SIEMENS

Configuration Workshop 2015 Vienna, Austria

A Heuristic, Replay-based Approach for Reconfiguration

Alois Haselböck, Gottfried Schenner

A Heuristic, Replay-based Approach for Reconfiguration
Motivation

SIEMENS

Why this paper?

- Describe how to implement reconfiguration with current solver technologies (CSP, ASP)
- Try out a heuristic approach to reconfiguration especially suited for "large-scale" configuration (many component types and instances)

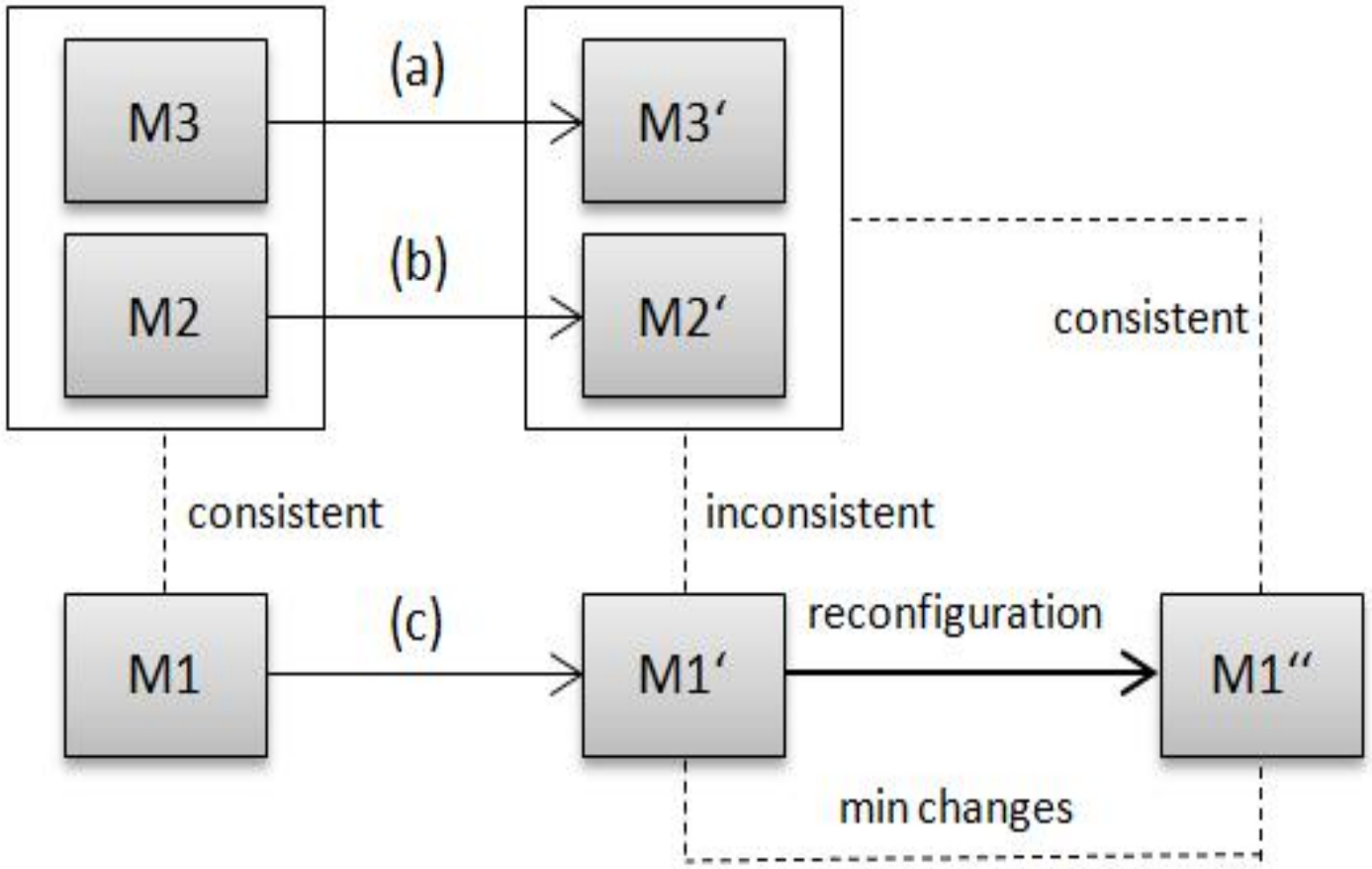
Examples for reconfiguration

- Extension and modification of long-living technical system (interlocking systems, power grid, production systems, etc)
- Integration of new technologies into existing systems (e.g. ETCS European Train Control System)
- Sales configurator (consider existing devices owned by customer)
- Software configuration (package manager, plugin dependencies)

A Heuristic, Replay-based Approach for Reconfiguration
Reconfiguration – Modelling view

SIEMENS

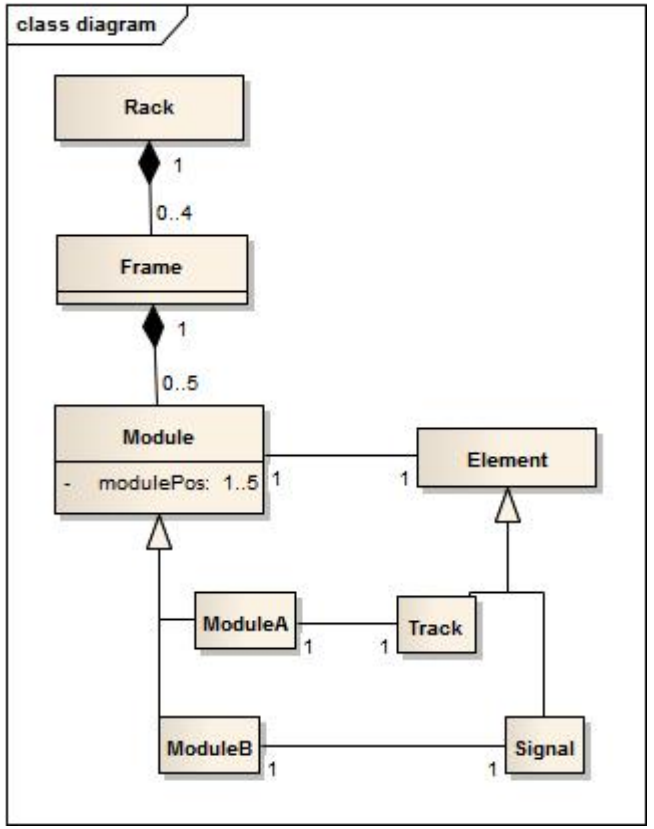
	Levels of models
M4	Modeling language - UML,CSP, ASP…
M3	Problem domain - Generic specification of the component catalogue
M2	Problem Instance - Requirements, specification of a concrete configuration problem
M1	Configuration - Solution to M2 – a consistent configuration



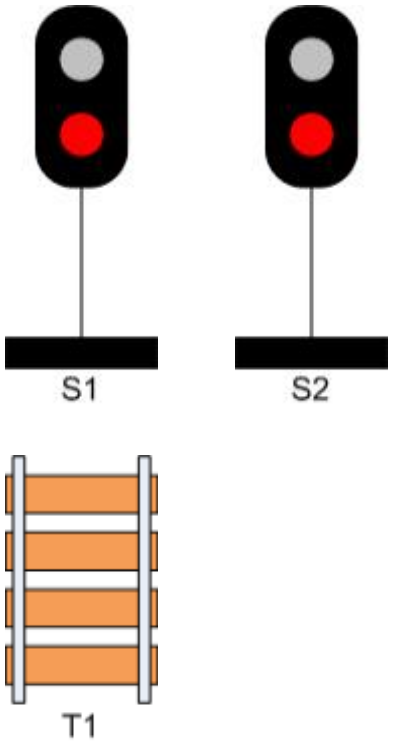
A Heuristic, Replay-based Approach for Reconfiguration
Example Domain Hardware Configuration

SIEMENS

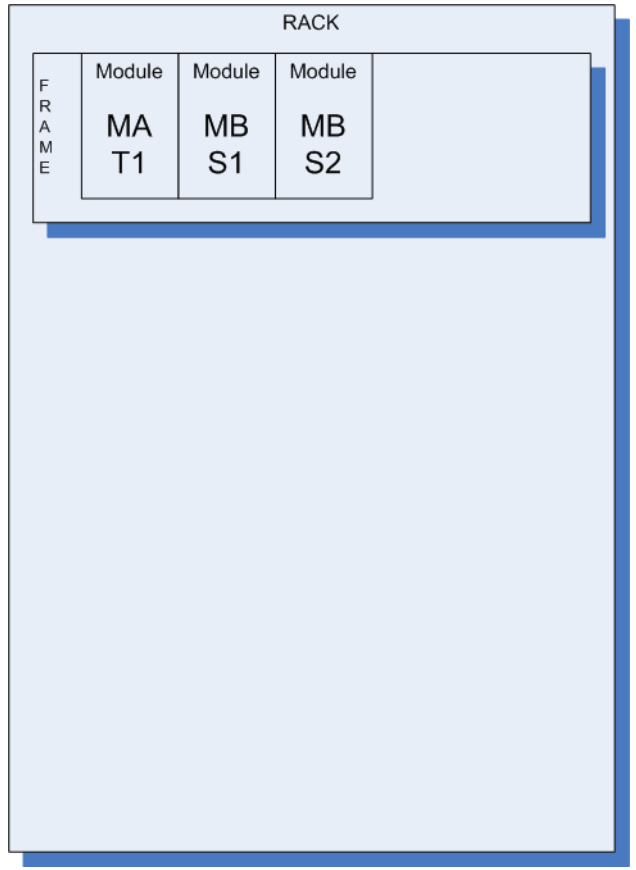
M3 Problem domain



M2 Problem instance



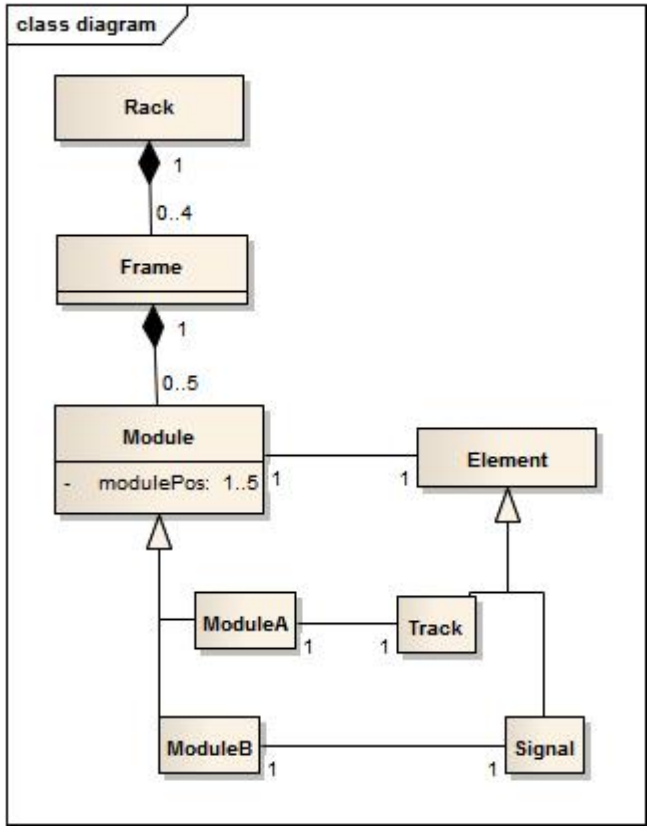
M1 Configuration



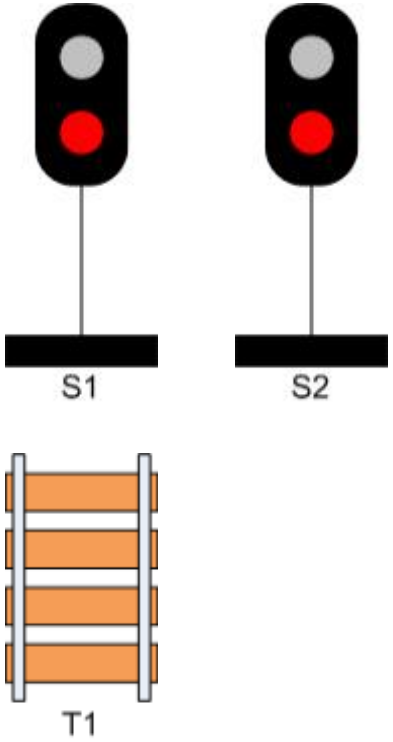
A Heuristic, Replay-based Approach for Reconfiguration
Example Domain Hardware Configuration

SIEMENS

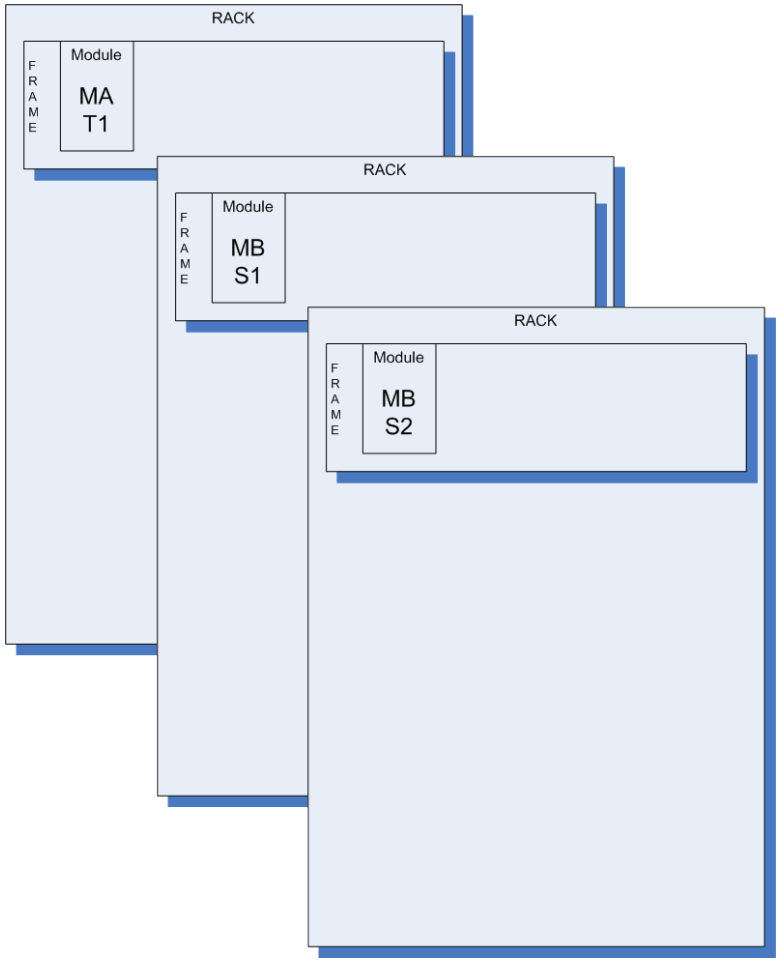
M3 Problem domain



M2 Problem instance



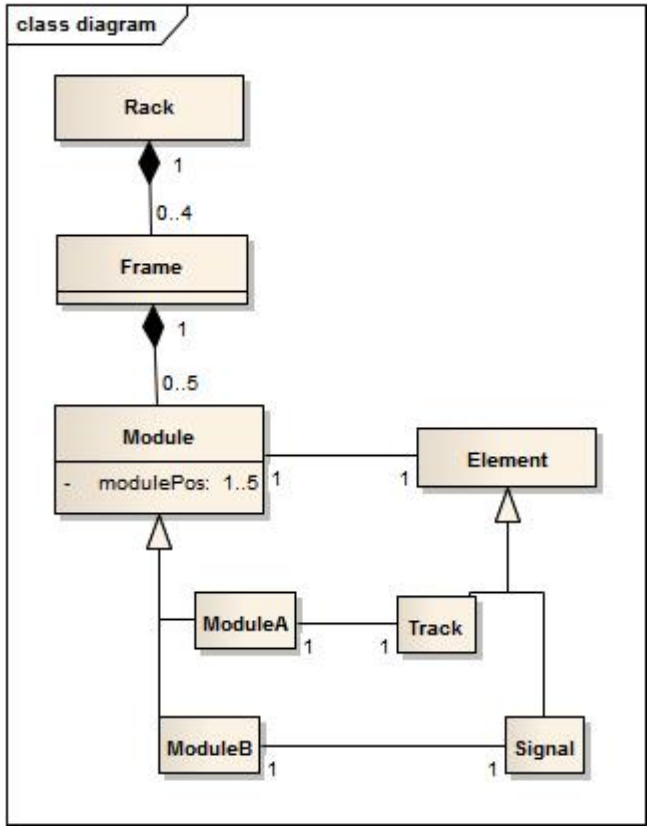
M1 Configuration



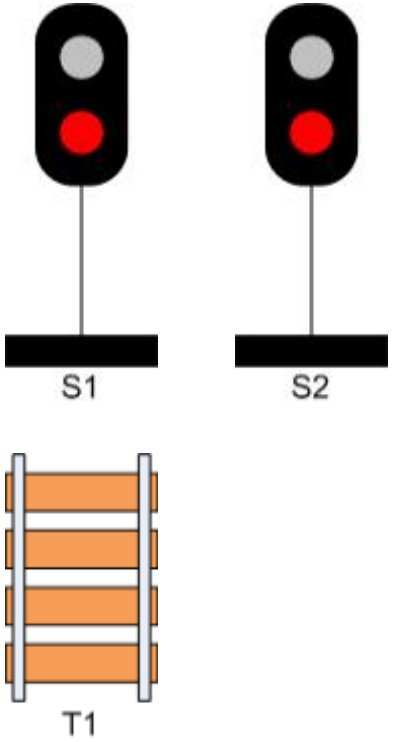
A Heuristic, Replay-based Approach for Reconfiguration
Example Domain Hardware Configuration

SIEMENS

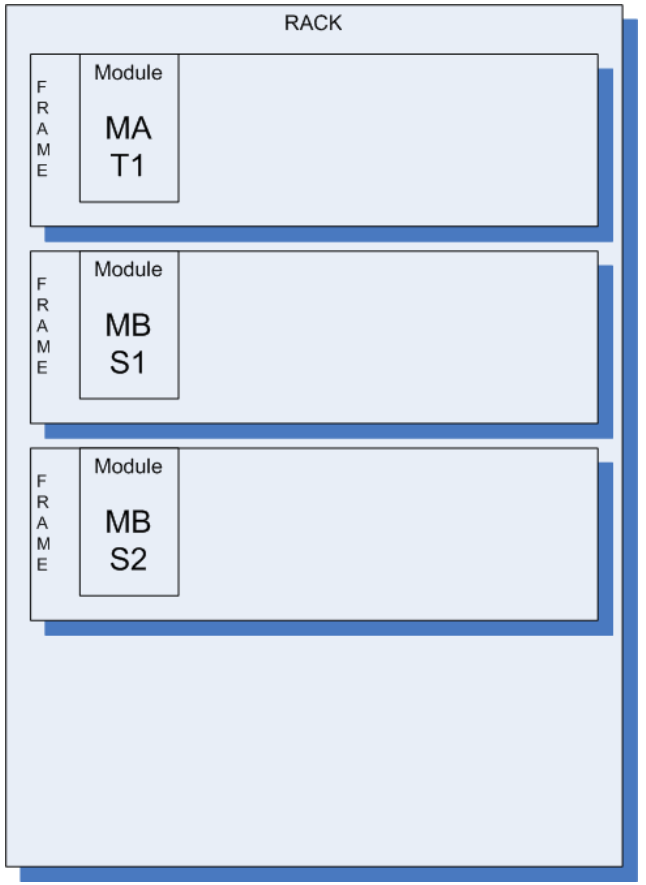
M3 Problem domain



M2 Problem instance



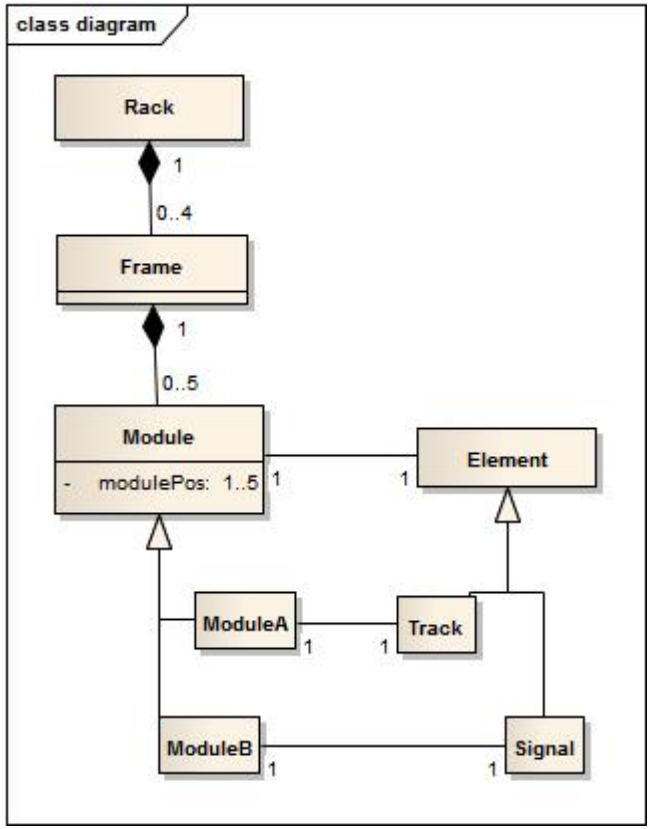
M1 Configuration



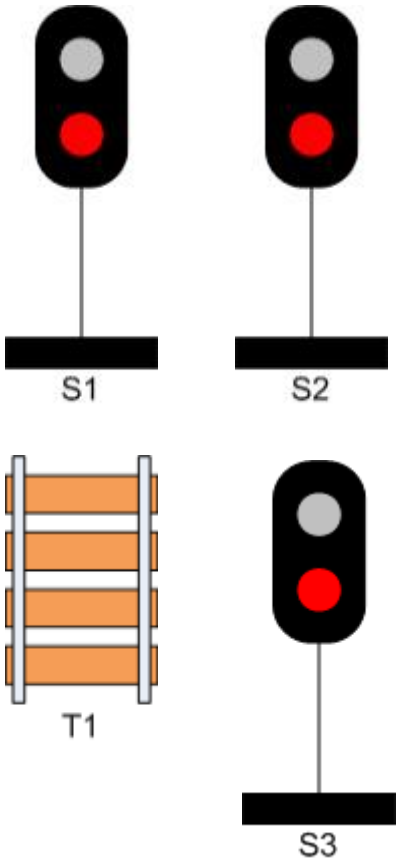
A Heuristic, Replay-based Approach for Reconfiguration
Example Domain Hardware Reconfiguration

SIEMENS

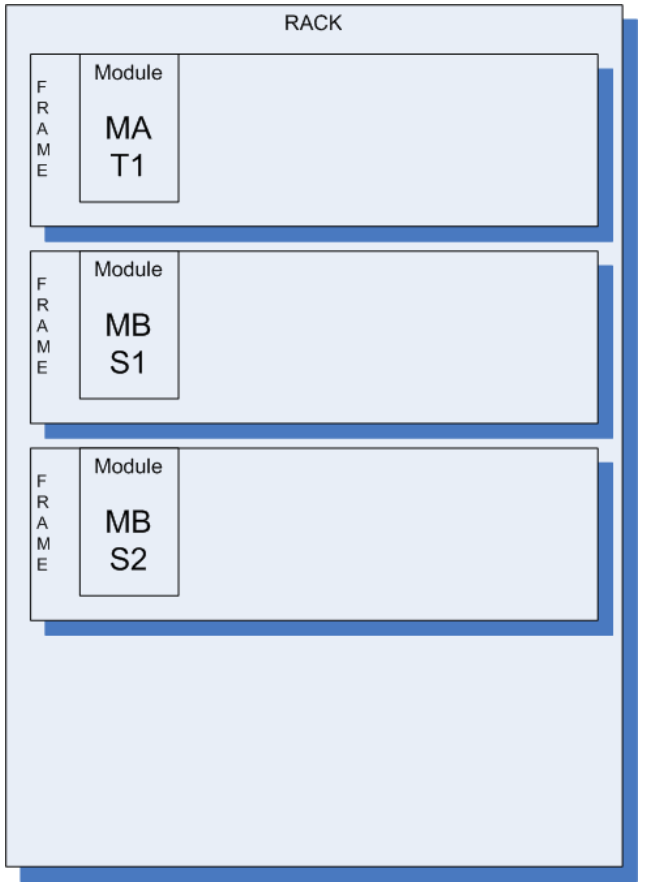
M3 Problem domain



M2' Problem instance



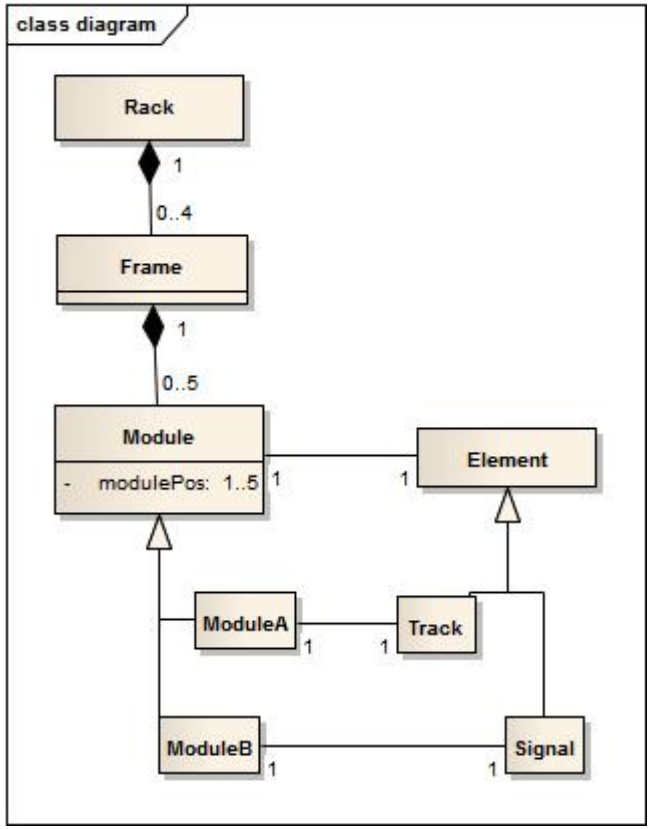
M1' Configuration



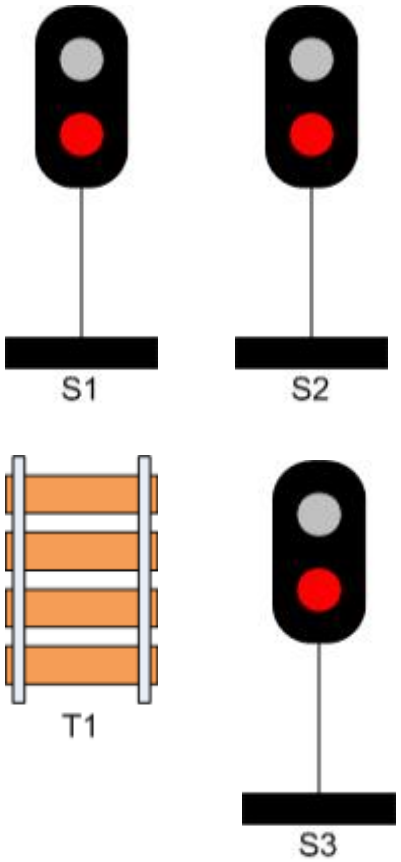
A Heuristic, Replay-based Approach for Reconfiguration
Example Domain Hardware Reconfiguration

SIEMENS

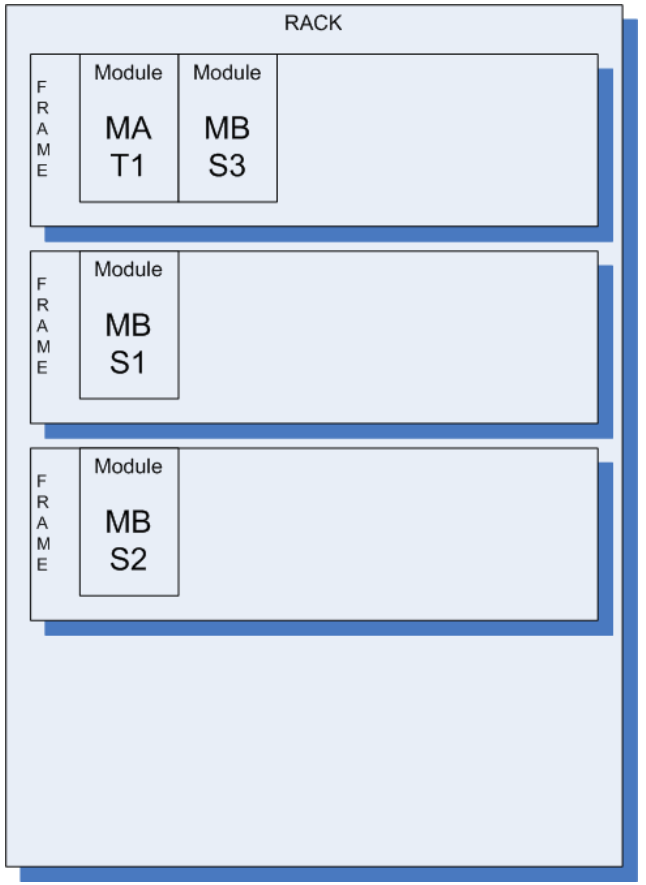
M3 Problem domain



M2' Problem instance



M1'' Configuration



A Heuristic, Replay-based Approach for Reconfiguration
CSP Encoding of object-oriented model

Encoding as static CSP

- Define maximal number of objects MAXOBJX for every class X
- Use integers 1..MAXOBJX as object identifier
- One boolean array for every class, defining which objects are actually used in a configuration
- One integer array for every attribute
- Additional constraints to ensure that unused objects have null-values

```
% MiniZinc example
% encoding of class Module
int: MAXNROFMODULES;
set of int: MODULES = 1..MAXNROFMODULES;

% which object is used
array[MODULES] of var bool: module_used;

% attribute position
% 0 is null value
array[MODULES] of var 0..5 : module_position;

constraint forall(i in MODULES)
        (module_used[i] <-> module_position[i] != 0);
```

A Heuristic, Replay-based Approach for Reconfiguration
CSP Encoding of object-oriented model - Associations

SIEMENS

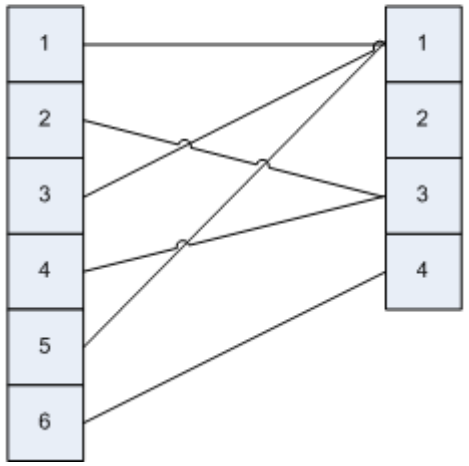
Encode 1-N Associations between class A and B

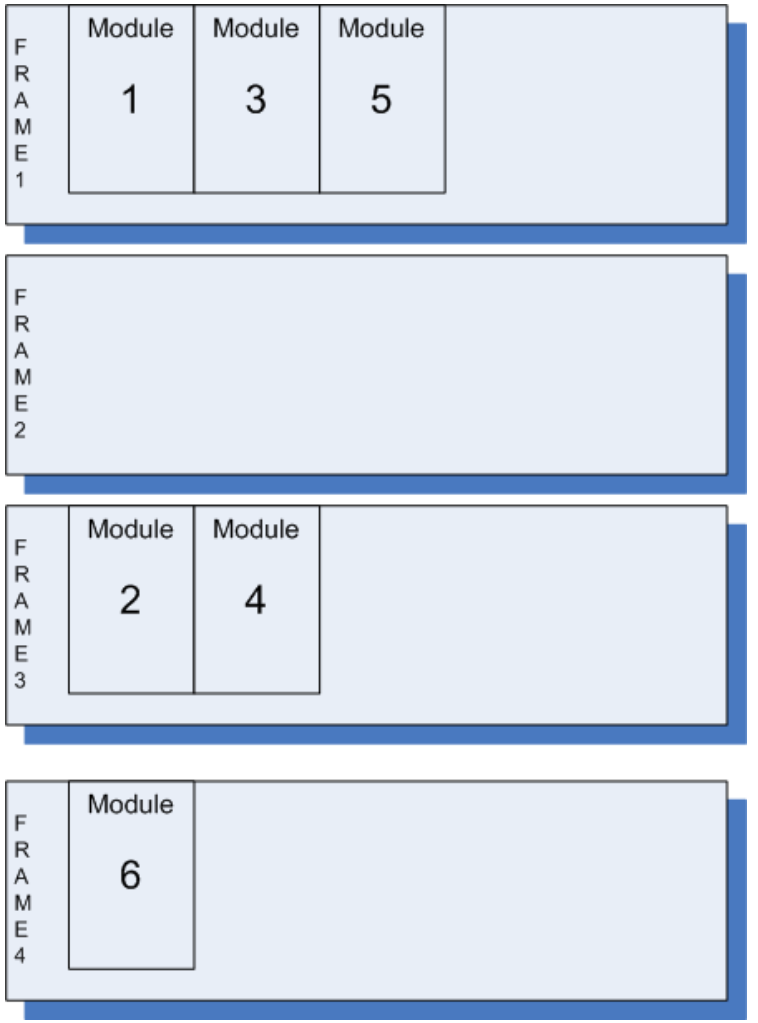
- One integer array for the links from B to A
- Use global cardinality constraint to ensure the cardinality restrictions of the association
- Other encodings possible (set variables, matrix etc.)

```
% Association Frame 1-N Module

array[MODULES] of var FRAMES: module_frame;
array[FRAMES] of var 0..MAXNROFMODULES:
frame_modulecount;

% restrict cardinalities
constraint
global_cardinality(module_frame,
        FRAMES,
        frame_modulecount);

% no frame can reference more than 5 modules
constraint forall (f in FRAMES)
        (frame_modulecount[f]<=5);
```

A Heuristic, Replay-based Approach for Reconfiguration
CSP Encoding of object-oriented model – Associations Example

module_frame = [1, 3, 1, 3, 1, 4]
frame_modulecount = [3,0,2,1]





SIEMENS

Heuristic Reconfiguration for CSP

1. Encode legacy configuration as a (solved) CSP
2. Use this CSP-encoding of legacy configuration to derive value heuristic
3. Solve configuration task with CSP-solver using the value heuristic

- In a way the solver “replays” the legacy configuration
- If the requirements are unchanged, the first solution found by the solver is the same as the legacy configuration
- If changes are small, the solution is expected to be “close” to the legacy configuration
- Implementation in CHOCO, by defining a special IntValueSelector

A Heuristic, Replay-based Approach for Reconfiguration
ASP Encoding of object-oriented model

Encoding uses OOASP

- OOASP is a framework for describing object oriented models in ASP
- New reconfiguration task based on heuristic reconfiguration

```
% OOASP version of hw example
ooasp_class("hw","Frame").
ooasp_class("hw","Module").

ooasp_assoc("hw","Frame_modules",
   "Frame",1,1,
   "Module",0,5).

ooasp_attribute("hw","Module","position","integer").

ooasp_attribute_minInclusive("hw","Module","position",1).
ooasp_attribute_maxInclusive("hw","Module„,"position",5).

ooasp_assoc("hw","Module_element",
  "Module",1,1,
  "Element",1,1).
…
```

SIEMENS

Heuristic Reconfiguration for CSP

1. Encode legacy configuration as ASP
2. Convert facts of ASP-encoding to heuristic facts
3. Solve standard OOASP configuration task with heuristic facts

- Special _heuristic-predicate of clingo
- _heuristic(ATOM,true,LEVEL) influences ASP solver to prefer ATOMs with higher LEVEL
- Only order of solutions (answer sets) is affected

```
ooasp_isa("c","Track",„T1")
% legacy configuration converted to heuristic
_heuristic(
  ooasp_isa("c","ModuleA","M1"),
  true,1).

_heuristic(
  ooasp_attribute_value("c„,"position„,"M1",4),
  true,1).

_heuristic(
  ooasp_associated("c„,"Module_element„,"M1",„T1"),
  true,1).
```

A Heuristic, Replay-based Approach for Reconfiguration Time

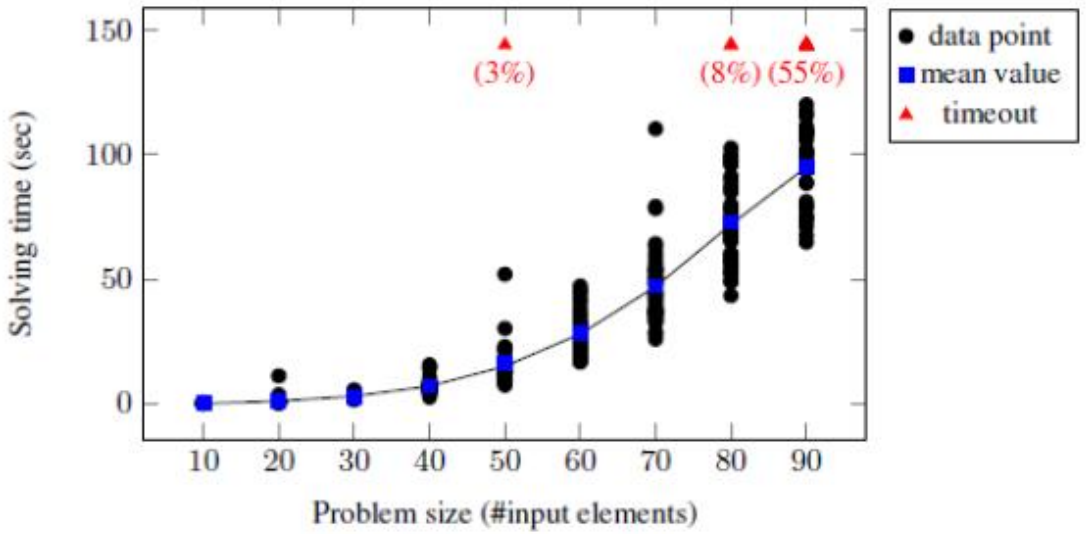


Figure 3. ASP solving times.

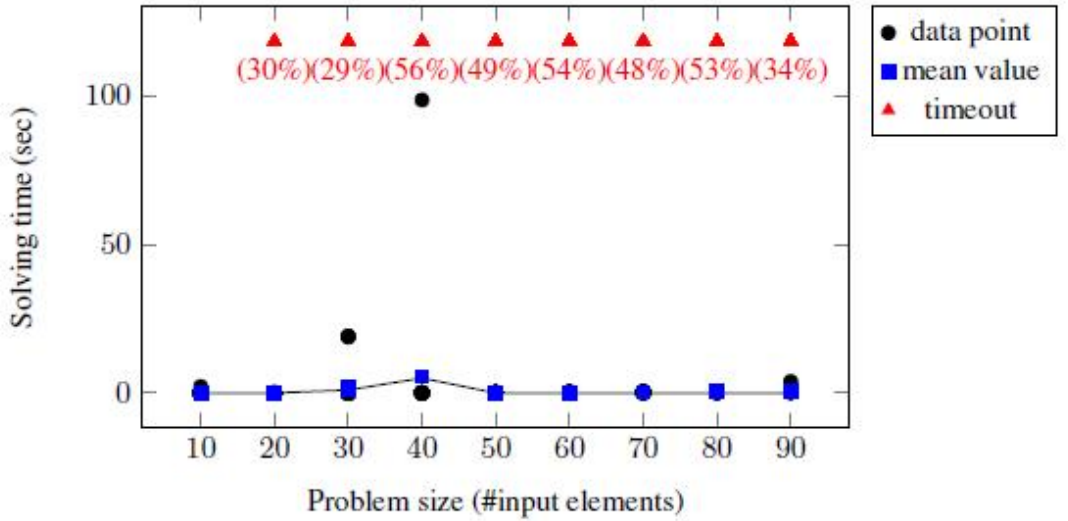


Figure 4. CSP solving times.

A Heuristic, Replay-based Approach for Reconfiguration Memory and Reconfiguration Quality

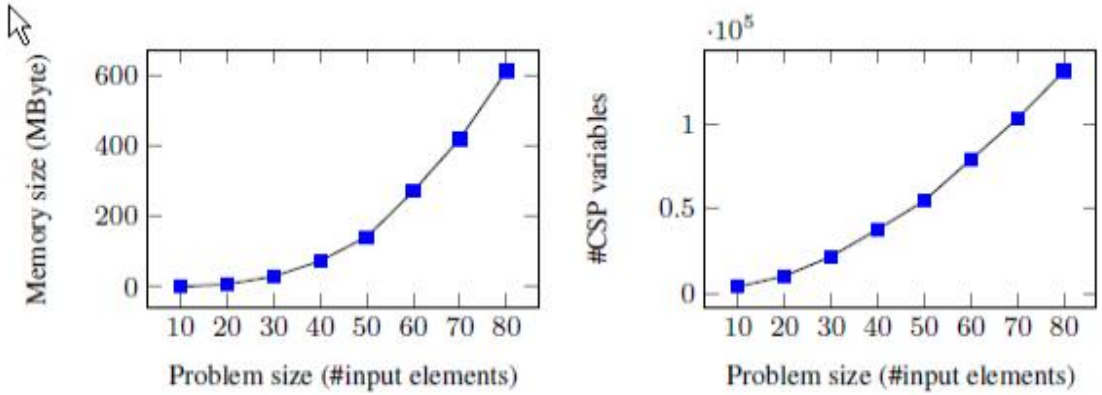


Figure 5. (a) ASP grounding memory size. (b) CSP number of variables.

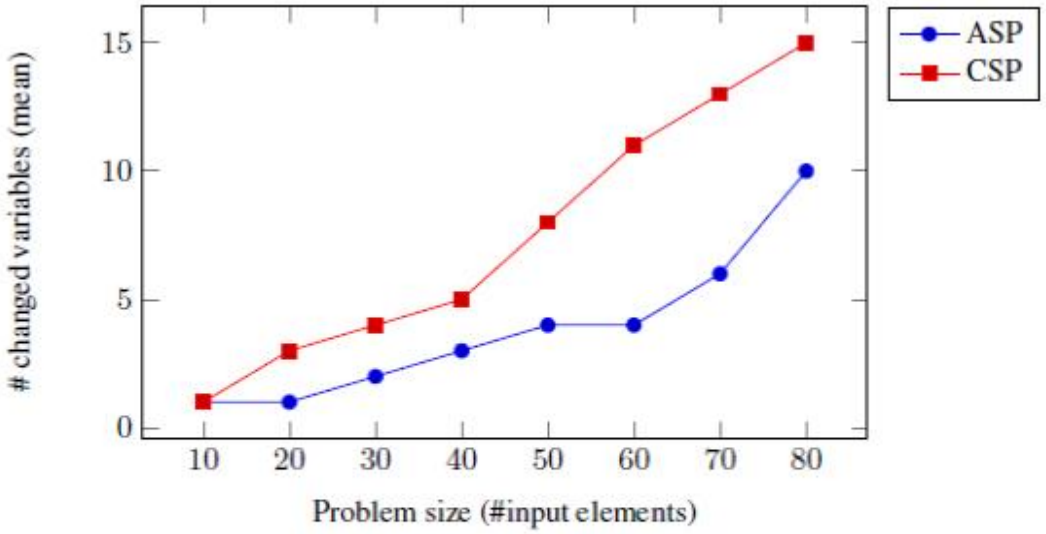


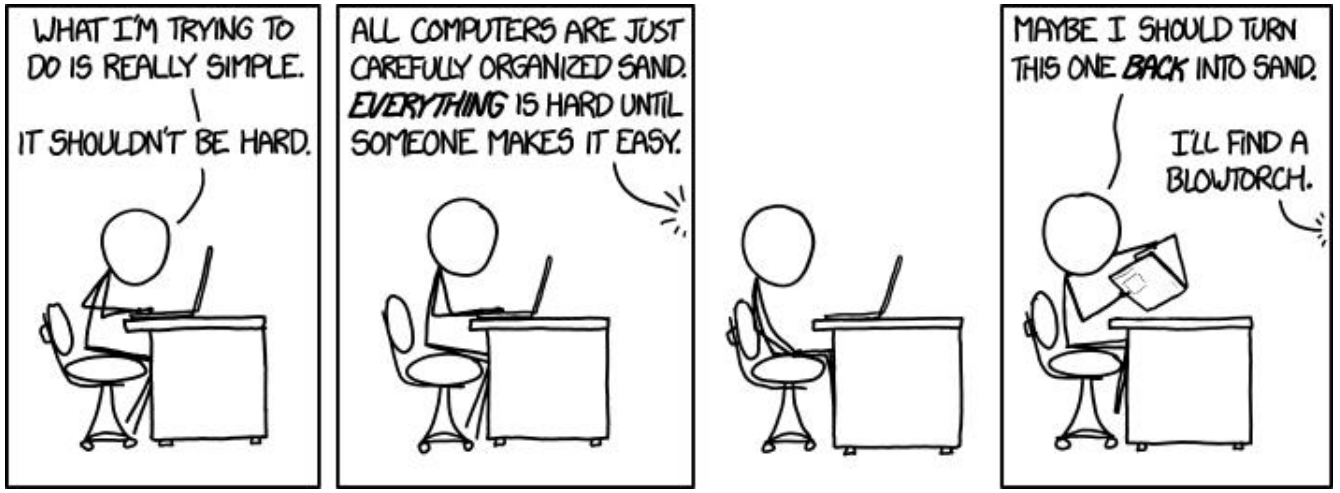
Figure 6. ASP and CSP reconfiguration quality.

A Heuristic, Replay-based Approach for Reconfiguration Results

SIEMENS

	ASP	CSP
Robustness	High predictable	Low Very sensitive to input constellation
Performance	Good for small to middle sized problems	Good, with the right heuristic
Memory	High (grounding)	Medium (can be tuned by reformulating constraints)
Solution quality	Very good	Most of the time the optimum or close to the optimum
Problem encoding	Favored is an automatic transformation from the object-oriented problem description to ASP. Direct ASP encoding is also possible.	Direct encoding of an object-oriented data model with a CSP system is quite intricate and error-prone. Automatic transformation is highly recommended.

A Heuristic, Replay-based Approach for Reconfiguration Discussion



HTTP://XKCD.COM/1349

- How would you solve reconfiguration?

A Heuristic, Replay-based Approach for Reconfiguration
Contact page

SIEMENS

Thank you for your attention!

Gottfried Schenner
CT RTC BAM CON-AT
Siemensstrasse 90

1210 Vienna,

Austria

E-mail:
gottfried.schenner@siemens.com

siemens.com/answers