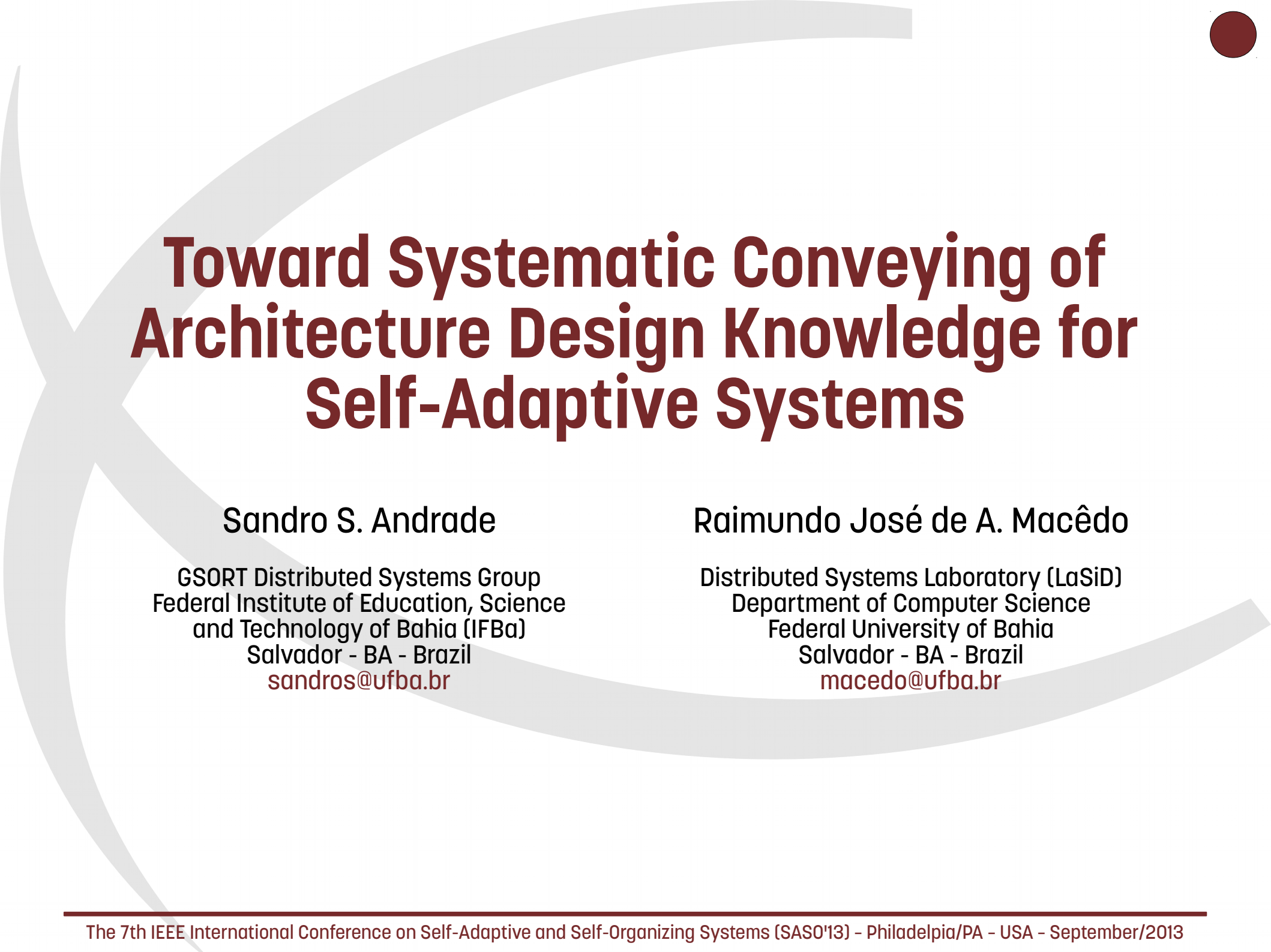




Toward Systematic Conveying of Architecture Design Knowledge for Self-Adaptive Systems

Sandro S. Andrade

GSORT Distributed Systems Group
Federal Institute of Education, Science
and Technology of Bahia (IFBa)
Salvador - BA - Brazil
sandros@ufba.br

Raimundo José de A. Macêdo

Distributed Systems Laboratory (LaSiD)
Department of Computer Science
Federal University of Bahia
Salvador - BA - Brazil
macedo@ufba.br

Pole
placement ?



Highly specialized domain

Lack of support for
well-informed trade-off
analysis

Self-Adaptive
Systems

Effective
solutions highly
tailored to specific problems

Large design/solution
space

Pole
placement ?

Root locus ?

Highly specialized domain

Lack of support for
well-informed trade-off
analysis

Self-Adaptive
Systems

Effective
solutions highly
tailored to specific problems

Large design/solution
space



A close-up photograph of a man with dark hair, looking directly at the camera with a confused or overwhelmed expression. His hand is running through his hair. In the top right corner, there is a small red circle. A thought bubble is positioned in the upper left area of the image.

What am I supposed
to do with all these
stuff ?

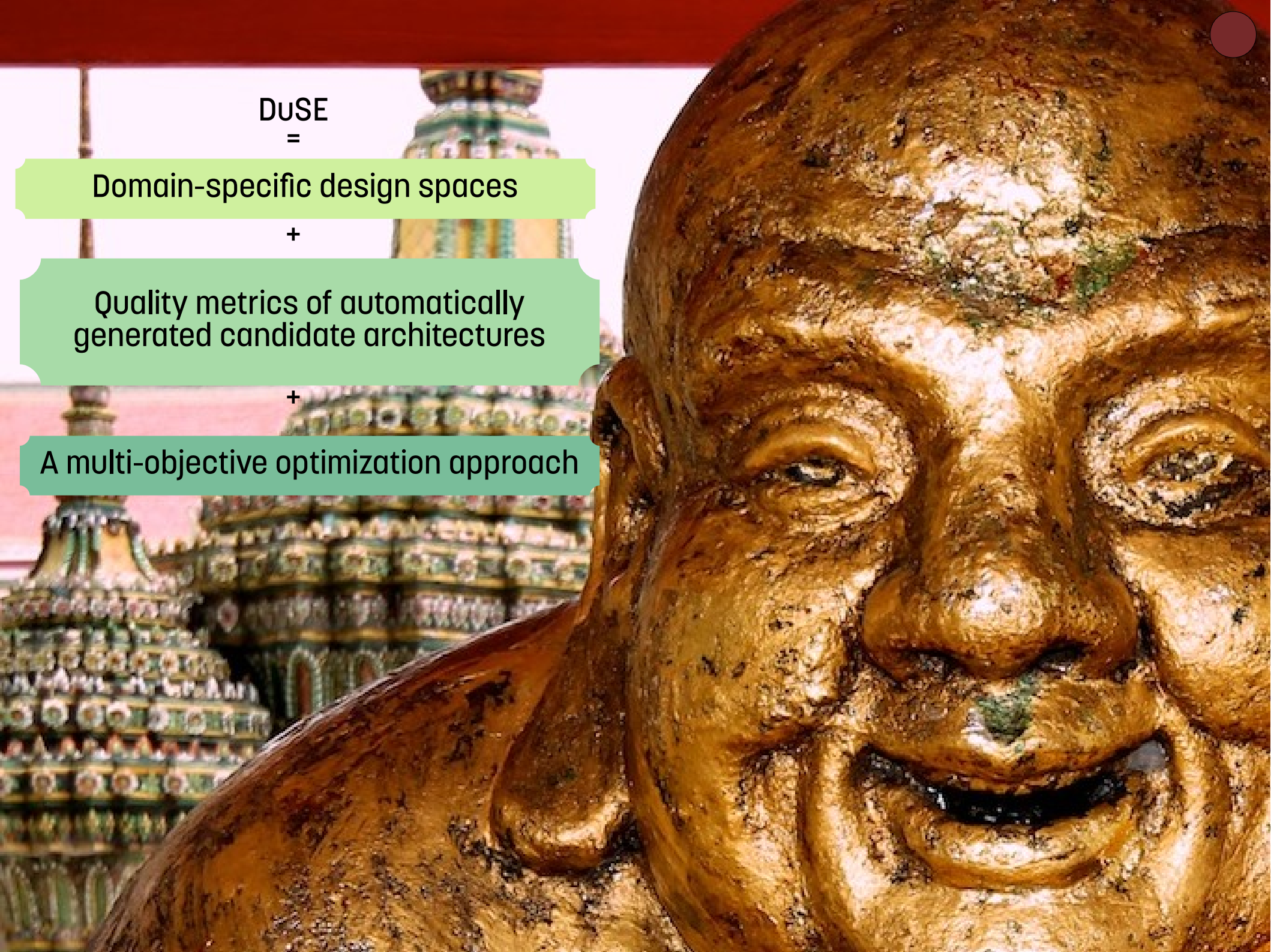
Model-Based Software
Design & Development

Formal Reference
Models / Frameworks

Designing
Self-Adaptive
Systems

Architecture Design
Handbooks

Reference Architectures
Architectural Styles



DuSE
=

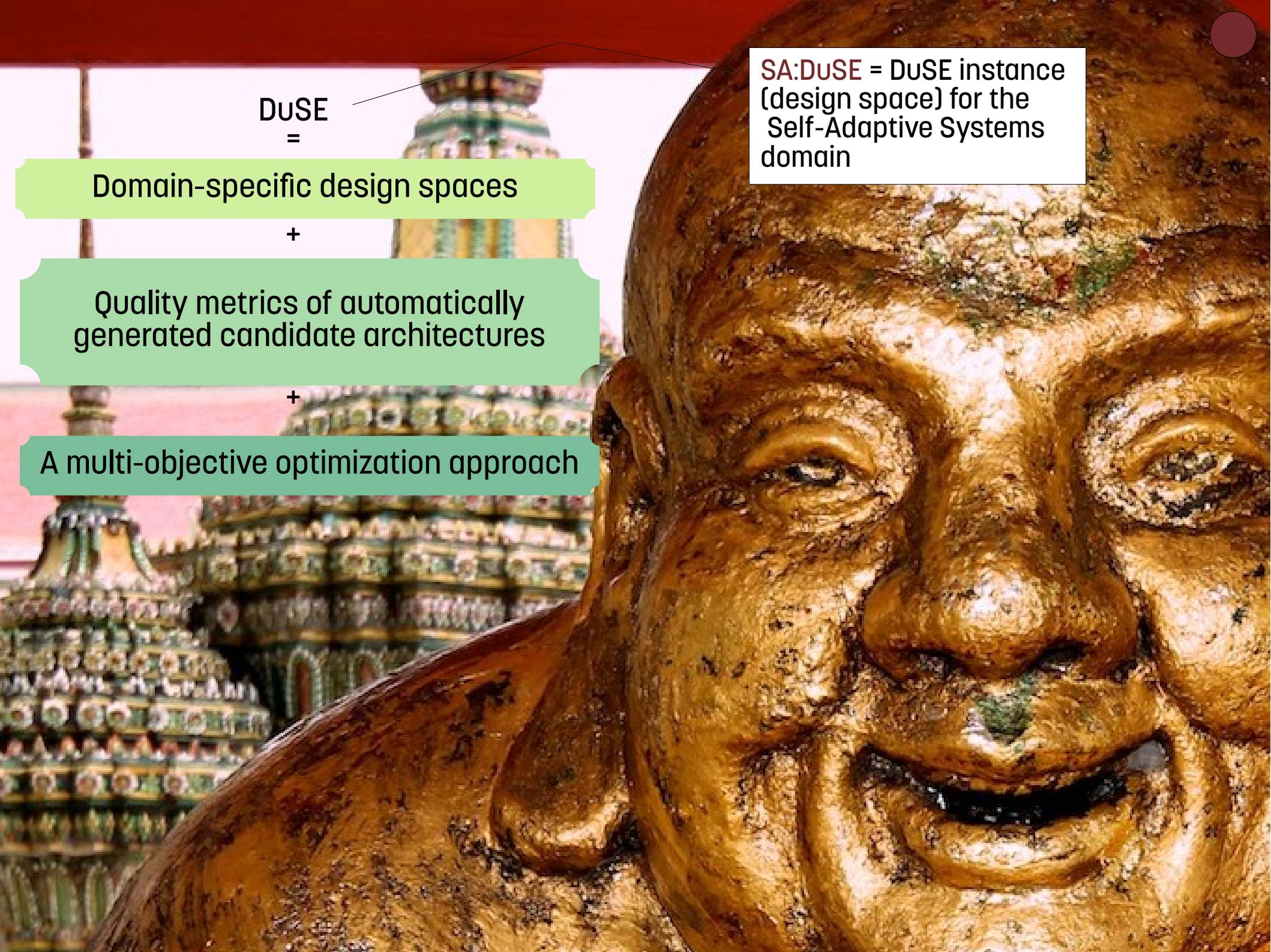
Domain-specific design spaces

+

Quality metrics of automatically
generated candidate architectures

+

A multi-objective optimization approach



DuSE
=

Domain-specific design spaces

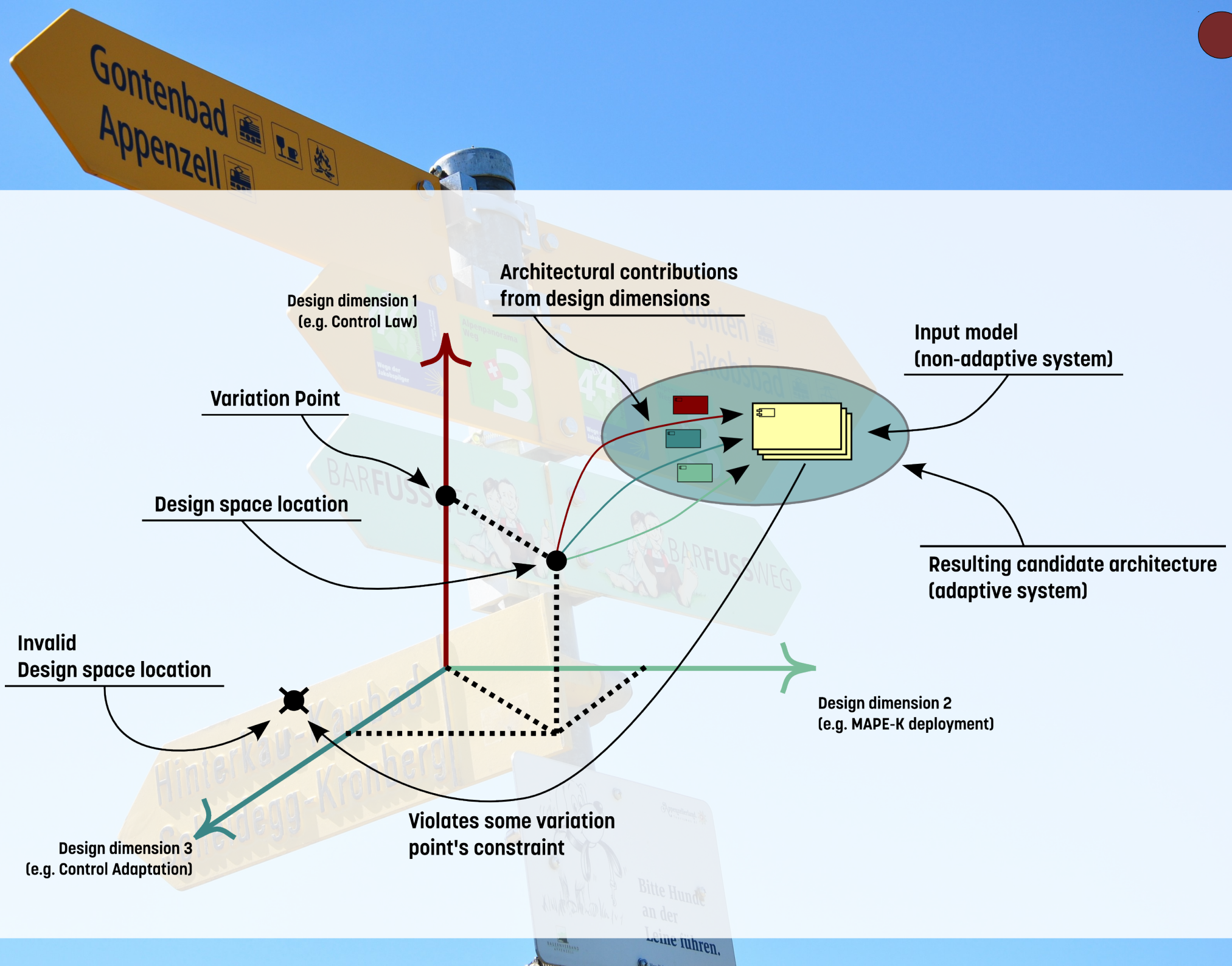
+

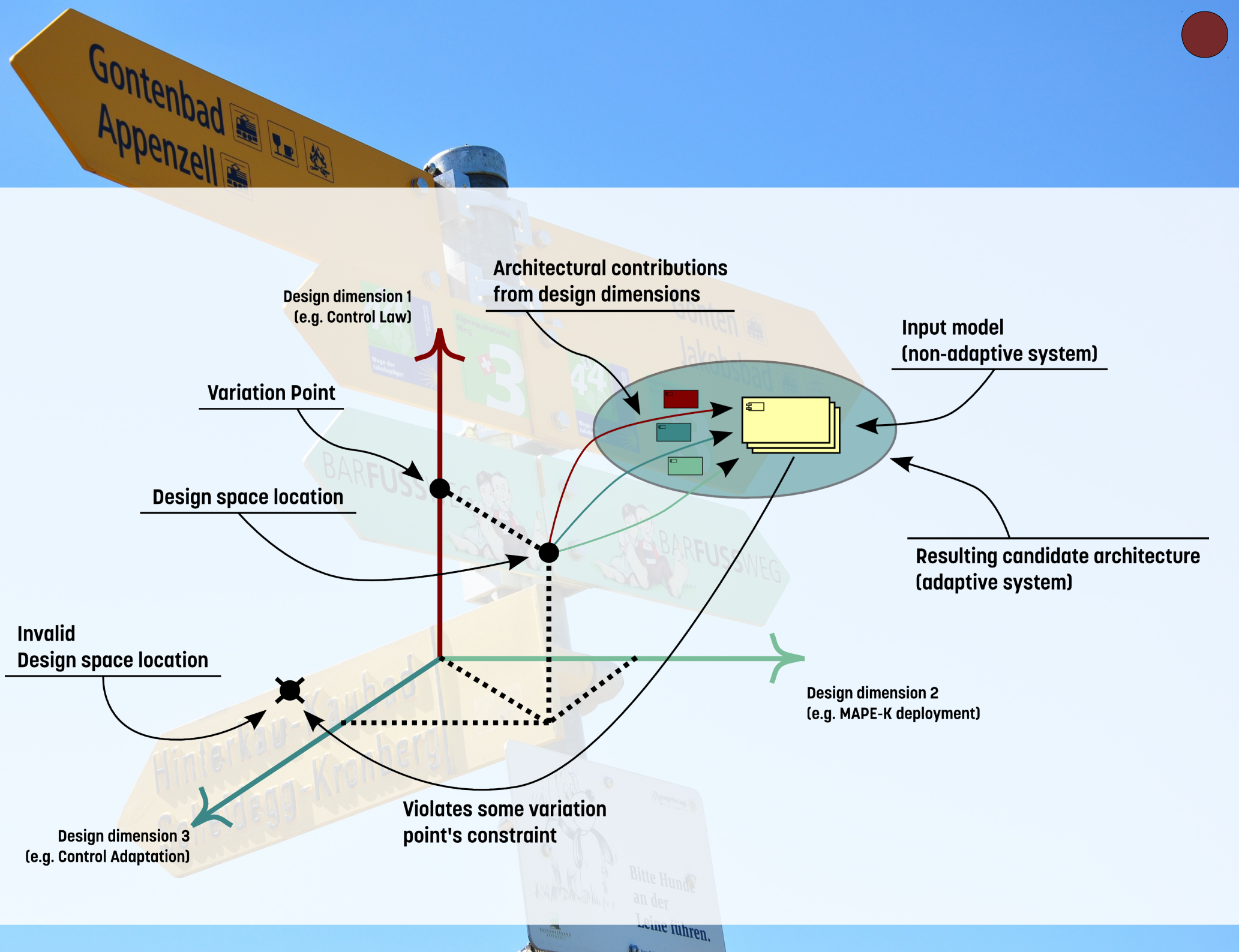
Quality metrics of automatically
generated candidate architectures

+

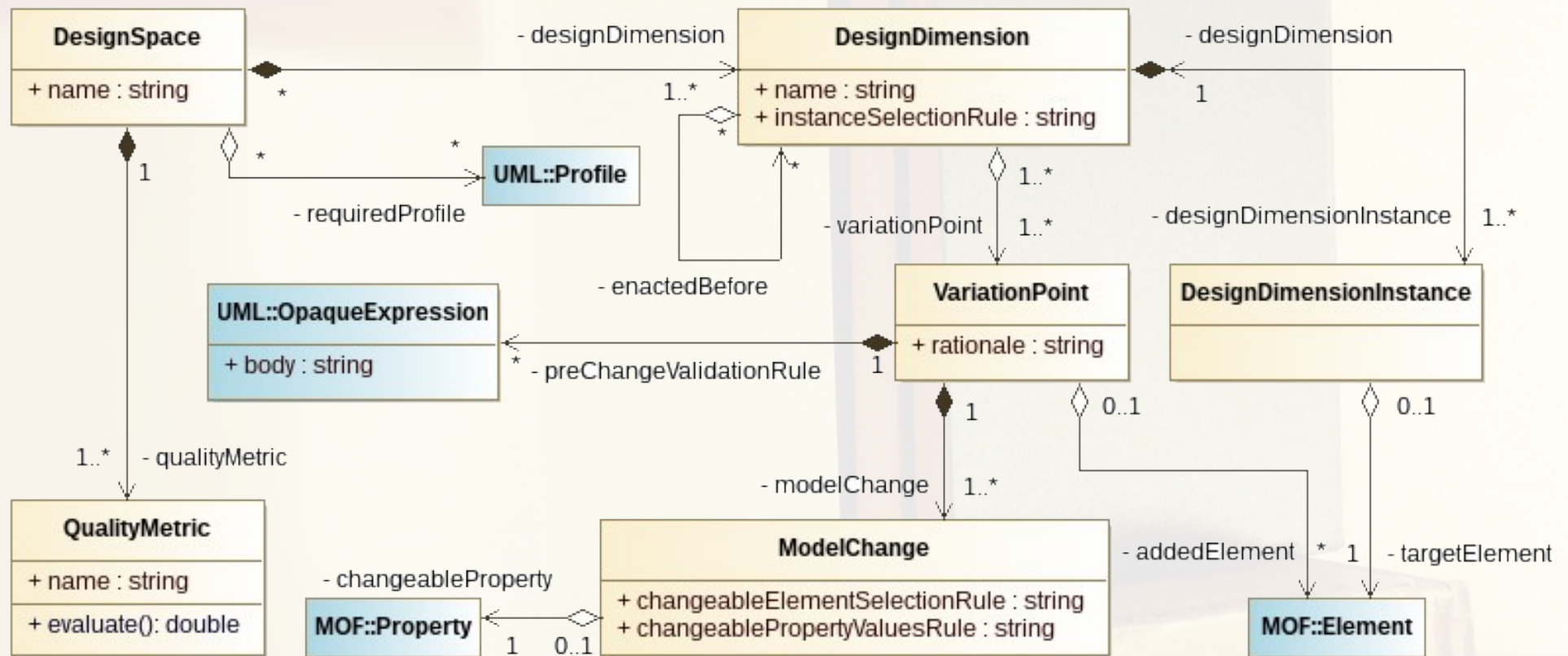
A multi-objective optimization approach

SA:DuSE = DuSE instance
(design space) for the
Self-Adaptive Systems
domain





DuSE Metamodel



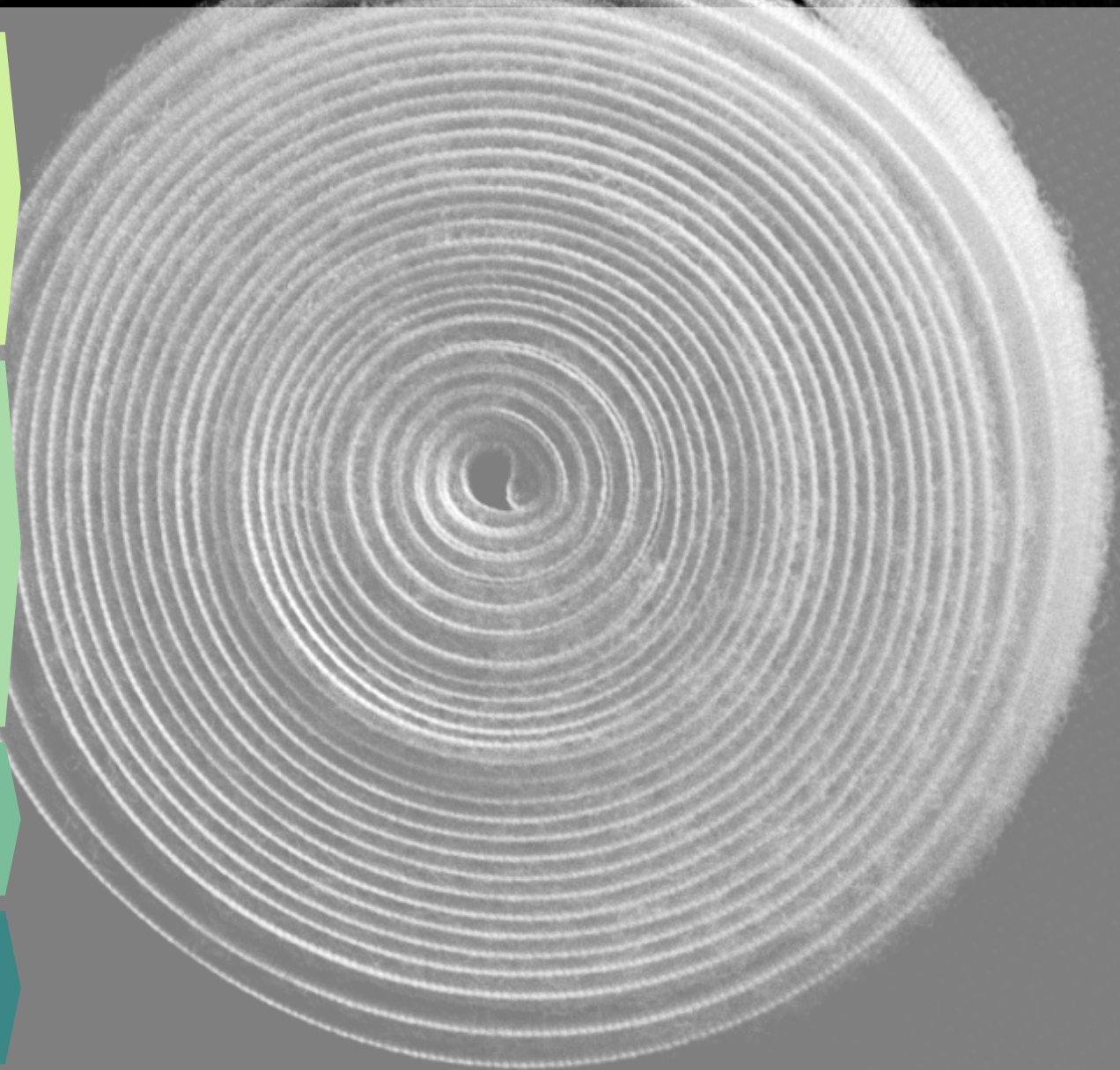
SA:DuSE Design Space

DD₁: Control Law

DD₂: Tuning Approach

DD₃: Control Adaptation

DD₄: MAPE Deployment



SA:DuSE Design Space

DD₁: Control Law

VP11: Proportional
VP12: Proportional-Integral
VP13: Proportional-Integral-Derivative
VP14: Static State Feedback
VP15: Precompensated Static State Feedback
VP16: Dynamic State Feedback

DD₂: Tuning Approach

VP21: Chien-Hrones-Reswick, 0 OS, Dist. Rejection
VP22: Chien-Hrones-Reswick, 0 OS, Ref. Tracking
VP23: Chien-Hrones-Reswick, 20 OS, Dist. Rejection
VP24: Chien-Hrones-Reswick, 20 OS, Ref. Tracking
VP25: Ziegler-Nichols
VP26: Cohen-Coon
VP27: Linear Quadratic Regulator

DD₃: Control Adaptation

VP31: Fixed Gain (no adaptation)
VP32: Gain Scheduling
VP33: Model Identification Adaptation Control

DD₄: MAPE Deployment

VP41: Global Control
VP42: Local Control + Shared Reference
VP43: Local Control + Shared Error

SA:DuSE Quality Metrics

M_1 : Control
Overhead

M_2 : Average
Settling Time

M_3 : Average
Maximum Overshoot

M_4 : Control
Robustness



SA:DuSE Quality Metrics

M_1 : Control
Overhead

$$ME_1 = \frac{\text{allOwnedElements}() \rightarrow \text{selectAsType}(QController) \rightarrow \text{collect}(\text{overhead}()) \rightarrow \text{sum}()}{\text{allOwnedElements}() \rightarrow \text{selectAsType}(QController) \rightarrow \text{size}()}$$

; $QController::\text{overhead}()$ increasingly penalizes VP_{32} , VP_{33} , VP_{41} and VP_{43}

M_2 : Average
Settling Time

$$ME_2 = \frac{\text{allOwnedElements}() \rightarrow \text{selectAsType}(QParametricController) \rightarrow \text{sum}(\text{stime}())}{\text{allOwnedElements}() \rightarrow \text{selectAsType}(QParametricController) \rightarrow \text{size}()}$$

$$; \text{ where } QParametricController::\text{stime}() = \frac{-4}{\log(\max_i |p_i|)}$$

; and $\max_i |p_i|$ is the magnitude of the largest closed-loop pole

M_3 : Average
Maximum Overshoot

$$ME_3 = \frac{\text{allOwnedElements}() \rightarrow \text{selectAsType}(QParametricController) \rightarrow \text{sum}(\text{maxOS}())}{\text{allOwnedElements}() \rightarrow \text{selectAsType}(QParametricController) \rightarrow \text{size}()}$$

$$; \text{ where } QParametricController::\text{maxOS}() = \begin{cases} 0; \text{ real dominant pole } p_1 \geq 0 \\ |p_1|; \text{ real dominant pole } p_1 < 0 \\ r^{\pi/|\theta|}; \text{ dominant poles } p_1, p_2 = r.e^{\pm j.\theta} \end{cases}$$

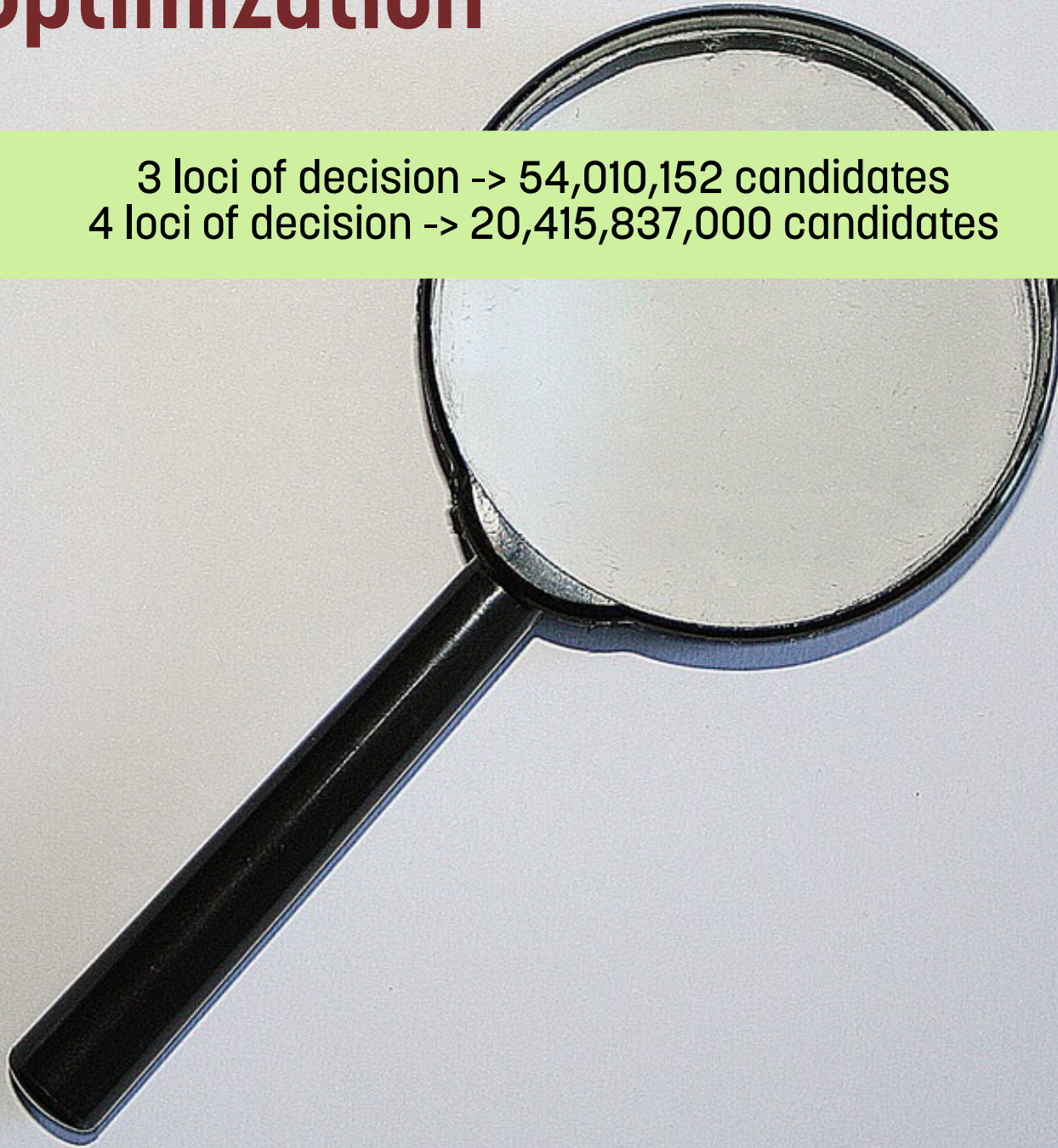
M_4 : Control
Robustness

$$ME_4 = \frac{\text{allOwnedElements}() \rightarrow \text{selectAsType}(QController) \rightarrow \text{collect}(\text{robustness}()) \rightarrow \text{sum}()}{\text{allOwnedElements}() \rightarrow \text{selectAsType}(QController) \rightarrow \text{size}()}$$

; $QController::\text{robustness}()$ increasingly penalizes VP_{31} and VP_{32}

DuSE Optimization

3 loci of decision -> 54,010,152 candidates
4 loci of decision -> 20,415,837,000 candidates



DuSE Optimization

3 loci of decision -> 54,010,152 candidates
4 loci of decision -> 20,415,837,000 candidates

A search-based approach enables effective design space exploration
and helps preventing false intuition and technology bias

```
1: procedure DUSE_OPT( $i, di$ )  
2:           ▷  $i$  = input model;  $di$  = DuSE instance  
3:    $s \leftarrow globalSettings()$   
4:    $dii \leftarrow createAppDSInstance(i, di)$   
5:    $p \leftarrow randomPopulation(dii, s.populationSize)$   
6:    $curIteration \leftarrow 0$   
7:   while  $curIteration \leq s.maxIterations$  do  
8:      $p \leftarrow nsga2Rank(p, dii, s.populationSize)$   
9:      $p \leftarrow p \cup nsga2Mate(p, s.offspringSize)$   
10:     $curIteration \leftarrow curIteration + 1$   
11:  end while  
12:  return  $nsga2highestParetoRank(p)$   
13: end procedure
```

Tool Support (DuSE-MT)

The screenshot displays the DuSE-MT software interface, titled "example2.xmi - DuSE-MT". The interface is divided into several sections:

- Quality Metrics:** A horizontal bar at the top containing four gauges labeled "Control Robustness", "Average Maximum Overshoot", "Average Settling Time", and "Control Overhead".
- Model Inspector:** A tree view on the left showing the model structure. The root is "MyModel" (class "QUmlModel"). It contains "PackageImport.0" (class "QUmlPackageImport") and "Component1" (class "QUmlComponent"). "Component1" has four ports ("port1", "port2", "port3", "port4", all class "QUmlPort") and two properties ("c2", "c3", both class "QUmlProperty"). "Component1" is connected to "Component2" (class "QUmlComponent") via a "connector" (class "QUmlConnector"). "Component2" has a port "port2" (class "QUmlPort") and a property "name" (class "QUmlProperty"). "Component2" is associated with "Component2ProfileA..." (class "QUmlProfileApplication"). "Component2" is also associated with "Component2ProcessC..." (class "QSADuseProfileProc") and "Component1ProcessC..." (class "QSADuseProfileProc"). "PrimitiveTypes (importe..." (class "QUmlPackage") contains "Boolean" (class "QUmlPrimitiveType") and "Integer" (class "QUmlPrimitiveType"). "Boolean" has a property "Boolean-_ownedC..." (class "QUmlComment"). "Integer" has a property "Integer-_ownedC..." (class "QUmlComment").
- Welcome to DuSE-MT:** A central panel with a large "D" logo and the text "Welcome to DuSE-MT !".
- Property Editor:** A panel on the right showing the properties of the selected object. The filter is "MyModel: QUmlModel". The table shows the following properties and values:

Property	Value
objectName	MyModel
ownedElements (RO)	[Component2, _Pa
owner (RO)	
ownedComments (RO)	
name	MyModel
qualifiedName (RO)	MyModel
nameExpression	
namespace (RO)	
clientDependencies (RO)	
- JavaScript Browser:** A panel at the bottom left showing the JavaScript code "self.ownedElements".
- JavaScript Editor:** A panel at the bottom right showing the JavaScript code "self.ownedElements".
- JavaScript Output:** A panel at the bottom right showing the output of the JavaScript code: "QUmlComponent(name = \"Component2\"),QUmlPackageImport(name = \"_PackageImport.0\"),QUmlComponent(name = \"Component1\"),QUmlProfileApplication(name = \"Component2ProfileApplication\")".

The interface also includes a menu bar (File, Edit, Actions, Window, Help) and a toolbar with icons for opening, saving, and closing files. At the bottom, there are tabs for "Issues", "XPath Browser", "OCL Browser", and "JavaScript Browser".

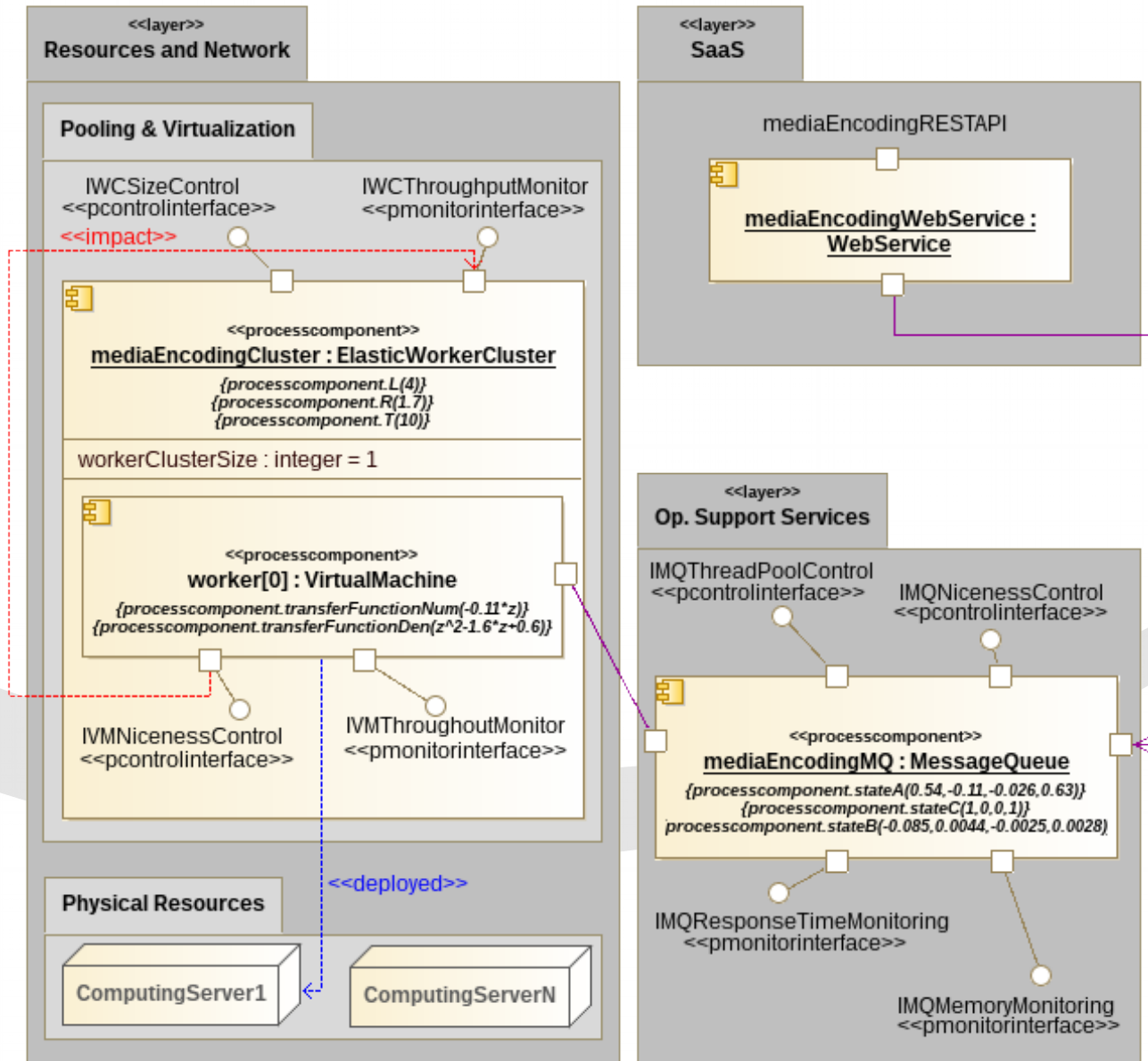
Case Study

Cloud-based media encoding service

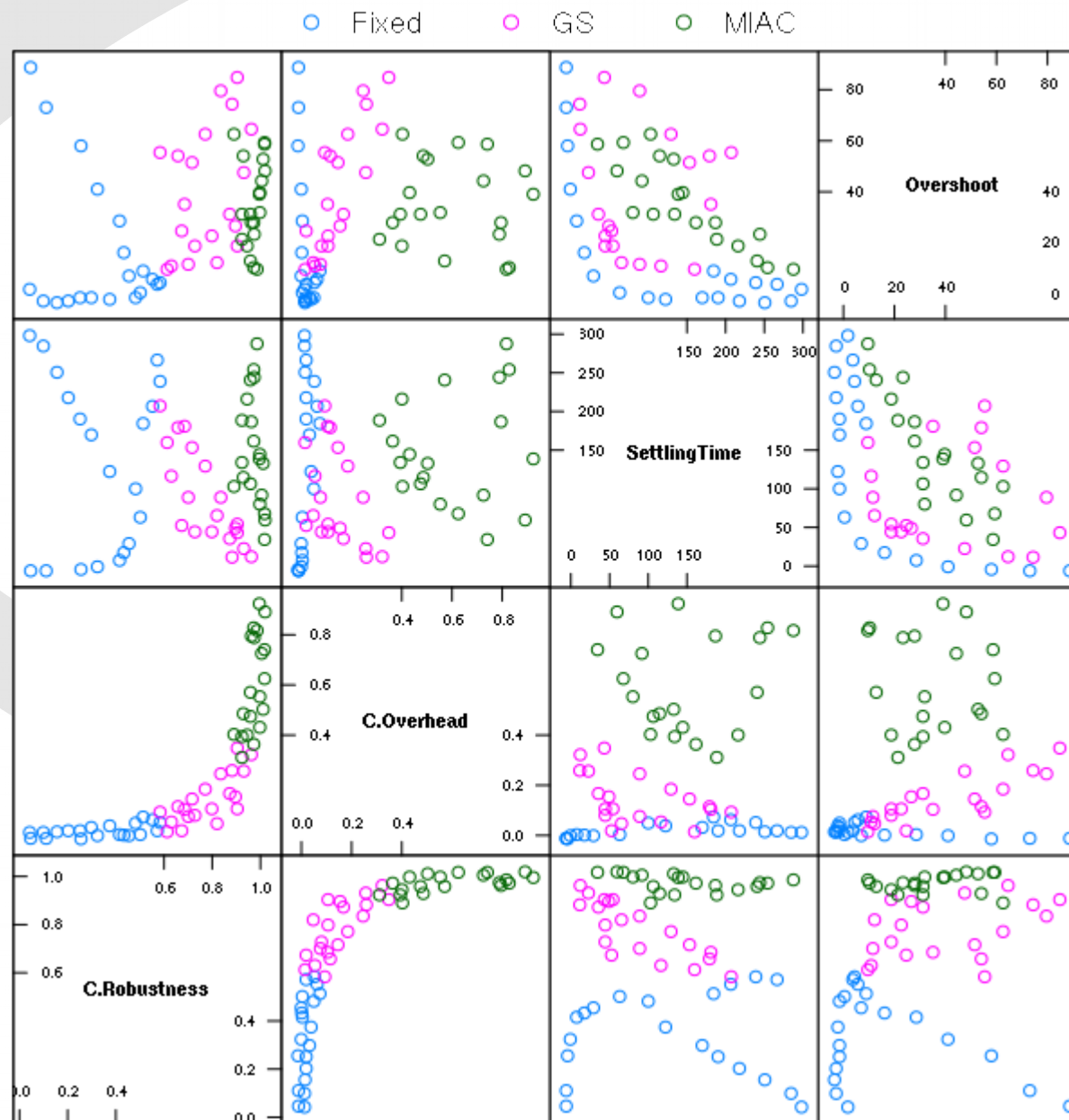
Three loci of decision (controllable components)

Annotations from Qemu + Hadoop + CloudStack experiments

Control goal: enforce encoding throughput



Findings



Conclusion

Contributions

Systematic gathering of architecture design knowledge in the field of Self-Adaptive Systems

A search-based approach for endowing architectures with self-adaptative behaviour and explicit support for well-informed design trade-off analysis

A supporting tool (DuSE-MT)

Conclusion

Contributions

Systematic gathering of architecture design knowledge in the field of Self-Adaptive Systems

A search-based approach for endowing architectures with self-adaptive behaviour and explicit support for well-informed design trade-off analysis

A supporting tool (DuSE-MT)

Limitations

Requires an initial annotated architectural model

No guaranteed optimality (local-optimal Pareto fronts)

Still requires *a posteriori* preference articulation

Current & Future Work

Second case study

Resulting Parent front evaluation (indicators)

From Design Spaces to Design Theories