"From Model-Based Development to Formal Verification of Adaptive Embedded Systems"

Ina Schaefer

inschaef@informatik.uni-kl.de

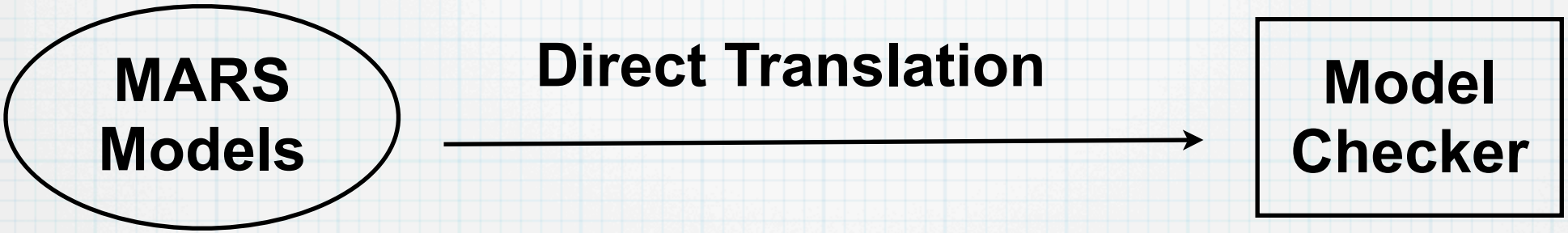
AG Softwaretechnik

University of Kaiserslautern

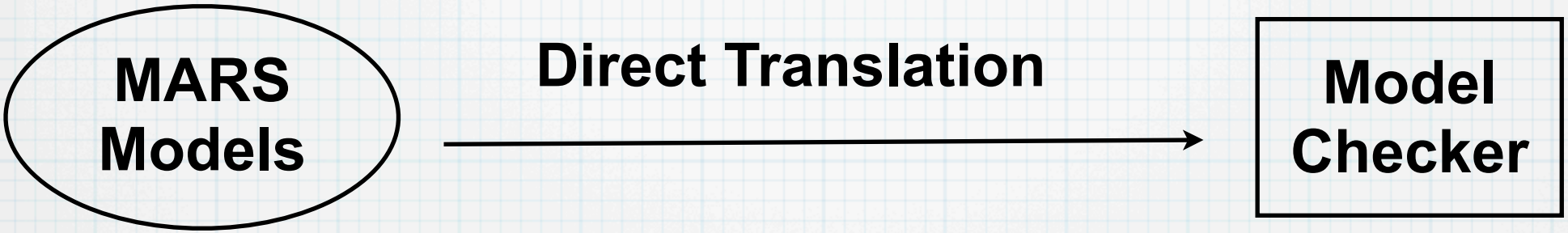
(joint work with Jan Olaf Blech and Arnd Poetzsch-Heffter)

VerAS-Workshop, Kaiserslautern, 14.09.2007

Motivation



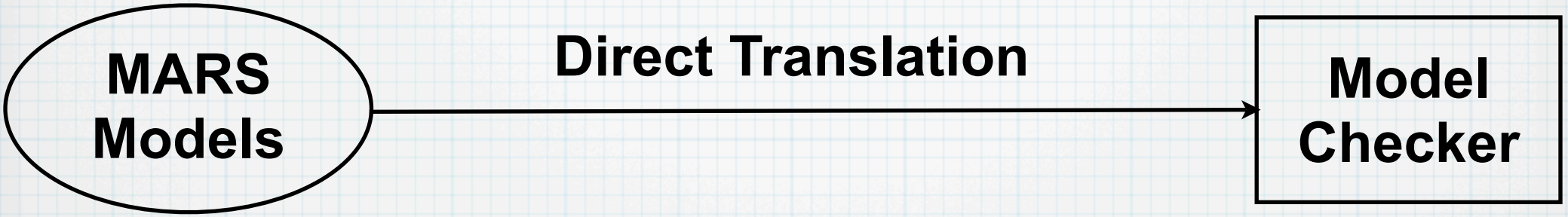
Motivation



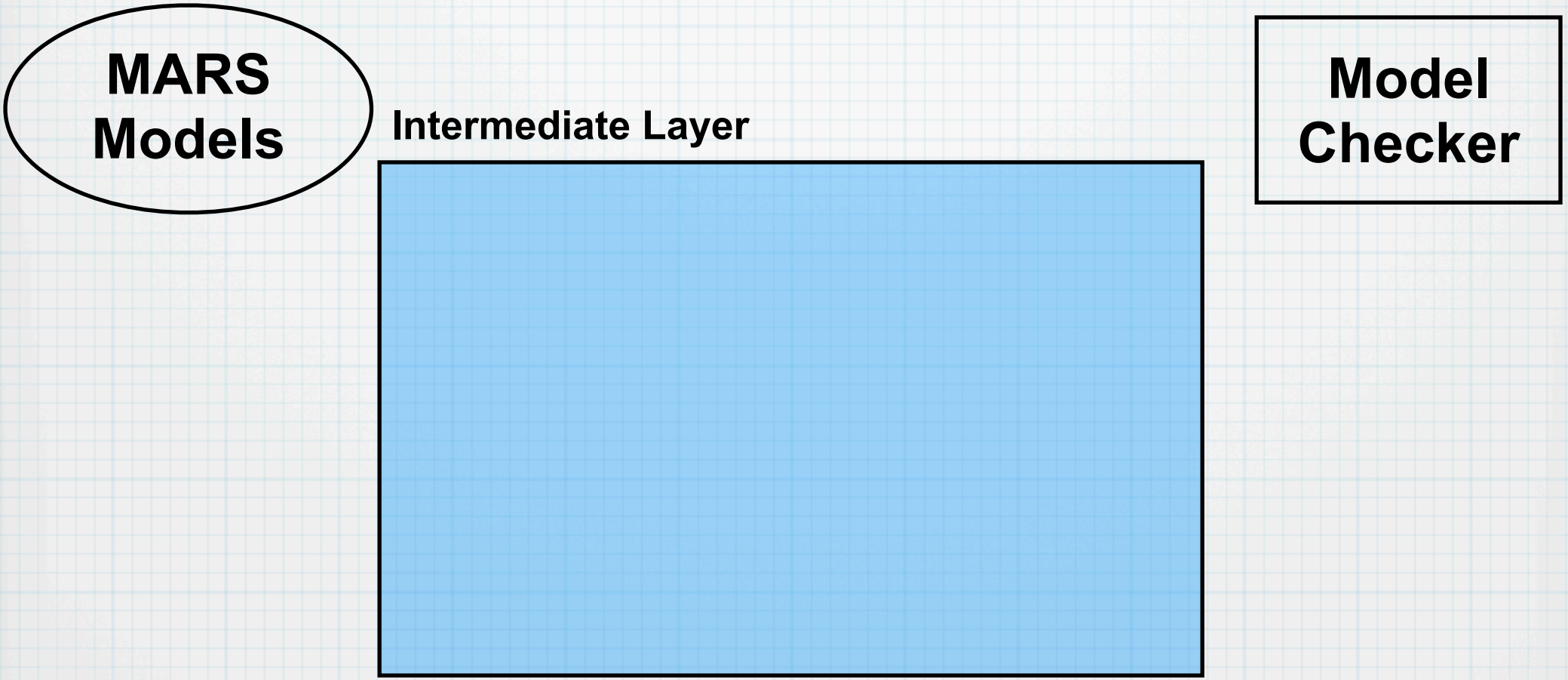
Problems:

- Modelling concepts may have to be restricted.
- Models may be very large.
- Verification is limited to capabilities of model checker.

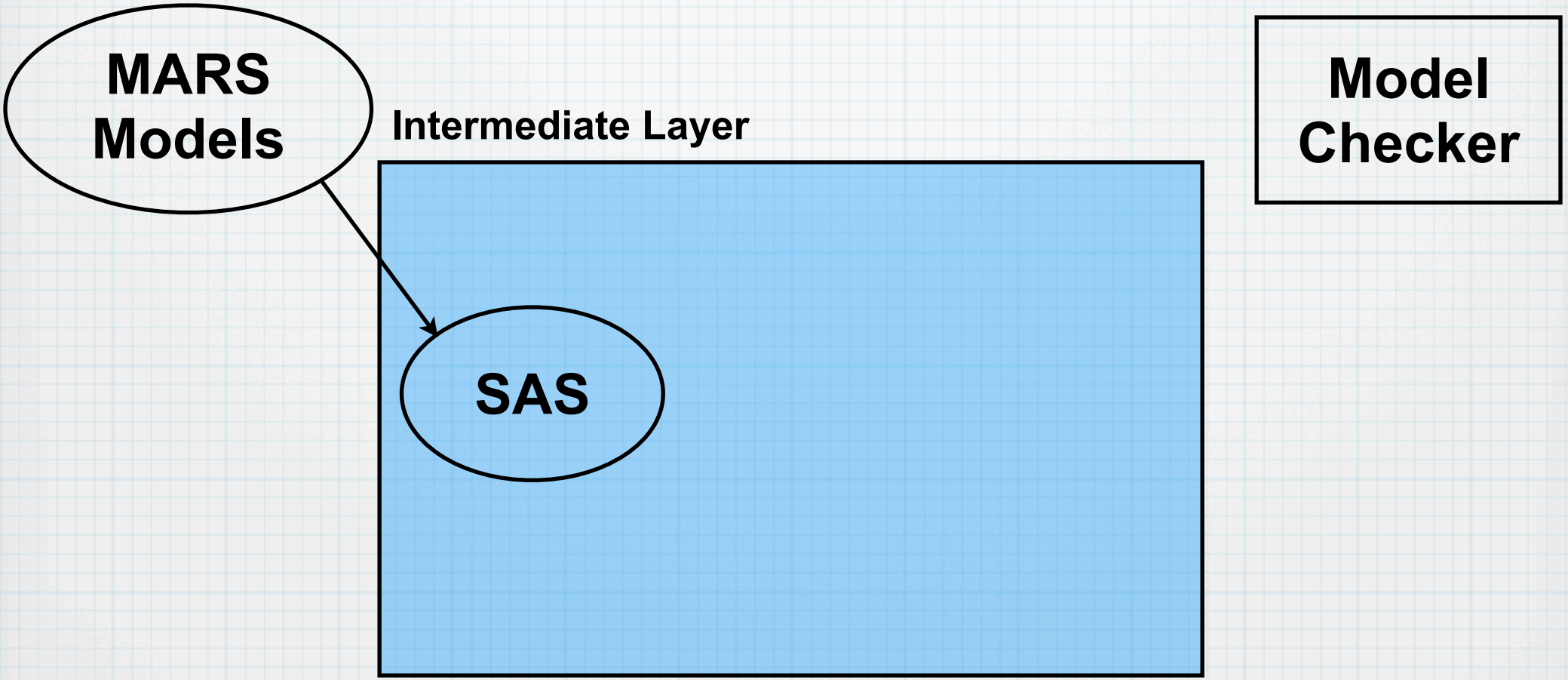
Motivation (2)



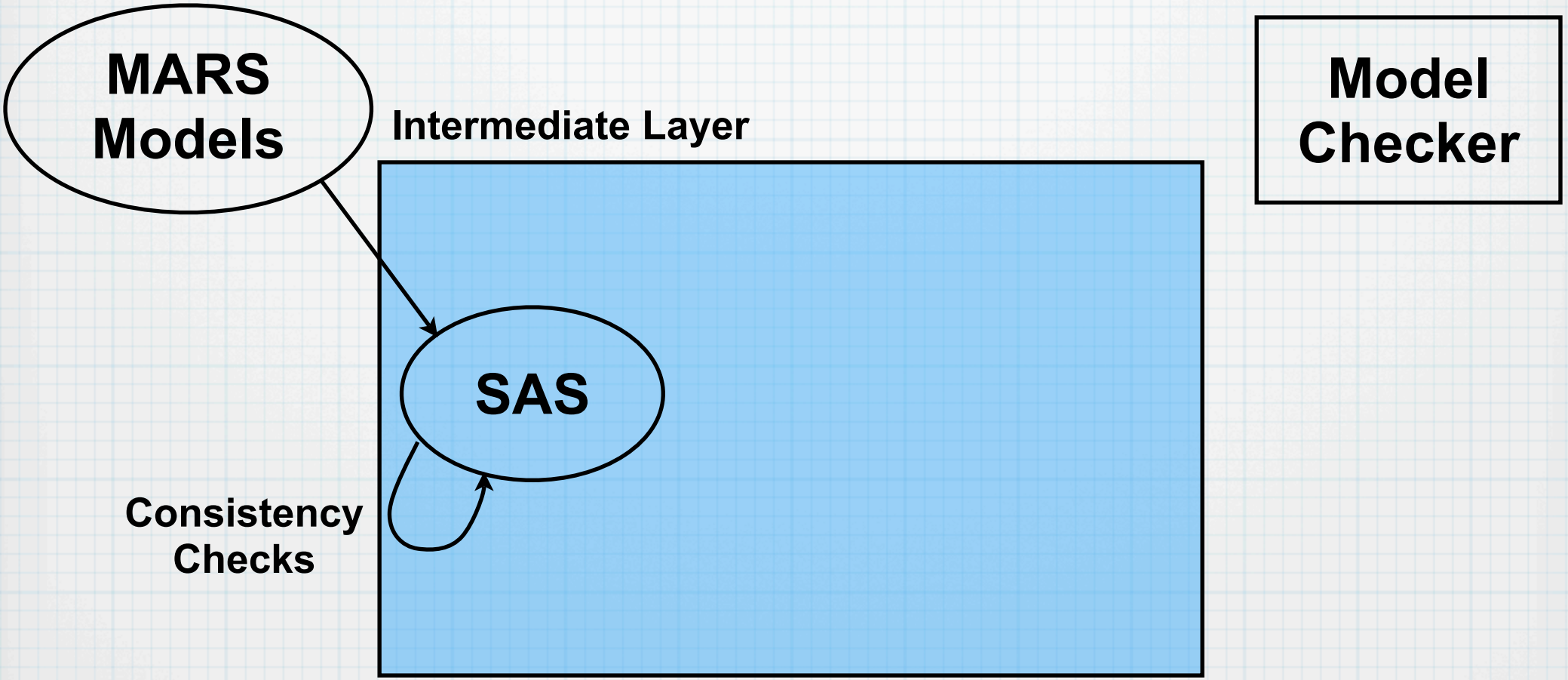
Motivation (2)



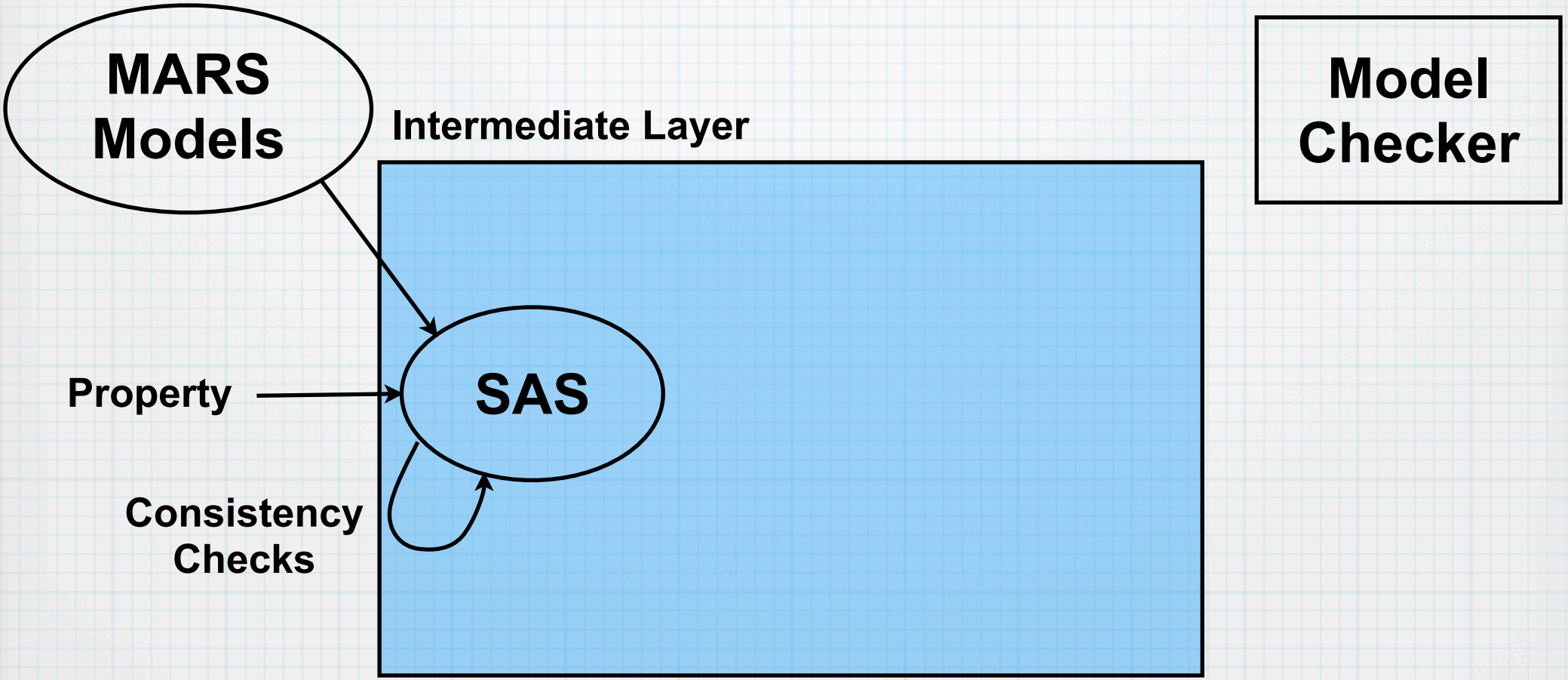
Motivation (2)



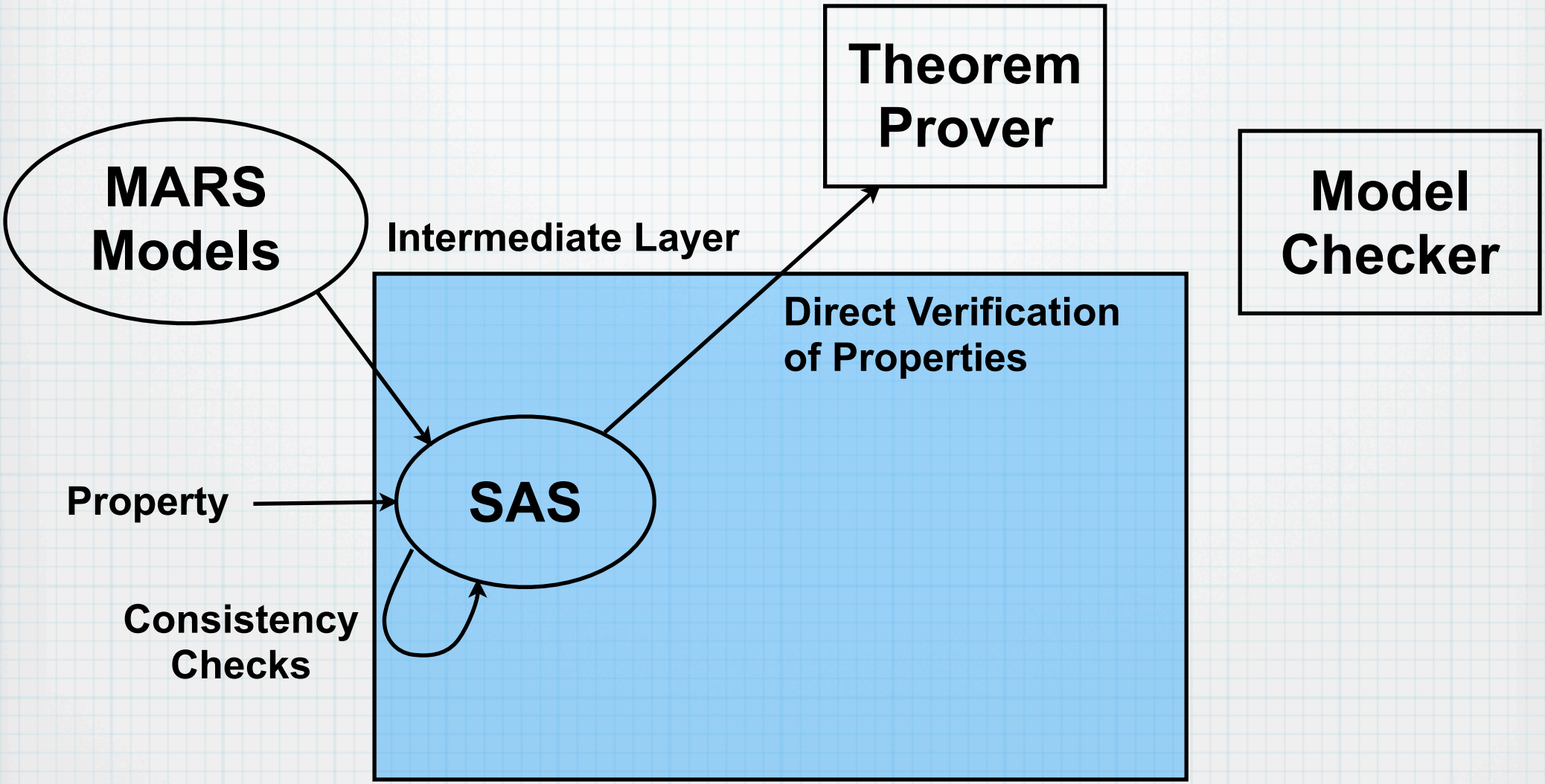
Motivation (2)



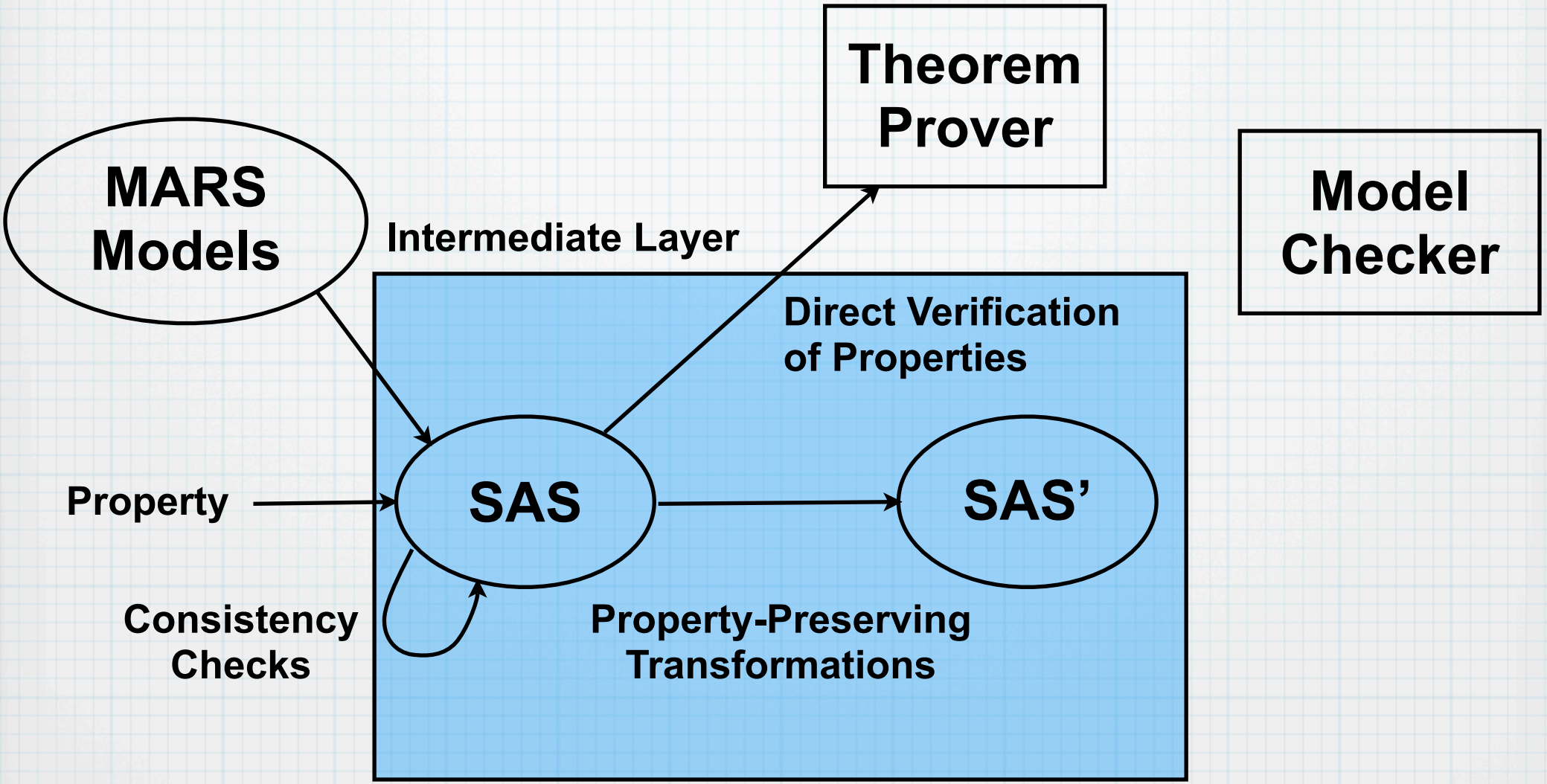
Motivation (2)



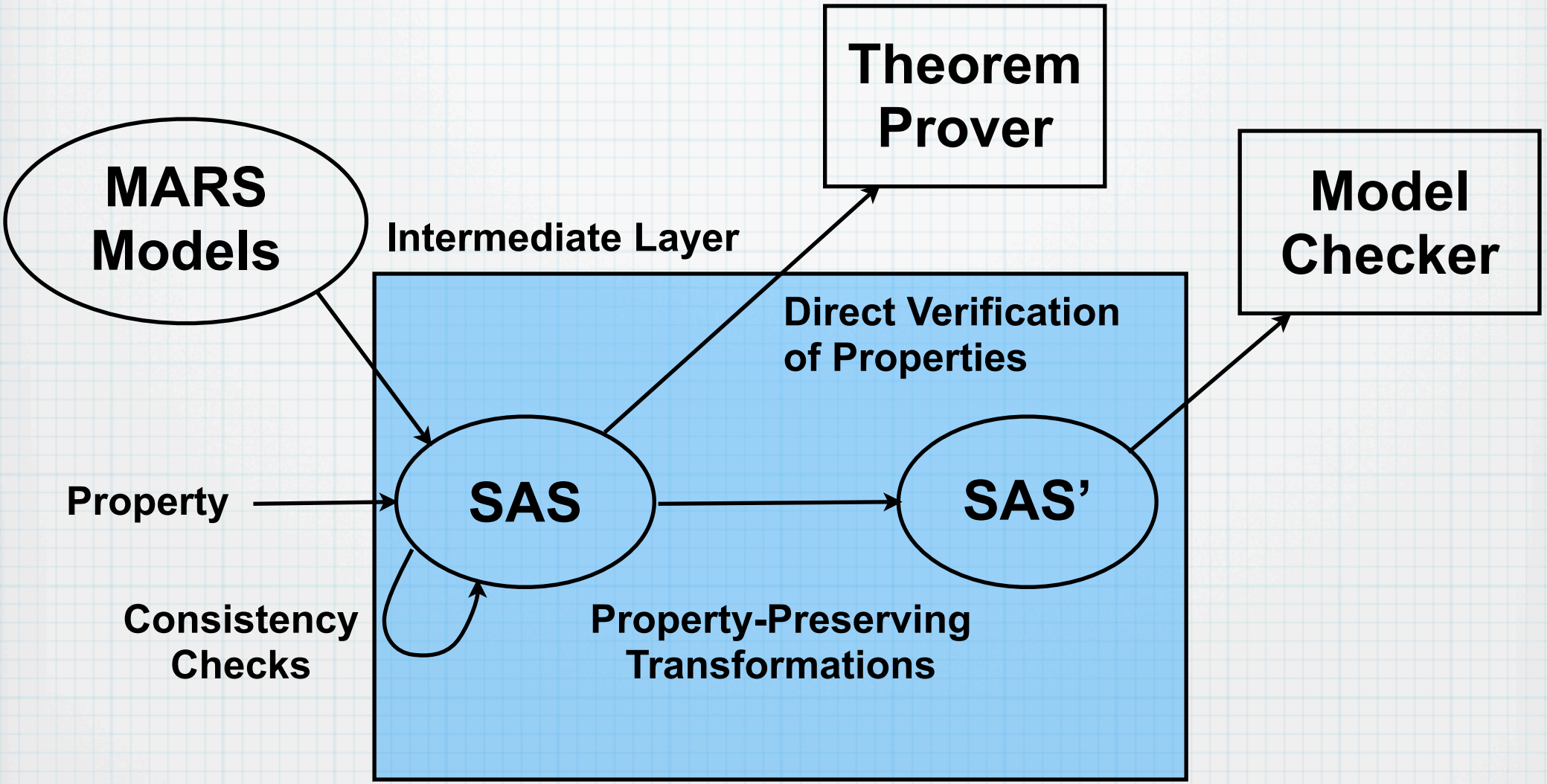
Motivation (2)



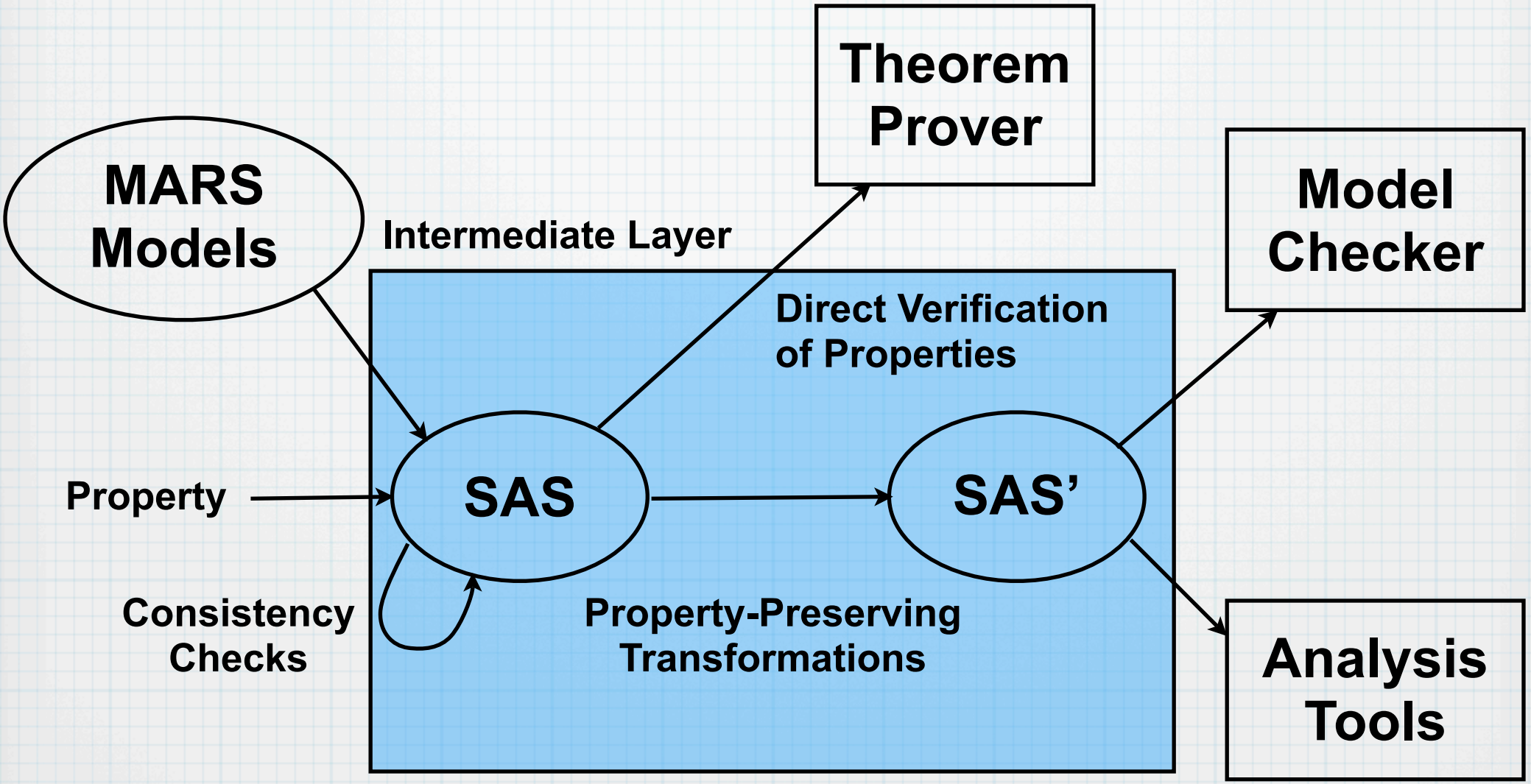
Motivation (2)



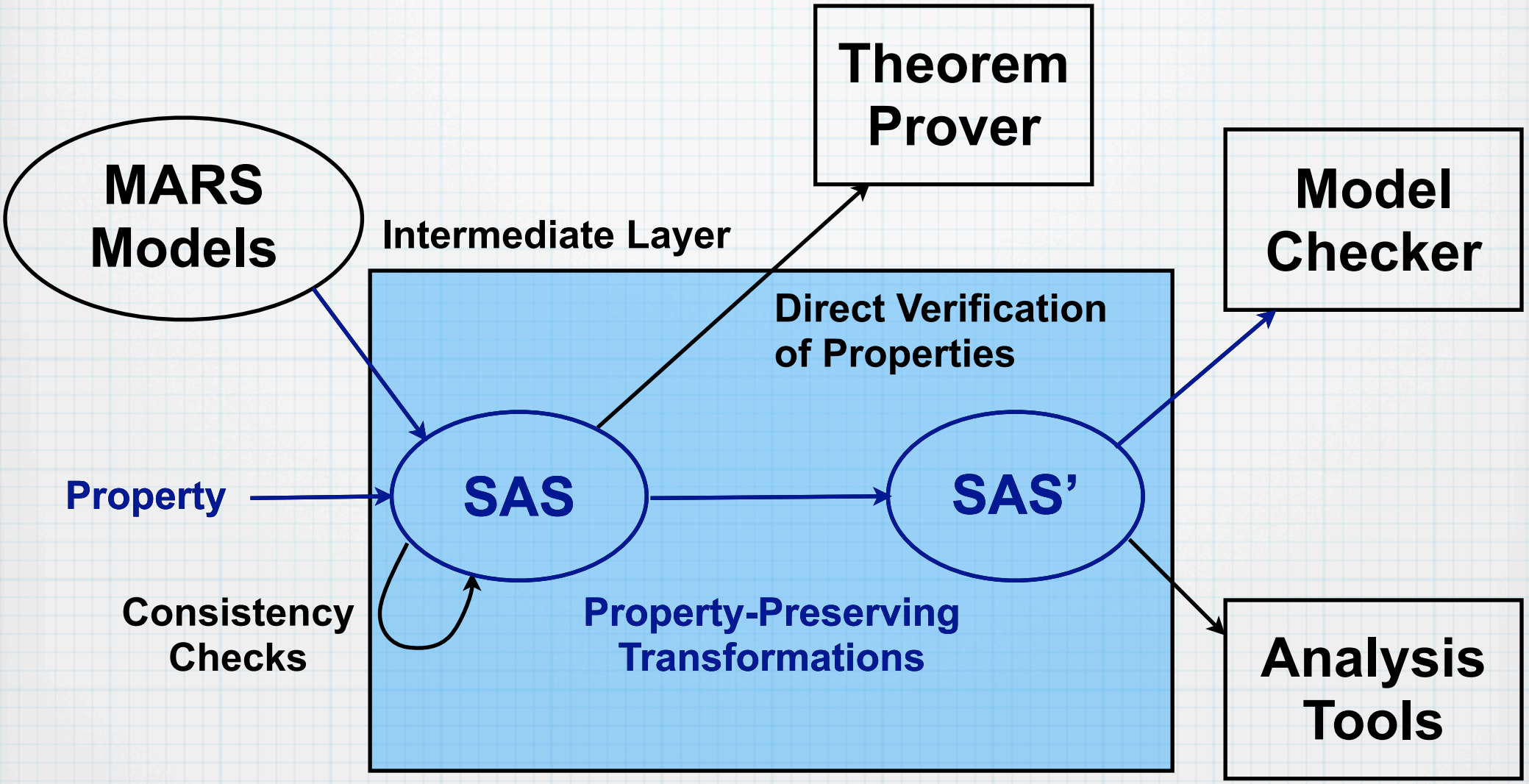
Motivation (2)



Motivation (2)



Motivation (2)



Outline

- SAS Models
- Specification Logic
- Property Preserving Transformations for Model Reduction and Translation Validation
- Conclusion and Future Work

SAS Models

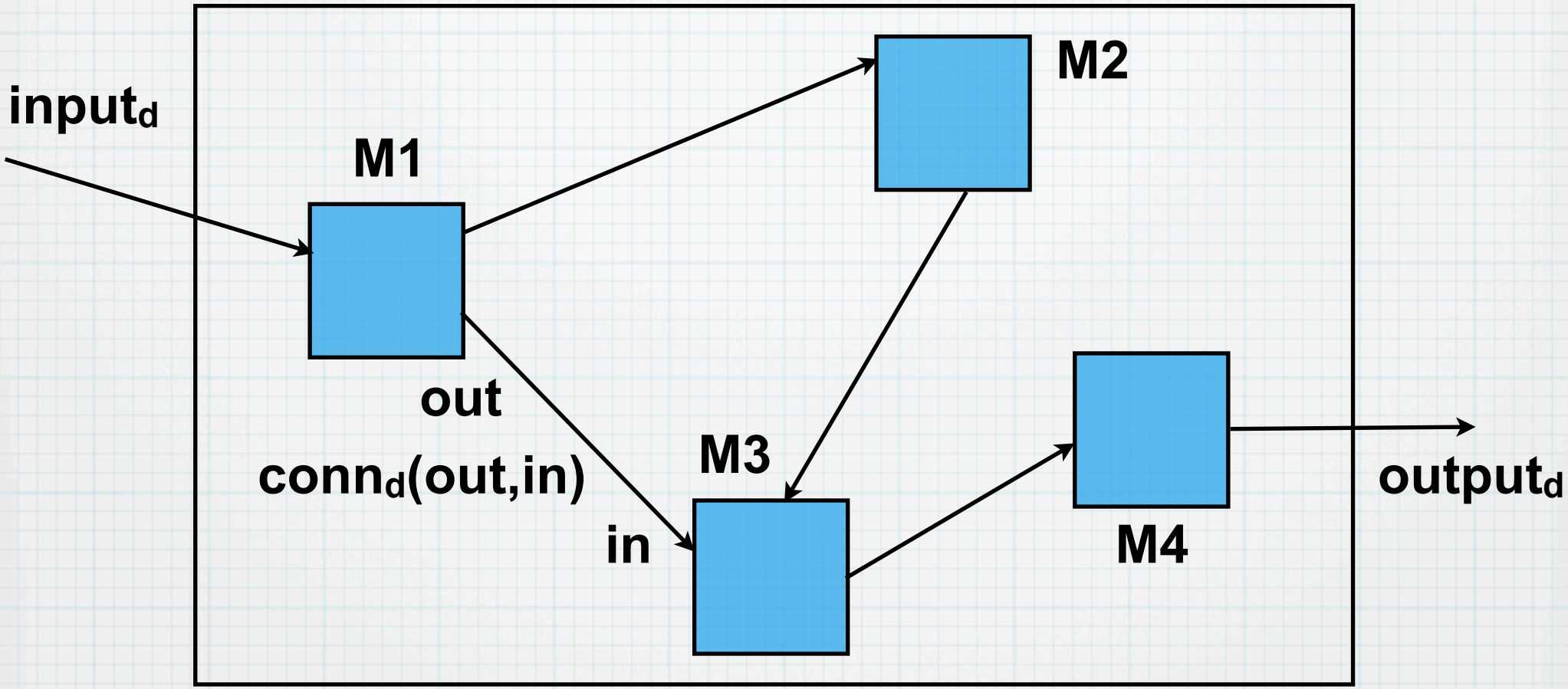
Design Goals:

- Explicit and Separated Modelling of Adaptation and Functionality
- Modular System Structure

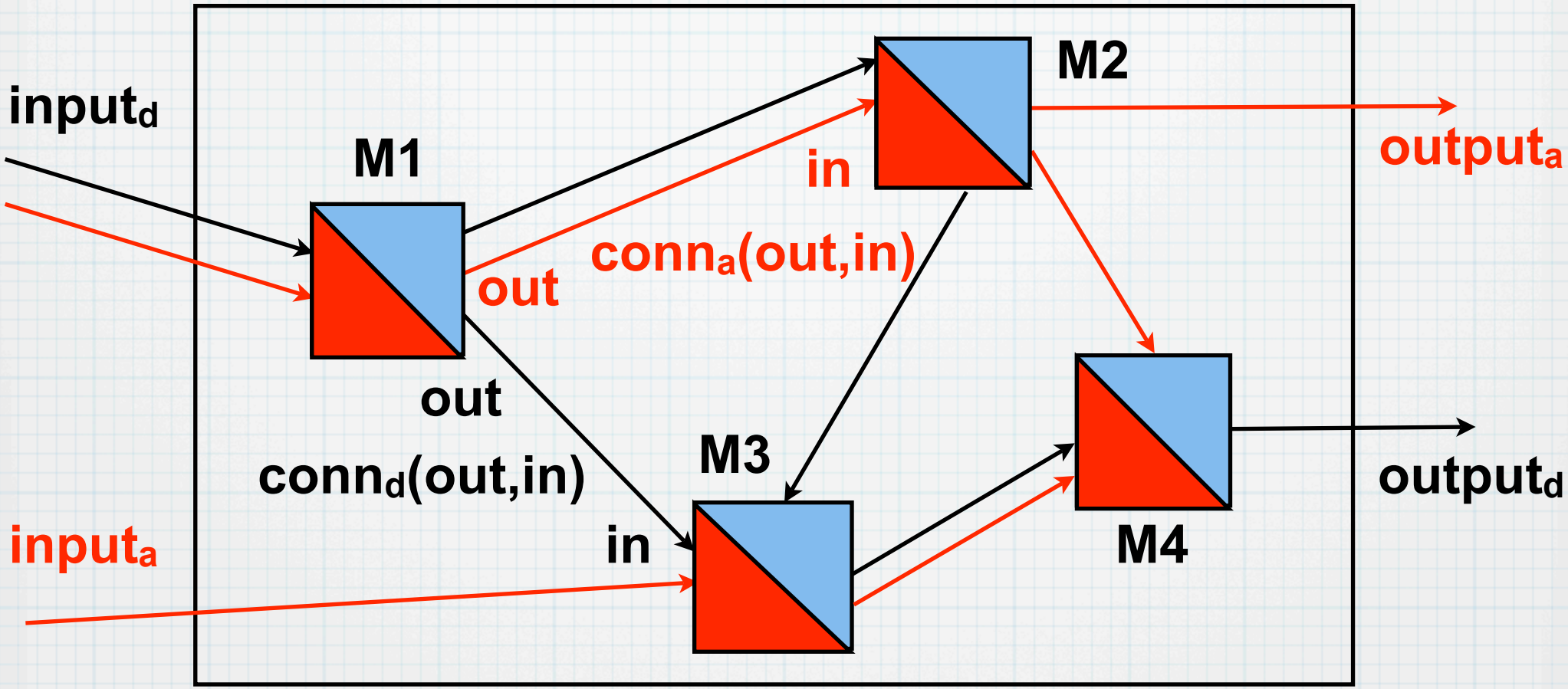
Main Concepts:

- Based on State-Transition Systems
- Uses Adaptation Aspect

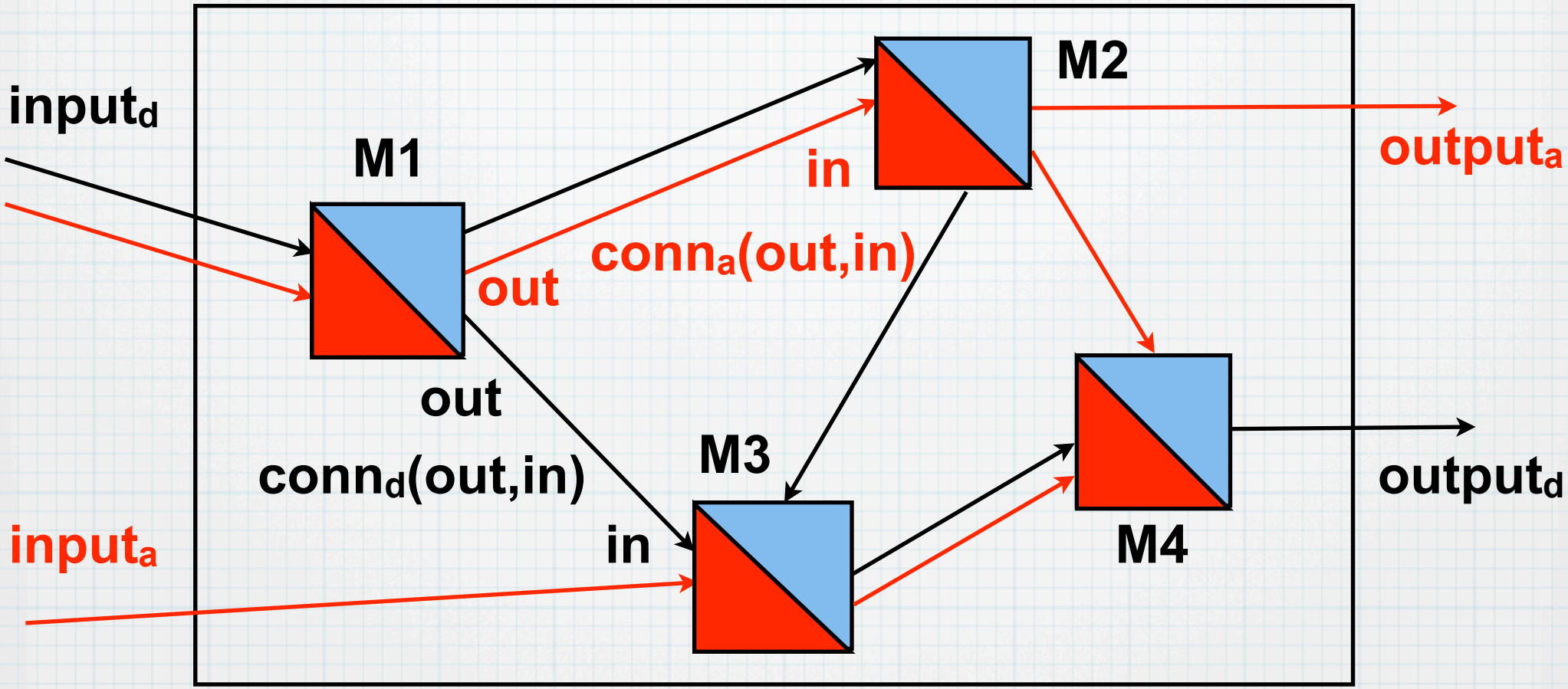
Synchronous Adaptive Systems



Synchronous Adaptive Systems



Synchronous Adaptive Systems



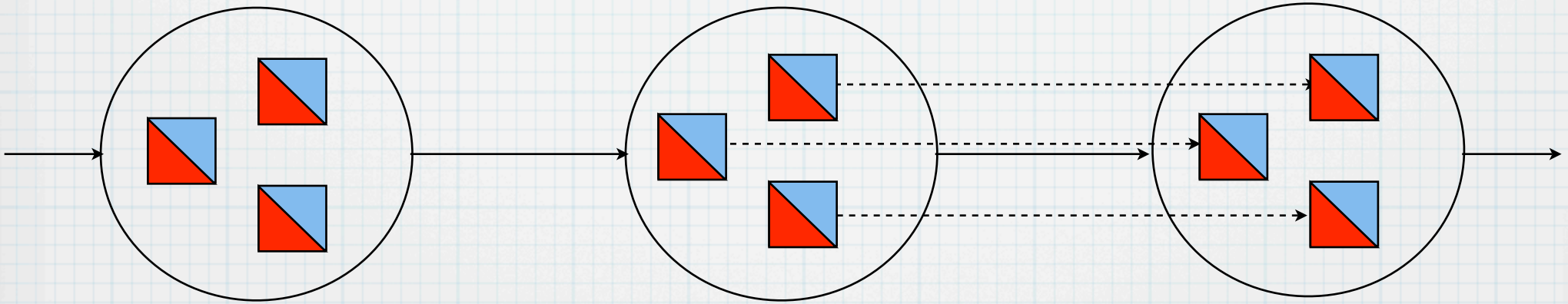
SAS = (M,conn$_d$,conn$_a$,input$_d$,input$_a$,output$_d$,output$_a$)

Global SAS Semantics

Global States: Set of local states of single modules

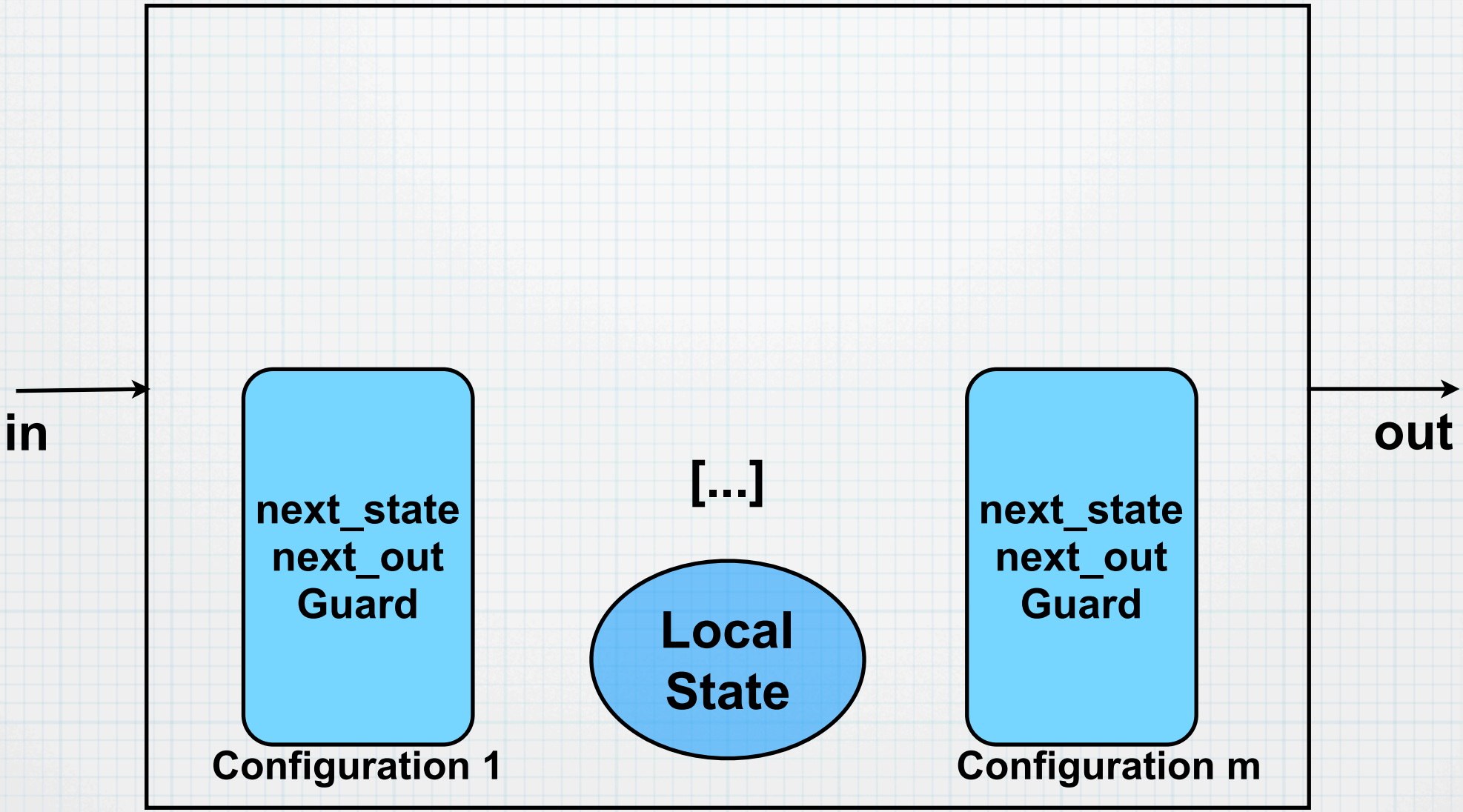
$$\sigma = (s_1, s_2, \ldots, s_n)$$

System Traces: Sequences of global states

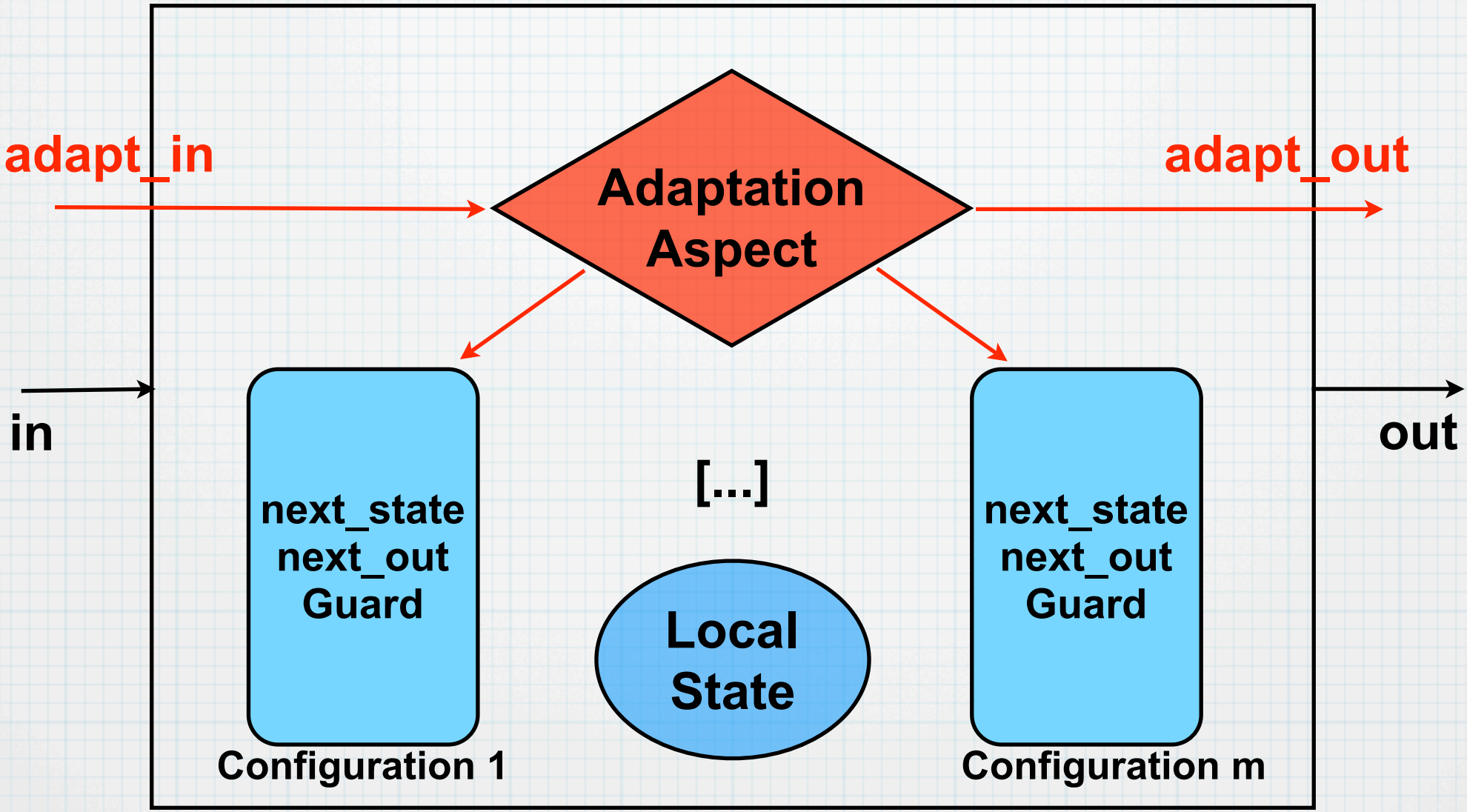


SAS Module

SAS Module

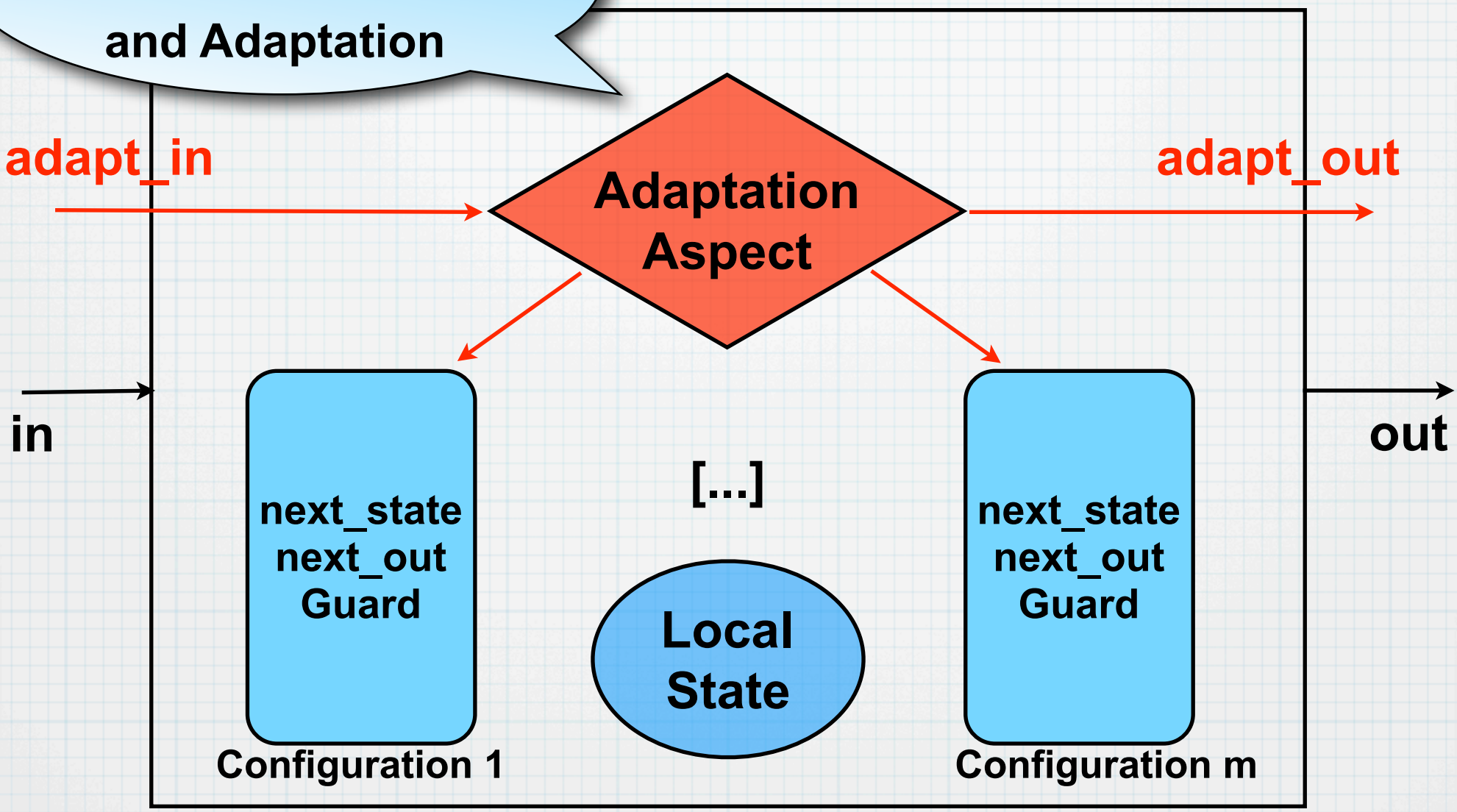

in
next_state
next_out
Guard
Configuration 1
[...]
Local
State
next_state
next_out
Guard
Configuration m
out

SAS Module


adapt_in
Adaptation Aspect
adapt_out
in
out
next_state
next_out
Guard
[...]
Local State
next_state
next_out
Guard
Configuration 1
Configuration m

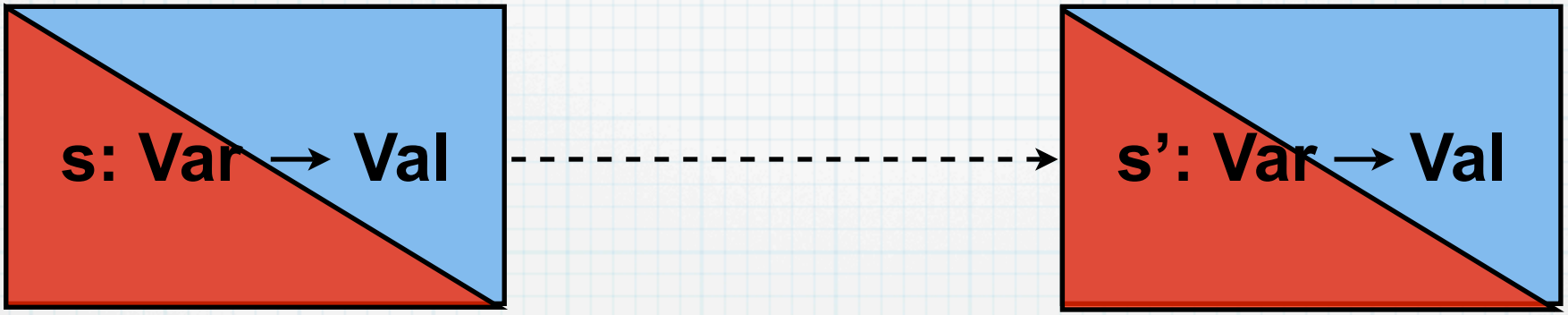
AS Module

Independent Modelling of Functionality and Adaptation



Local SAS Semantics

Local State: Evaluation of in, out and local variables and of adapt_loc, adapt_in and adapt_out variables

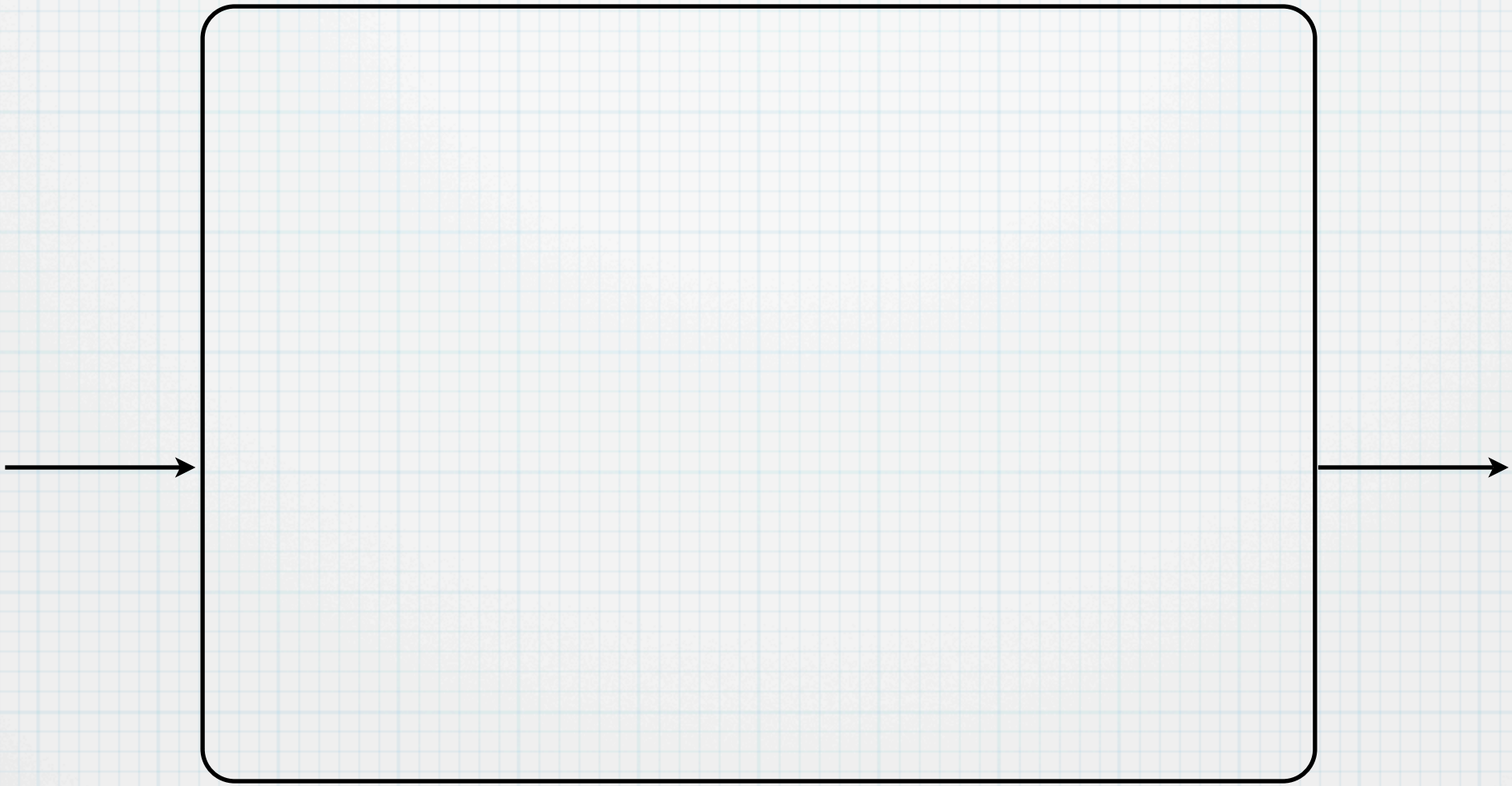


Local Transition:

1. Adaptation aspect determines applicable configuration and computes adaptation parameters.
2. Next_state and next_out function of respective configuration compute next functional state.

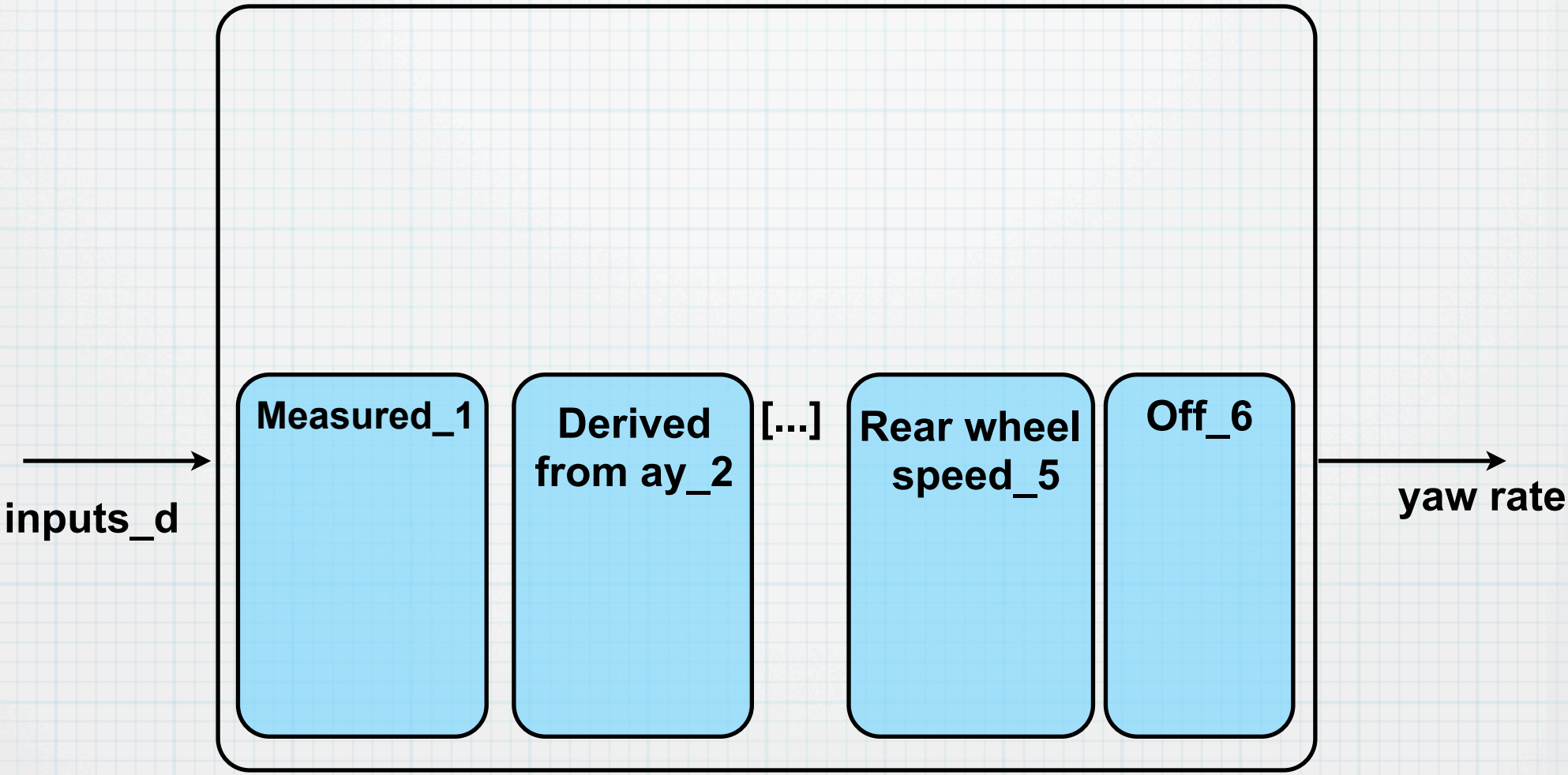
Traction Control System

Example: Yaw Rate Module



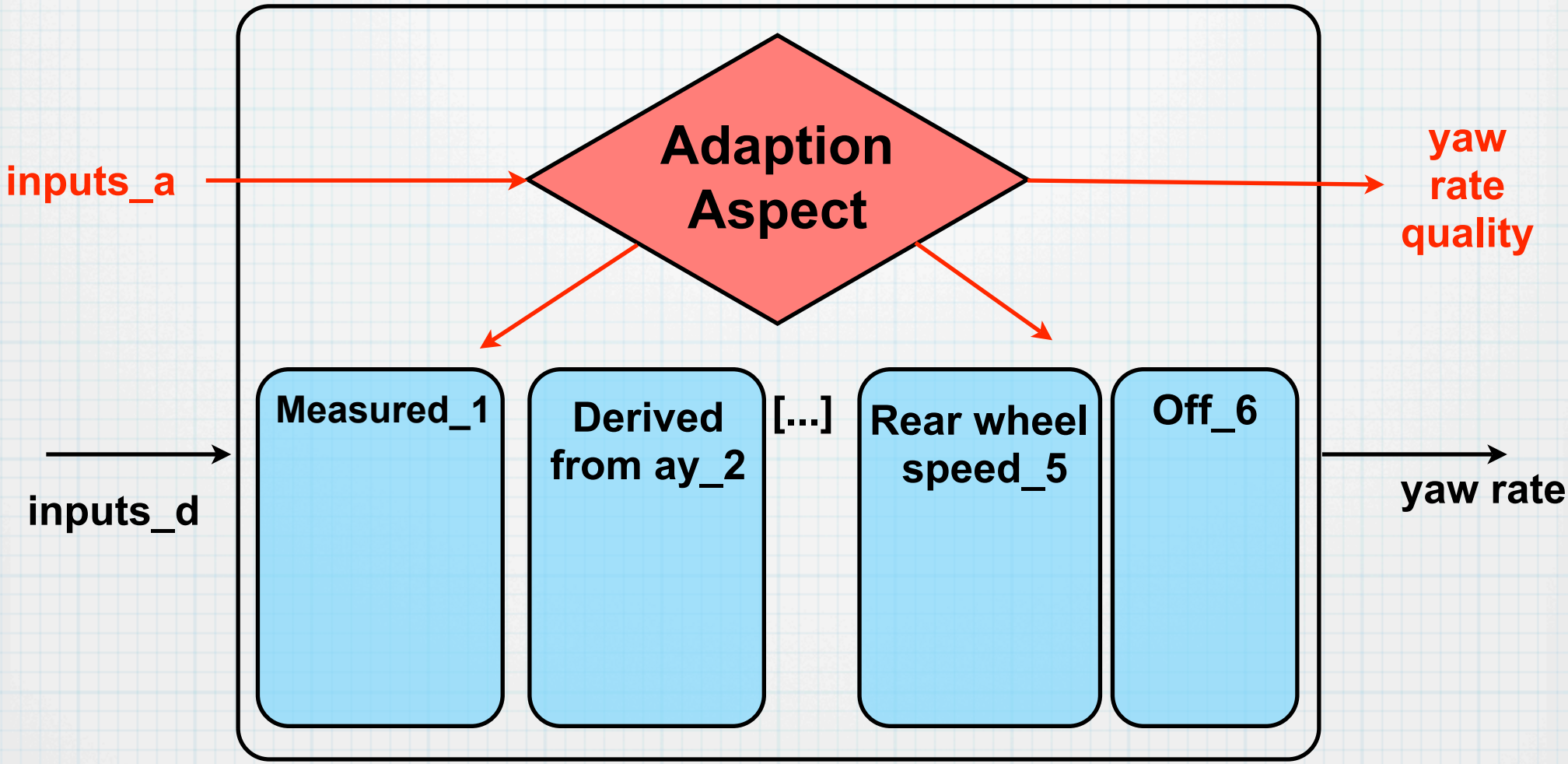
Traction Control System

Example: Yaw Rate Module



Traction Control System

Example: Yaw Rate Module



System Properties

- Adaptive Properties
  - Generic Properties
  - Application-Specific Properties
- Functional Properties

Specification Logic

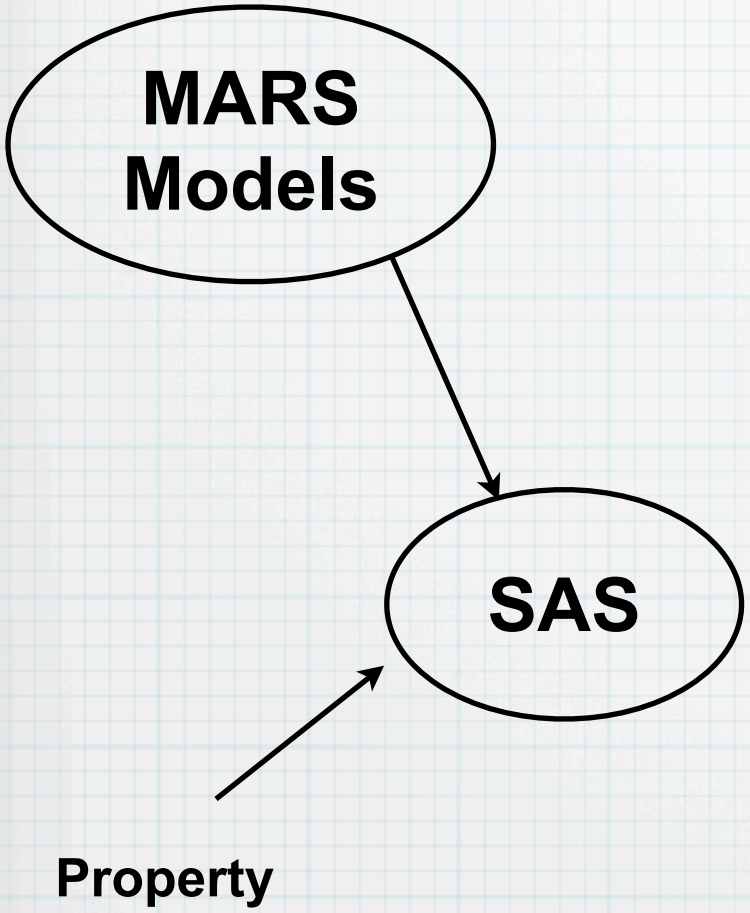
Variant of temporal logic CTL*

- Atomic propositions: `x == 3, y < 5, x < y,...`
- Boolean connectives: `p && q, !q`
- Temporal operators: `X p, G p, F p`
- Path quantifiers: `E q, A q`

Example:

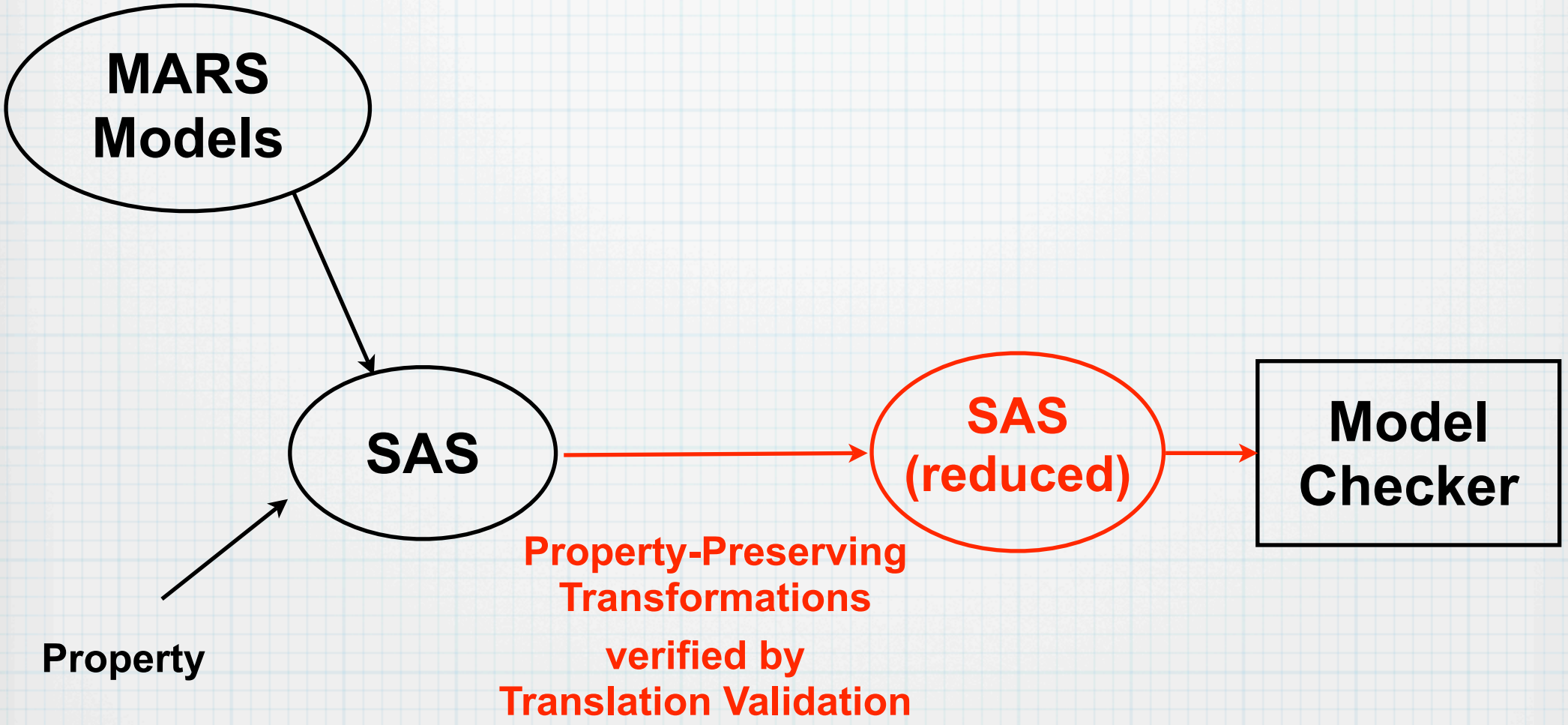
```
EG ( useconf = derived && EF useconf = derived)
```

Verification Complexity Reduction

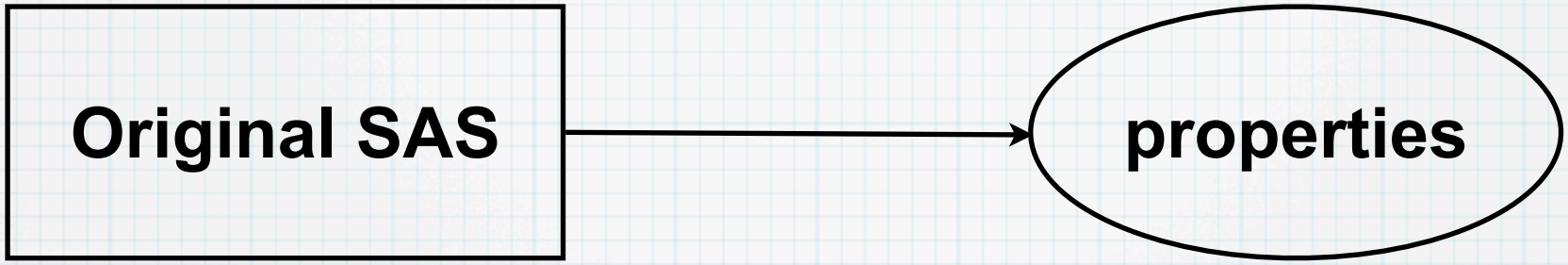


Model Checker

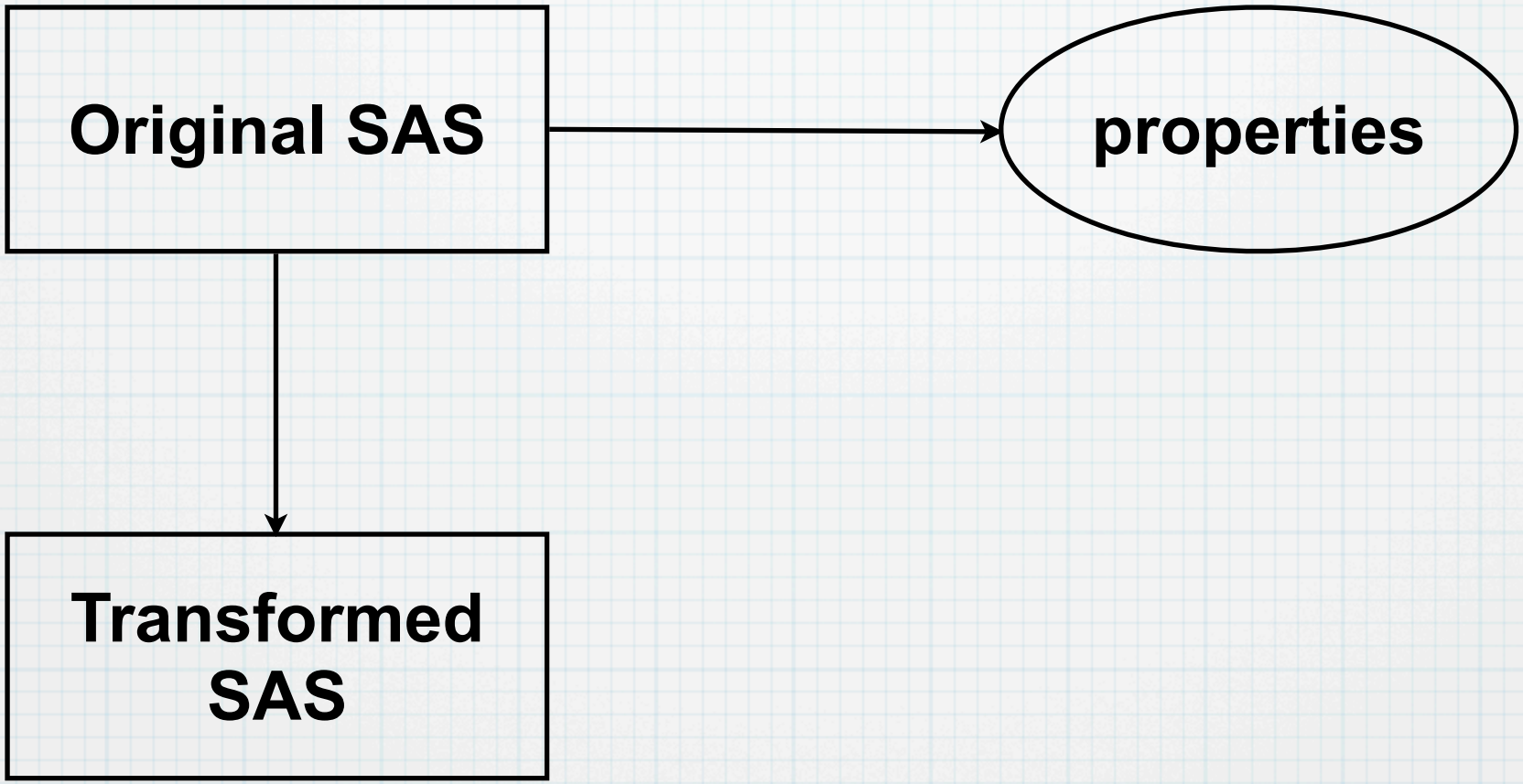
Verification Complexity Reduction



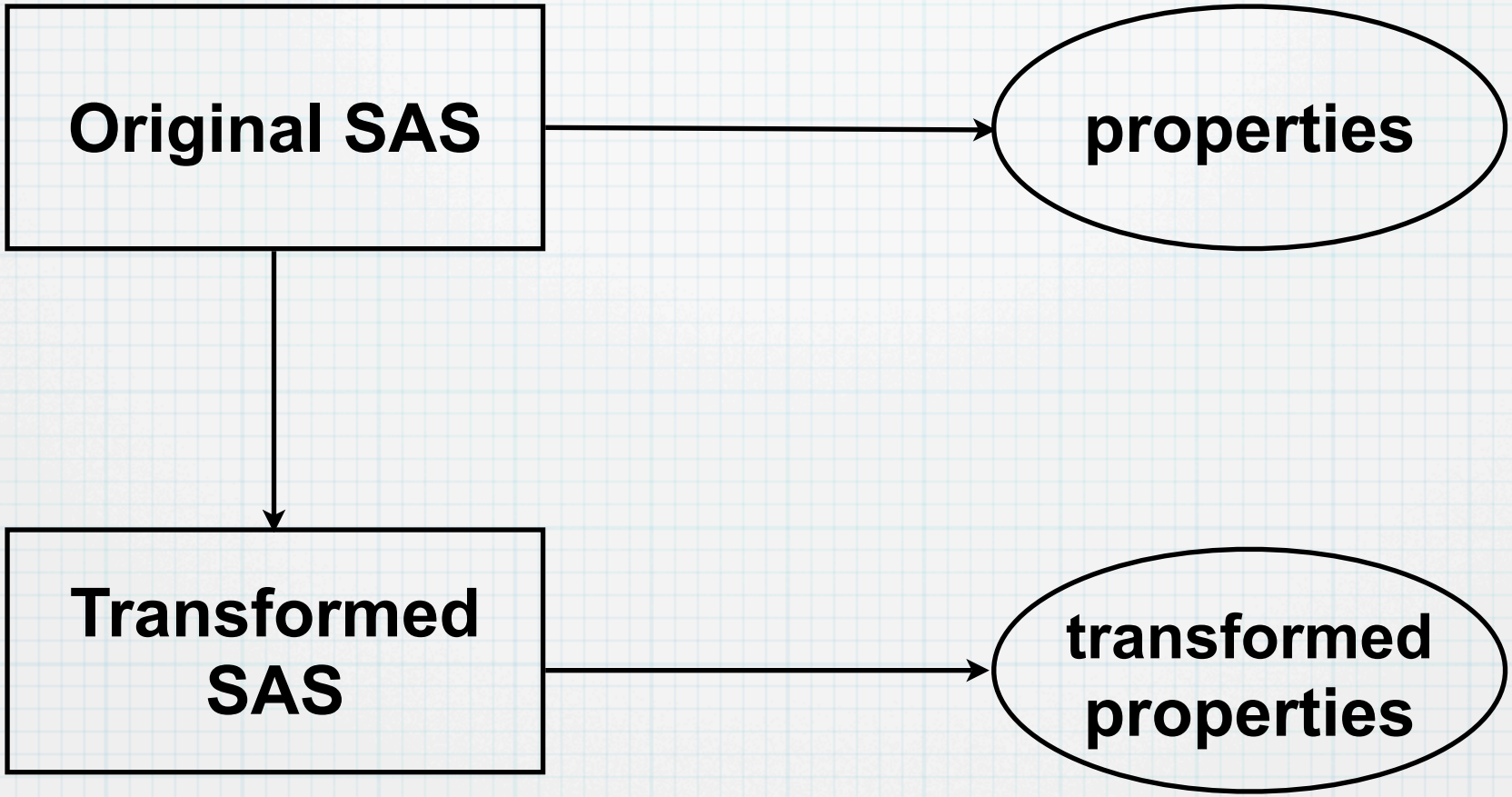
Property-Preserving Transformations



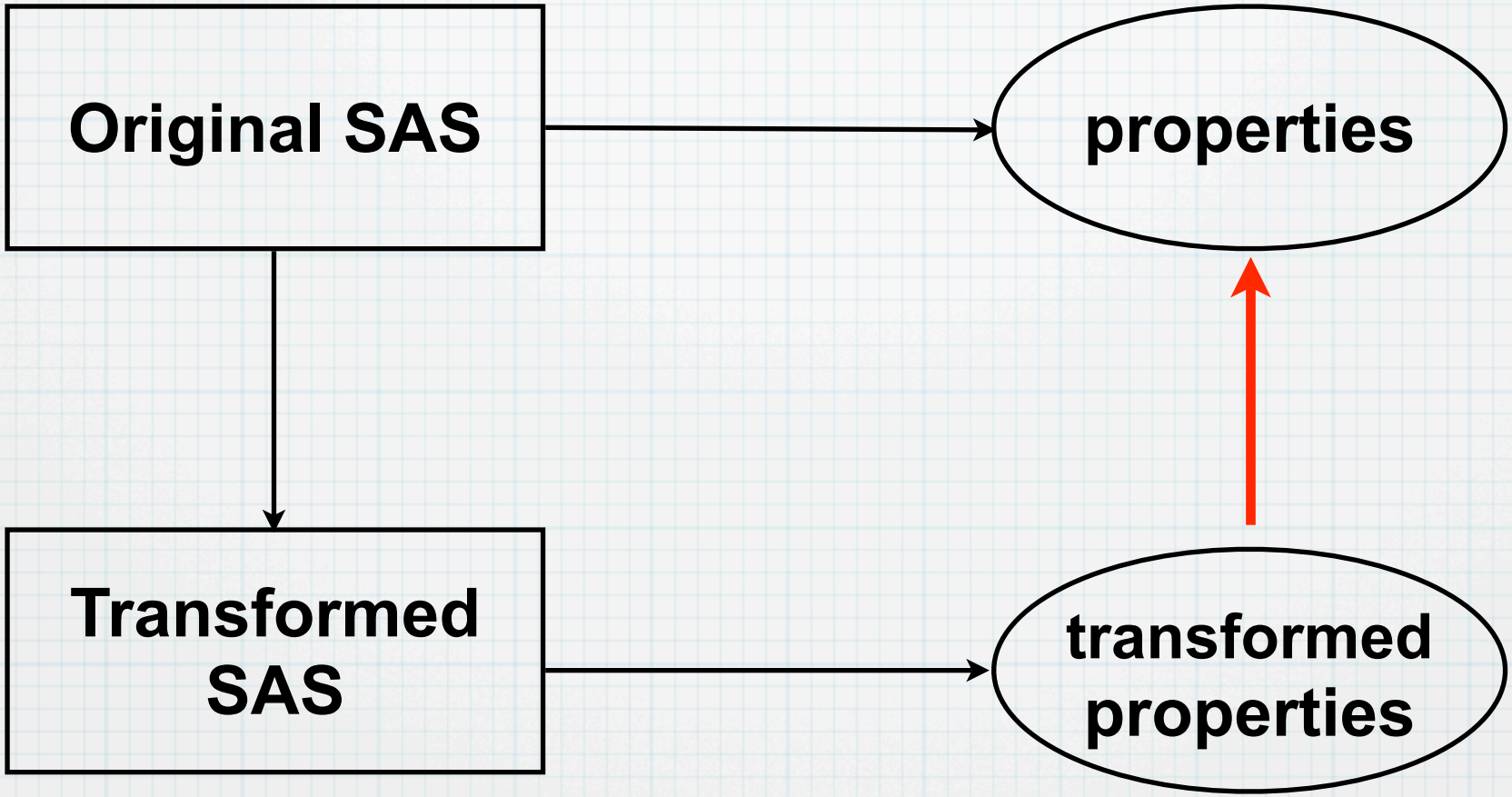
Property-Preserving Transformations



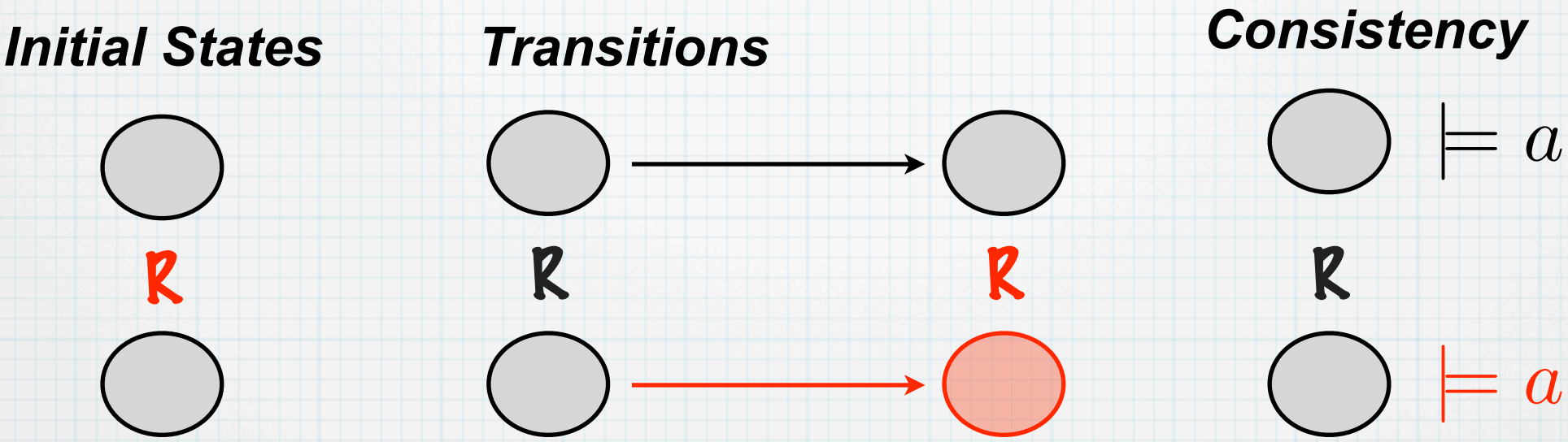
Property-Preserving Transformations



Property-Preserving Transformations



Property-Preservation by Simulation

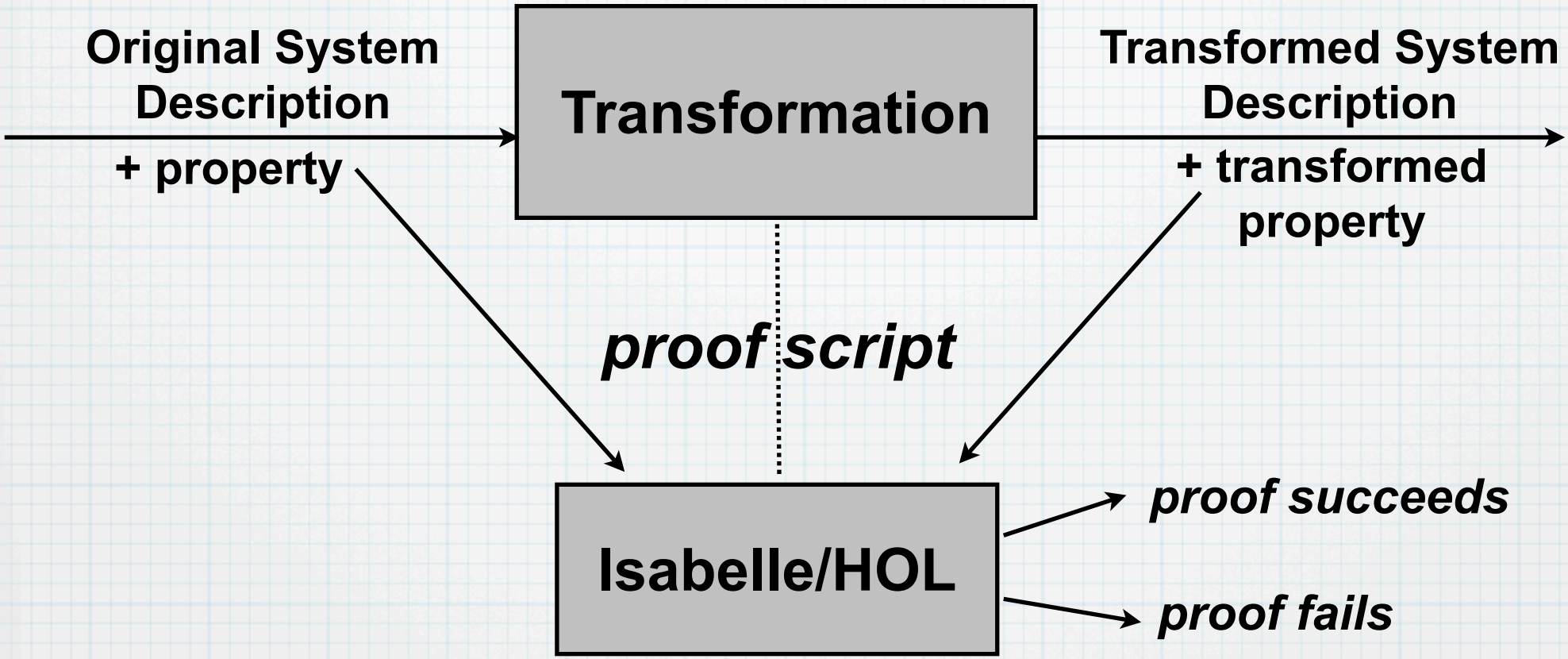


Theorem:

Consistent Bisimulation $\longrightarrow$ Preservation of full logic

Consistent Simulation $\longrightarrow$ Preservation of universal fragment

Translation Validation



J.O. Blech, I. Schaefer, A. Poetzsch-Heffter. Translation Validation for System Abstractions.
in: Proc. of 7th Workshop on Runtime Verification (RV'07), Vancouver, Canada, March 2007

System Transformations

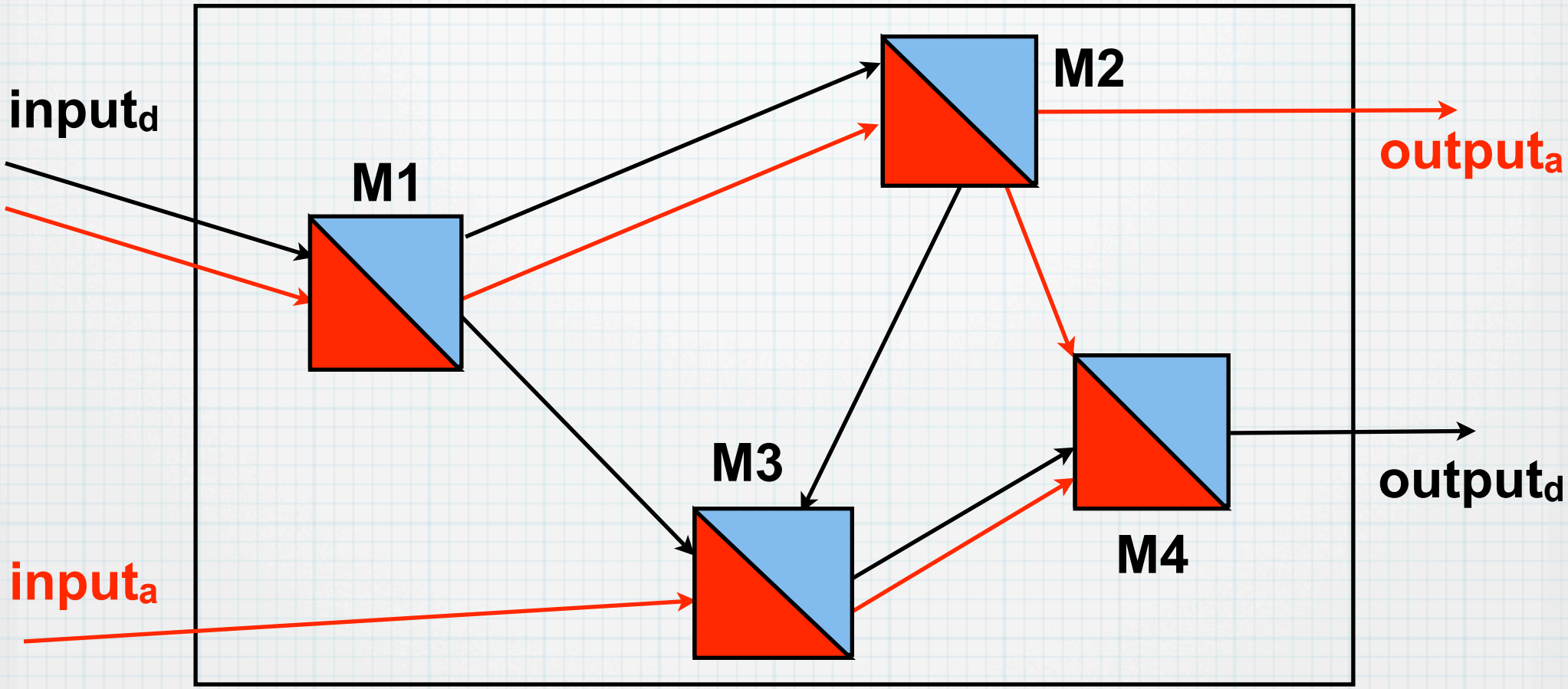
Transformations used for SAS:

- **Slicing**
- **Abstractions**
- **Decomposition**

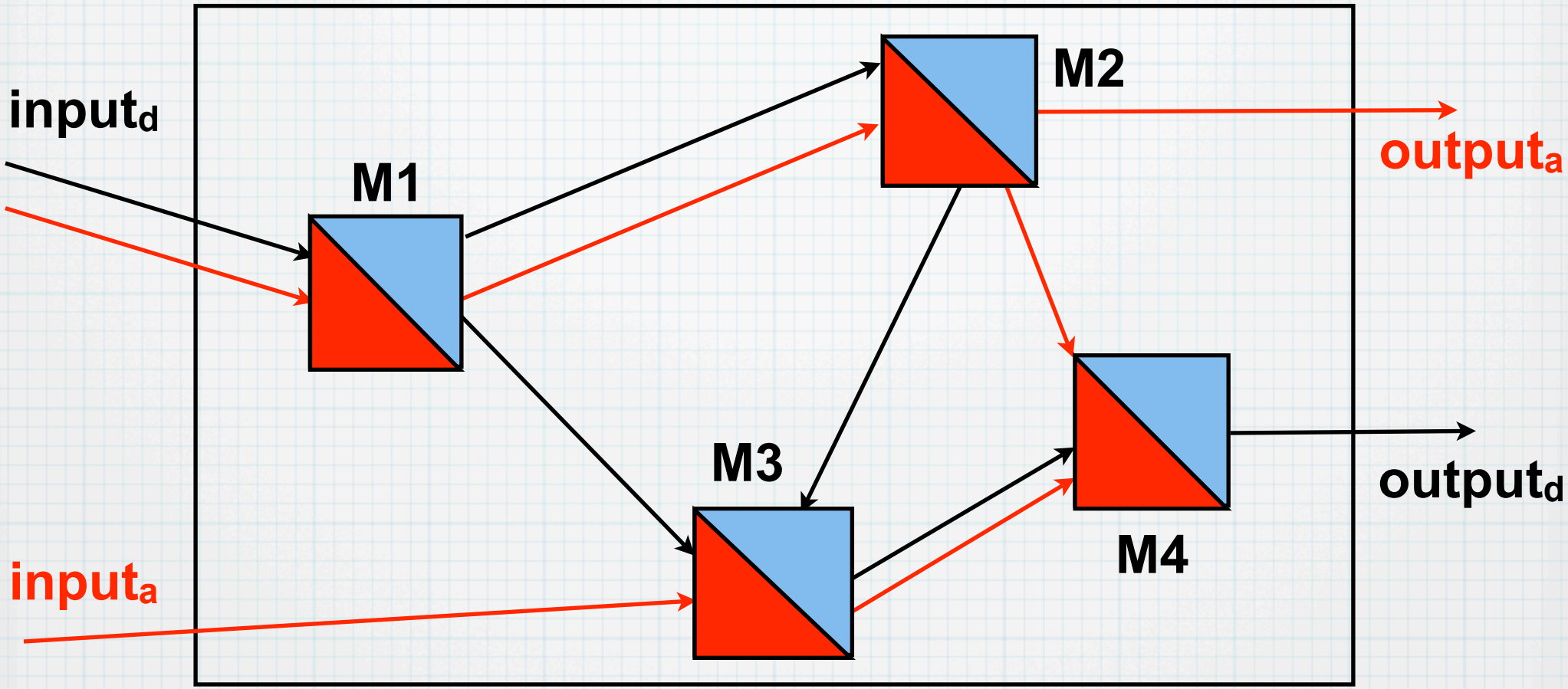
Slicing for Model Reduction

- Omitting Unused Variables
- Slicing on Module Level
- Slicing on System Level
- Adaptive Slicing

System Slicing

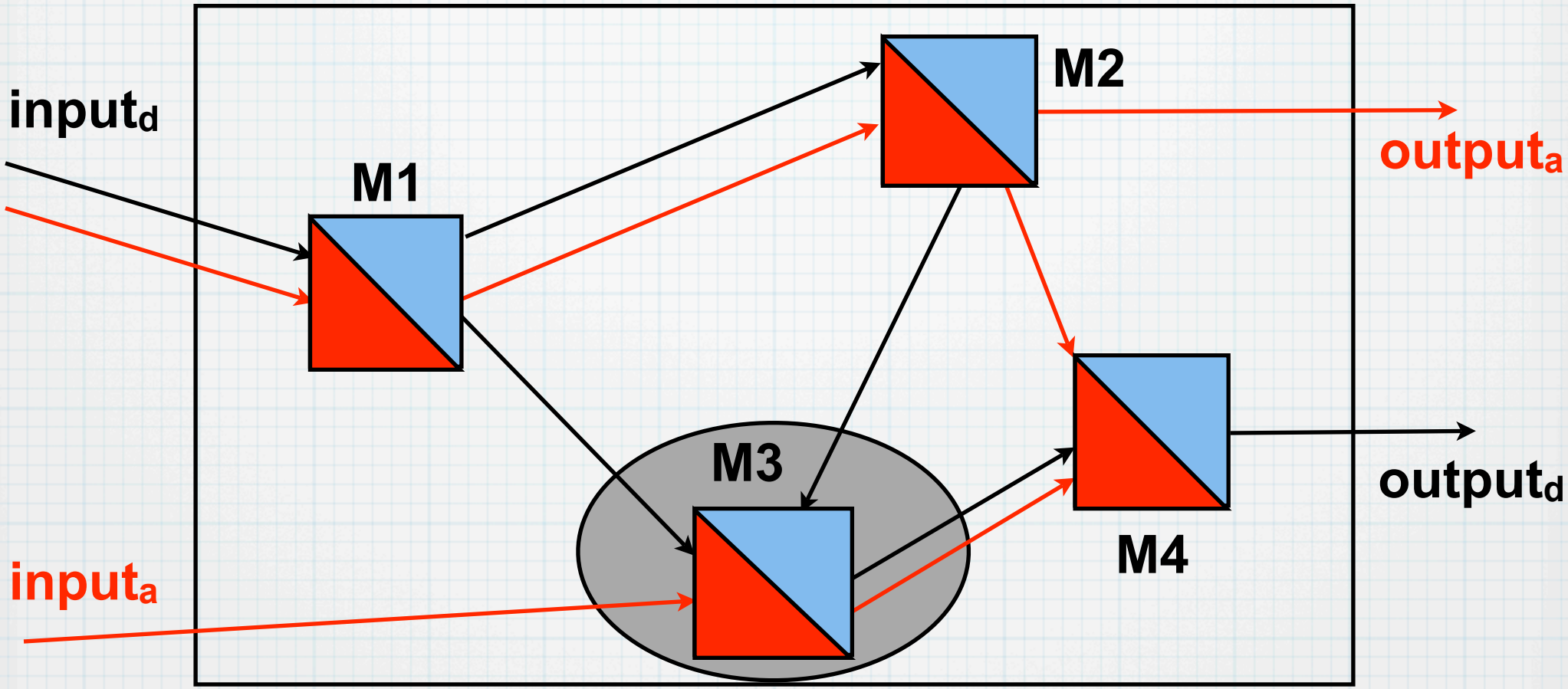


System Slicing



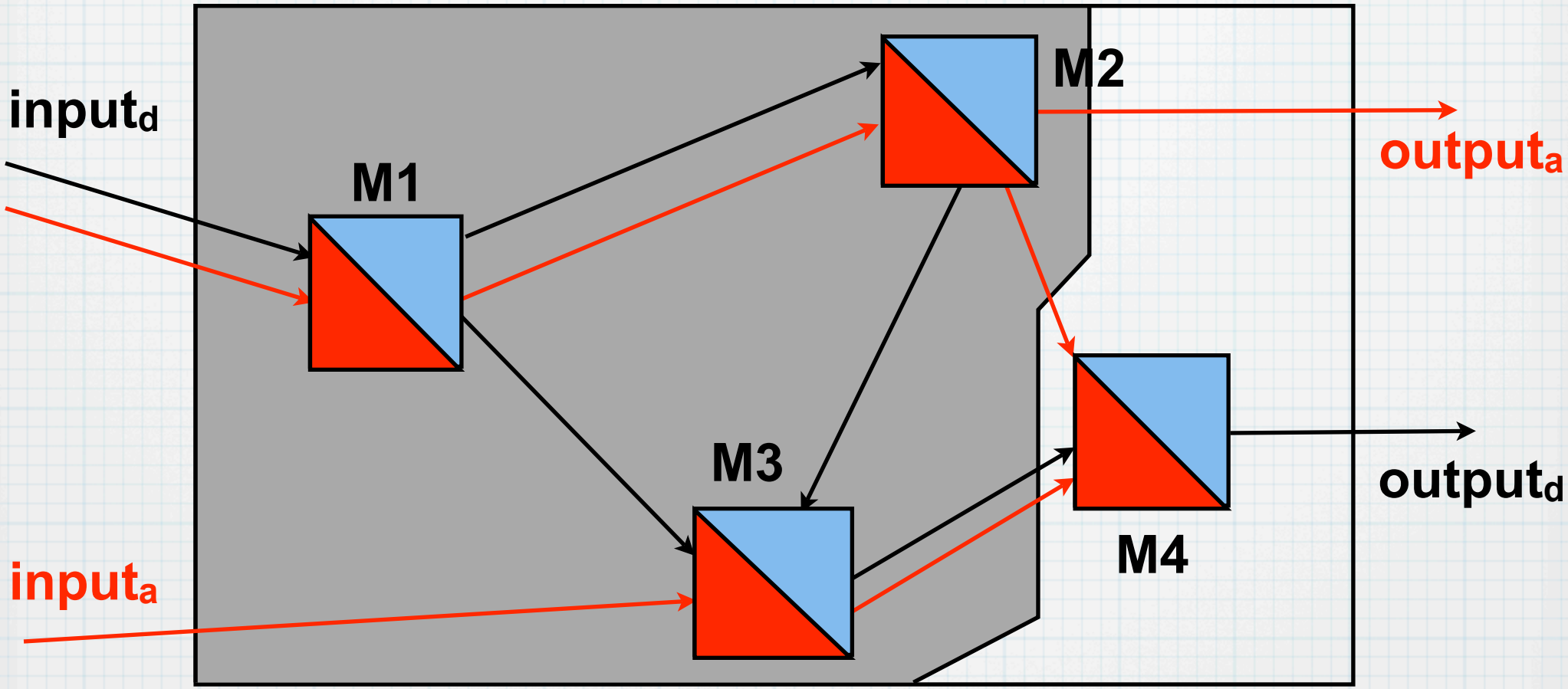

```
AG ( x3 > 0 && useconf_3 = derived)
```

System Slicing



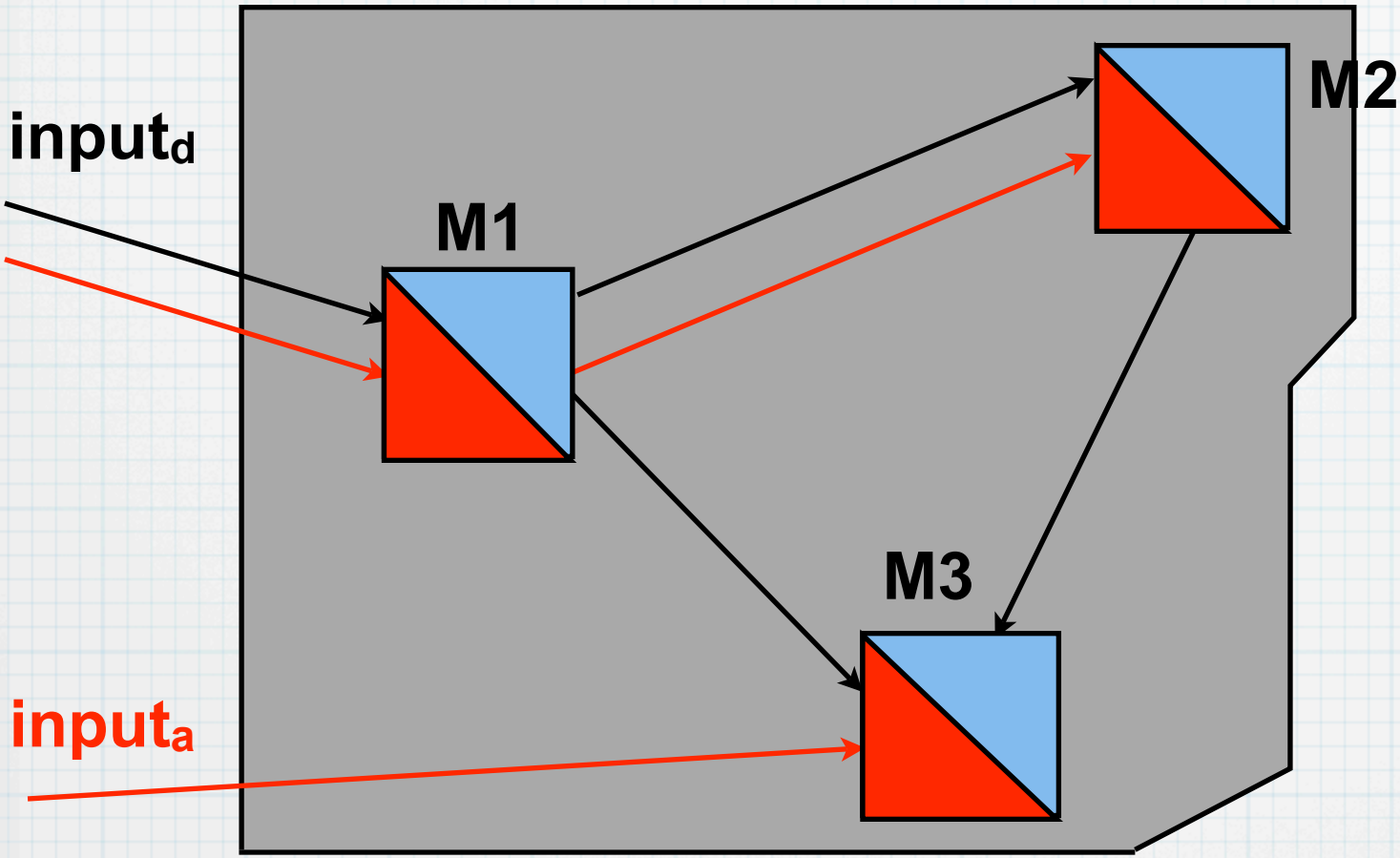

```
AG ( x3 > 0 && useconf_3 = derived)
```

System Slicing



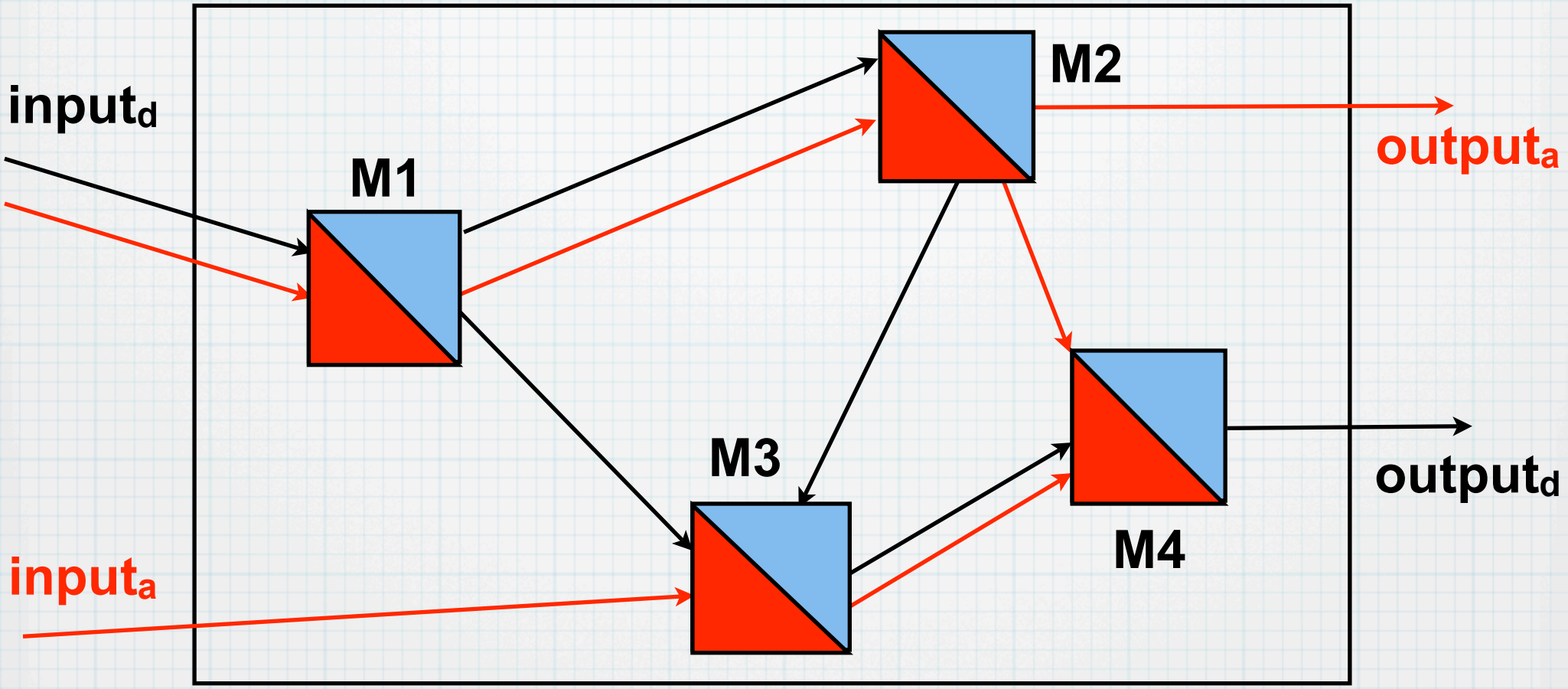

```
AG ( x3 > 0 && useconf_3 = derived)
```

System Slicing

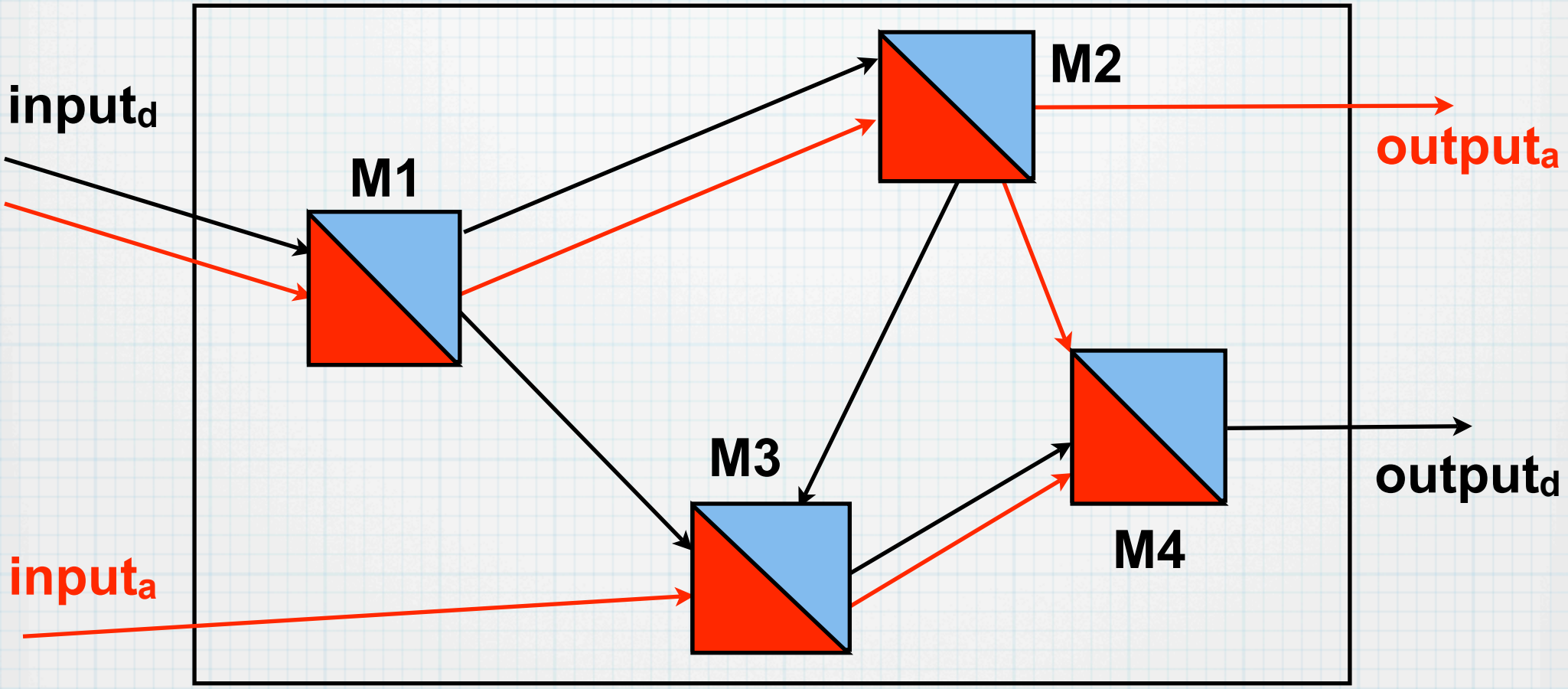



```
AG ( x3 > 0 && useconf_3 = derived)
```

Adaptive Slicing

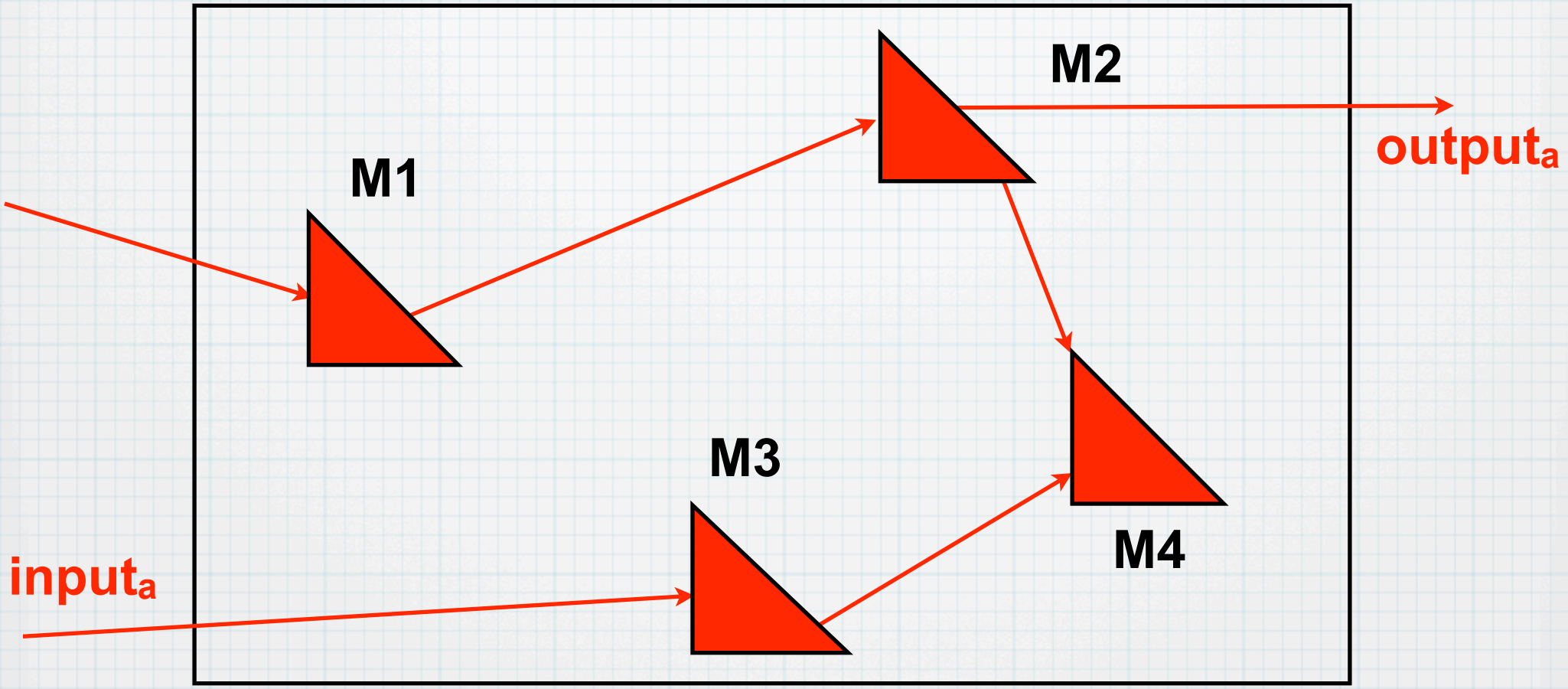


Adaptive Slicing




```
EG ( useconf = derived && EF useconf = derived)
```

Adaptive Slicing




```
EG ( useconf = derived && EF useconf = derived)
```

Experimental Evaluation

Traction Control System

System	Variables	Initial States	Reachable States	Time
Original	522	$\sim$ 10e+29	$\sim$ 10e+156	0.4
Mod Red	264	$\sim$ 10e+25	$\sim$ 10e+79	0.08
Adapt Mod Red	96	$\sim$ 10e+06	$\sim$ 10e+28	0.03

Figures according to Averest 1.9.2

Abstractions

Example: Sign Abstraction

Original System

```
x: int

if (x > 0) y := x + 5
```

Transformed System

```
x : pos, zero, neg

if (x in pos) y := pos
```

Original Property

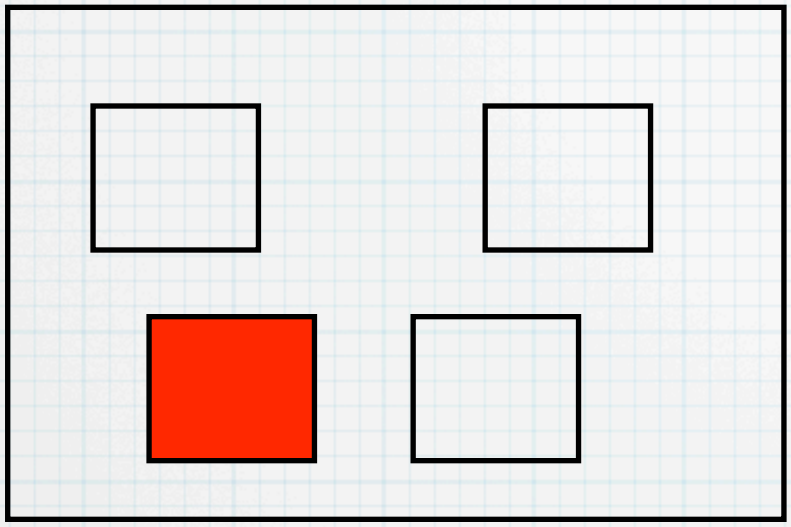
```
AG (y >= 0)
```

Transformed Property

```
A G ( y == zero &&
      y == pos)
```

Consistent Simulation ⟶
Preservation of universal properties

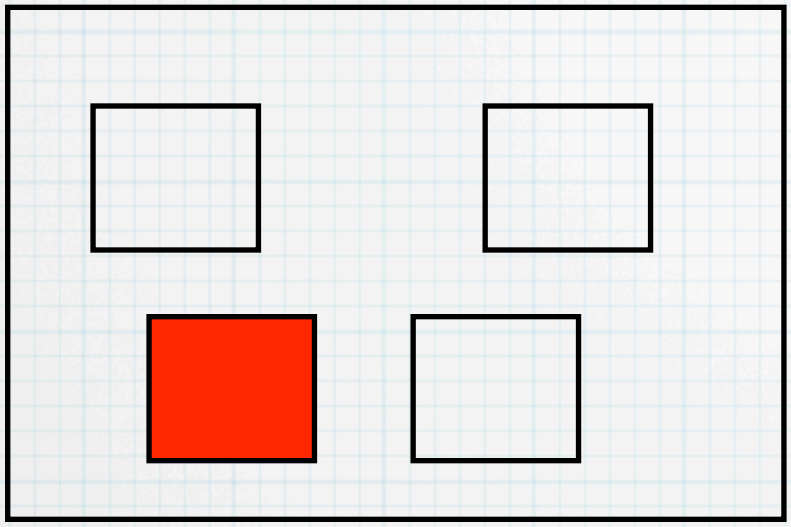
Decomposition



?

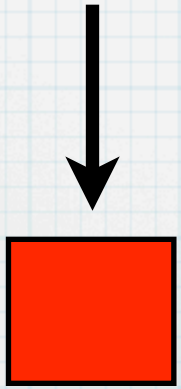
```
AG ( useconf = Off ||
useconf = measured ||
useconf = derived )
```

Decomposition



?

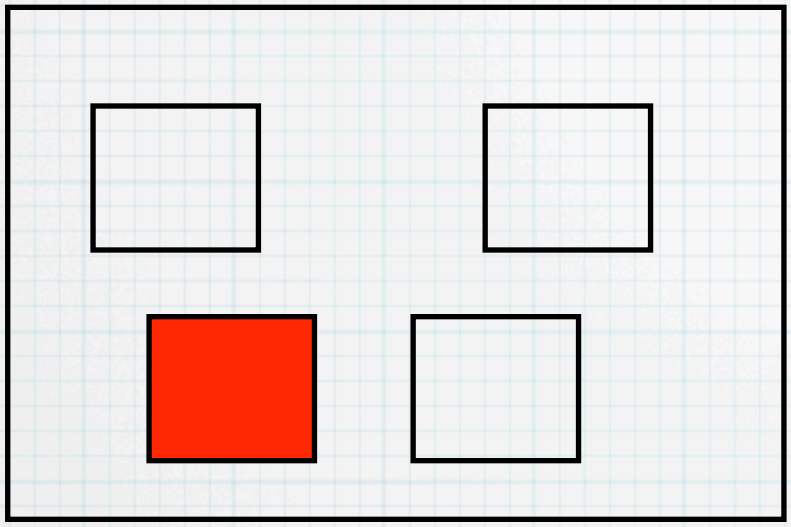
```
AG ( useconf = Off ||
useconf = measured ||
useconf = derived )
```



?

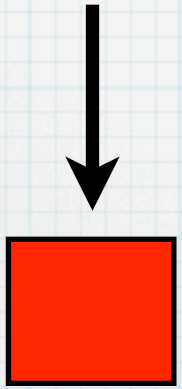
```
AG ( useconf = Off ||
useconf = measured ||
useconf = derived )
```

Decomposition



?

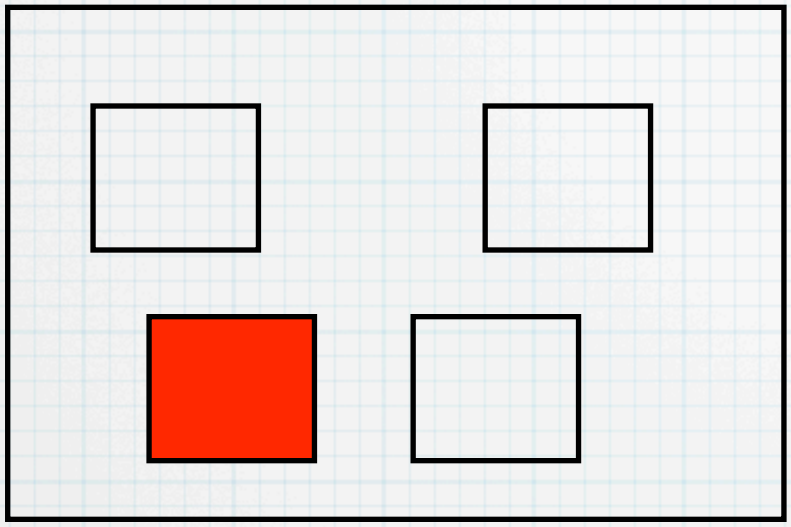
```
AG ( useconf = Off ||
useconf = measured ||
useconf = derived )
```



!

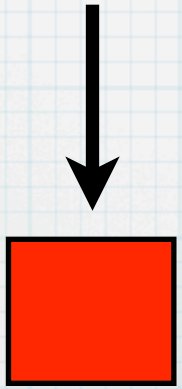
```
AG ( useconf = Off ||
useconf = measured ||
useconf = derived )
```

Decomposition



!

```
AG ( useconf = Off ||
useconf = measured ||
useconf = derived )
```



!

```
AG ( useconf = Off ||
useconf = measured ||
useconf = derived )
```

Conclusion

- Approach for Integrating Model-based Development with Formal Verification
- SAS as Semantical Model for Adaptive Systems with Clear Separation of Functional and Adaptive Behaviour and Modular Structure
- Property Preserving Transformations verified by Translation Validation

Future Work

- Propagation of Verification Results back to Modelling Level
- Intuitive Way for Property Specification at Modelling Level
- Further Automatisation of Reductions