Rigorous Component-based Design in BIP

18th of October, 2019

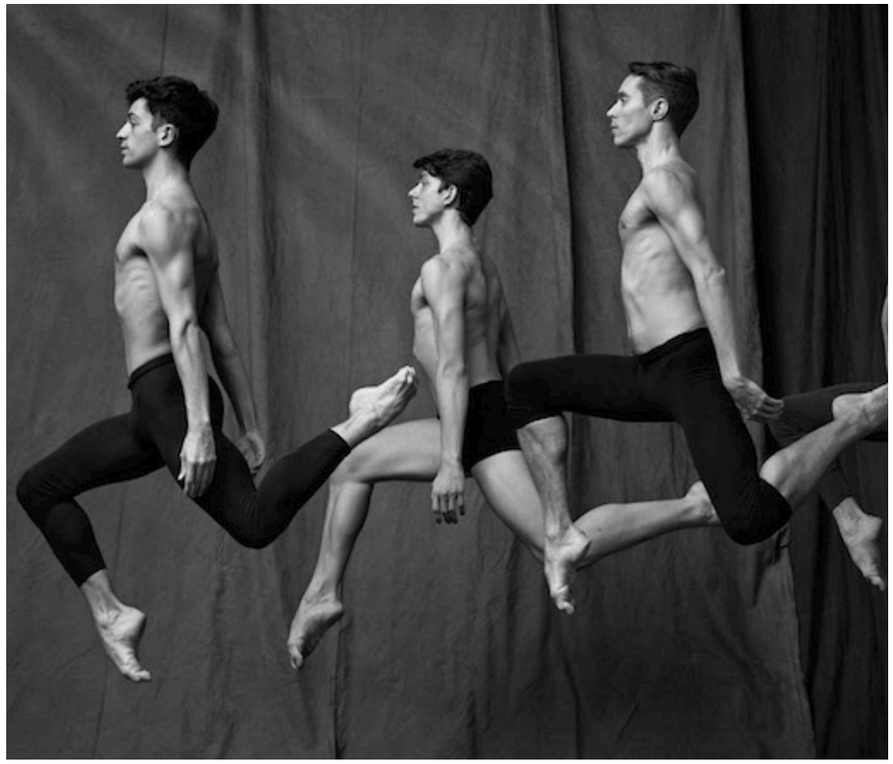
IMBSA @ Thessaloniki

Simon Bliudze
Inria Lille – Nord Europe

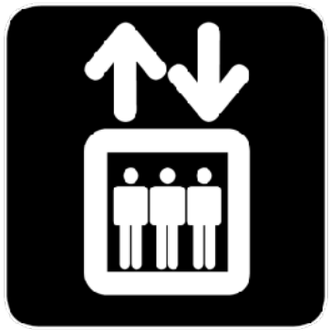
Région Hauts-de-France

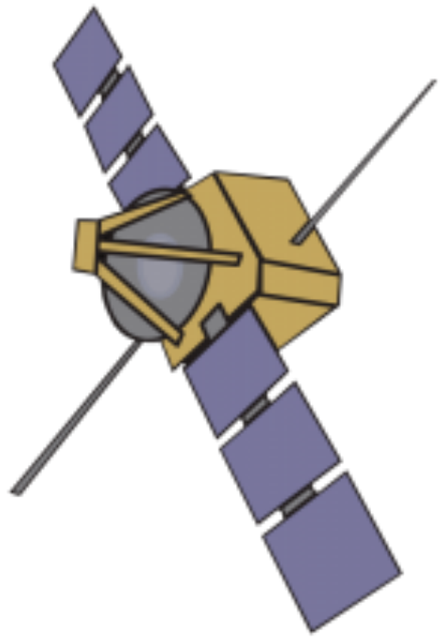
informatics mathematics

Concurrency...



...is everywhere!





Embedded

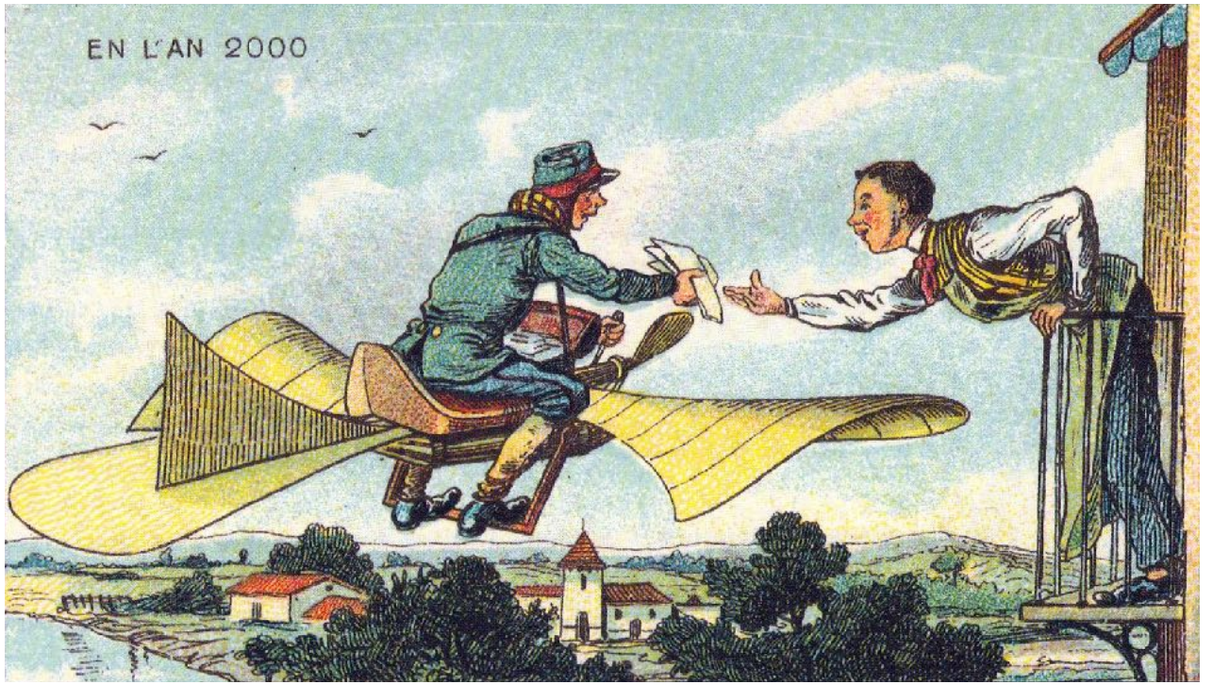
Infrastructure

Platform

Services

...you name it!

Coordination



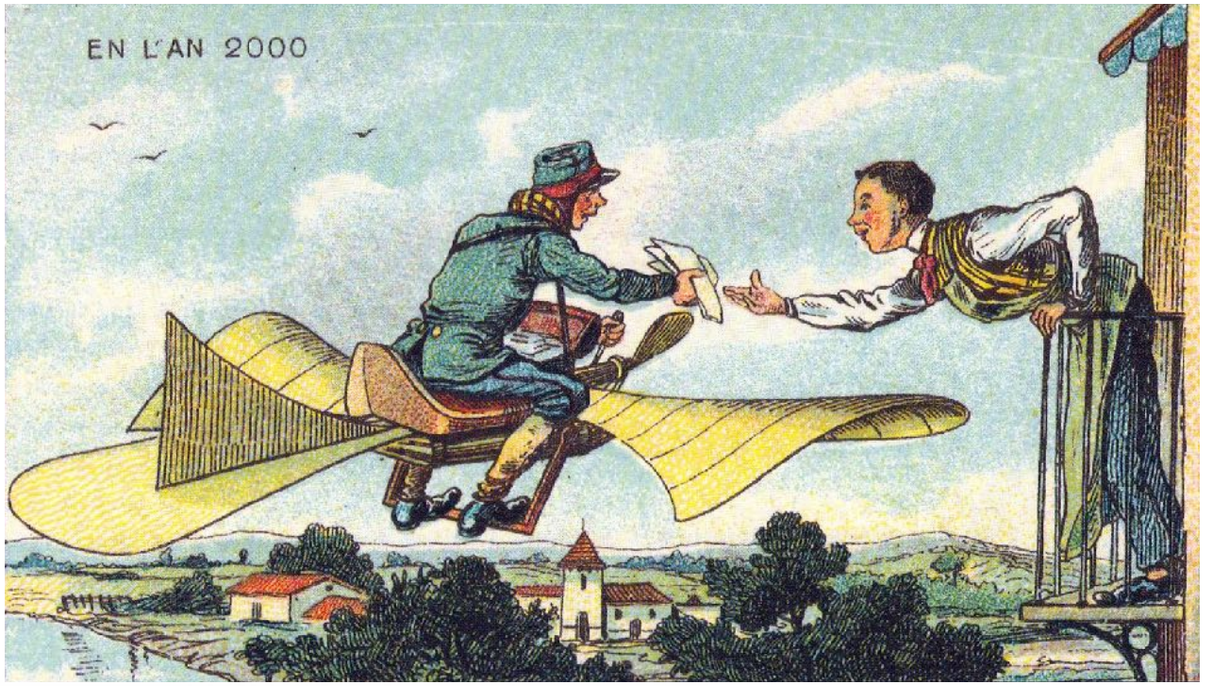
Control-centric

- Synchronisation is primitive
- Locks, semaphores etc.
- Concurrent execution
- Critical systems

Data-centric

- Data exchange is primitive
- Messages, split-join etc.
- Distributed execution
- Data-intensive computation

Coordination



The two views are complementary

Control-centric

- Synchronisation is primitive
- Locks, semaphores etc.
- Concurrent execution
- Critical systems

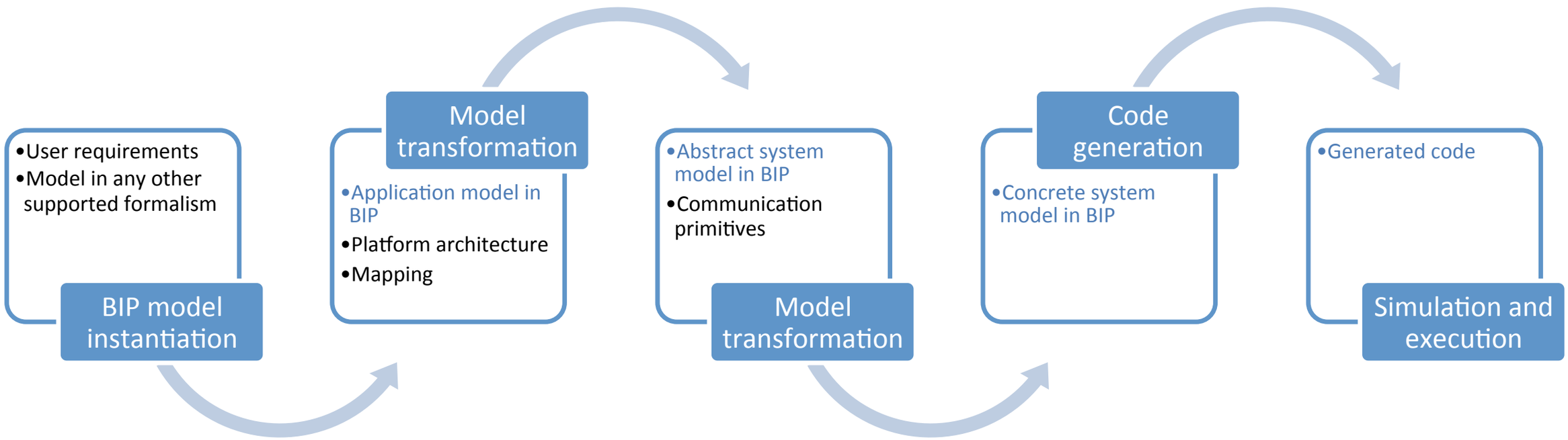
Data-centric

- Data exchange is primitive
- Messages, split-join etc.
- Distributed execution
- Data-intensive computation

Semaphores, locks, monitors, etc.

Coordination based on low-level primitives rapidly becomes unpractical.

Rigorous System Design flow

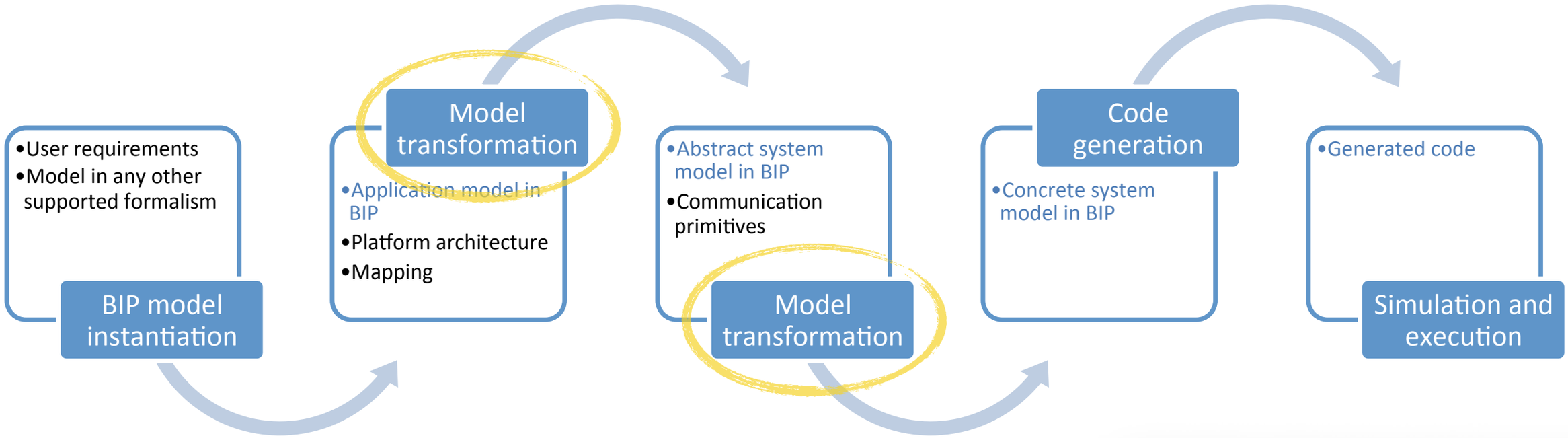


A series of semantics-preserving transformations

Correctness decomposed into
- correctness of transformations
- correctness of high-level models

Final implementation is **correct by construction**

Rigorous System Design flow



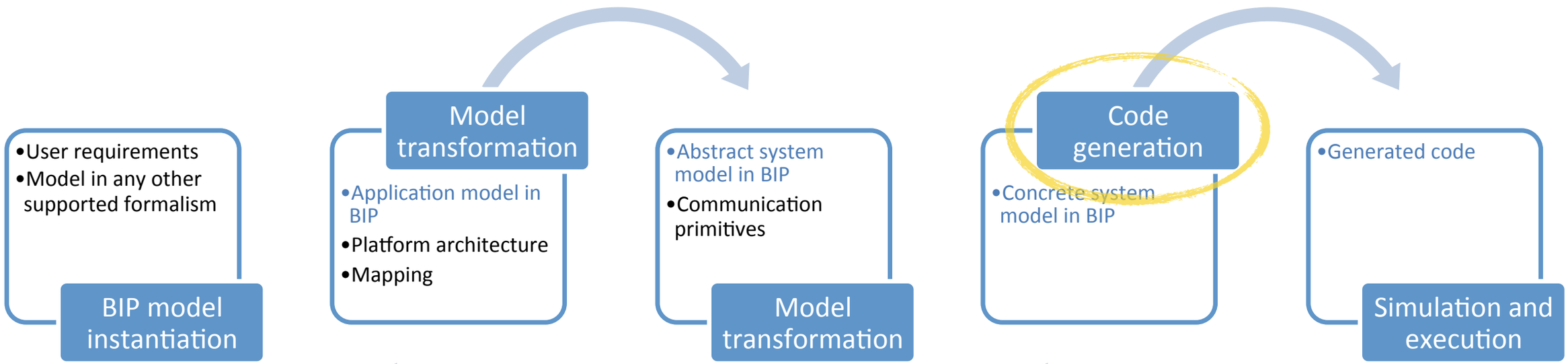
A series of semantics-preserving transformations

Correctness decomposed into
- correctness of transformations
- correctness of high-level models

Final implementation is **correct by construction**

- ☐ Unifying modelling framework

Rigorous System Design flow



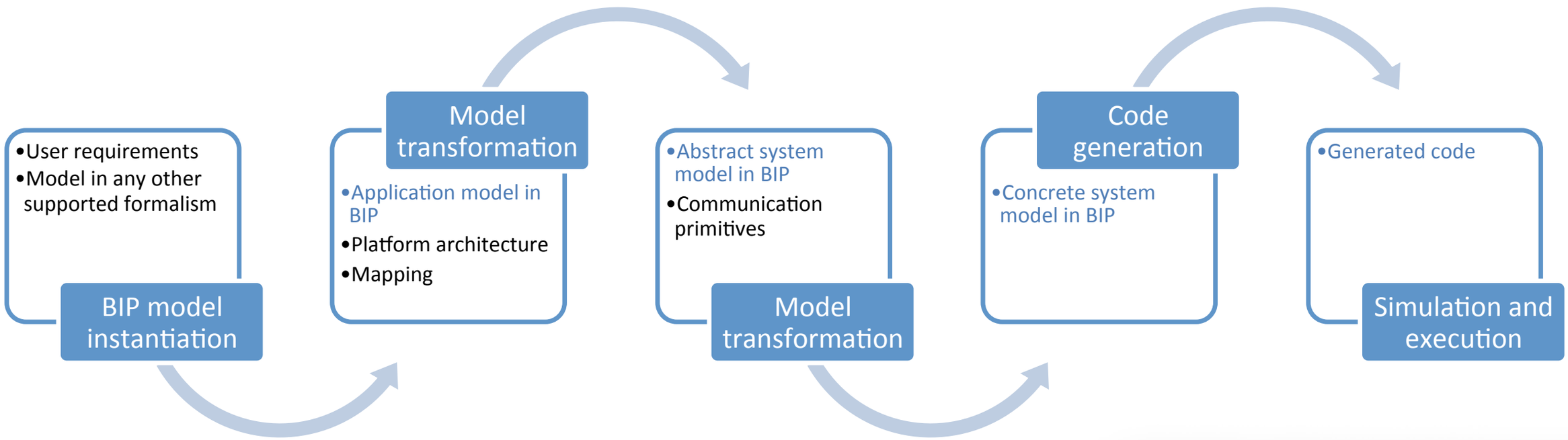
A series of semantics-preserving transformations

Correctness decomposed into
- correctness of transformations
- correctness of high-level models

Final implementation is **correct by construction**

- ☐ Unifying modelling framework
- ☐ Operational semantics

Rigorous System Design flow



A series of semantics-preserving transformations

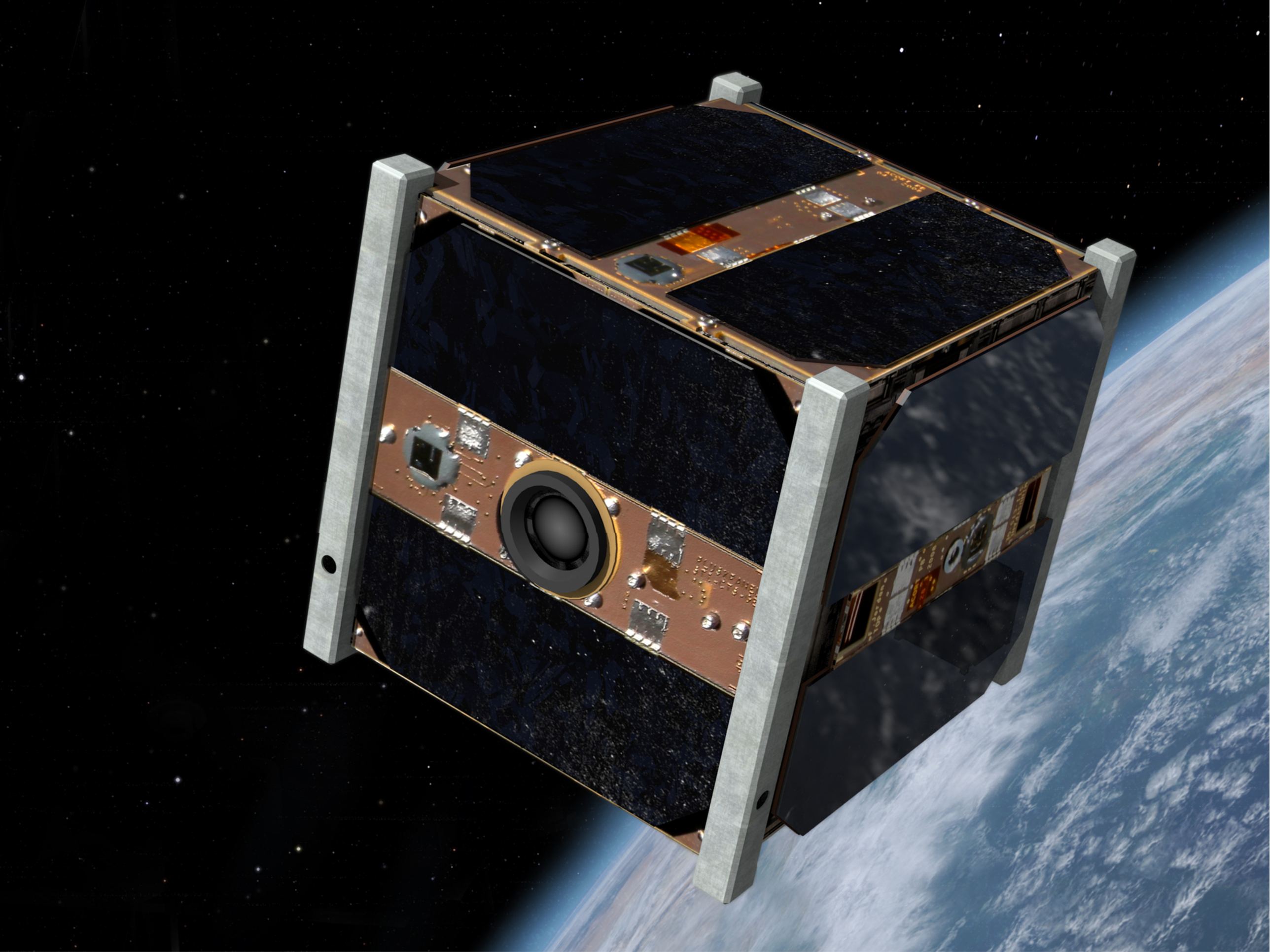
Correctness decomposed into

correctness of transformations

correctness of high-level models

Final implementation is **correct by construction**

- ☐ Unifying modelling framework
- ☐ Operational semantics
- ☐ Method(s) to design correct models



Satellite software design

A collaboration with the EPFL Space Engineering Center

Component-based design in BIP of the control software for a nano-satellite

Control and Data Management System (CDMS)

Communication with other subsystems through an I^2C bus

A collaboration with ThalesAlenia Space (France) and Aristotle University of Thessaloniki (Greece)

"Catalogue of System and Software Properties"

Funded by ESA

eSpace SPACE ENGINEERING CENTER

ThalesAlenia Space — A Thales / Finmeccanica Company

ARISTOTLE UNIVERSITY OF THESSALONIKI

Satellite software design

A collaboration with the EPFL Space Engineering Center

Component-based design in BIP of the control software for a nano-satellite

Control and Data Management System (CDMS)

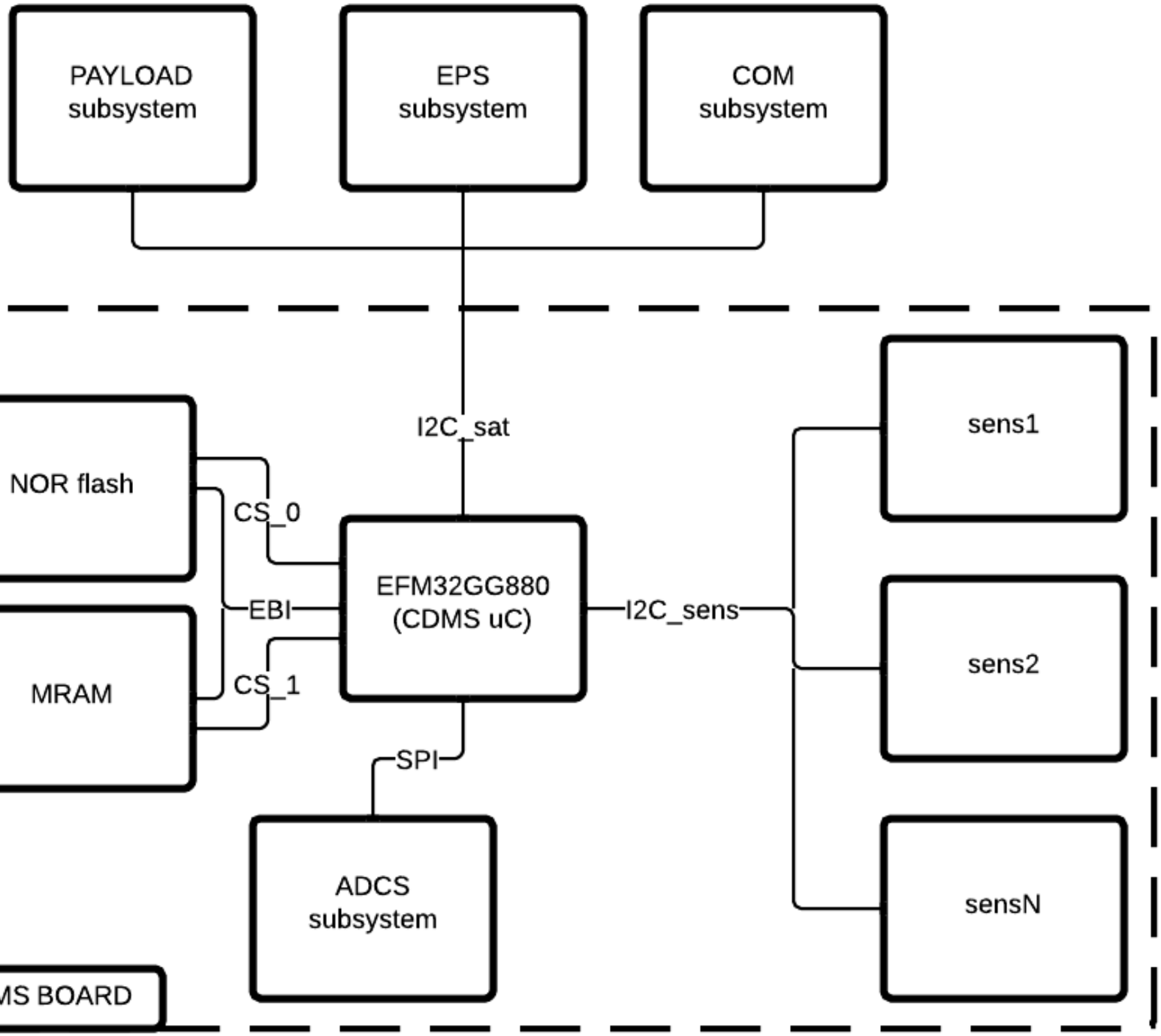
BIP = Behaviour-Interaction-Priority

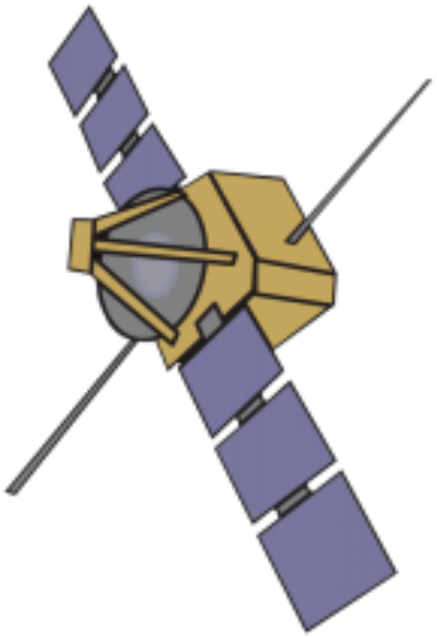
A collaboration with ThalesAlenia Space (France) and Aristotle University of Thessaloniki (Greece)

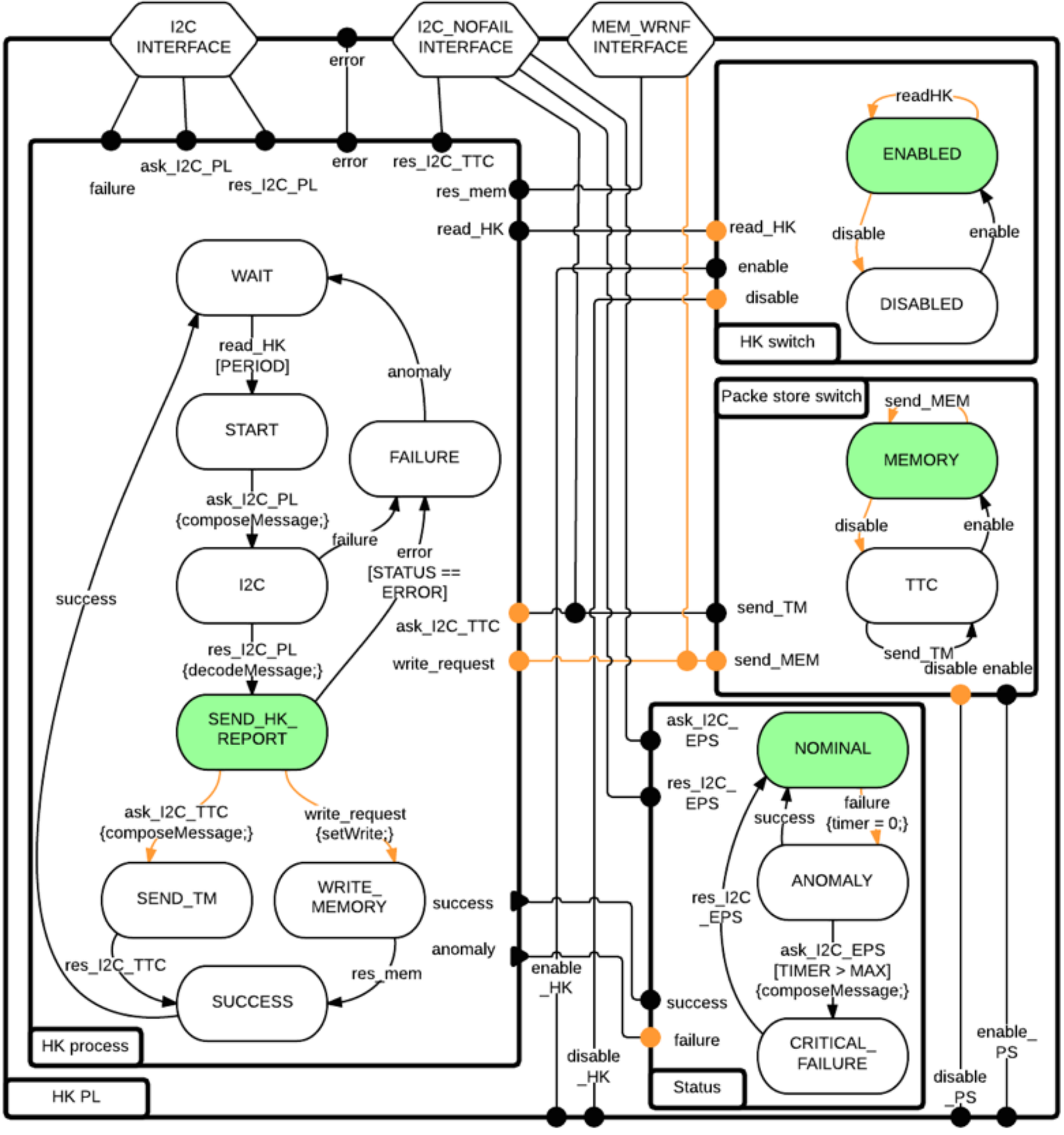
"Catalogue of System and Software Properties"

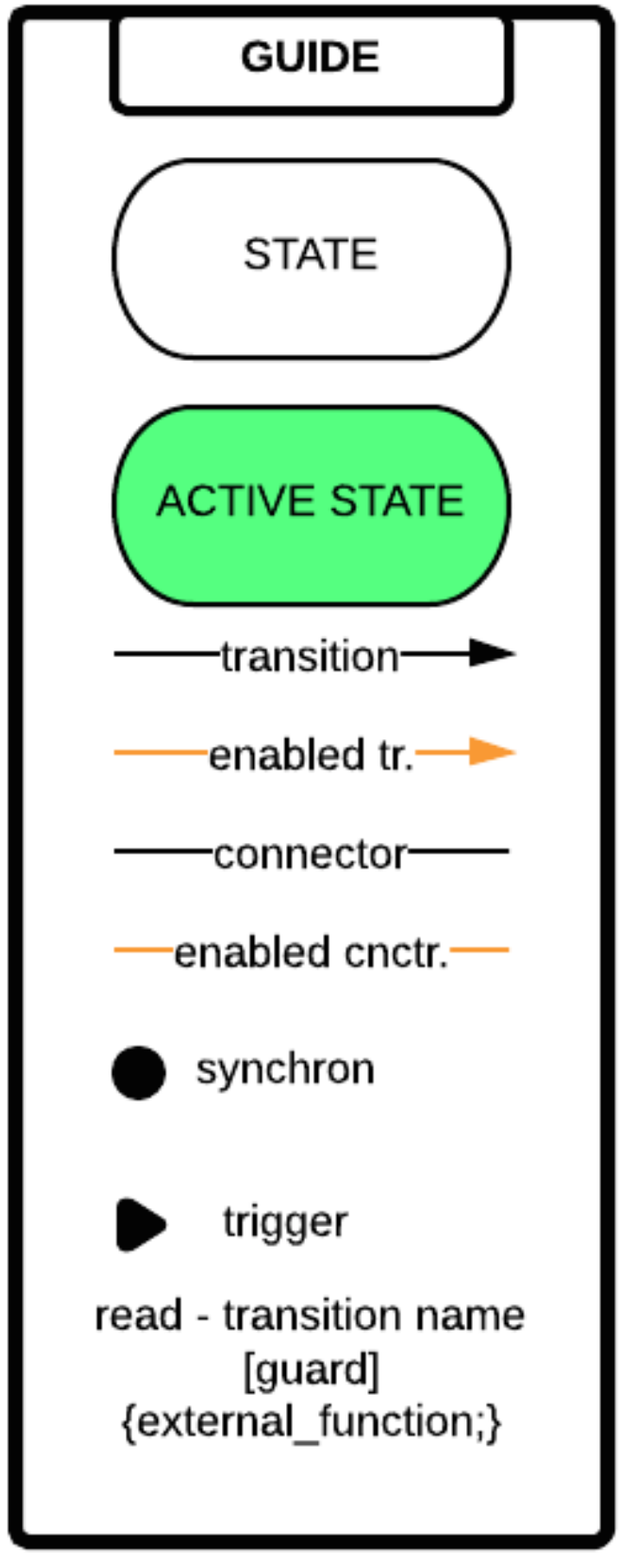
Funded by ESA

CubETH: CDMS architecture



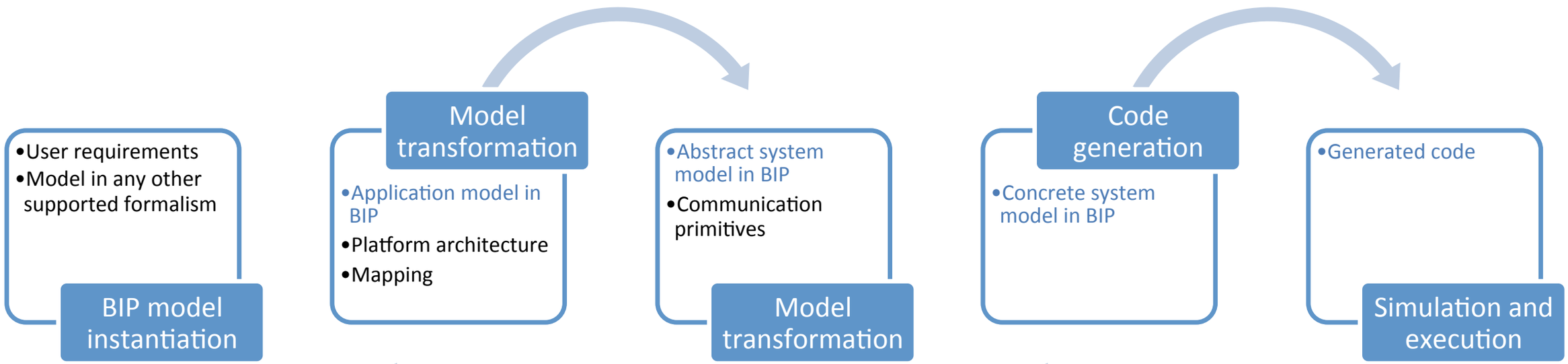






[Pagnamenta MSc 2014]

Rigorous System Design flow



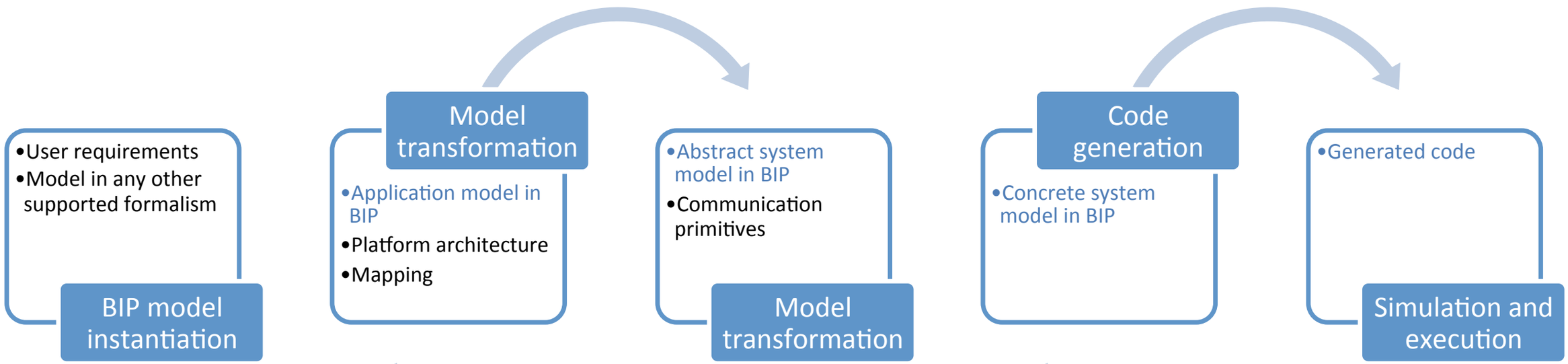
A series of semantics-preserving transformations

Correctness decomposed into
- correctness of transformations
- correctness of high-level models

Final implementation is **correct by construction**

- ☐ Unifying modelling framework
- ☐ Operational semantics
- ☐ Method(s) to design correct models

Rigorous System Design flow



A series of semantics-preserving transformations

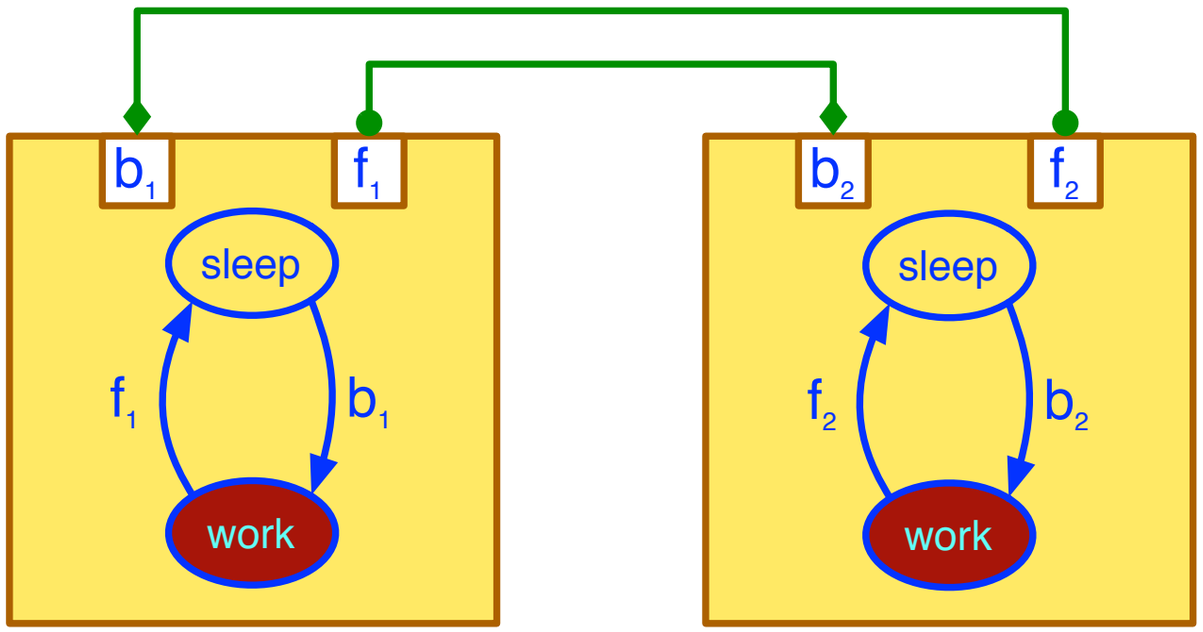
Correctness decomposed into
- correctness of transformations
- correctness of high-level models

Final implementation is **correct by construction**

- ☑ Unifying modelling framework
- ☐ Operational semantics
- ☐ Method(s) to design correct models

se·man·tic to meaning or arising from
symbols; semantic change; semanti
semantic(ós) adj. [1655–65; < Gk sēmanti
-tics (si man/tiks), n. (
dealing with the su
structured in langua
the branch of se
signs or sy
se·man·tics
linguistics
meaning is stu
over time. 2. the
tionship between signs
meaning, or an interpretation
Let's not argue
—**se·man/ti·c**

BIP by example: Mutual exclusion



Interaction model:

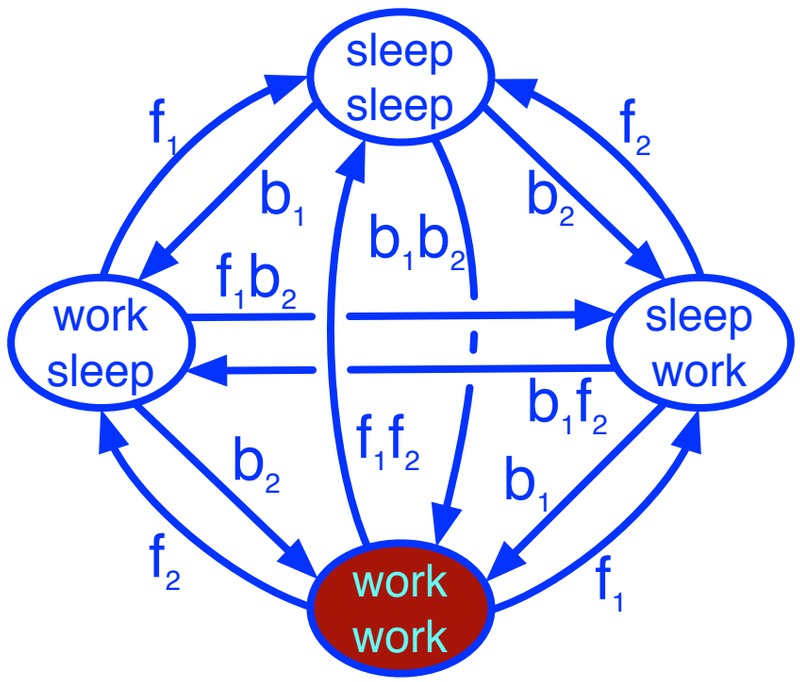
$$\{b_1,\ f_1,\ b_2,\ f_2,\ b_1f_2,\ b_2f_1\}$$

Maximal progress:

$$b_1 < b_1f_2,\ b_2 < b_2f_1$$

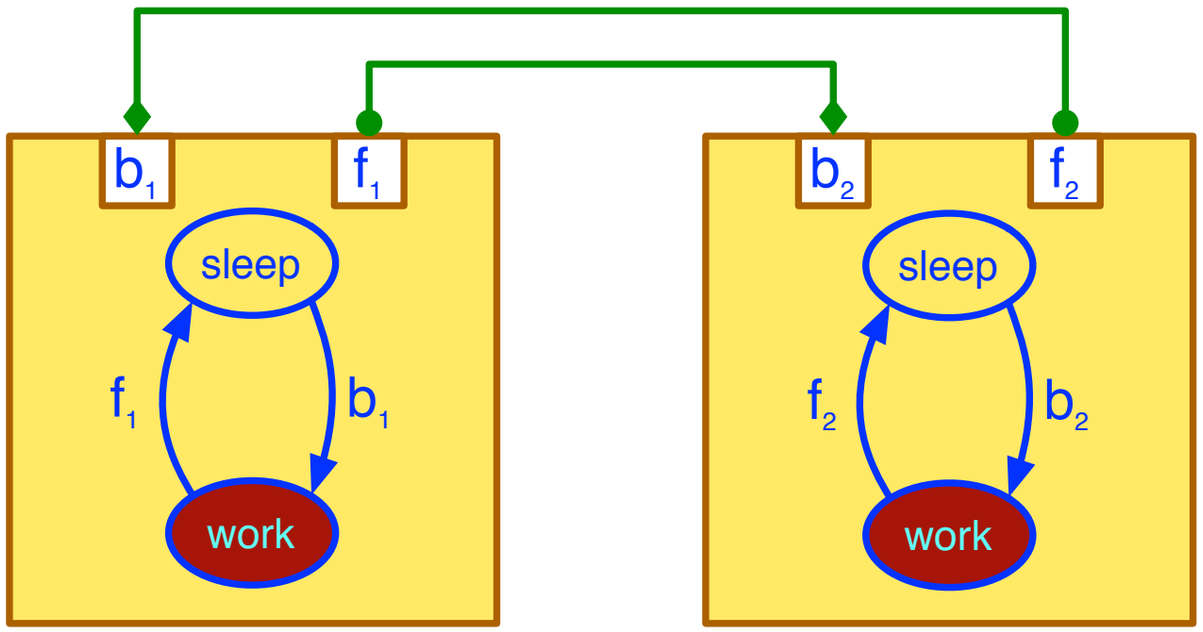
Design view

Semantic view



[Bliudze, Sifakis, EMSOFT '07]

BIP by example: Mutual exclusion



Interaction model:

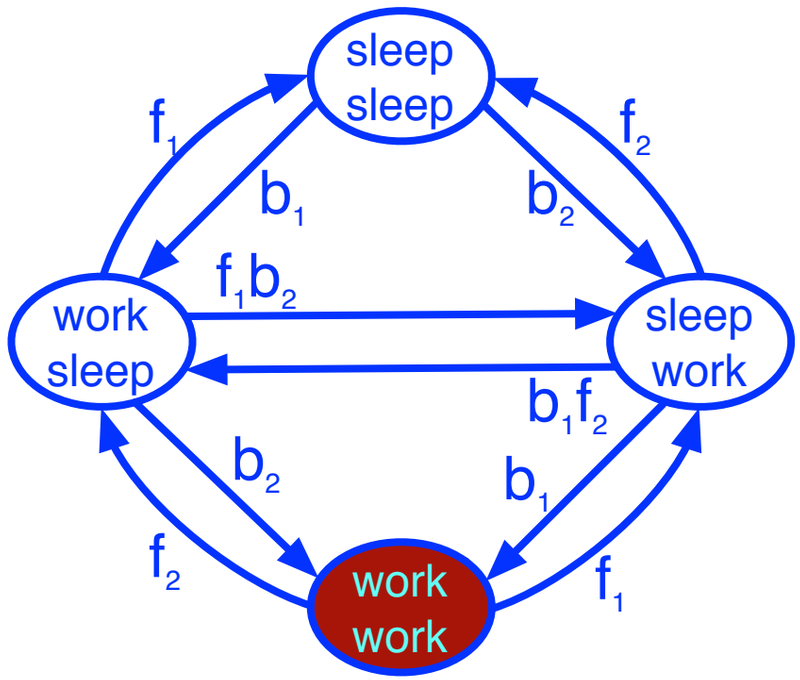
$$\{b_1,\ f_1,\ b_2,\ f_2,\ b_1f_2,\ b_2f_1\}$$

Maximal progress:

$$b_1 < b_1f_2,\ b_2 < b_2f_1$$

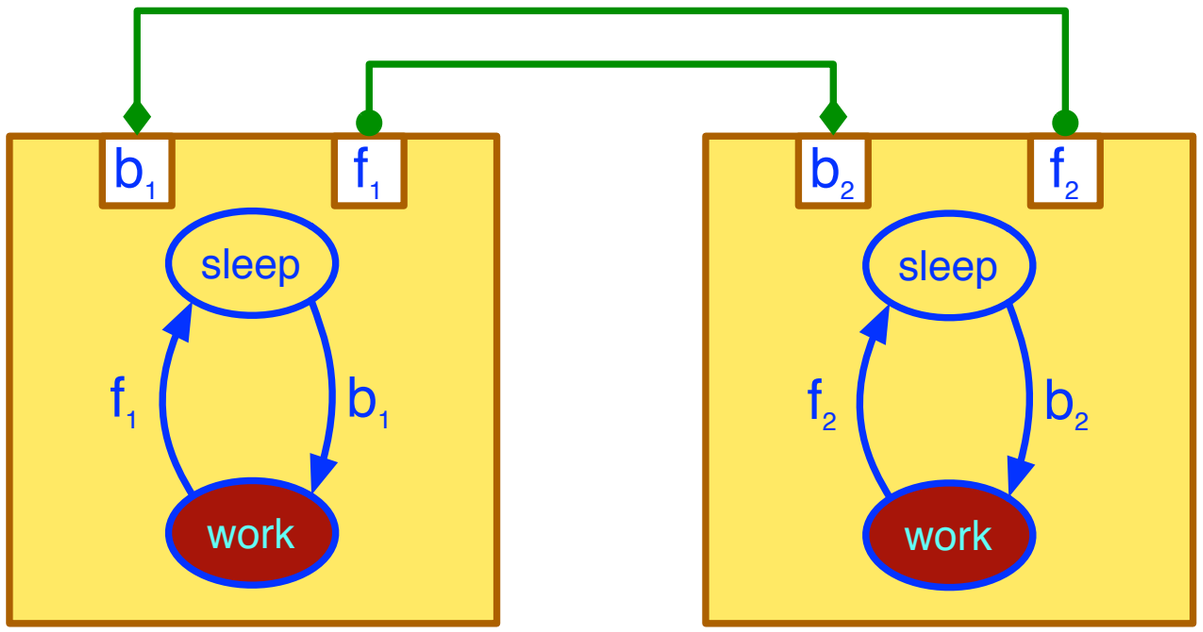
Design view

Semantic view



[Bliudze, Sifakis, EMSOFT '07]

BIP by example: Mutual exclusion



Interaction model:

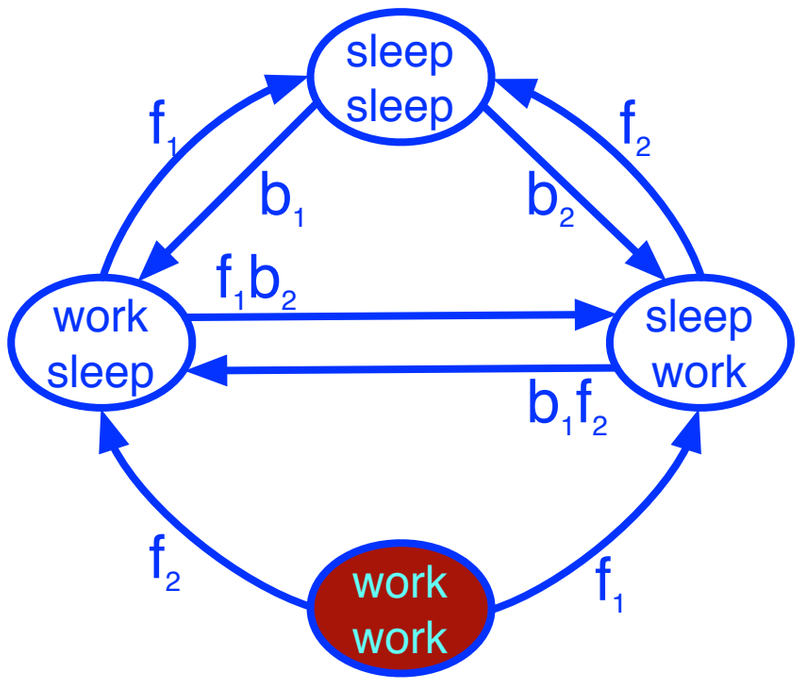
$$\{b_1,\ f_1,\ b_2,\ f_2,\ b_1f_2,\ b_2f_1\}$$

Maximal progress:

$$b_1 < b_1f_2,\ b_2 < b_2f_1$$

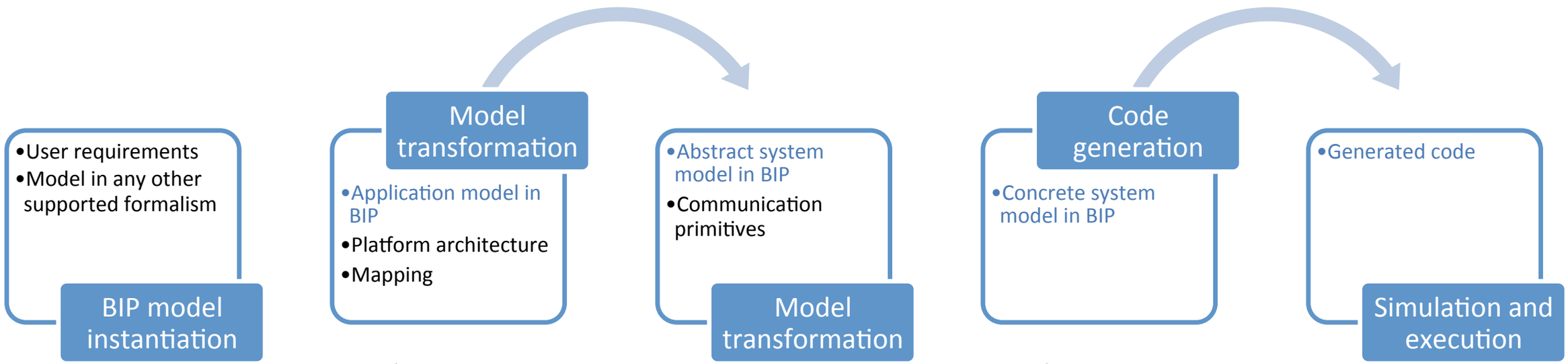
Design view

Semantic view



[Bliudze, Sifakis, EMSOFT '07]

Rigorous System Design flow



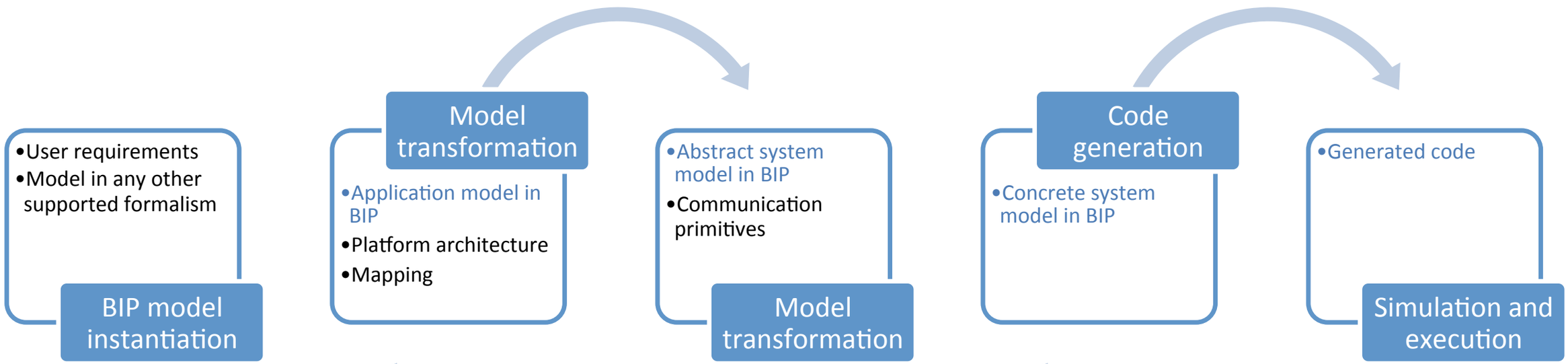
A series of semantics-preserving transformations

Correctness decomposed into
- correctness of transformations
- correctness of high-level models

Final implementation is **correct by construction**

- ☑ Unifying modelling framework
- ☐ Operational semantics
- ☐ Method(s) to design correct models

Rigorous System Design flow



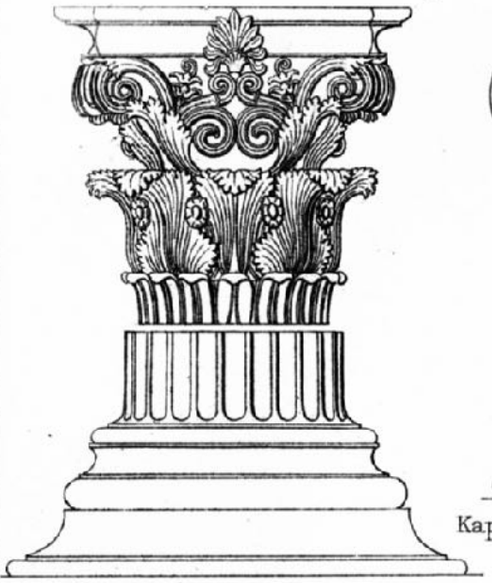
A series of semantics-preserving transformations

Correctness decomposed into
- correctness of transformations
- correctness of high-level models

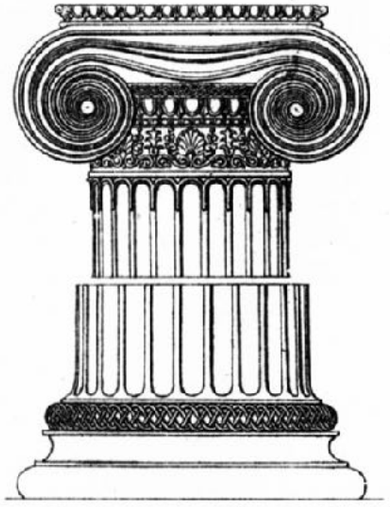
Final implementation is **correct by construction**

- ☑ Unifying modelling framework
- ☑ Operational semantics
- ☐ Method(s) to design correct models

Korinthische Ordnung

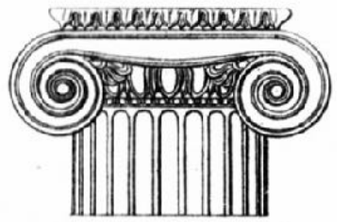


Kapitäl u. Basis vom Monument des Lysikrates zu Athen.

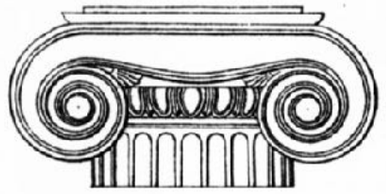


Kapitäl u. Basis vom Tempel der Athene zu Athen.

Jonische Ordnung



Kapitäl vom Tempel der Athene zu Priene.

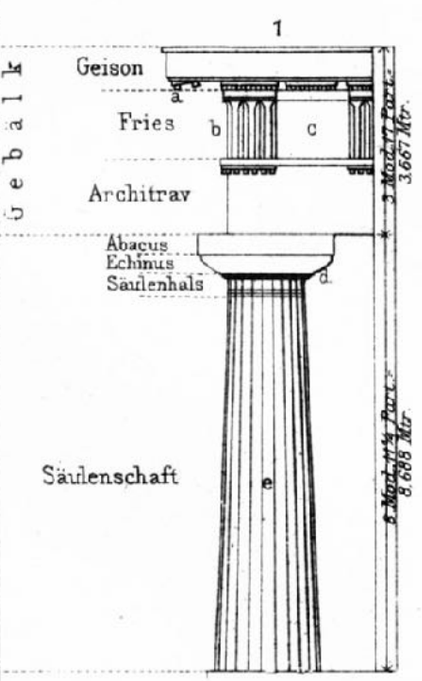


Kapitäl vom Tempel am Ilissos zu Athen.

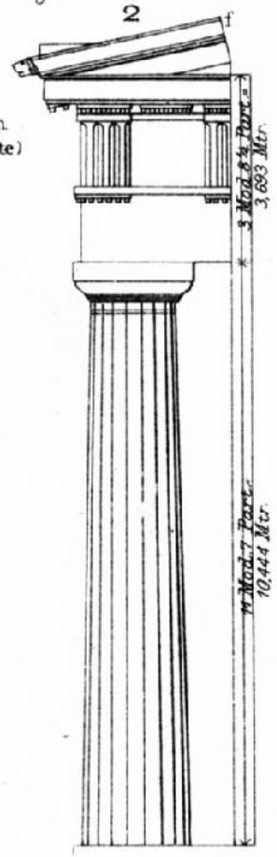
Korinthisch Römisch-Korinthisch.

Zu 1. 2. 3.

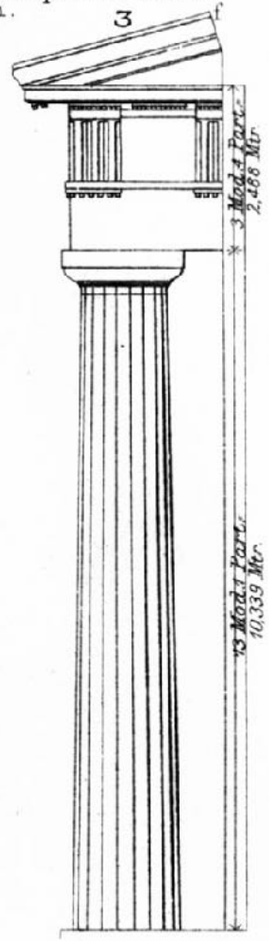
a Mutuli (Dielenköpfe)
b Triglyphen (Dreischlitze)
c Metopen
d Riemchen
e Kannelirungen
f Sima (Rinnleiste)



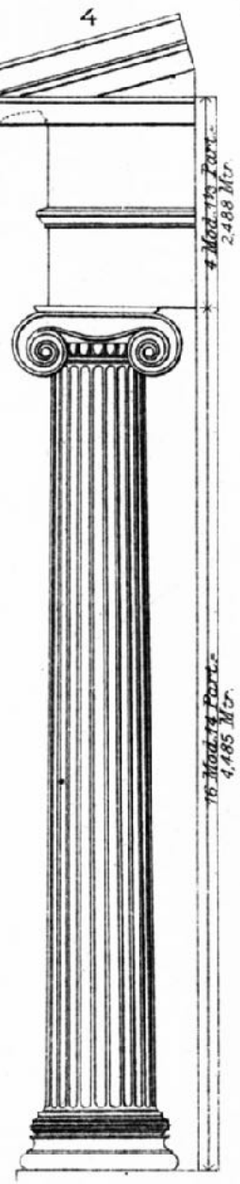
Vom Tempel in Pästum



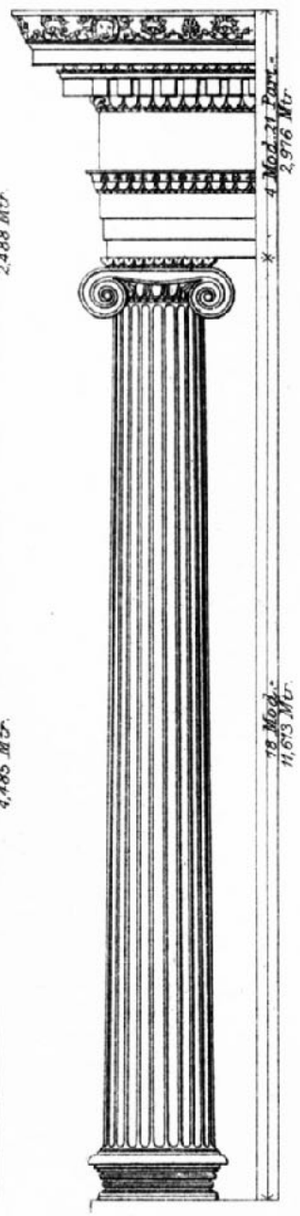
Vom Parthenon in Athen



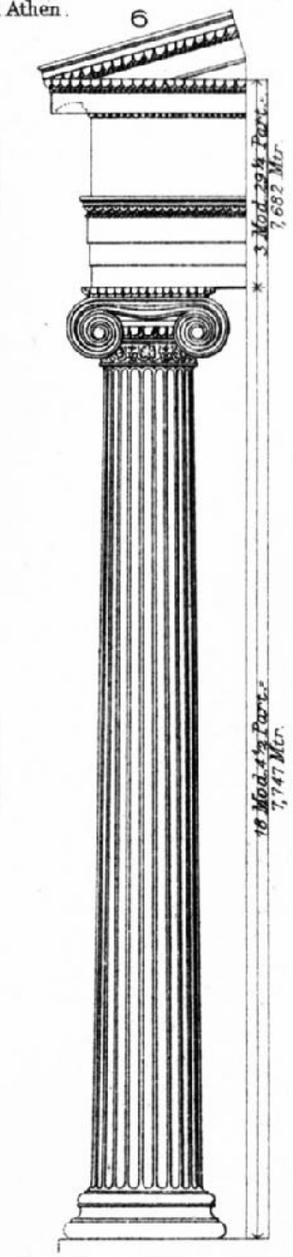
Vom Tempel des Nemëischen Zeus.



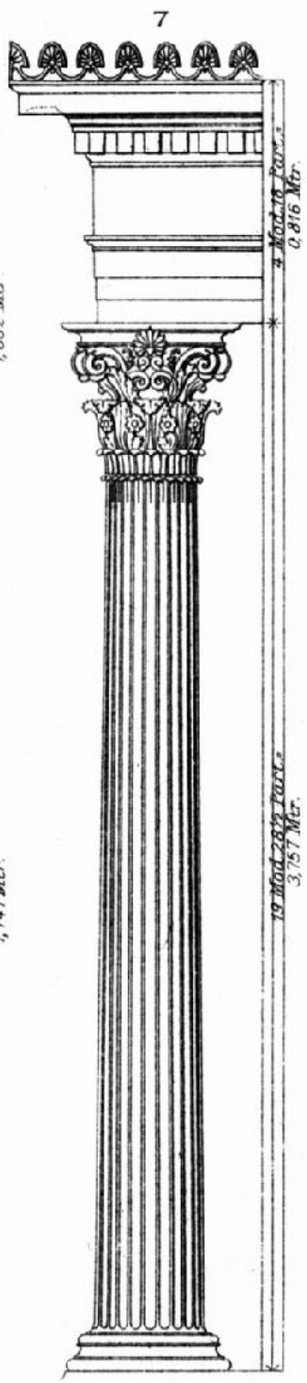
Vom Tempel am Ilissos in Athen



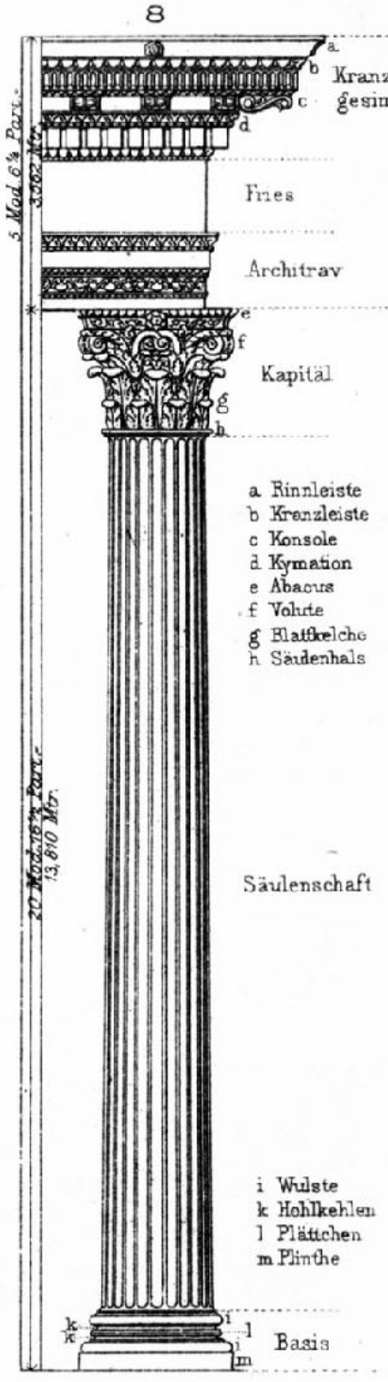
Vom Tempel d. Athene Polias in Priene.



Vom Tempel d. Athene Polias in Athen.

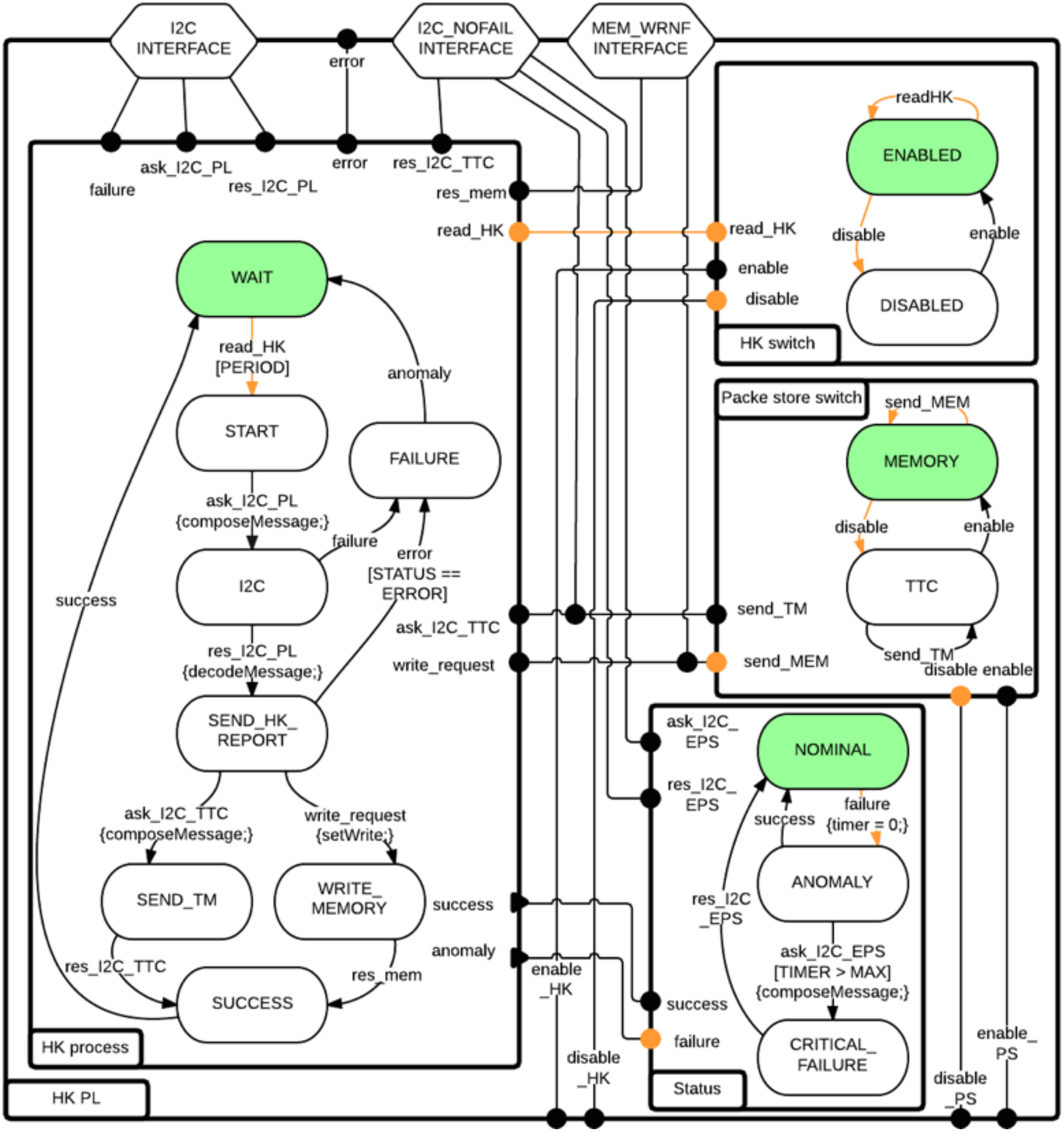


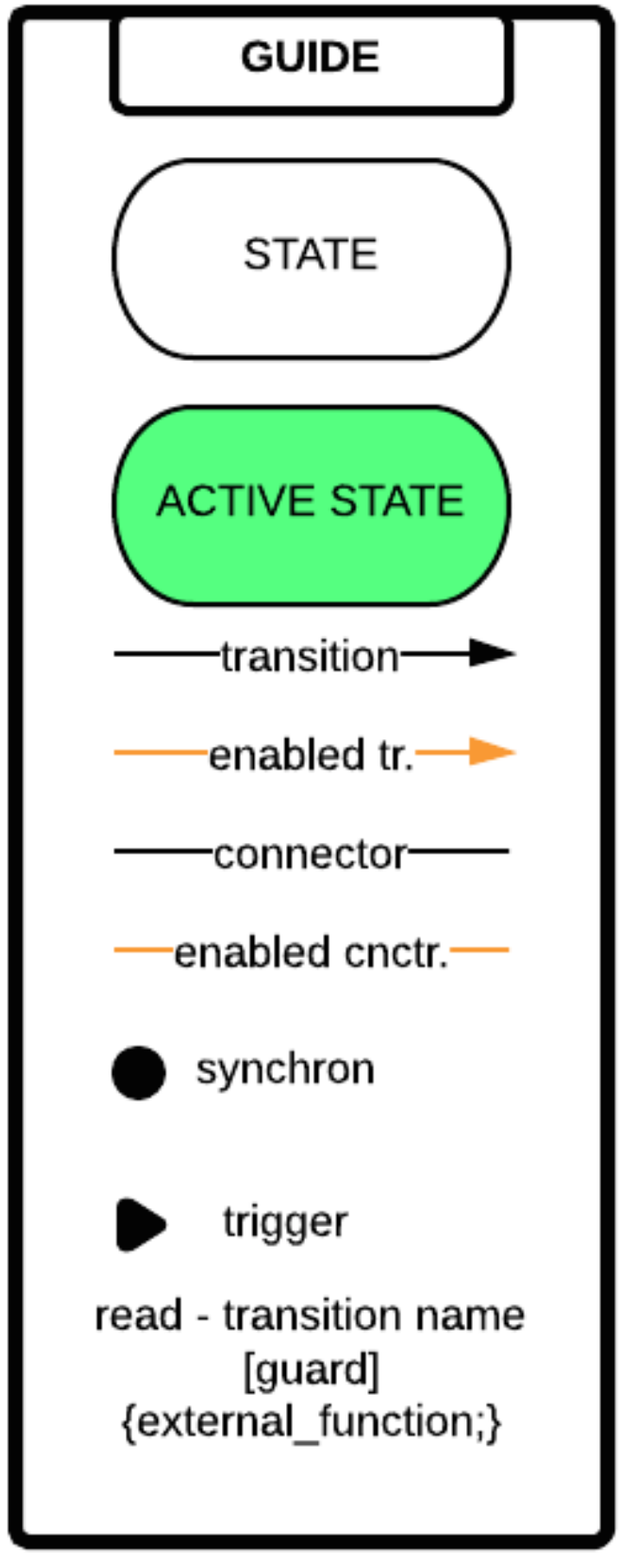
Vom Monument des Lysikrates in Athen.



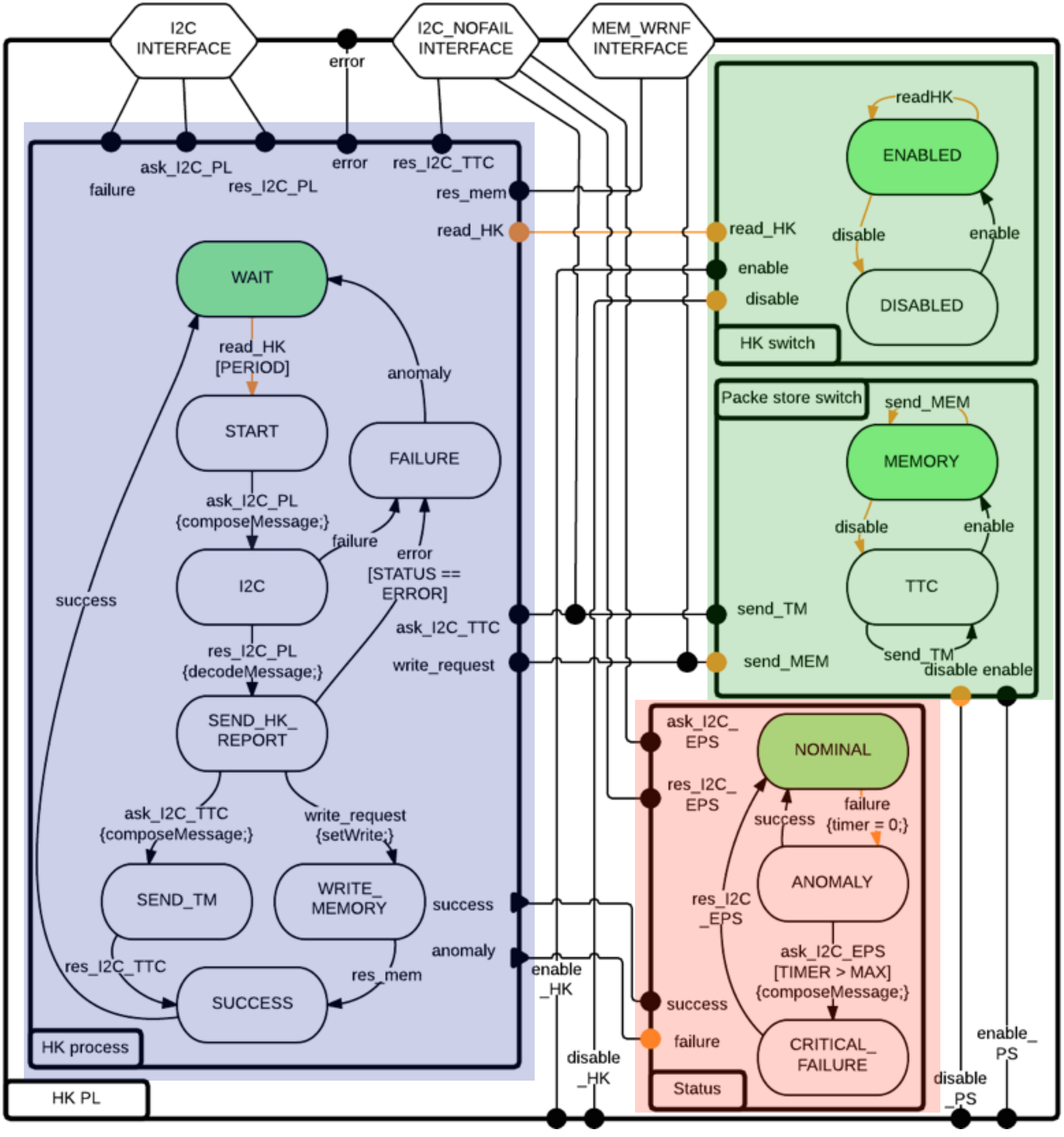
Vom Tempel d. Jupiter-Stator in Rom.

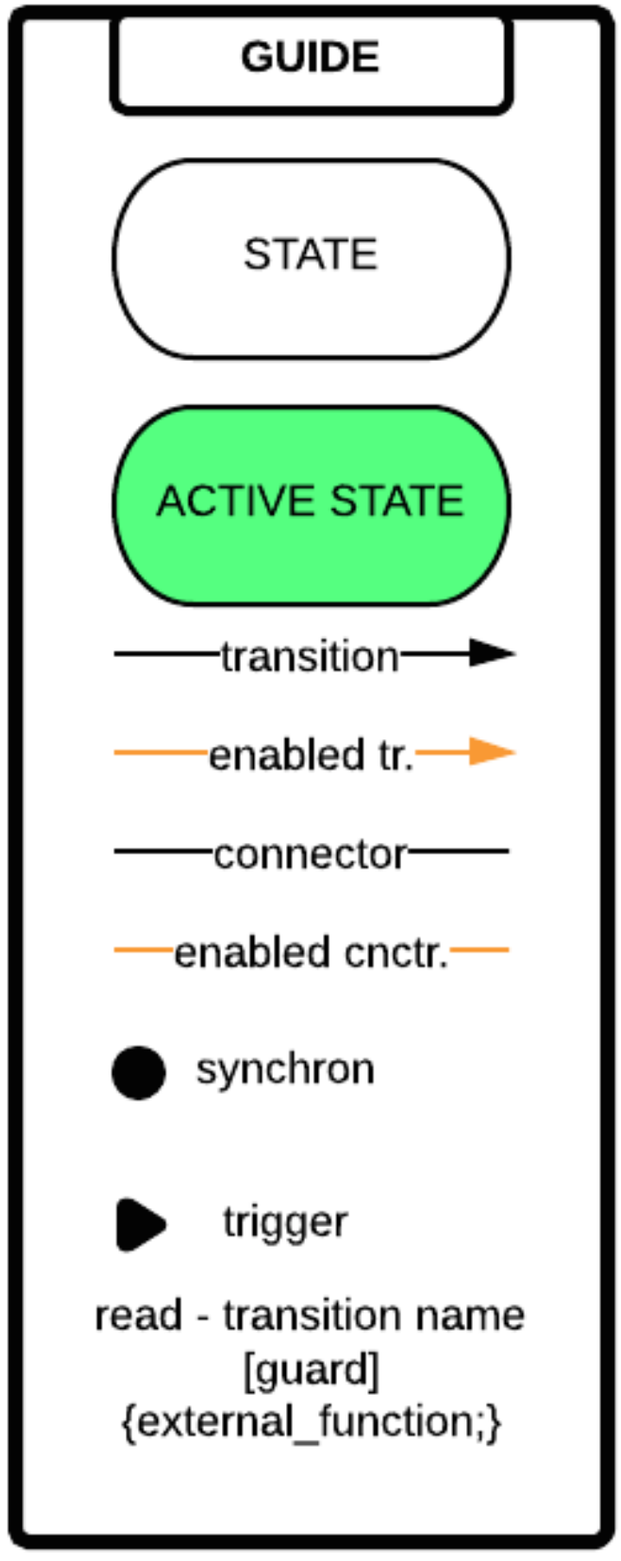
Dorische Säulenordnung. Jonische Säulenordnung. Korinthisch u. Römisch-Korinthisch.





slide courtesy of
Marco Pagnamenta





slide courtesy of
Marco Pagnamenta

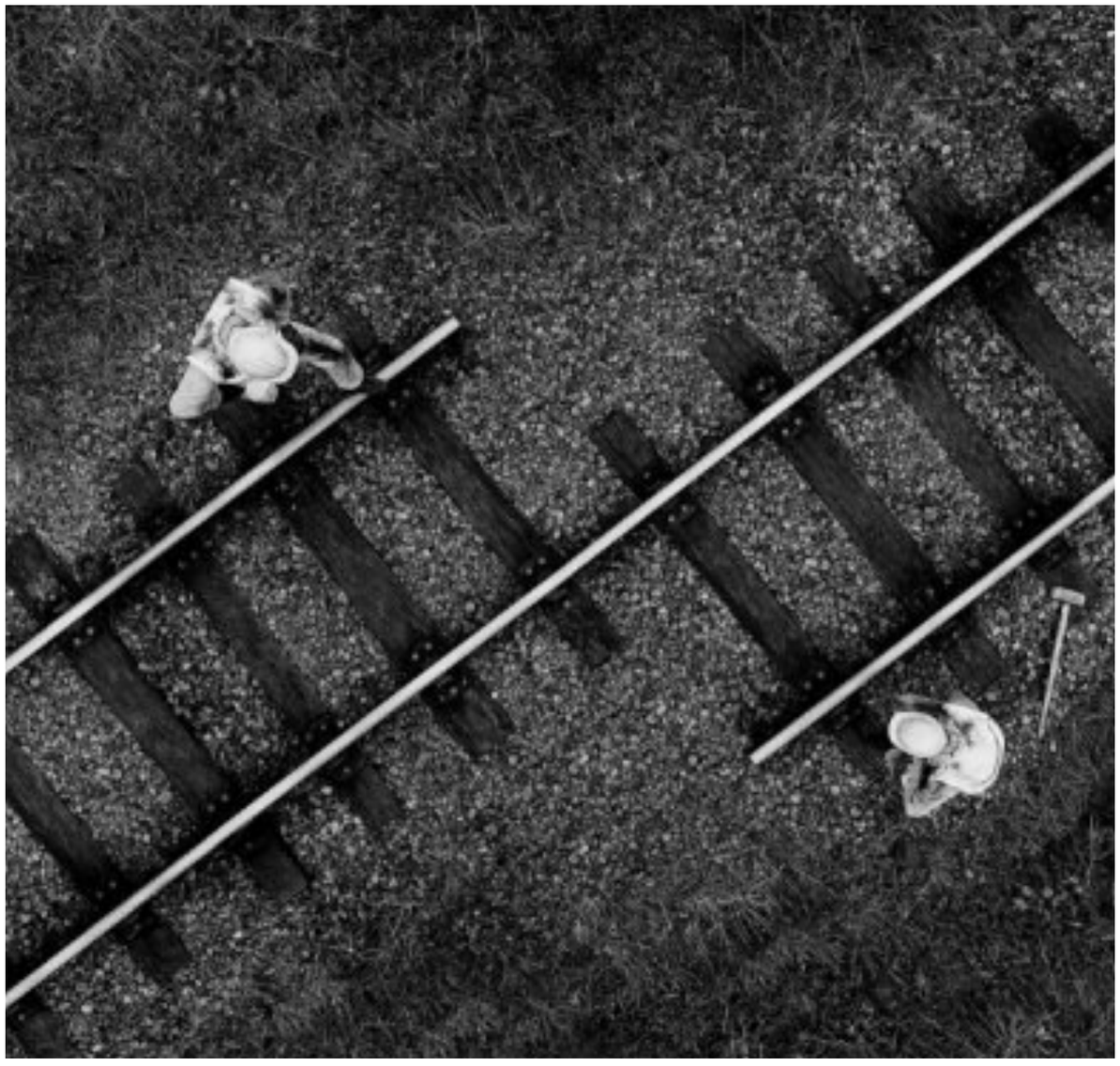
Theory of architectures

Design patterns for BIP

How to model?

How to combine?

How to specify?



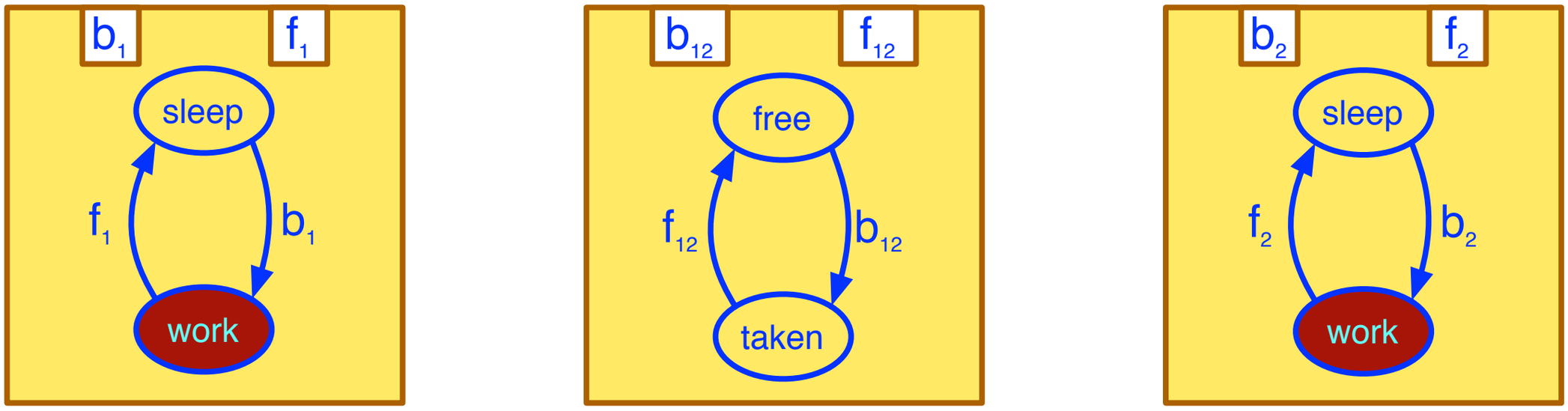
Architectures enforce characteristic properties. The crucial question is whether these are preserved by composition?

[Attie et al, SEFM '14]

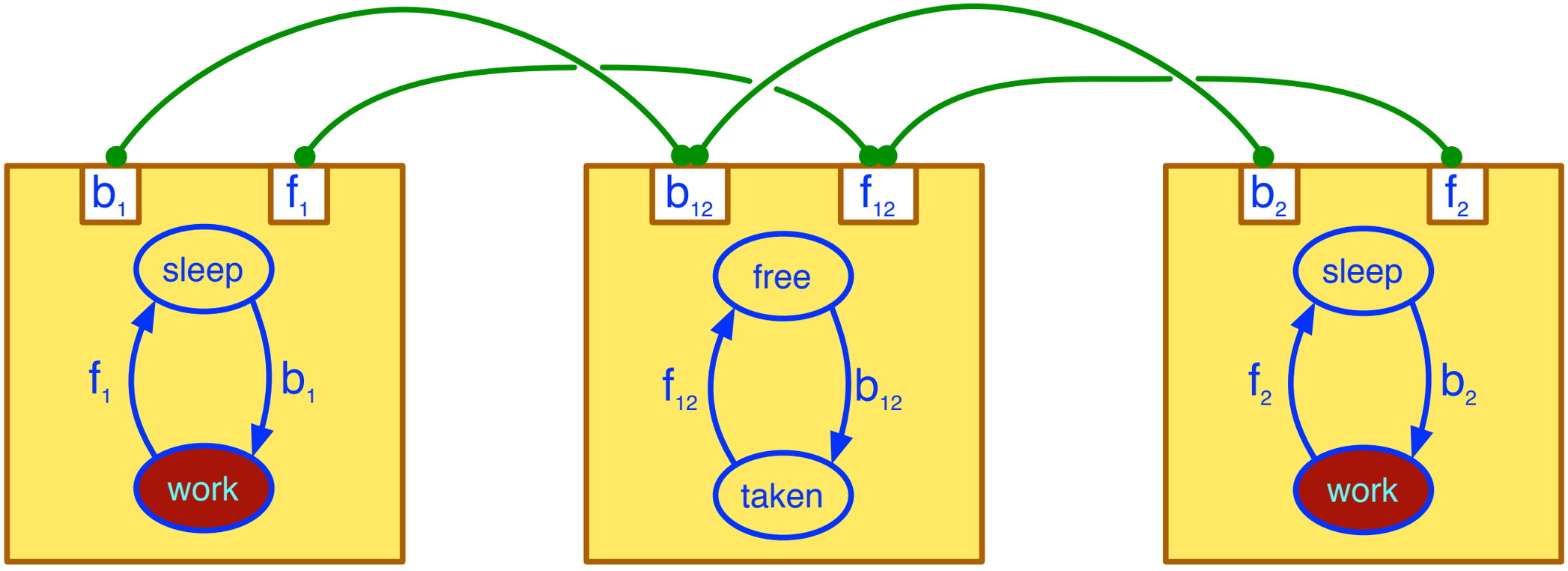
How to model?

Example: Lock

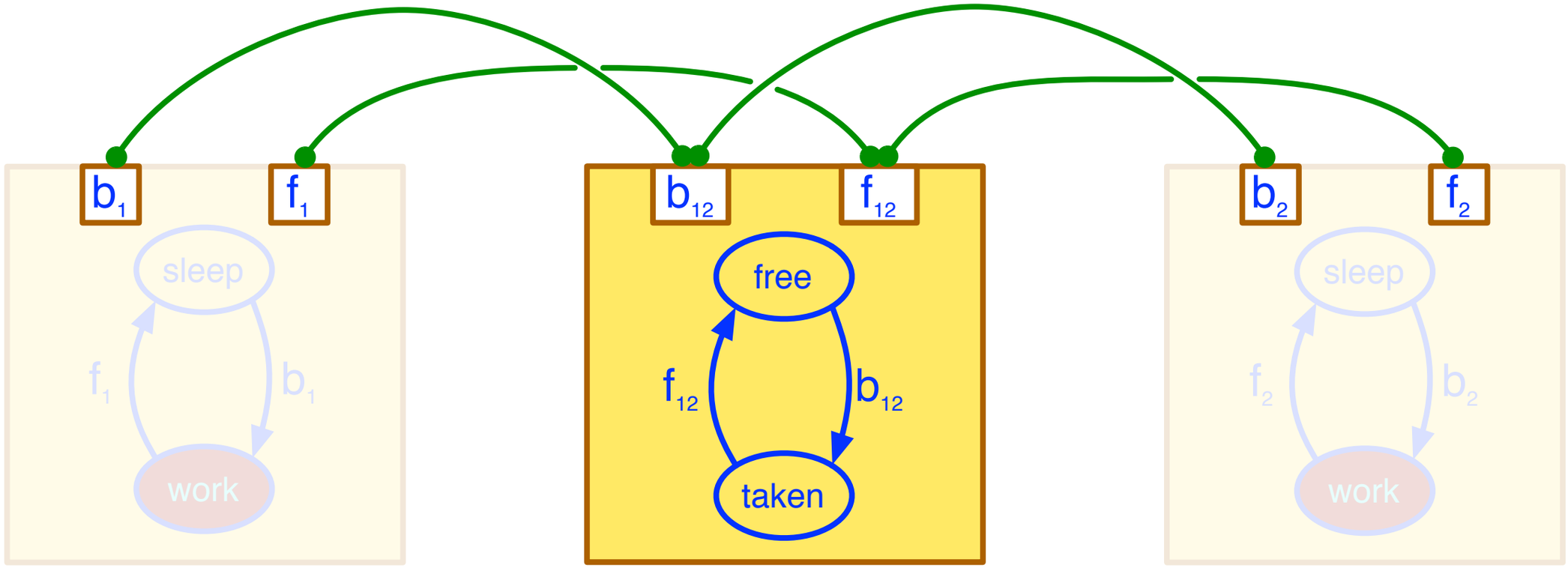
Example: Lock



Example: Lock



Example: Lock



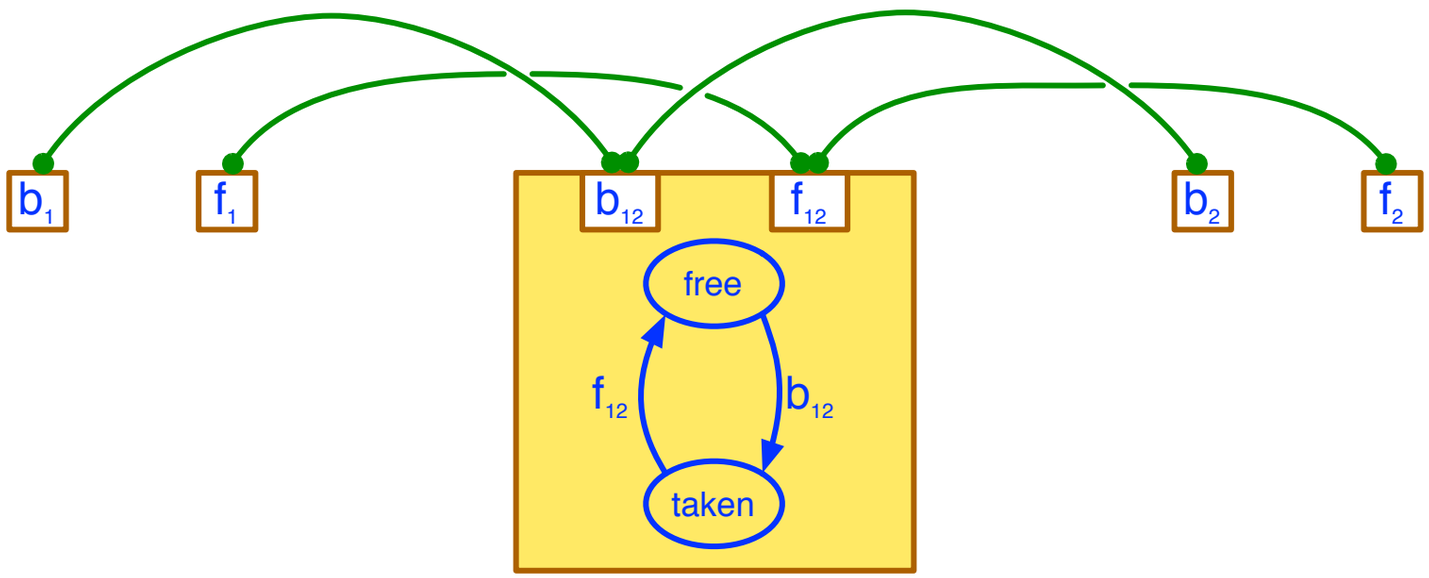
An architecture is...

$$A = (\mathcal{C}, P_A, \gamma)$$

Set of coordinating behaviours

Interface (ports)

Interaction model



…an operator…

$$A = (\mathcal{C}, P_A, \gamma)$$

…transforming

a set of components $\mathcal{B}$

into a composed BIP system $A(\mathcal{B}) \stackrel{def}{=} \Big(\gamma \ltimes P\Big)(\mathcal{B} \cup \mathcal{C})$

where $P \stackrel{def}{=} \bigcup_{B \in \mathcal{B} \cup \mathcal{C}} P_B, \quad \gamma \ltimes P \stackrel{def}{=} \Big\{a \subseteq 2^P \,\Big|\, a \cap P_A \in \gamma\Big\}$

Nice properties

Under suitable conditions

Architectures can be composed before applying

$$A_2(A_1(\mathcal{B})) = (A_1 \oplus A_2)(\mathcal{B})$$

Architecture application can be restricted

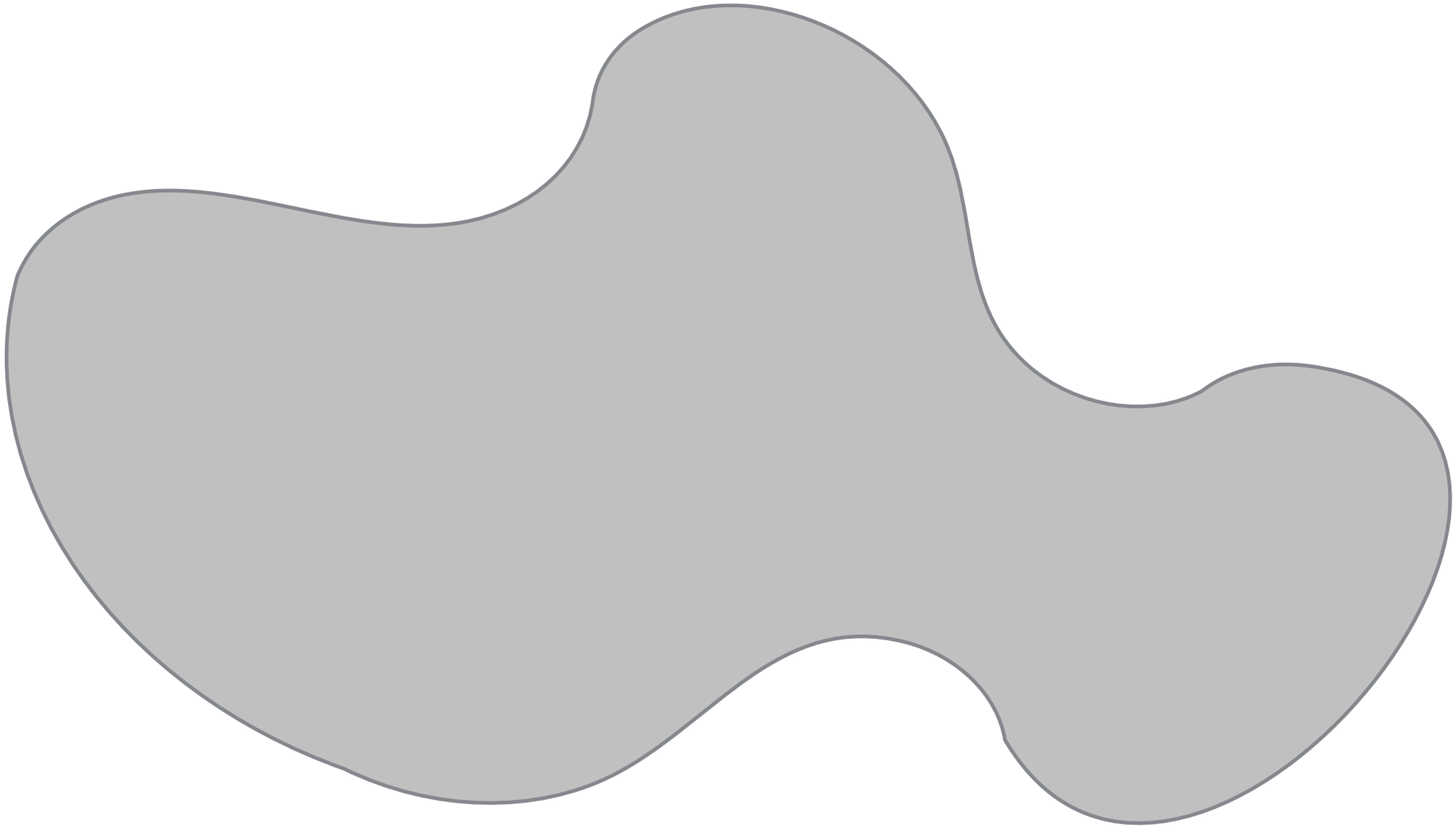
$$A_2(A_1(\mathcal{B}_1, \mathcal{B}_2)) = A_2(A_1(\mathcal{B}_1), \mathcal{B}_2)$$

Architecture can be applied partially

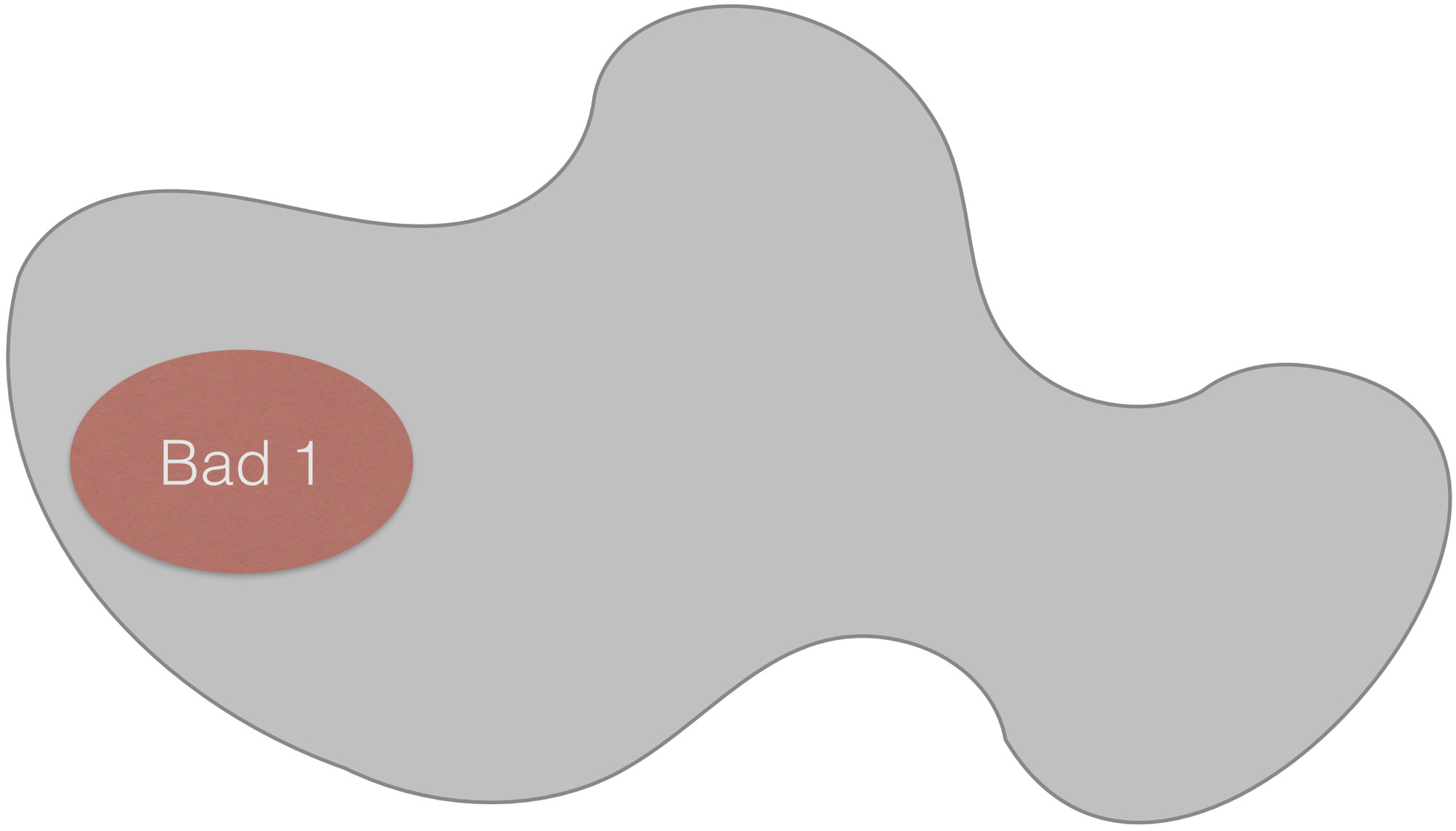
$$A(\mathcal{B}_1, \mathcal{B}_2) = A[\mathcal{B}_1](\mathcal{B}_2)$$

How to combine?

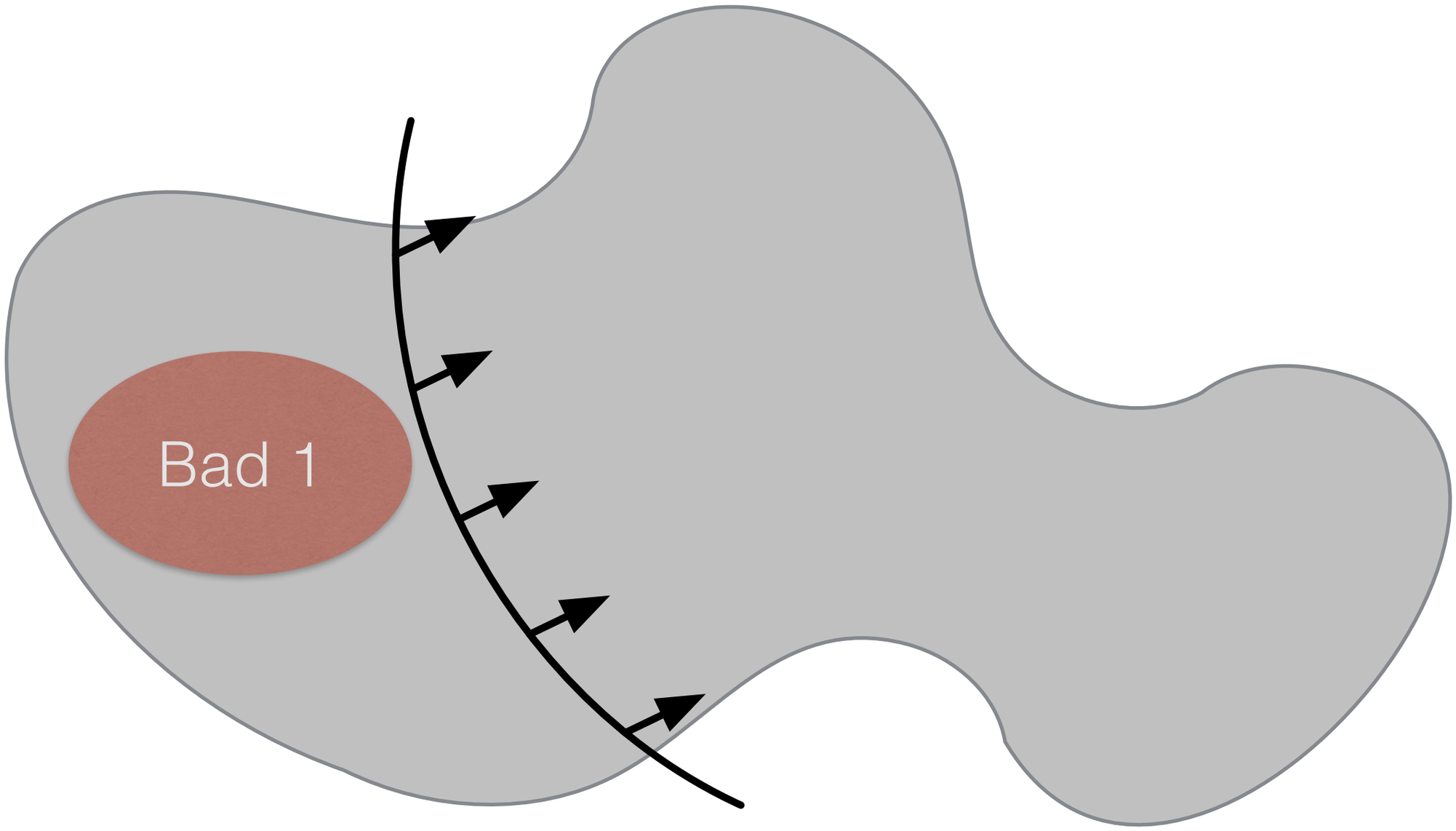
Constraints intuition



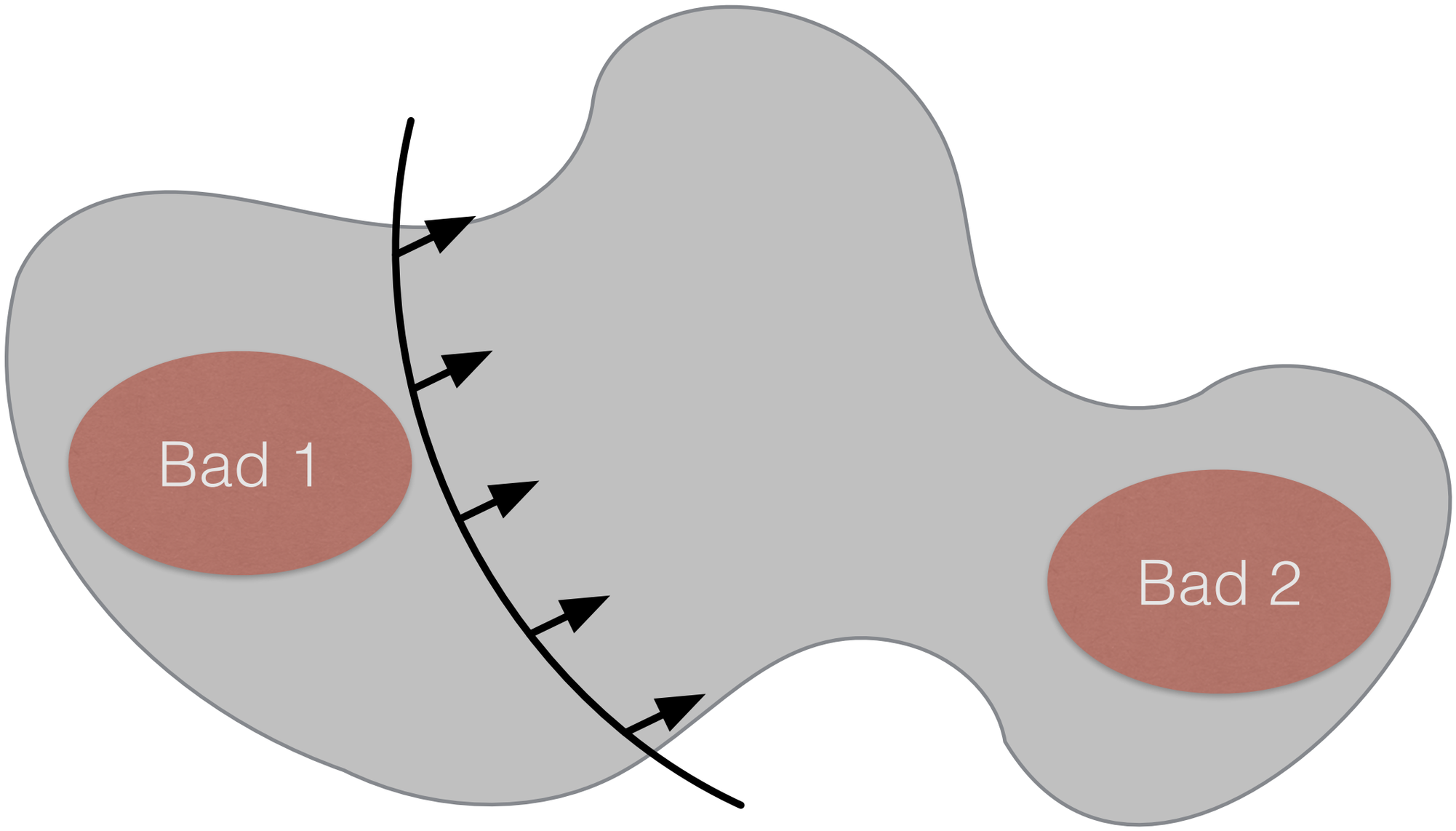
Constraints intuition



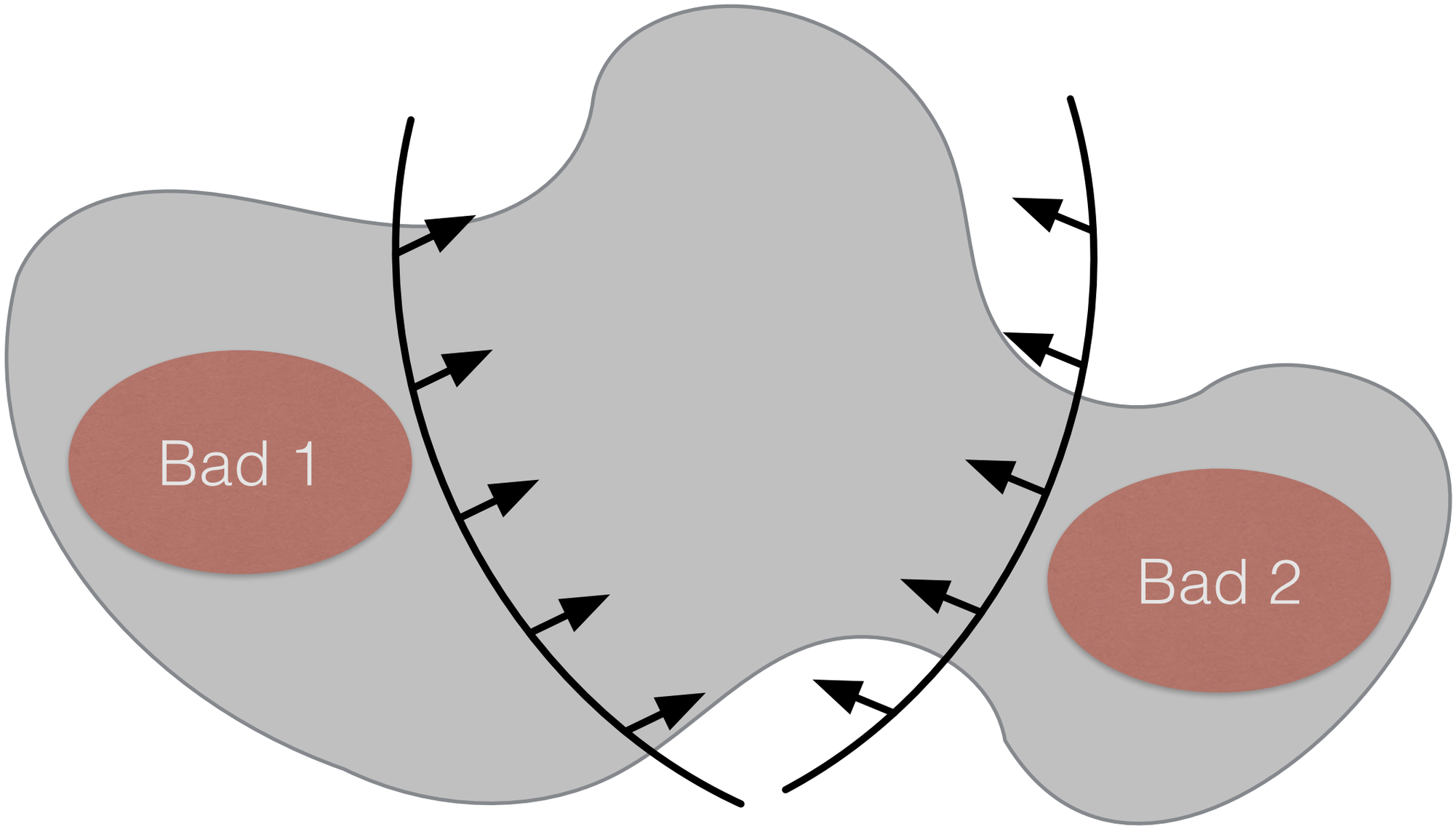
Constraints intuition



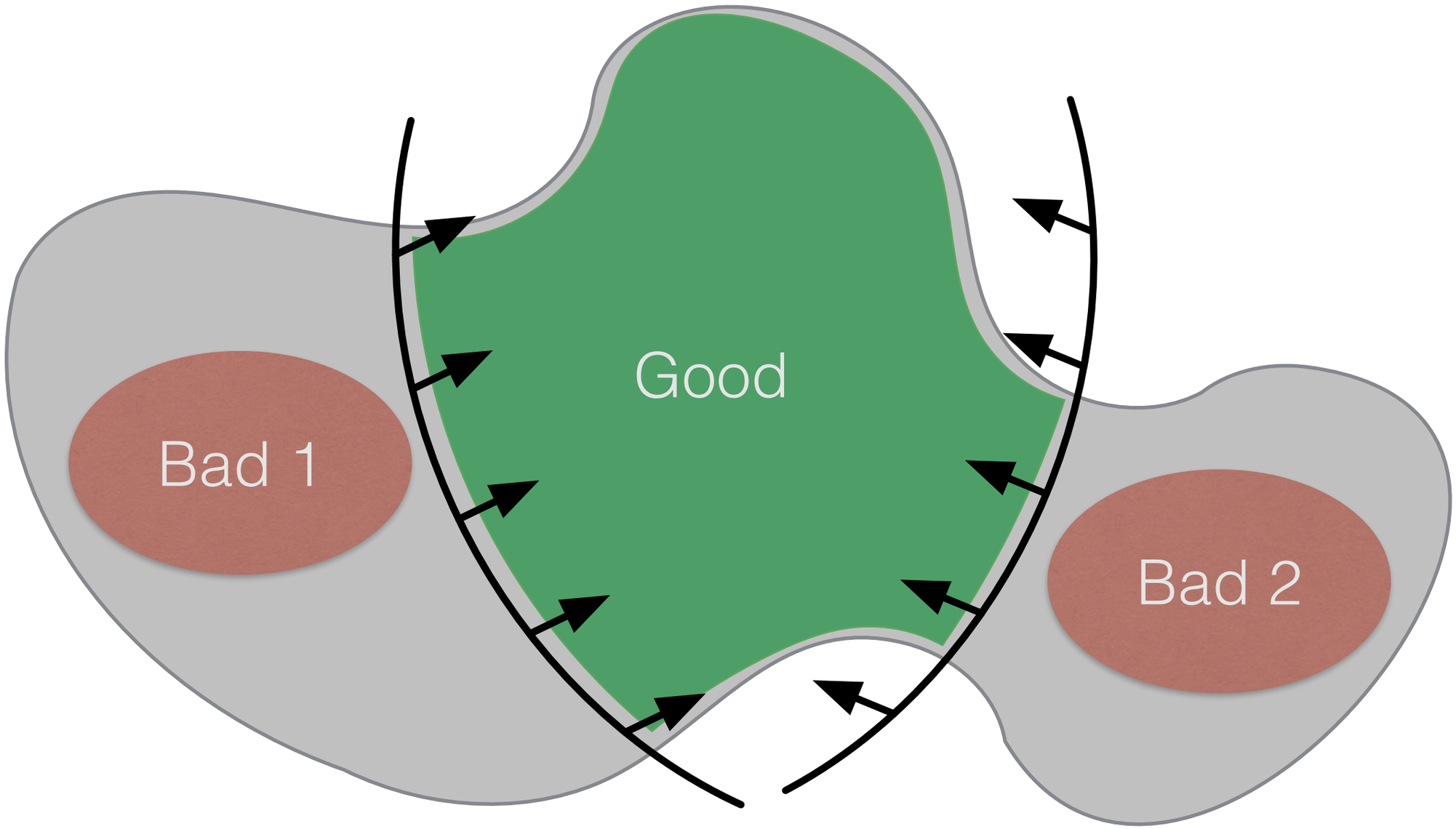
Constraints intuition



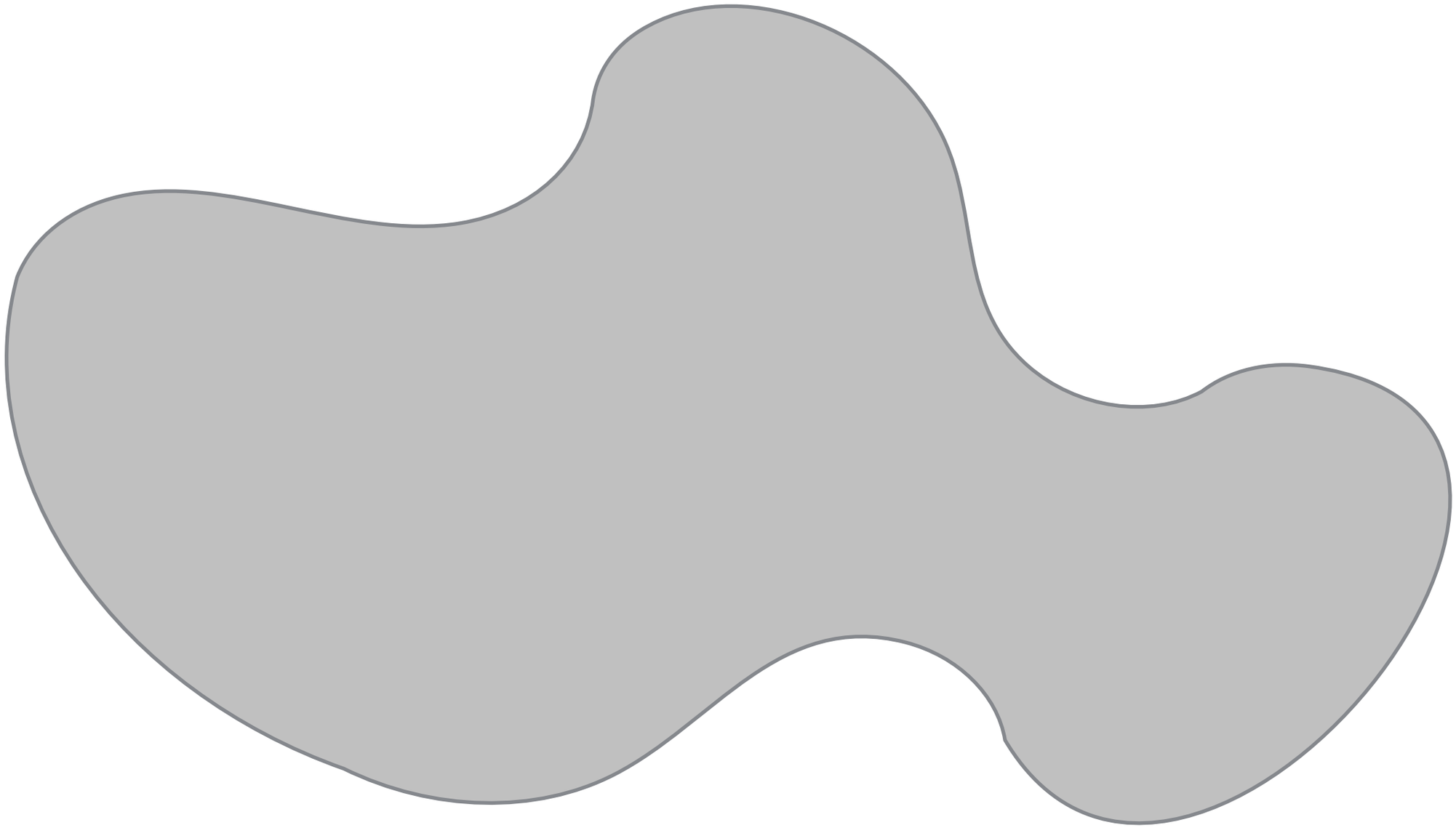
Constraints intuition



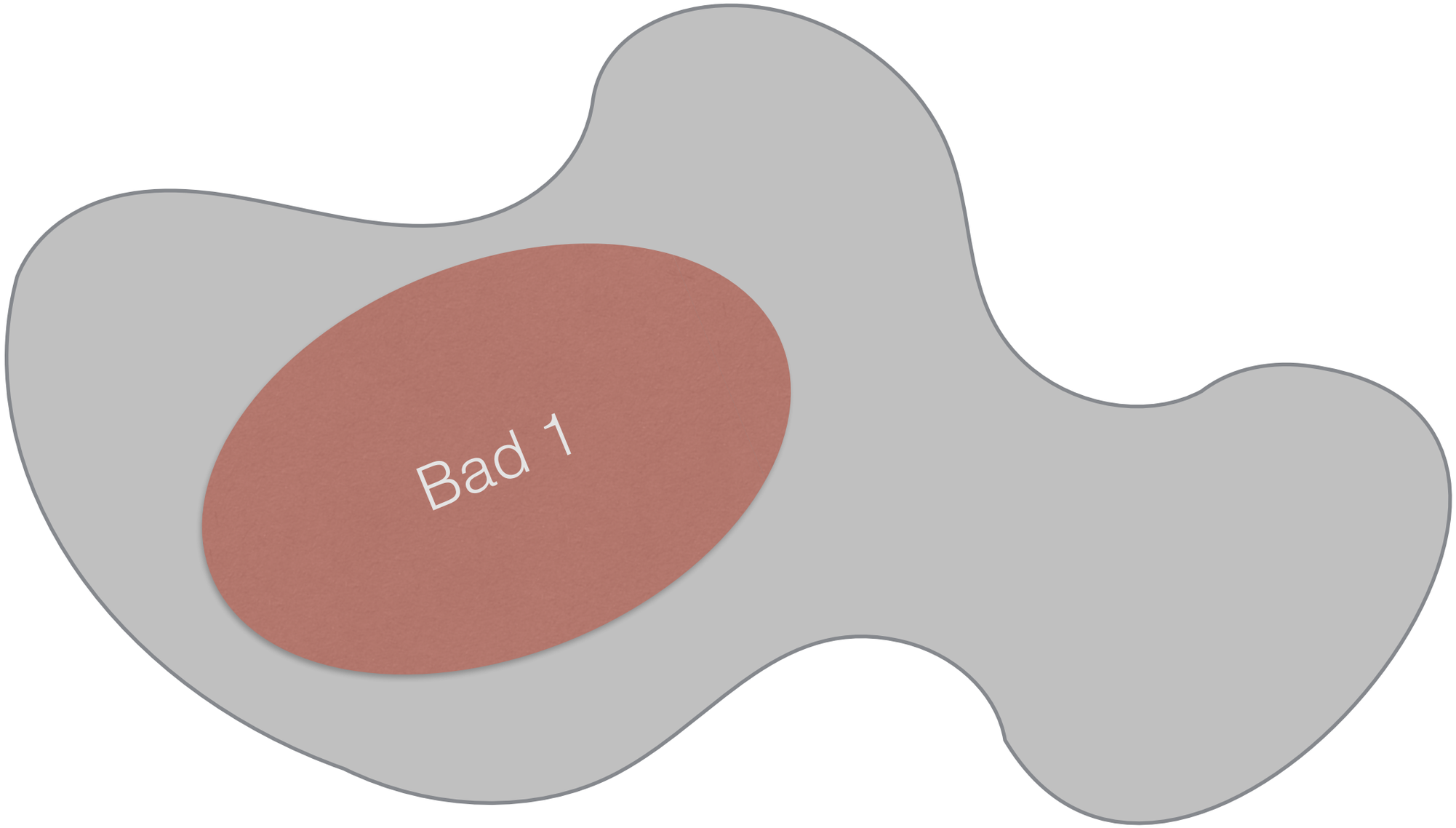
Constraints intuition



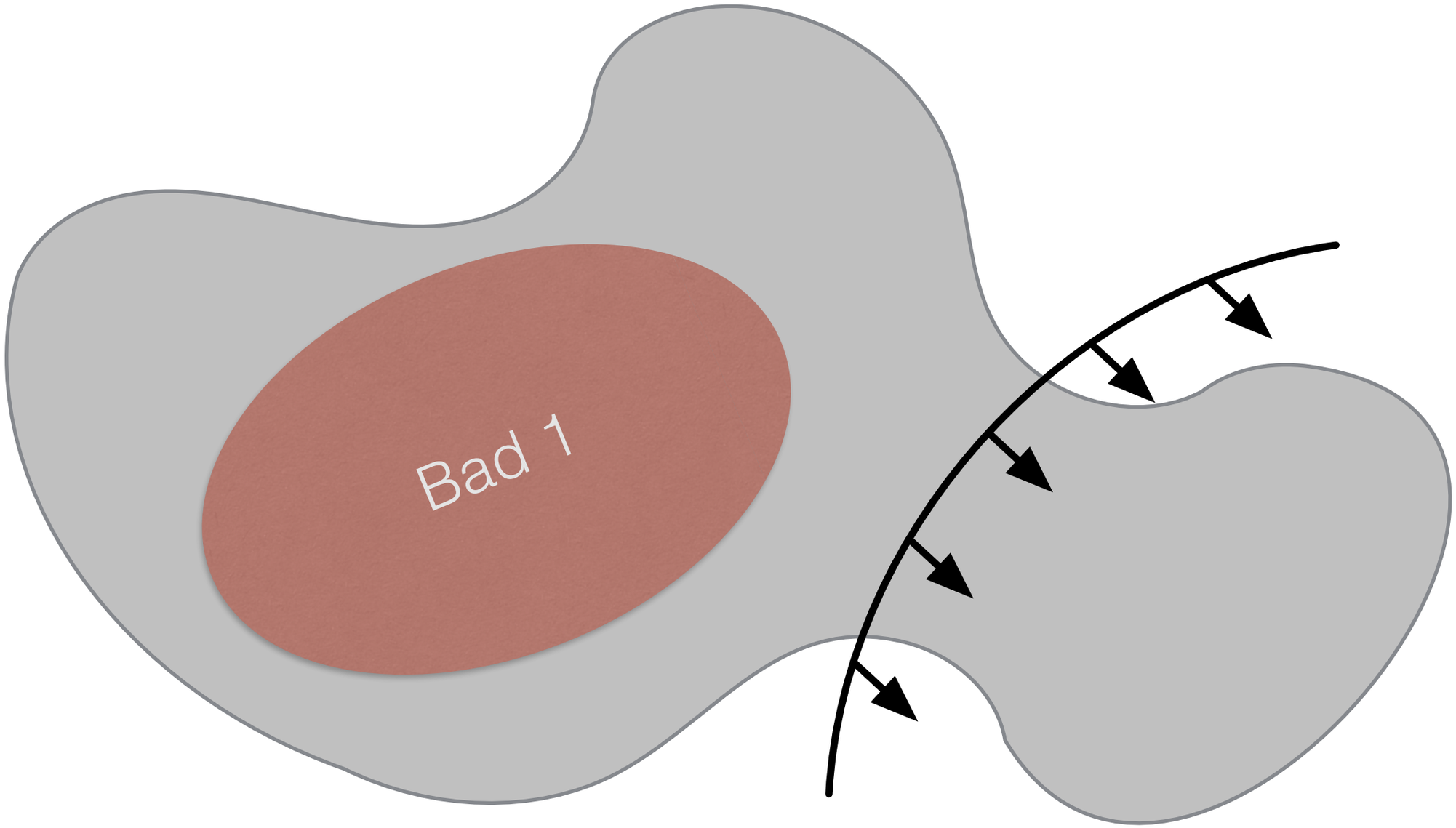
Limits of white magic



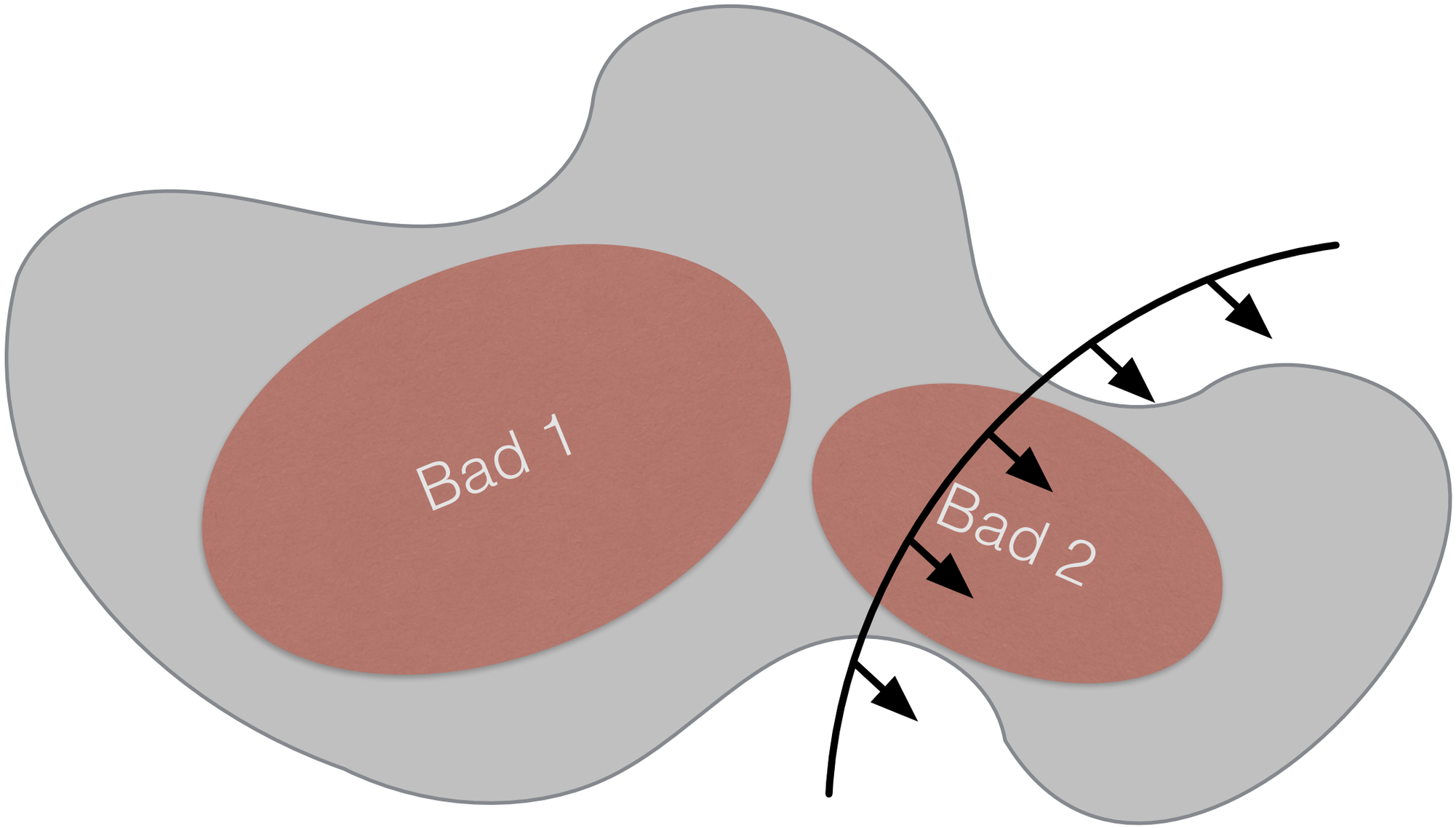
Limits of white magic



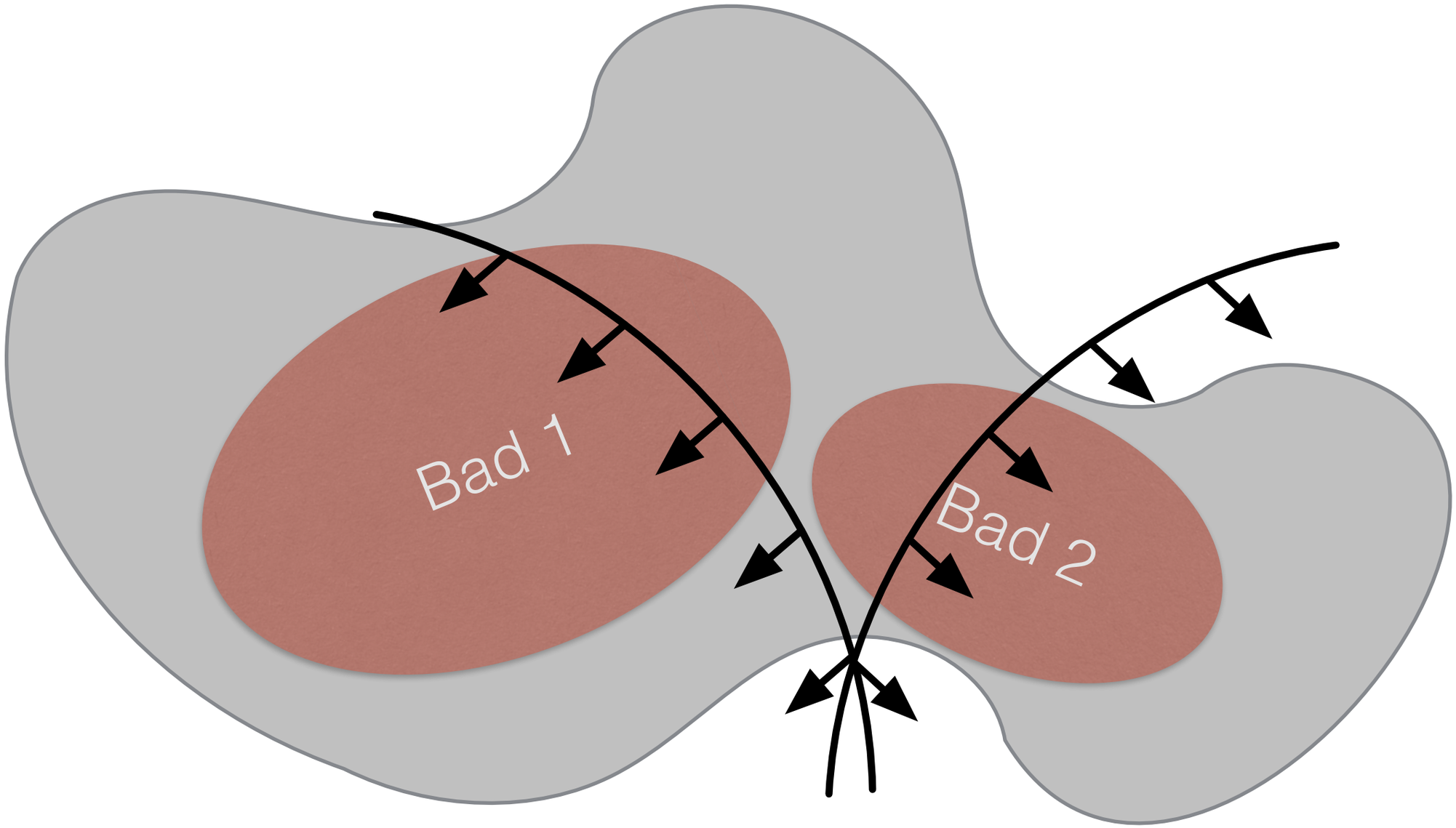
Limits of white magic



Limits of white magic



Limits of white magic



Formally

$$A_1 \oplus A_2 \stackrel{def}{=} (\mathcal{C}_1 \cup \mathcal{C}_2, P_1 \cup P_2, \gamma)$$

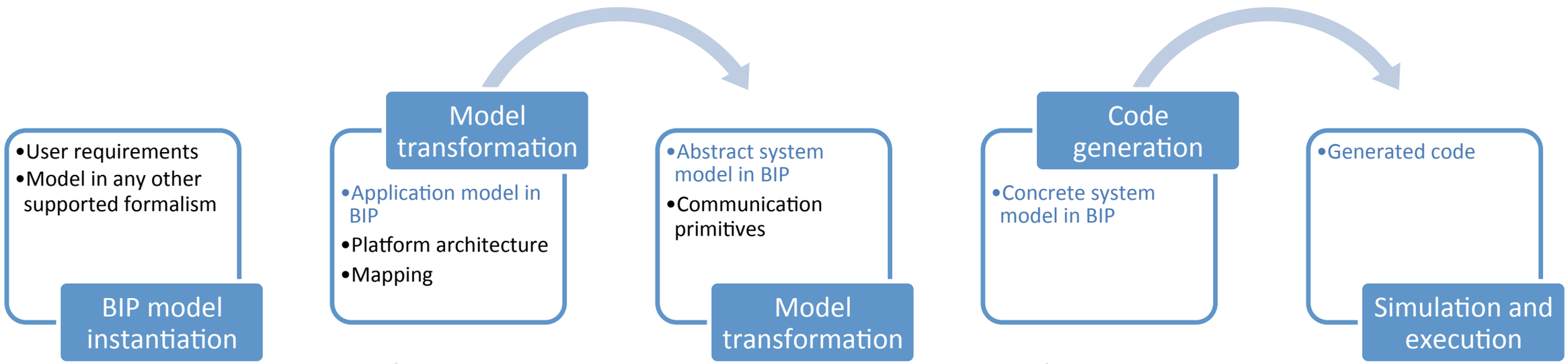
$$\begin{aligned}\gamma &\stackrel{def}{=} \left\{ a \subseteq 2^P \,\middle|\, a \cap P_1 \in \gamma_1 \;\wedge\; a \cap P_2 \in \gamma_2 \right\} \\ &= (\gamma_1 \ltimes P) \cap (\gamma_2 \ltimes P)\end{aligned}$$

Main results: Safety

$$\left.\begin{aligned} A_1(\mathcal{B}) &\models \Phi_1 \\ A_2(\mathcal{B}) &\models \Phi_2 \end{aligned}\right\} \quad \Longrightarrow \quad \big(A_1 \oplus A_2\big)(\mathcal{B}) \models \Phi_1 \wedge \Phi_2$$

Safety = *"Bad states never occur"*

Rigorous System Design flow



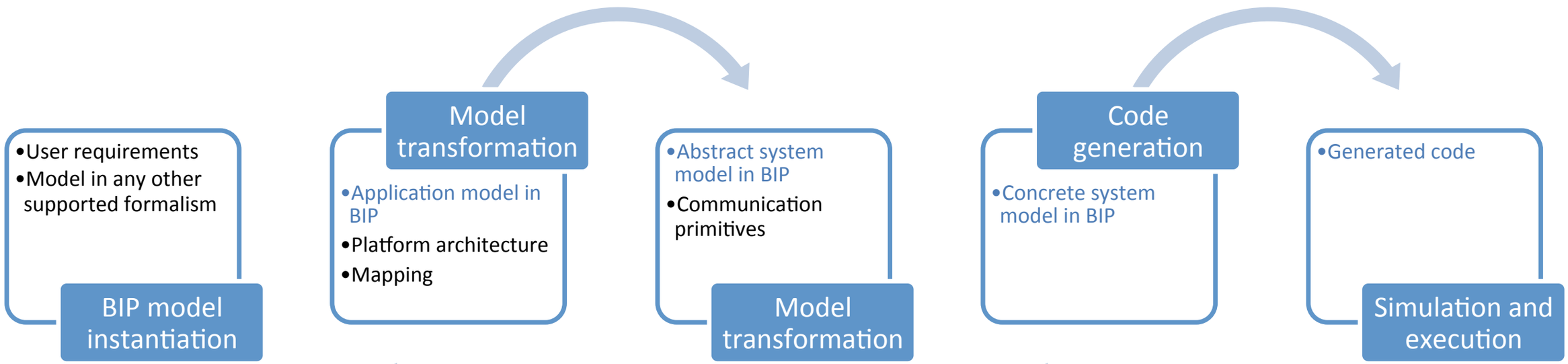
A series of semantics-preserving transformations

Correctness decomposed into
- correctness of transformations
- correctness of high-level models

Final implementation is **correct by construction**

- ☑ Unifying modelling framework
- ☑ Operational semantics
- ☐ Method(s) to design correct models

Rigorous System Design flow

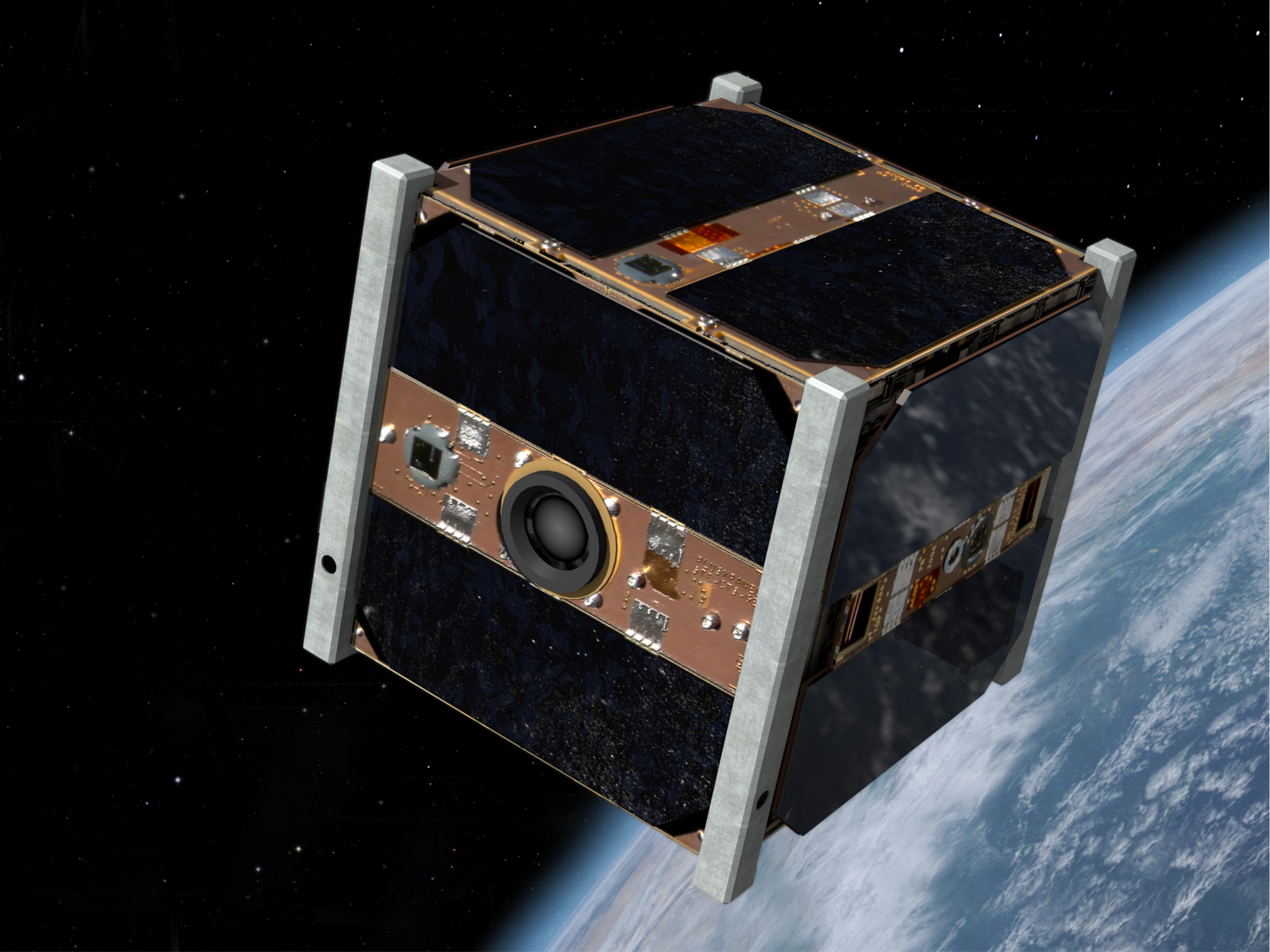


A series of semantics-preserving transformations

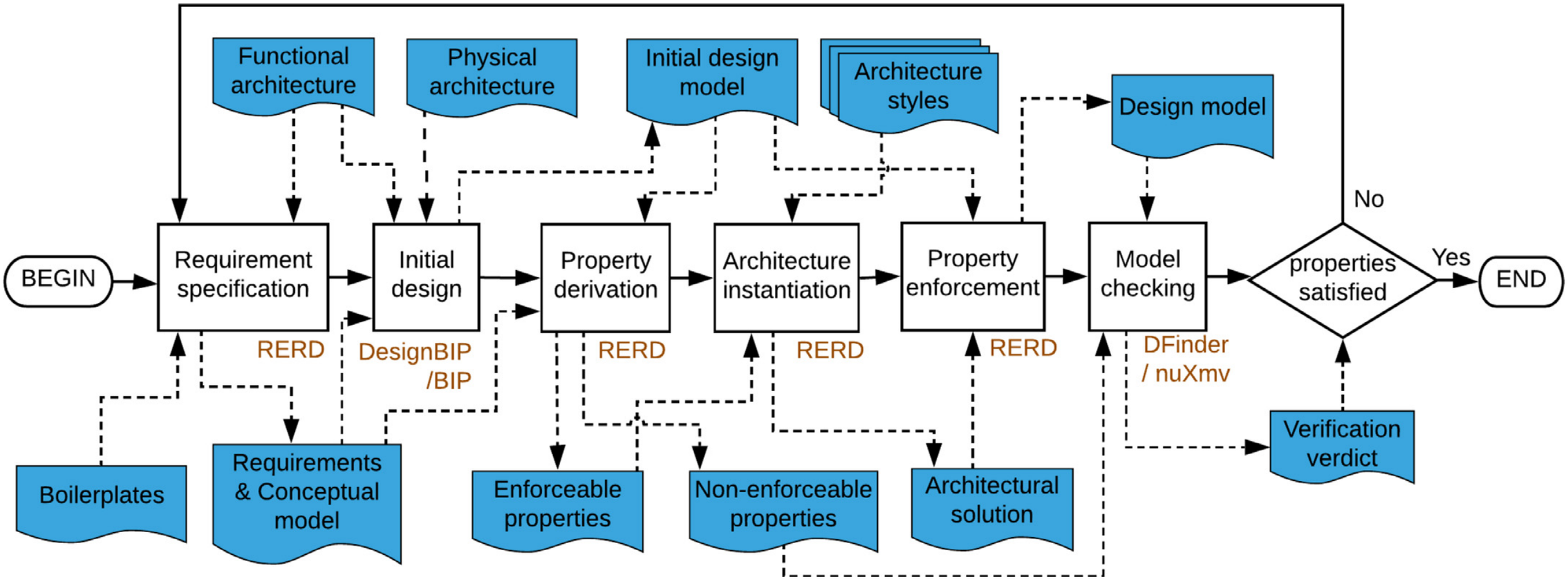
Correctness decomposed into
- correctness of transformations
- correctness of high-level models

Final implementation is **correct by construction**

- ☑ Unifying modelling framework
- ☑ Operational semantics
- ☑ Method(s) to design correct models



Requirements and design process



/Requirements Engineering for Rigorous Design/

[Stachtiari et al, JSS '18]

ARISTOTLE UNIVERSITY OF THESSALONIKI

CubETH case study

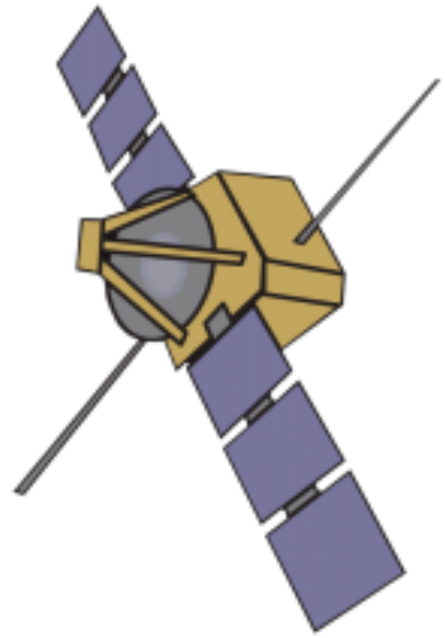


Table 1: Representative requirements for CDMS status and HK_PL

ID	Description
CDMS-007	The CDMS shall periodically reset both the internal and external watchdogs and contact the EPS subsystem with a “heartbeat”.
HK-001	The CDMS shall have a Housekeeping activity dedicated to each subsystem.
HK-003	When line-of-sight communication is possible, housekeeping information shall be transmitted through the COM subsystem.
HK-004	When line-of-sight communication is not possible, housekeeping information shall be written to the non-volatile flash memory.
HK-005	A Housekeeping subsystem shall have the following states: NOMINAL, ANOMALY and CRITICAL_FAILURE.

[Mavridou et al, FACS '16]

RERD tool

Requirement Editing | Property Formalization | Dictionary | Models

Abstraction Level : RB Category : ContextSavingRequirement

ID	Prefix
P2	While State : [...]
P3	If Event : [...] and State : [...
P1	If Event : [...]

ID	Main
M1	Function : [...] shall Action : [...]
M2	Function : [...] shall Action : [...] and Action : [...] : [...]
M3	Function : [...] shall State : [...]

ID	Suffix
S1	before Event : [...]
S2	sequentially
S3	atomically

Back to Categories

Console

If Event: a failure of the PL subsystem persists for [TBD] sec

Function: HK PL shall Action: contact the EPS for a restart of PL

HK-05

Generate Req ID

RB

ContextSavingRequirement

Invalid

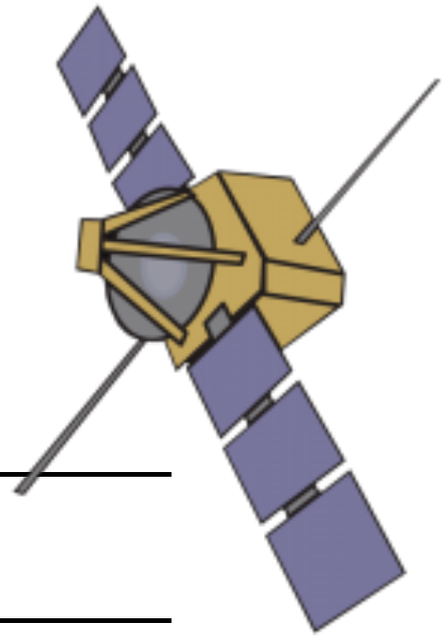
Refines | Refined By | Concretizes | Concretized By

Save | Validate | Clear | New

Search... | Ontology Validation

Req. ID	Status	Text	Category	AbsLevel	Edit	Delete
HK-02	🟡	If [TBD] seconds pass and HK for PL is enabled HK PL shall handle HK data from PL	ContextSavingRequirement	RB	Edit	Delete
HK-03	🟢	If HK has been read from PL and PS for PL is not enabled HK PL shall transmit HK data tl	ContextSavingRequirement	RB	Edit	Delete
HK-04	🔴	While PS for PL is enabled HK PL shall write HK data to the flash memory	ContextSavingRequirement	RB	Edit	Delete
HK-05	🔴	If a failure of the PL subsystem persists for [TBD] sec HK PL shall contact the EPS for a re	ContextSavingRequirement	RB	Edit	Delete

CubETH case study

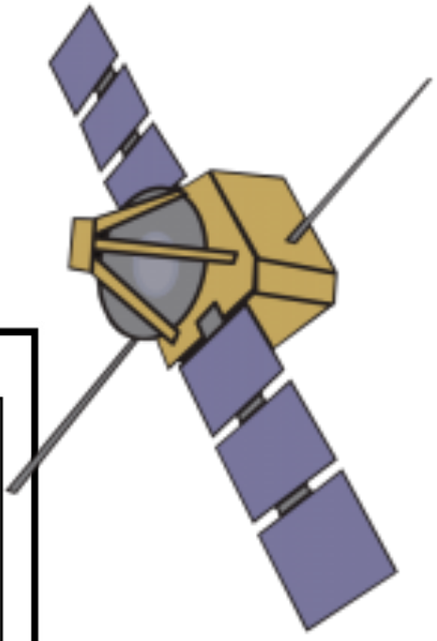


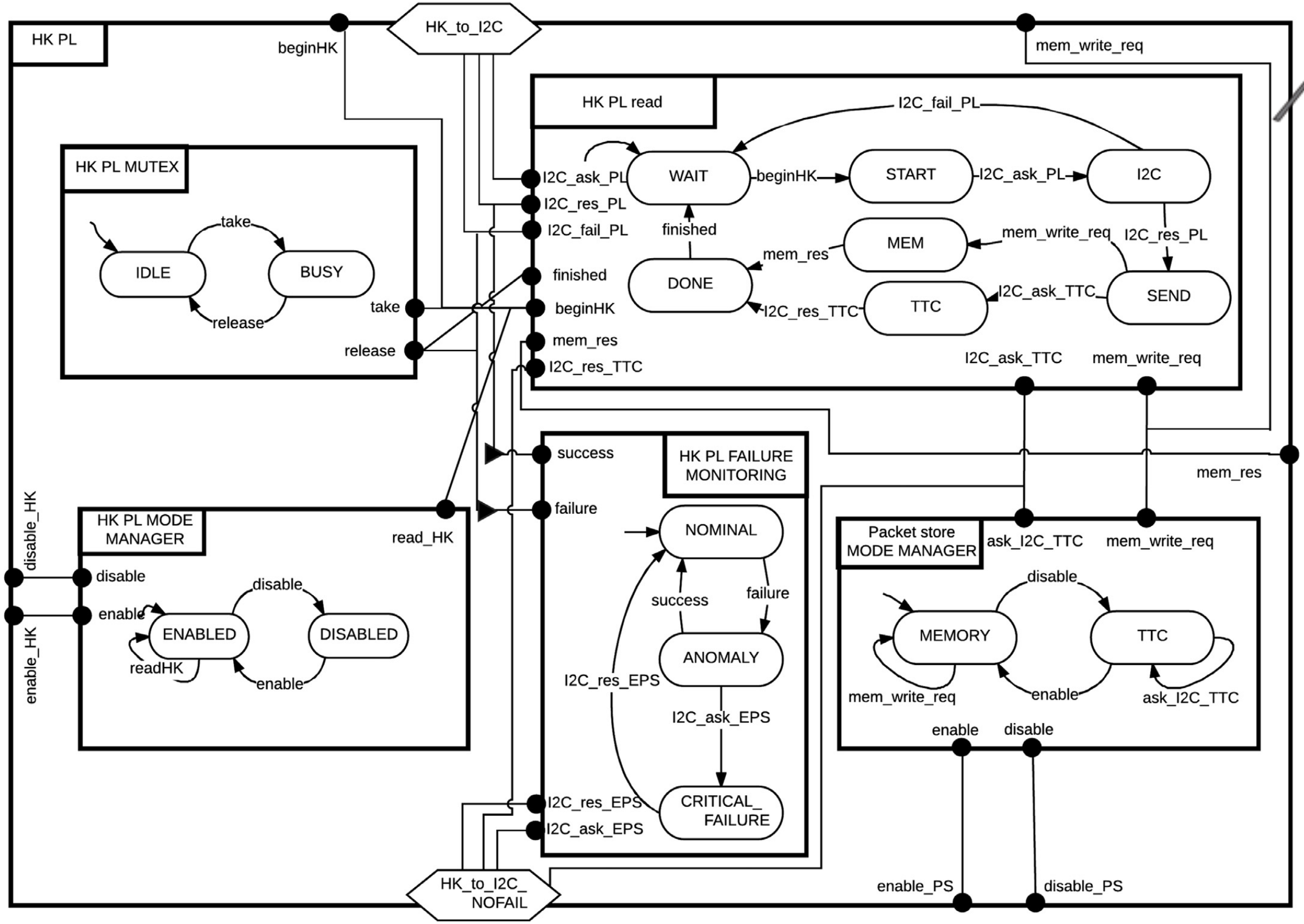
Requirements for the *HK PL* function.

ID	Requirement
HK-02	P2: if ⟨event-e003: [TBD] sec pass ⟩ and ⟨state-s003: HK collection is enabled for PL ⟩ M1: ⟨function: HK PL ⟩ shall ⟨action-a004: handle HK data from the PL ⟩
HK-03	P3: if ⟨state-s002: PS[a] for PL is not enabled ⟩ M1: ⟨function: HK PL ⟩ shall ⟨action-a002: transmit HK data through the TC/TM service ⟩
HK-04	P3: while ⟨state-s001: PS for PL is enabled ⟩ M1: ⟨function: HK PL ⟩ shall ⟨action-a001: write HK data to the flash memory ⟩
HK-05	P1: if ⟨event-e004: a PL failure persists for [TBD] sec ⟩ M1: ⟨function: HK PL ⟩ shall ⟨action-a003: contact the EPS for a restart of the PL ⟩

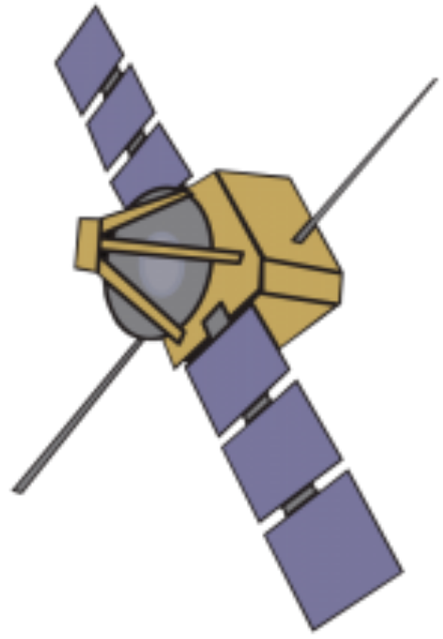
[a] PS stands for a packet store structure.

CubETH case study





CubETH case study



Durations and input sizes of the process steps.

Step	Duration	Input size
Requirement specification	8 h	38 requirements
Initial design	5 h	12 components
Architecture instantiation	3 h	47 enforced properties
Verification of deadlock freedom	12 s	46 components

Statistics of requirement formulation and property enforcement.

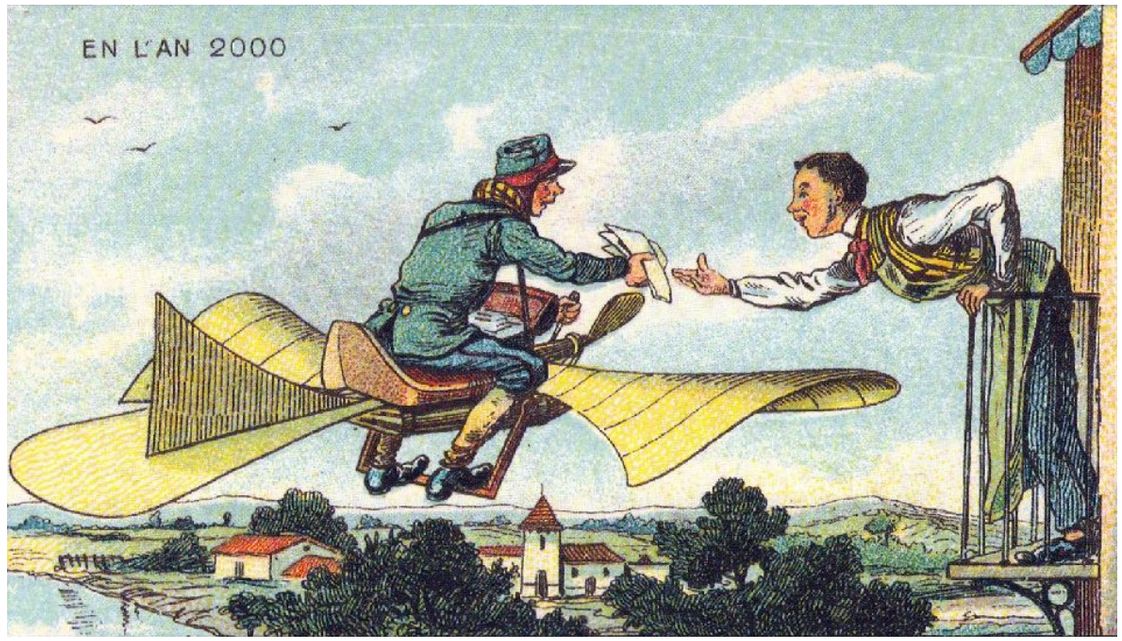
Model	Flow	Mode	Event	Mutex	Failure	Requir.	Deriv. Prop.	Assum. Prop.	Enforced
Payload	0	2	0	4	0	12	16	0	16
HK PL	0	2	1	1	1	4	6	0	6
HK EPS	0	2	1	1	1	4	6	0	6
HK COM	0	2	1	1	1	4	6	0	6
HK CDMS	0	2	1	1	0	3	4	0	4
Flash memory	0	1	0	1	0	8	13	4	3
CDMS status	1	0	0	0	0	1	3	0	3
Error logging	0	0	1	1	0	2	3	0	3
Total	1	11	5	10	3	38	57	4	47

Conclusion

Powerful theoretical tools to build systems that are correct by construction

Going from theory to practice requires a lot of effort and cross-domain collaborations

Future work



BIP

Dynamicity, distribution, self-adaptation

Architectures

Case studies, case studies, case studies and taxonomies (libraries)

Data, Real-time, Synthesis, Dynamicity, Probabilities etc.

DSLs for usability

Verification and proof of architectures and architecture styles

Tool support

RERD

Revamping the ontologies with better understanding of domain specificities

How to make generated BIP models understandable by developers?

Main results: Liveness

$$\underbrace{\left.\begin{array}{r}\mathcal{A}\\ \text{pairwise non-interfering}\end{array}\right\}\begin{array}{r}\text{live}\\ \end{array}}_{\text{w.r.t. } \mathcal{B}} \implies \bigoplus \mathcal{A} \quad \text{live}$$

Liveness = *"Good states occur infinitely often"*