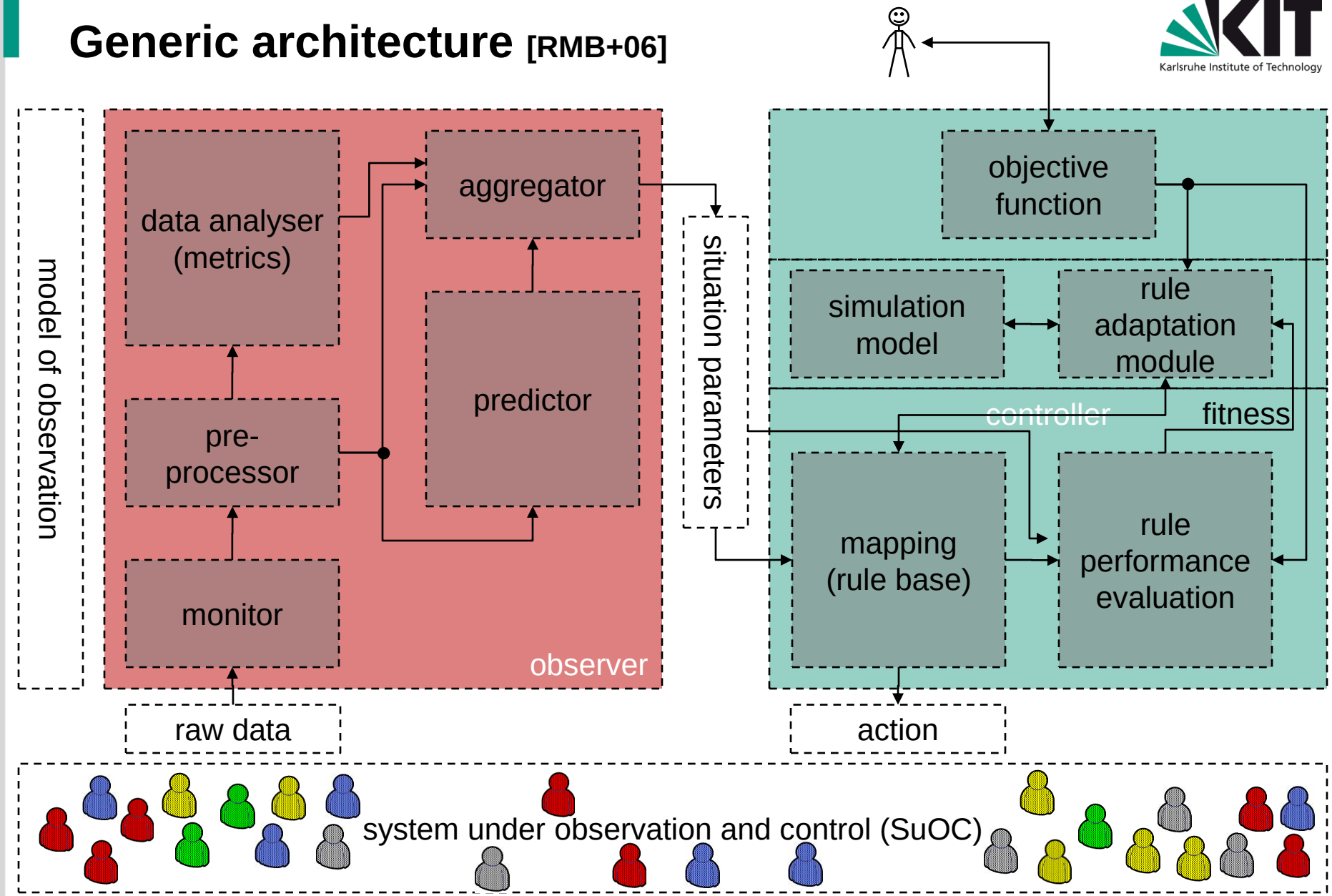
- Goal: Maintaining an equal spacing of the lifts.
- Possible control actions
  - Delaying/parking lifts at the lobby.
  - A lift drives past and passengers wait for another lift.
- Accelerate delayed lifts and/or slow down early lifts.
- A lift is considered to be delayed, if the distance to the next lift in travelling direction is significantly higher than the average distance.

Control strategies

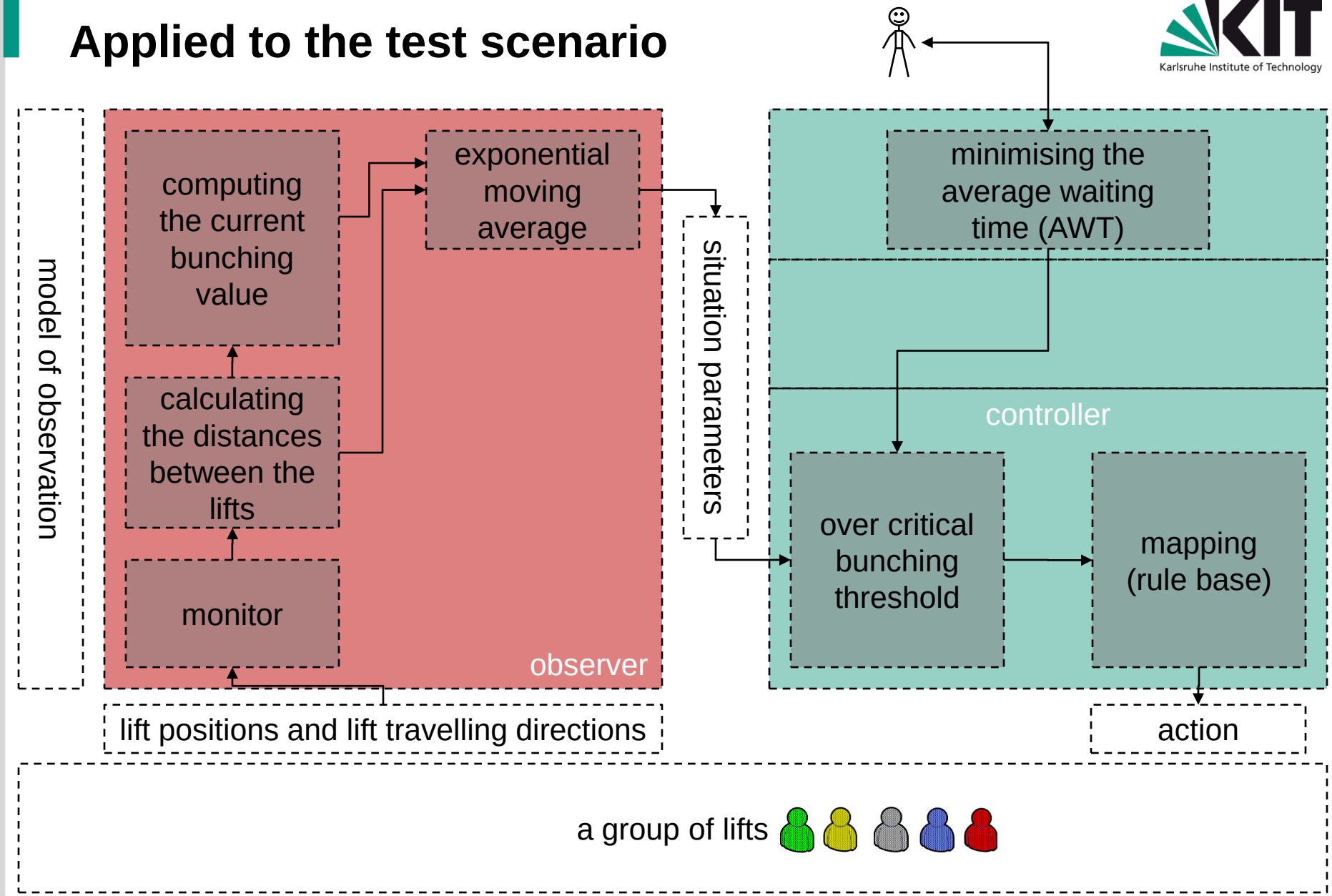
- **strongIntervention**
  - Goal: Accelerate delayed lifts.
  - Idea: Manipulate (a parameter of) the decision rules of the lifts.
  - Delayed lifts are blinded, i. e., they will not serve any hall calls in the building and will thus stop less frequent.

- **softIntervention**
  - Goal: Accelerate delayed lifts.
  - Idea: Change the lift's perception of its surrounding.
  - Acceleration is achieved by hiding a specific hall call from a certain lift.

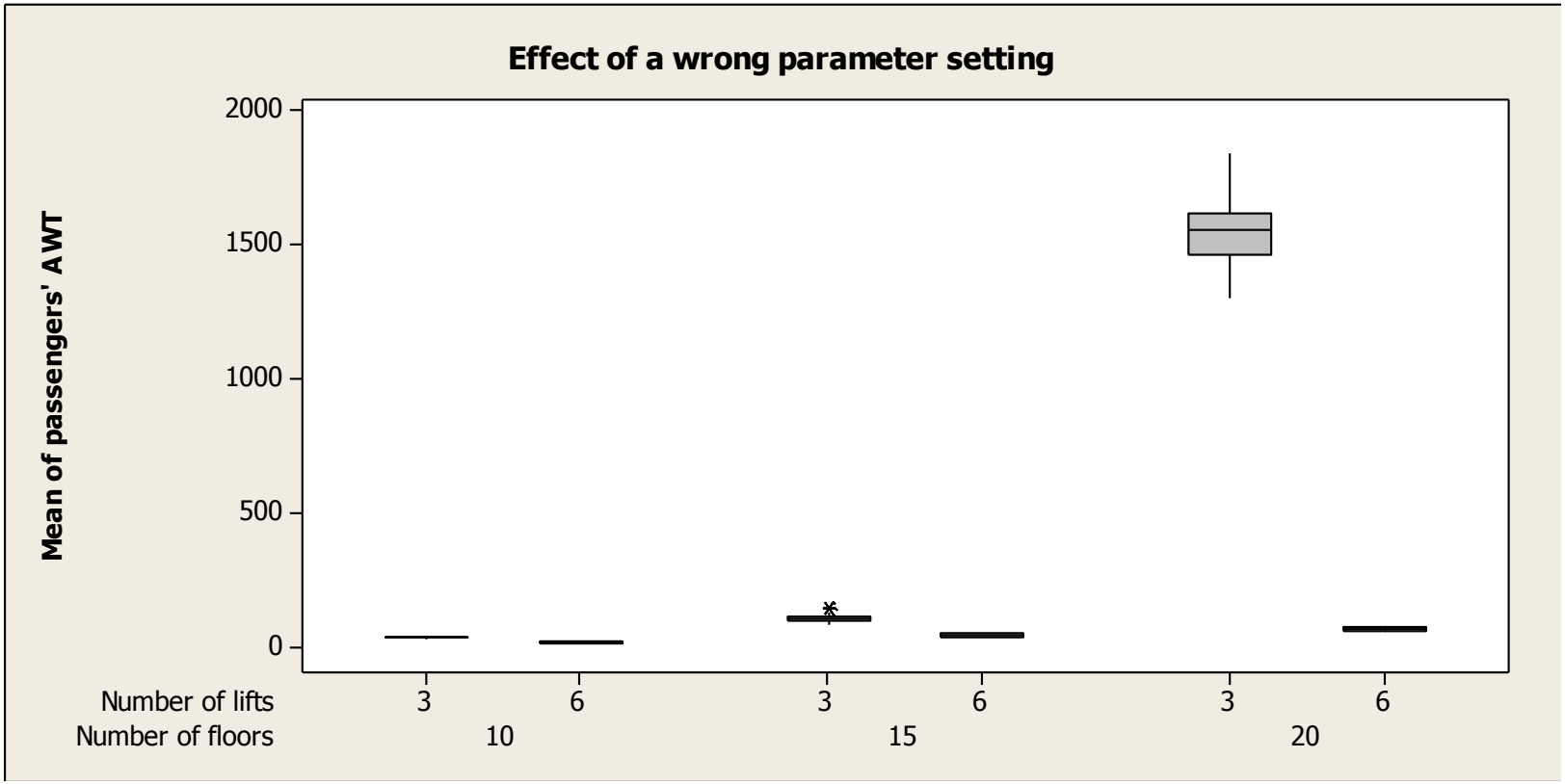
Generic architecture [RMB+06]



Applied to the test scenario

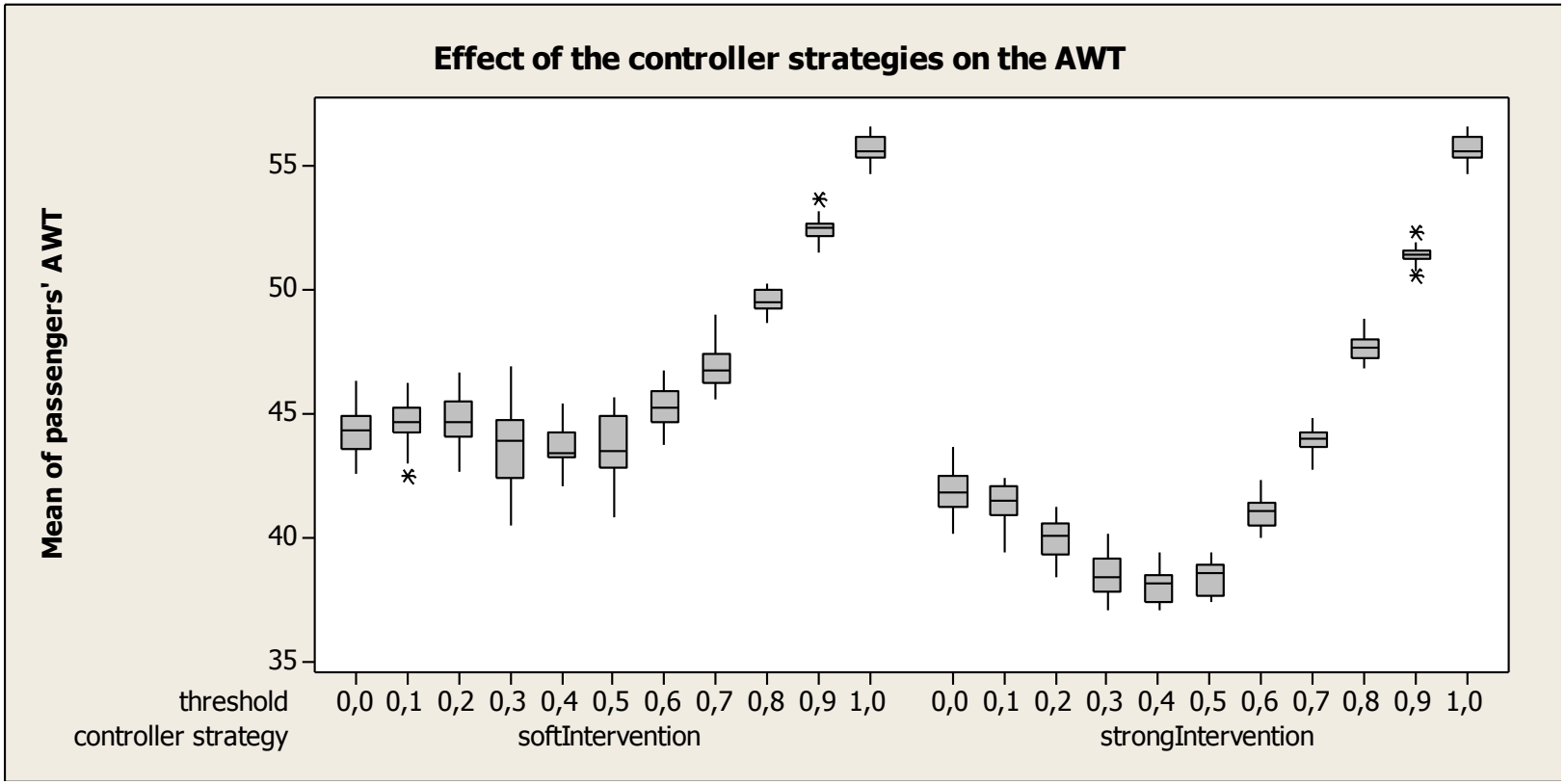


Preliminary investigations



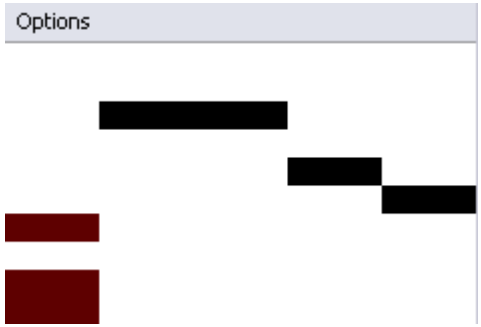
- The impact of variations of the number of floors, the number of lifts, and the traffic intensity is as intuitively estimated.
- Increasing the number of floors or the arrival rate results in a higher AWT. In contrast, more lifts lead to a lower AWT.

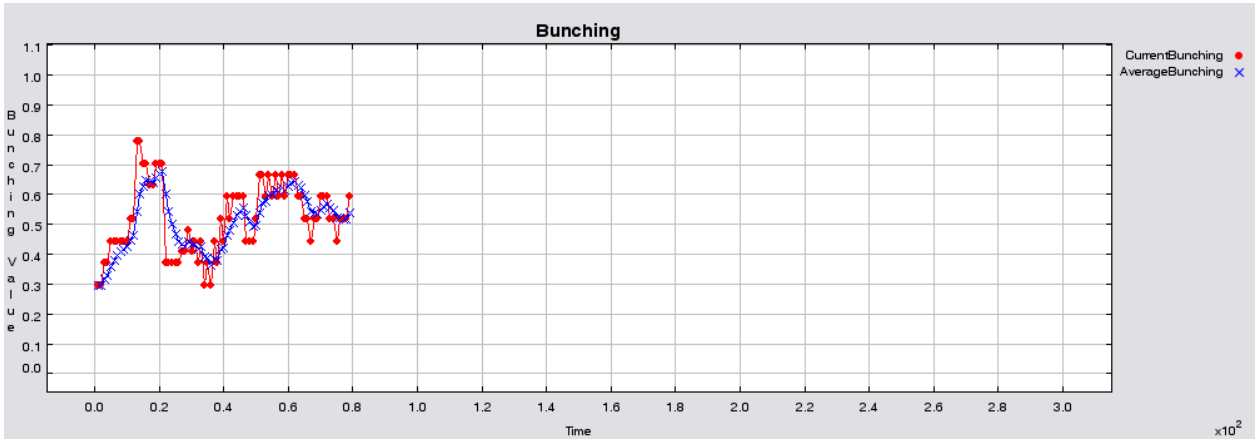
Effectiveness of the control strategies

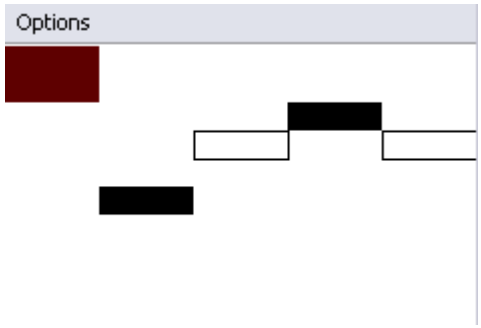


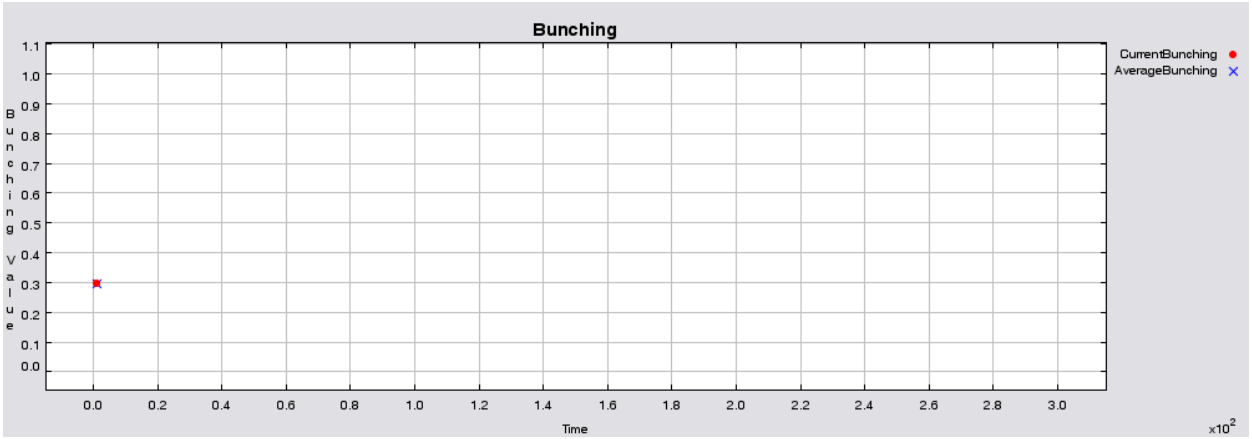
- The overall effectiveness of the control strategies was tested by varying the threshold of bunching for activating the controller.
- A threshold of 0.0 means that the controller is always active while a threshold of 1.0 corresponds to a deactivated controller.

Simulation with strongIntervention









Summary and outlook

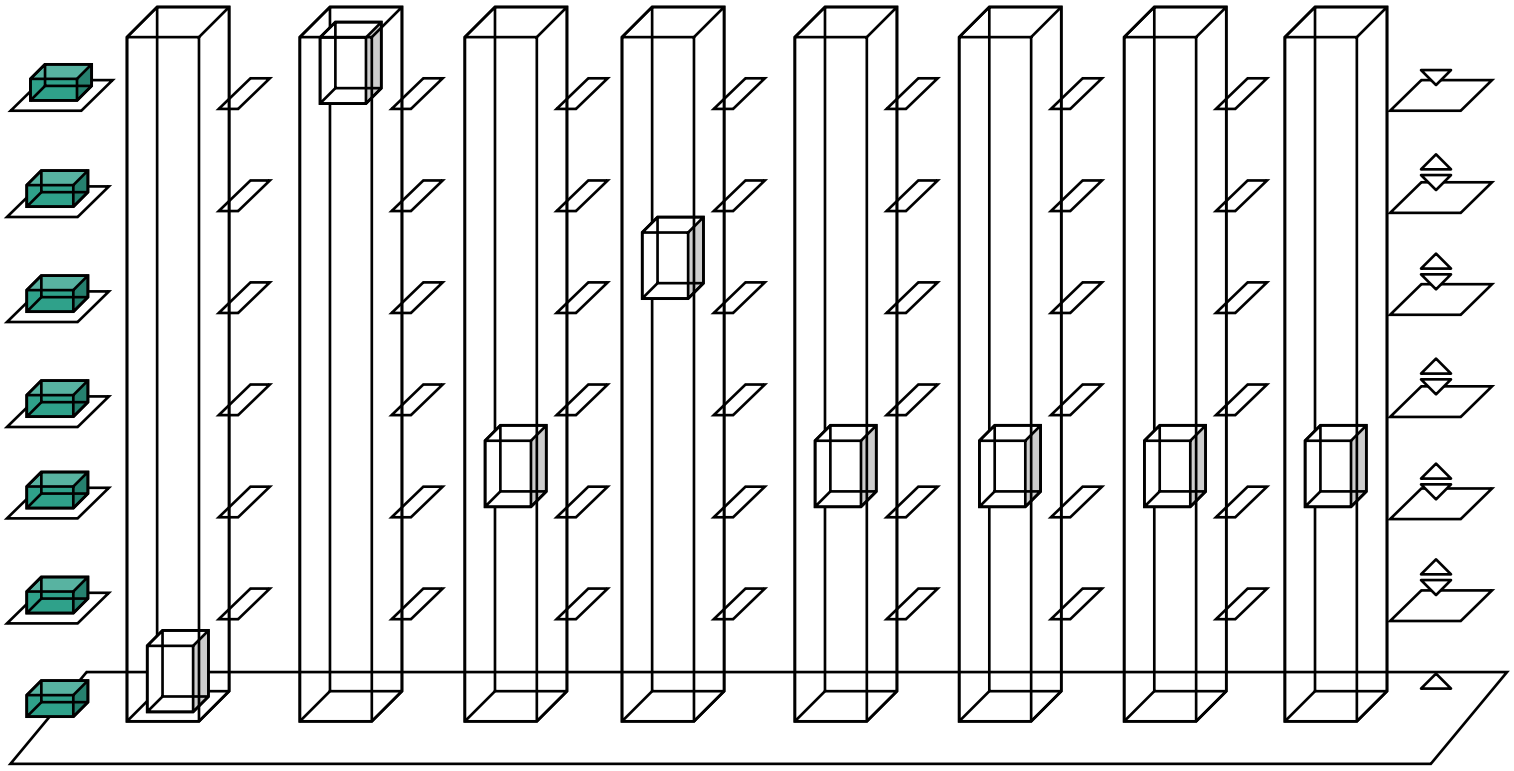
Summary

- Technical scenario of self-organising lifts which show bunching
- Applying an observer/controller architecture to realise controlled self-organisation
  - Metrics for quantification of bunching
  - Evaluation of 2 static control methods

Outlook

- Endowing the controller with rule adaptation and self-optimising capabilities to react to different traffic patterns
- Validating of learning methods
- Investigations of a prediction module to predict bunching

Thank you for your attention



Literature

[Pow95]

Bruce A. Powell. Measurement and reduction of bunching in elevator dispatching with multiple term objection function (USP 5447212). http://www.patentstorm.us/patents/5447212.html, 1995.

[RMB+06]

Urban Richter, Moez Mnif, Jürgen Branke, Christian Müller-Schloer, and Hartmut Schmeck. Towards a generic observer/controller architecture for Organic Computing. In Christian Hochberger and Rüdiger Liskowsky, INFORMATIK 2006 - Informatik für Menschen!, volume P-93 of GI-Edition - Lecture Notes in Informatics (LNI), pp. 112-119. Bonner Köllen Verlag, 2006.