



# PerOpteryx

**Automatically Improve Software Architecture Models  
for Performance, Reliability, and Costs using Evolutionary Algorithms**

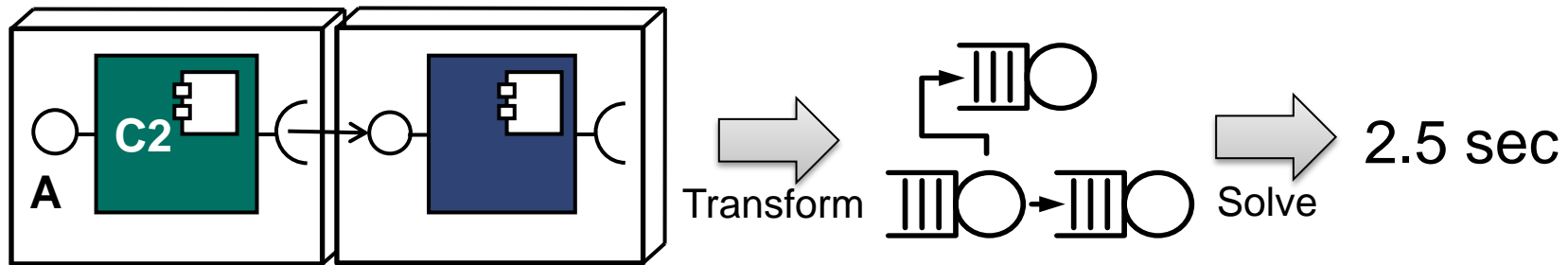
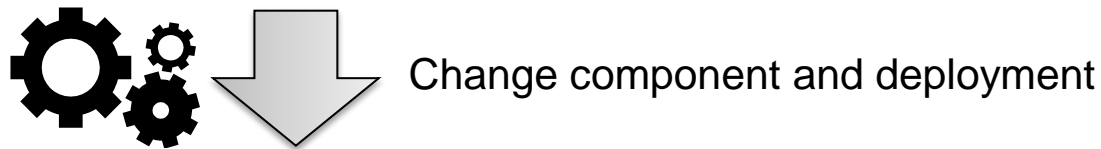
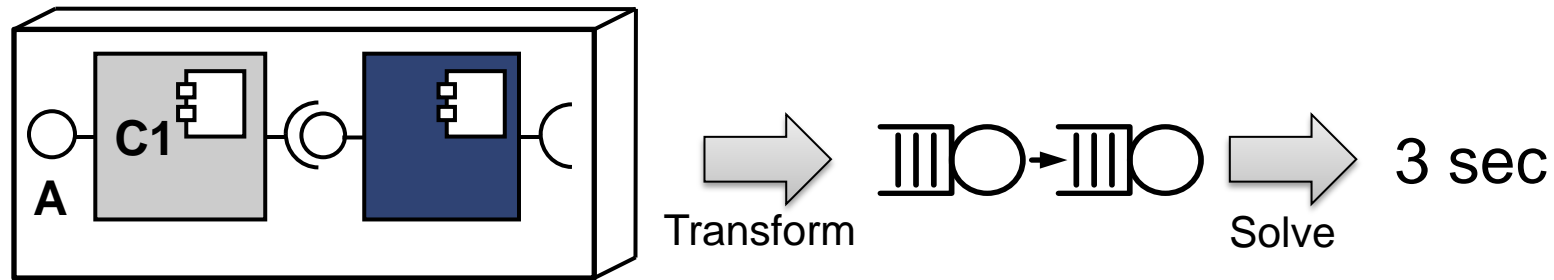
**Anne Martens**

**Karlsruhe Institute of Technology (KIT), Germany  
Heiko Koziolk, Steffen Becker, Ralf Reussner**

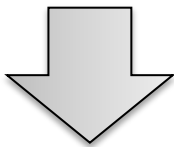
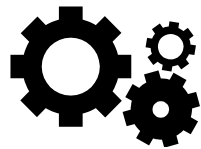
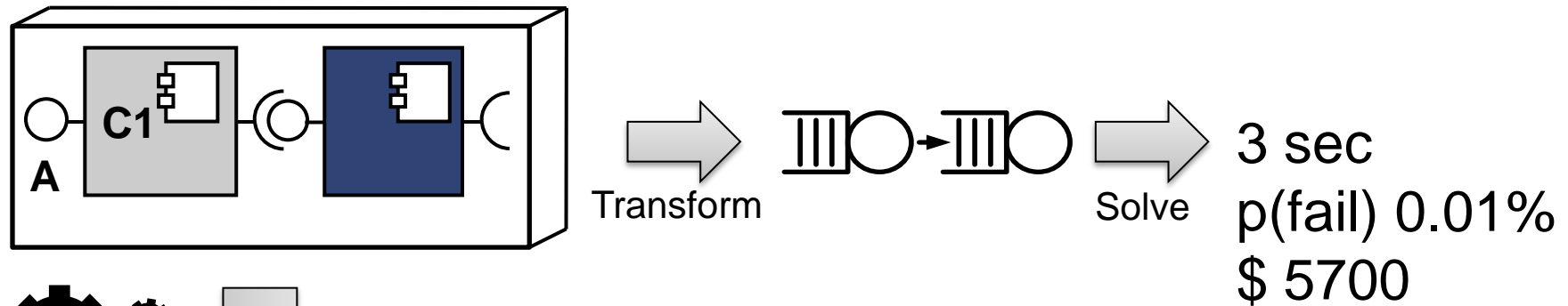
**WOSP / SIPEW 2010**

**An animated version of this slide set (as PowerPoint ppsx)  
can be found at <http://sdqweb.ipd.uka.de/wiki/PerOpteryx>**

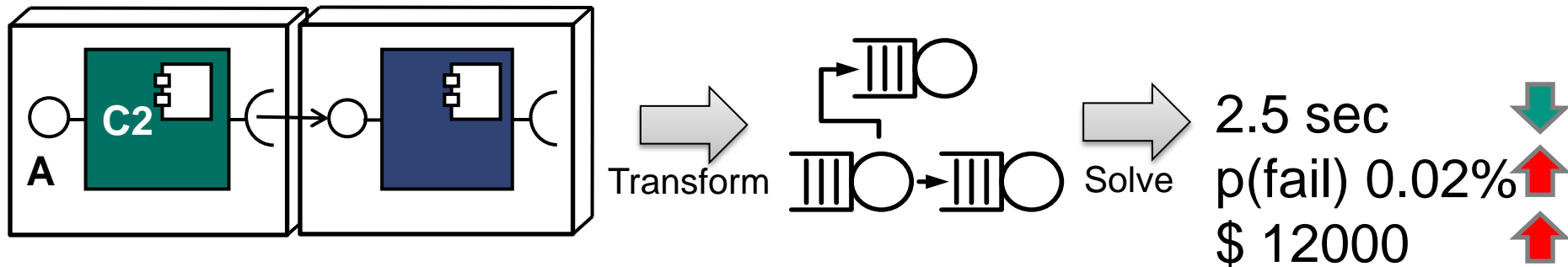
# Software Performance Engineering



# Not only Performance!

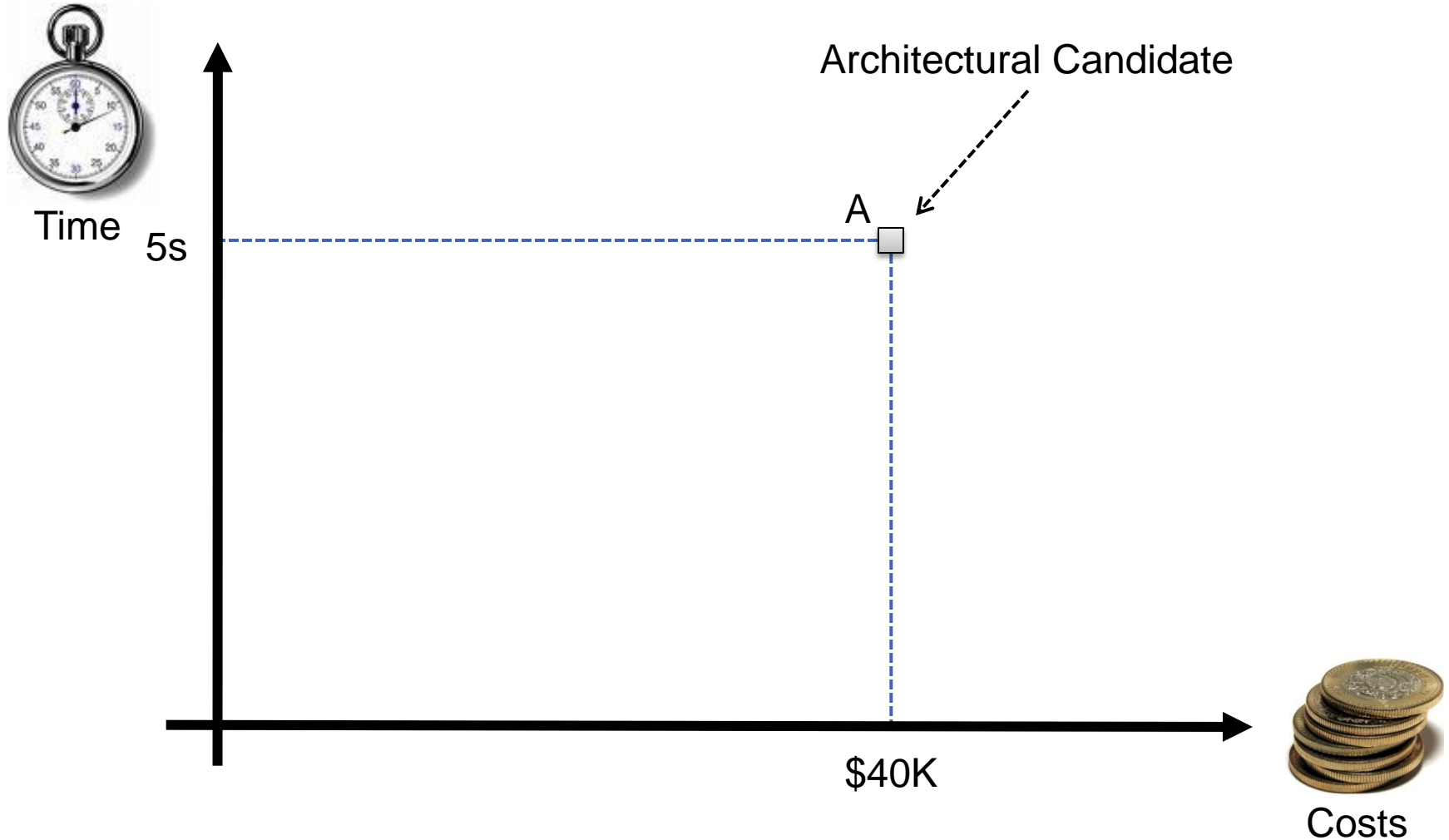


Change component and deployment



→ **Optimise multiple criteria at once**

# Multicriteria Optimisation

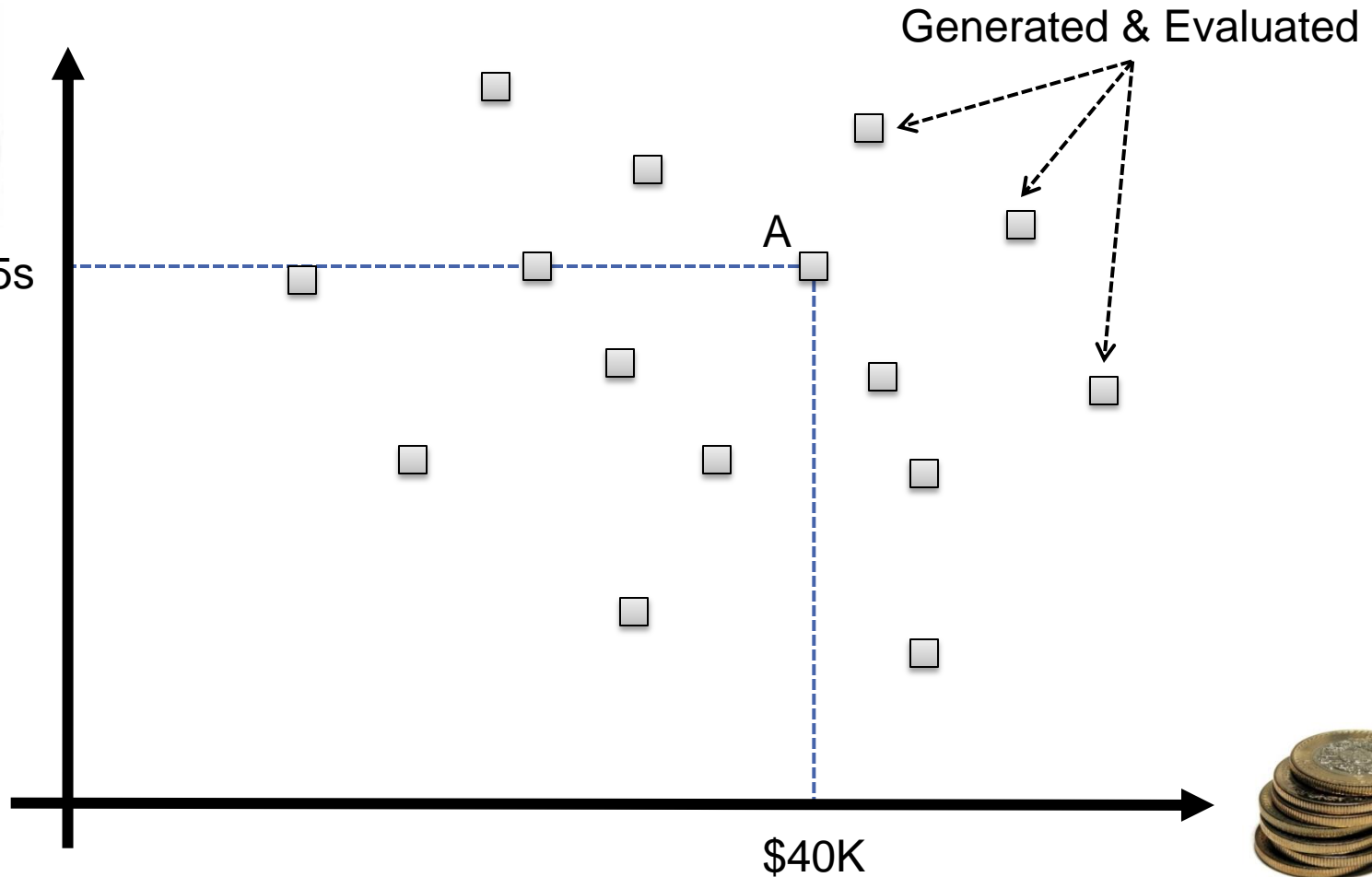


Motivation – Related Work – Approach – Case Study – Future Work – Conclusion

# Multicriteria Optimisation



Time  
5s



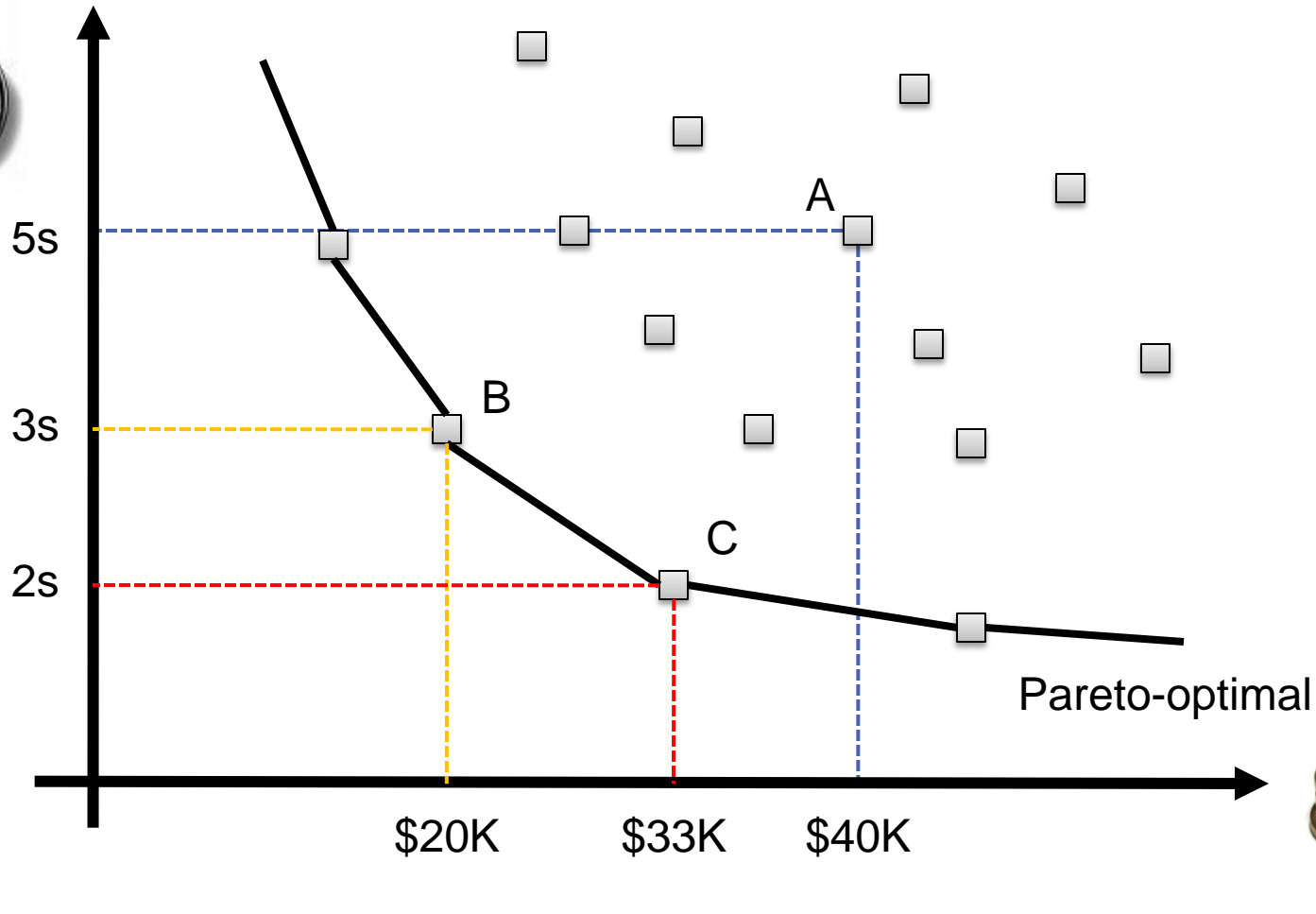
Costs

Motivation – Related Work – Approach – Case Study – Future Work – Conclusion

# Multicriteria Optimisation



Time



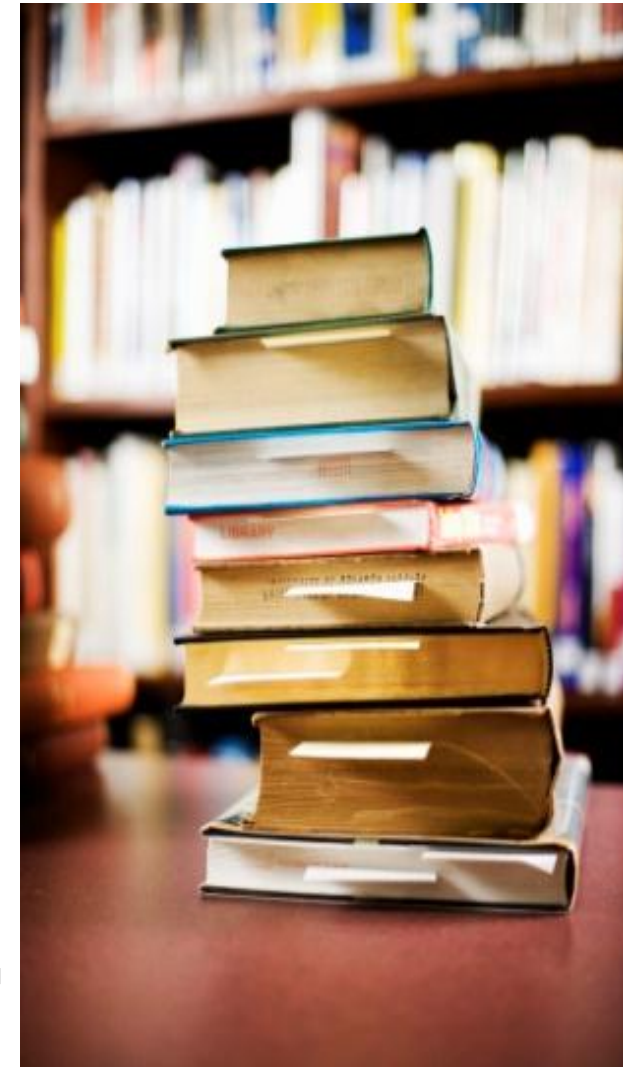
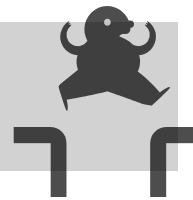
Costs

Motivation – Related Work – Approach – Case Study – Future Work – Conclusion

# Related Work: Quality Optimisation

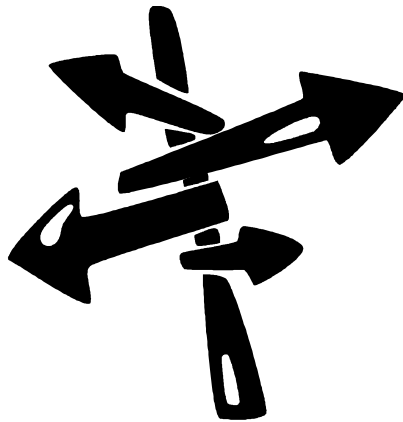
- Rule-based approaches: Single quality only
  - Parsons2008, Cortellessa2009, PerformanceBooster (Xu&Woodside2008), ArchE (McGregor2007)
- Multicriteria evaluation: No improvement
  - Bondarev2007, Grunske2007
- Optimisation: Limited degrees of freedom
  - ArcheOpteryx (Aleti2009), Canfora2005, Kavimandan2009, Sassy (Menascé2010)

**Missing: Flexible multicriteria optimisation at the design level**



# PerOpteryx Approach

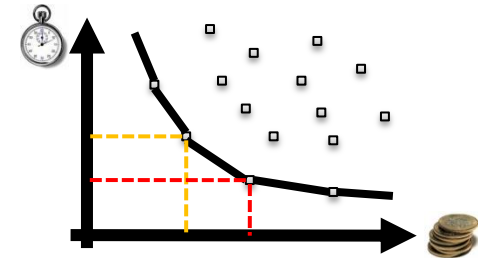
Flexible  
degrees  
of freedom



Multiple  
qualities



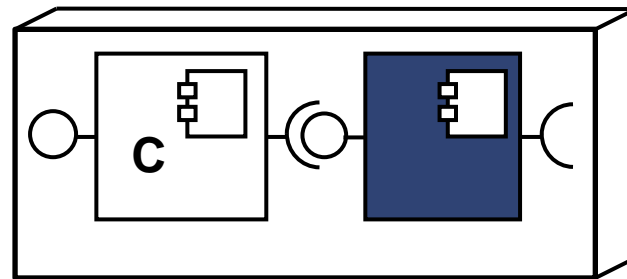
Multi-criteria  
optimization





# Degrees of Freedom

## Design decision that can still be made



<b>Variation point</b>	<b>Which instance to use for component type C?</b>	<b>Degree of freedom</b>
<b>Range of options</b>	<b>C1, C2, or C3</b>	

# Types of Degrees of Freedom in CBSE

## Software

Component selection

Middleware selection

Component replication

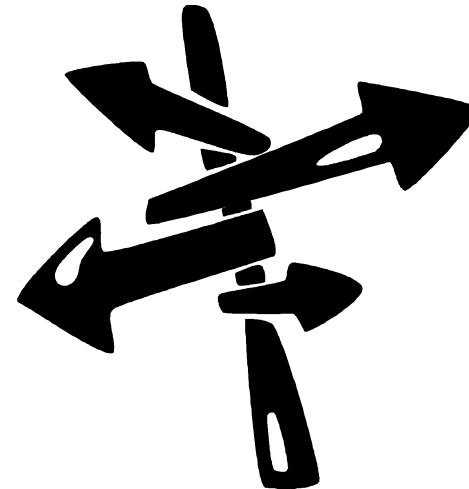
Software configuration

## Deployment

Allocation


Processing Rate

Number of Servers

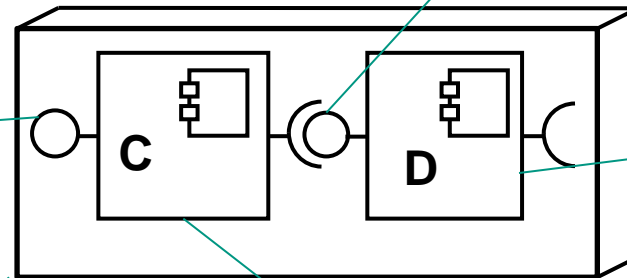


# Instances of Degrees of Freedom

Degree	Matching Rule
Allocation	Each component
Processor speed	Each server
Component selection	Search alternatives
...	...


**Component selection for D**

**Component selection for C**



**Allocation of D**

**Processor speed of server 1**

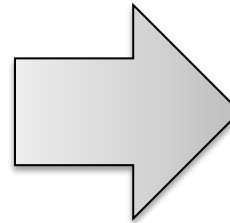
**Allocation of C**



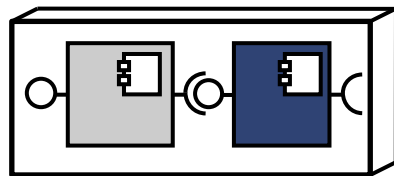
# Search Problem

Degree	Choice
Component selection C	C2
Allocation C	Server1
Speed server 1	2 GHz
...	...

evaluate



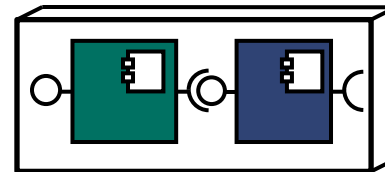
**Response in 2.5 s**  
**P(failure) 0.02%**  
**Cost \$6000**



transform

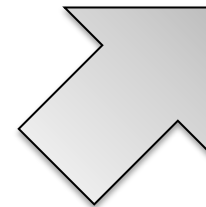


initial model

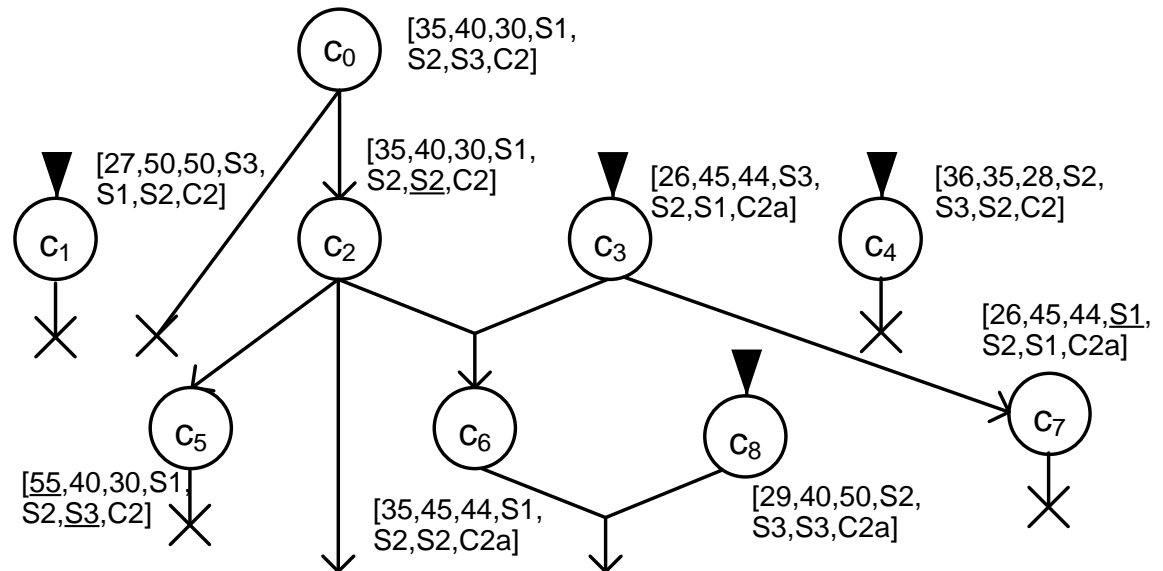
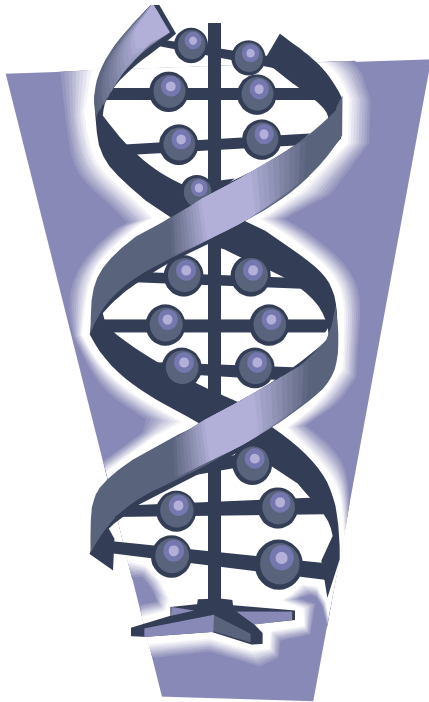


candidate model

evaluate



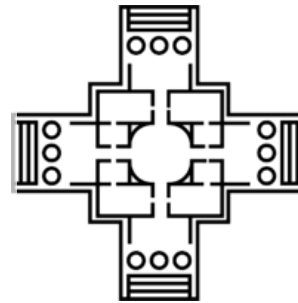
# Search Implementation



**NSGA-II**  
[Deb2002]

# Quality evaluation

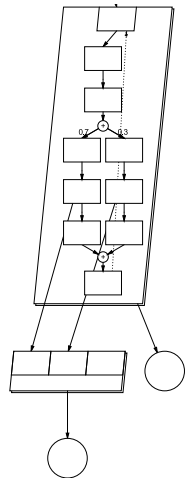
Palladio  
Component  
Model  
[Becker2007]



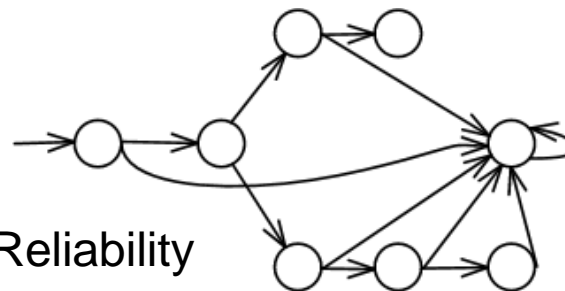
PCM2LQN  
[Koziolek2008]

PCM2DTMC  
[Brosch2009]

PCM2Cost  
[Martens2010]



Performance

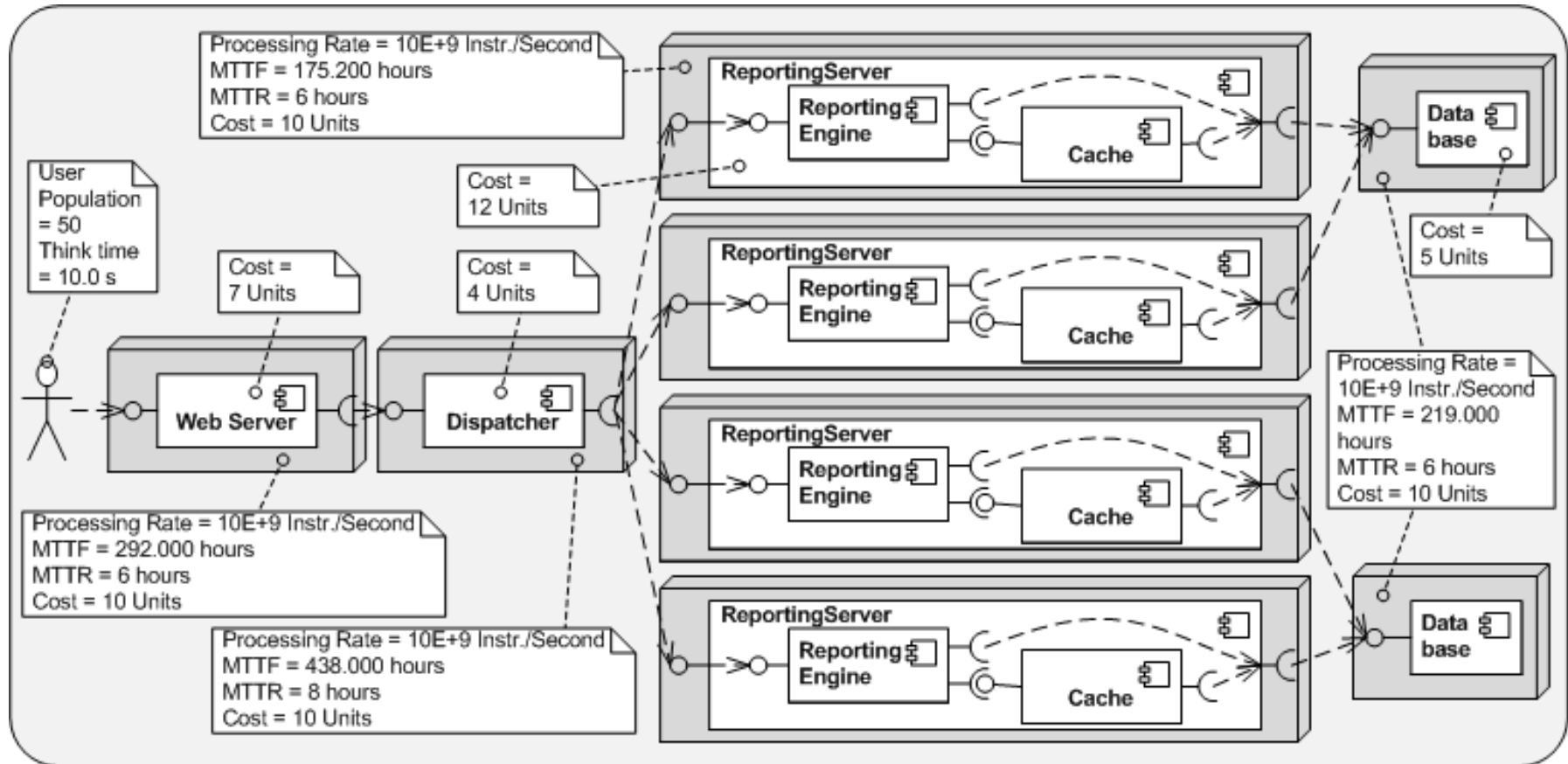


Reliability

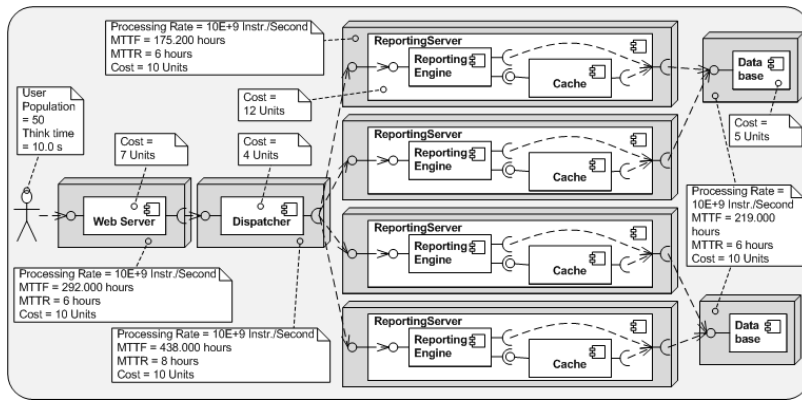
$\Sigma$

Cost

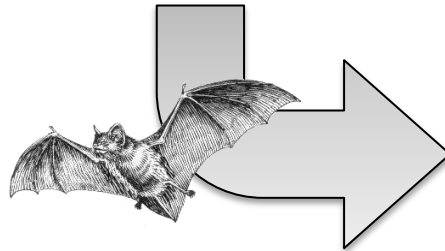
# Case Study with PerOpteryx (1/2)



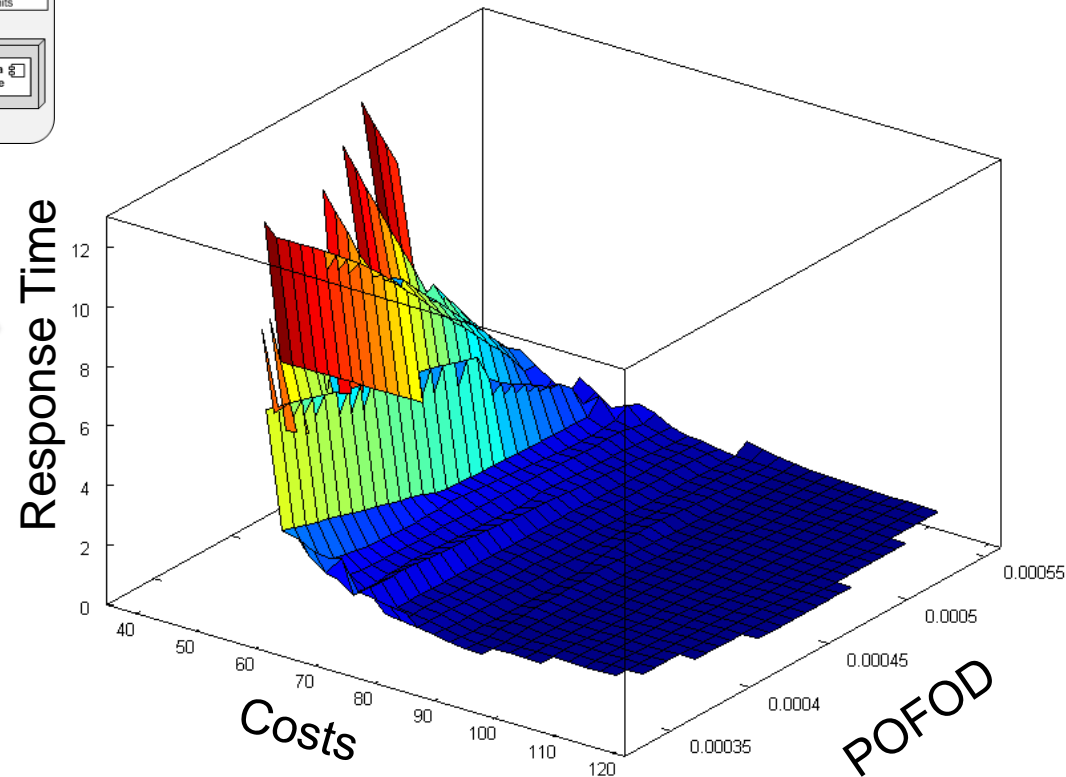
# Case Study with PerOpteryx (1/2)



- Component allocation
- Processing rates
- Component selection

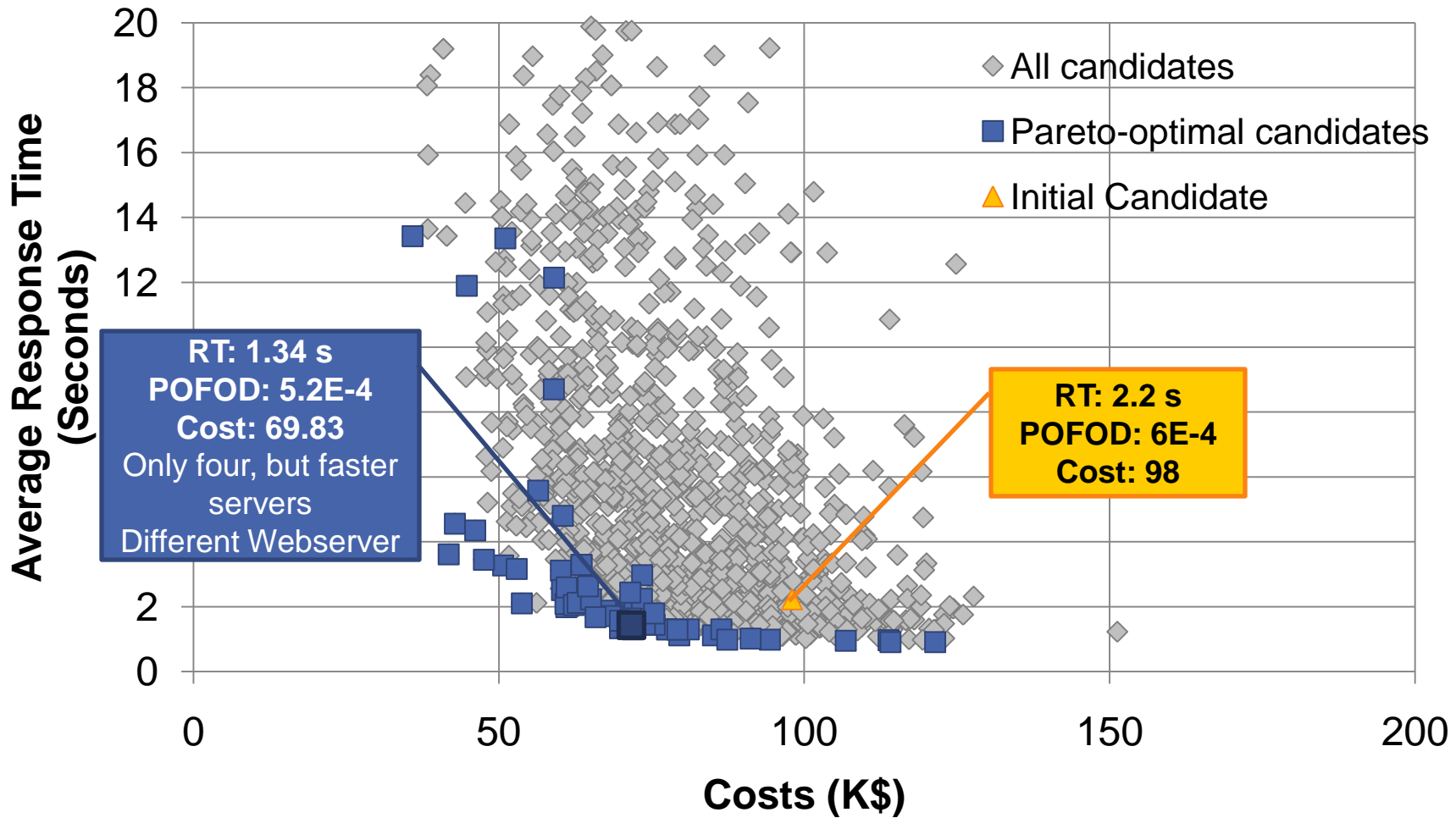


- 1235 candidates
- 58 Pareto optimal
- 8h running time





# Case Study with PerOpteryx (2/2)



# Future Work



## Short term

- Performance heuristics
- Requirement support
- More degrees of freedom



## Long term

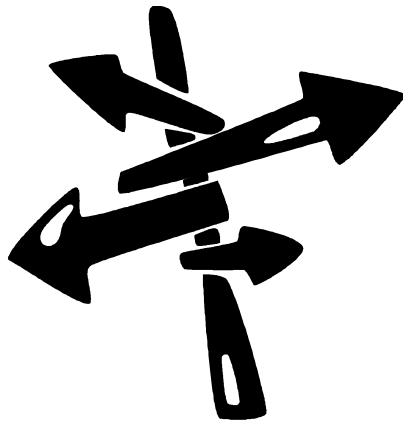
- Handle uncertainty of predictions
- QoS process integration



# Conclusions

## Automated Architecture Improvement

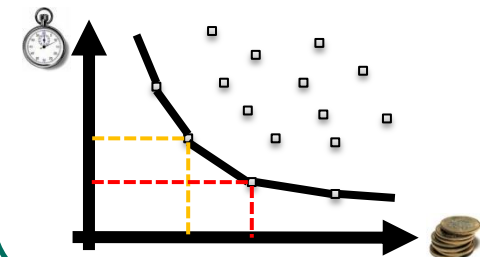
Flexible  
degrees  
of freedom



Multiple  
qualities



Multi-criteria  
Optimization



<http://sdqweb.ipd.kit.edu/wiki/PerOpteryx>