Comprehensive Systems for Software Interoperability Problems

Patrick Stünkel [a b] <past@hvl.no>

GReTA ∩ MDENet France Workshop 2023

2023-12-15

a Western Norway University of Applied Sciences - HVL
b Helse Bergen - Haukeland University Hospital

Agenda

About

Personal

- Associate Professor @ HVL
  - since: August 2023
  - topics:
    - (model-based) software engineering (& formal foundations thereof),
    - digital twins (process modelling),
    - data engineering,
    - health informatics
  - web: https://www.hvl.no/person/?user=Patrick.Stuenkel
- previously: PostDoc @ HelseBergen
  - project employment @ Patologi i Vest (PiV)
  - topics:
    - workflow optimization (automated planning)
    - process modelling
    - process mining
    - digital pathology
  - web: https://www.helse-bergen.no/piv

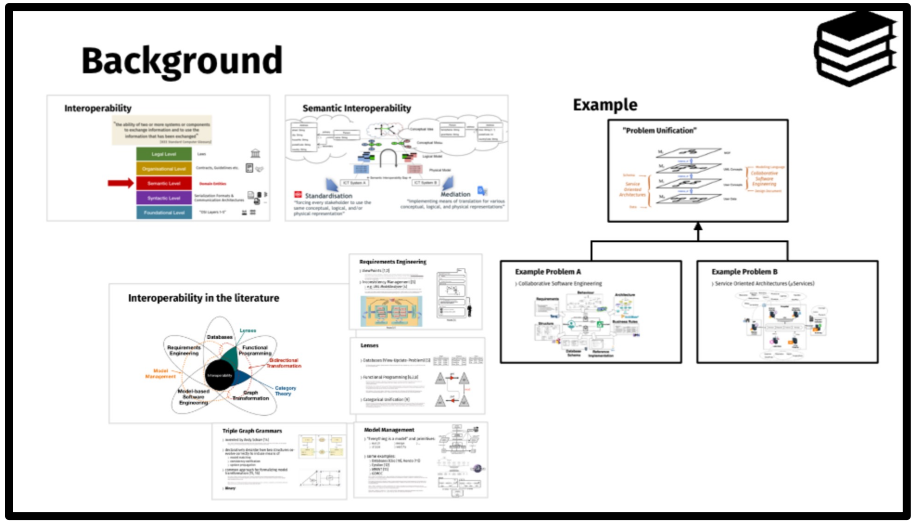
Education

- PhD
  - defence: 9th February 2022
  - title: A framework for multi-model consistency management
  - topics:
    - software interoperability & integration,
    - heterogeneous modelling (category theory),
    - inter-model consistency,

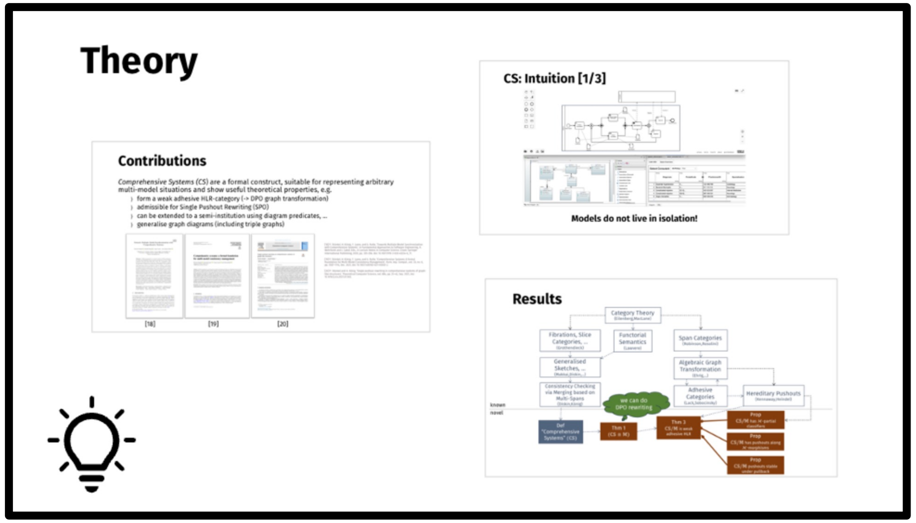
    todays talk
  - web: https://hvlopen.brage.unit.no/hvlopen-xmlui/handle/11250/2837743
- before that
  - Bachelor & Masters from FHDW Hannover
  - supervised by *Michael Löwe*
    - algebraic specification (partial algebras)
    - graph transformation (Single Pushout Approach)
    - category theory

https://past.corrlang.io/

https://github.com/webminz



Background



Theory

Contributions

Comprehensive Systems (CS) are a formal construct, suitable for representing arbitrary multi-model situations and show useful theoretical properties, e.g.

- form a weak adhesive HLR-category (-> DPO graph transformation)
- admissible for Single Pushout Rewriting (SPO)
- can be extended to a semi-institution using diagram predicates, …
- generalise graph diagrams (including triple graphs)

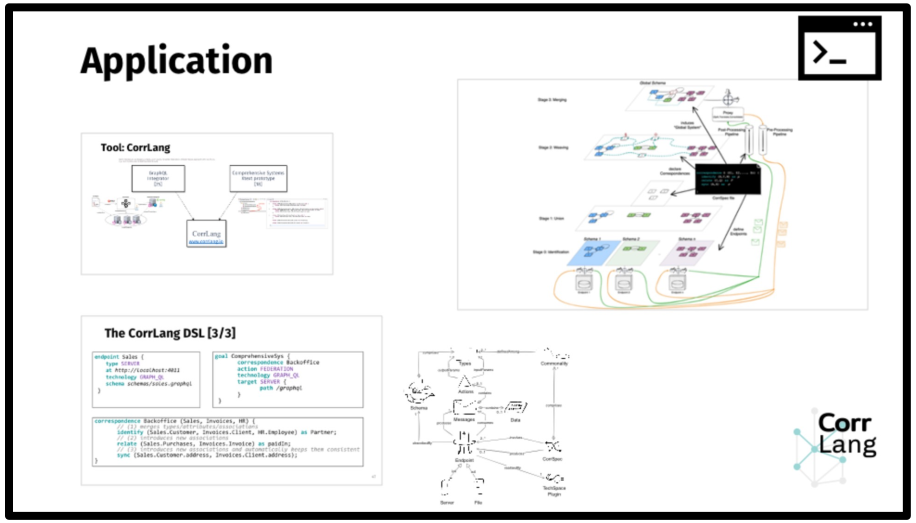
[18] [19] [20]

CS: Intuition [1/3]

Models do not live in isolation!

Results

Category Theory
Fibrations, Slice Categories, …
Functorial Semantics
Span Categories
Generalised Sketches, …
Algebraic Graph Transformation
Consistency Checking via Merging based on Multi-Spans
Adhesive Categories
Hereditary Pushouts
we can do DPO rewriting
known
novel
Def "Comprehensive Systems" (CS)
Thm 1 (CS ≅ M)
Thm 2 CS/M is weak adhesive HLR
Prop
Prop
Prop



Application

Tool: CorrLang

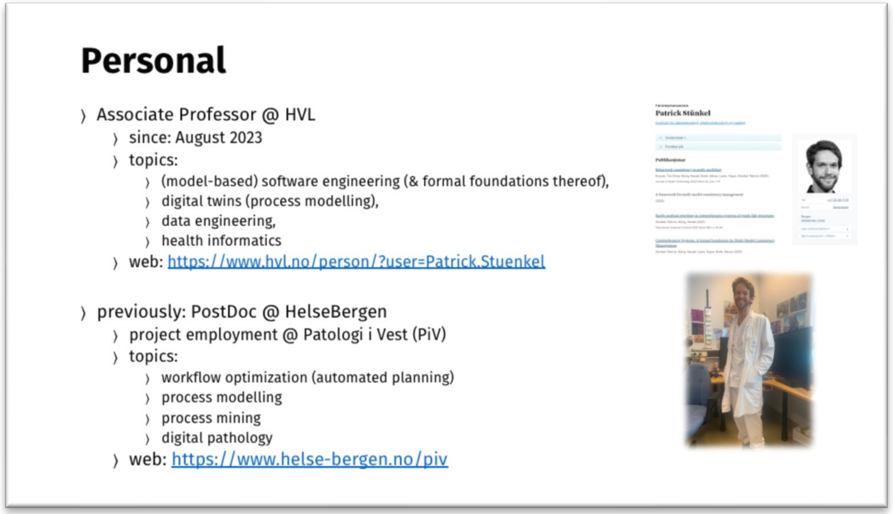
The CorrLang DSL [3/3]

```
endpoint Sales {
   type SERVER
   at http://localhost:4011
   technology GRAPH_QL
   schema schemas/sales.graphql
 }
```

```
goal ComprehensiveSys {
       correspondence Backoffice
       action FEDERATION
       technology GRAPH_QL
       target SERVER {
             path /graphql
       }
}
```

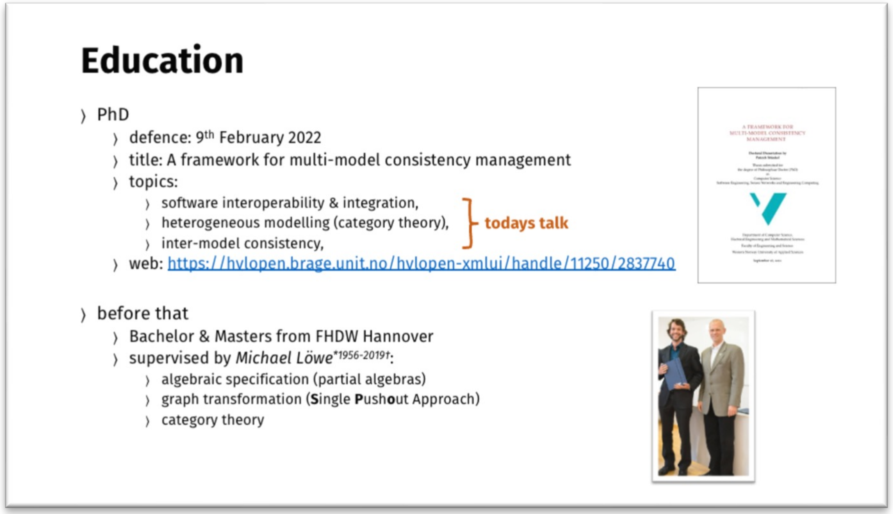
```
correspondence Backoffice (Sales, Invoices, HR) {
      // (1) merges types/attributes/associations
      identify (Sales.Customer, Invoices.Client, HR.Employee) as Partner;
      // (2) introduces new associations
      relate (Sales.Purchases, Invoices.Invoice) as paidIn;
      // (3) introduces new associations and automatically keeps them consistent
      sync (Sales.Customer.address, Invoices.Client.address);
}
```

About



Personal

- Associate Professor @ HVL
  - since: August 2023
  - topics:
    - (model-based) software engineering (& formal foundations thereof),
    - digital twins (process modelling),
    - data engineering,
    - health informatics
  - web: https://www.hvl.no/person/?user=Patrick.Stuenkel
- previously: PostDoc @ HelseBergen
  - project employment @ Patologi i Vest (PiV)
  - topics:
    - workflow optimization (automated planning)
    - process modelling
    - process mining
    - digital pathology
  - web: https://www.helse-bergen.no/piv



Education

- PhD
  - defence: 9th February 2022
  - title: A framework for multi-model consistency management
  - topics:
    - software interoperability & integration,
    - heterogeneous modelling (category theory),
    - inter-model consistency,
  - web: https://hvlopen.brage.unit.no/hvlopen-xmlui/handle/11250/2837740

todays talk

- before that
  - Bachelor & Masters from FHDW Hannover
  - supervised by *Michael Löwe**1956-2019†:
    - algebraic specification (partial algebras)
    - graph transformation (**S**ingle **P**ush**o**ut Approach)
    - category theory

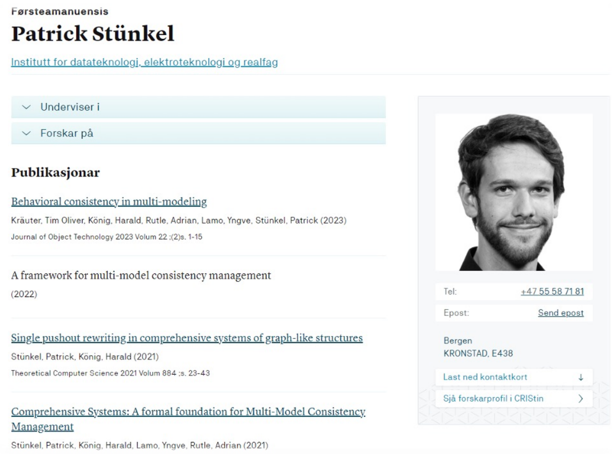
https://past.corrlang.io/

https://github.com/webminz

Personal

- Associate Professor @ HVL
  - since: August 2023
  - topics:
    - (model-based) software engineering (& formal foundations thereof),
    - digital twins (process modelling),
    - data engineering,
    - health informatics
  - web: https://www.hvl.no/person/?user=Patrick.Stuenkel

- previously: PostDoc @ HelseBergen
  - project employment @ Patologi i Vest (PiV)
  - topics:
    - workflow optimization (automated planning)
    - process modelling
    - process mining
    - digital pathology
  - web: https://www.helse-bergen.no/piv



Førsteamanuensis

Patrick Stünkel

Institutt for datateknologi, elektroteknologi og realfag

Publikasjonar

Behavioral consistency in multi-modeling

Kräuter, Tim Oliver, König, Harald, Rutle, Adrian, Lamo, Yngve, Stünkel, Patrick (2023)

Journal of Object Technology 2023 Volum 22 ;(2)s. 1-15

A framework for multi-model consistency management

(2022)

Single pushout rewriting in comprehensive systems of graph-like structures

Stünkel, Patrick, König, Harald (2021)

Theoretical Computer Science 2021 Volum 884 ;s. 23-43

Comprehensive Systems: A formal foundation for Multi-Model Consistency Management

Stünkel, Patrick, König, Harald, Lamo, Yngve, Rutle, Adrian (2021)

Tel: +47 55 58 71 81

Epost: Send epost

Bergen
KRONSTAD, E438

Education

- PhD
  - defence: 9th February 2022
  - title: A framework for multi-model consistency management
  - topics:
    - software interoperability & integration,
    - heterogeneous modelling (category theory),
    - inter-model consistency,

    todays talk
  - web: https://hvlopen.brage.unit.no/hvlopen-xmlui/handle/11250/2837740

A FRAMEWORK FOR
MULTI-MODEL CONSISTENCY
MANAGEMENT

Doctoral Dissertation by
Patrick Stünkel

Thesis submitted for
the degree of Philosophiae Doctor (PhD)
in
Computer Science:
Software Engineering, Sensor Networks and Engineering Computing

Department of Computer Science,
Electrical Engineering and Mathematical Sciences

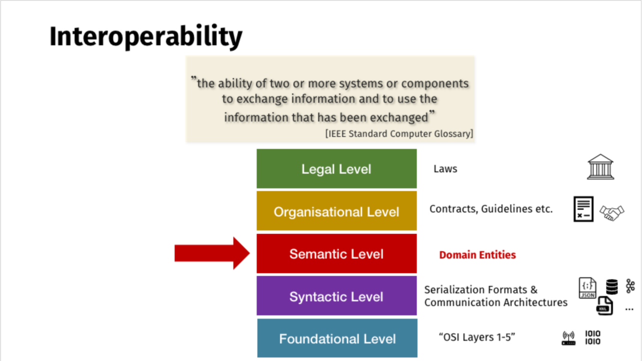
Faculty of Engineering and Science

Western Norway University of Applied Sciences

September 16, 2021

- before that
  - Bachelor & Masters from FHDW Hannover
  - supervised by *Michael Löwe*$^{*1956-2019\dagger}$:
    - algebraic specification (partial algebras)
    - graph transformation (**S**ingle **P**ush**o**ut Approach)
    - category theory

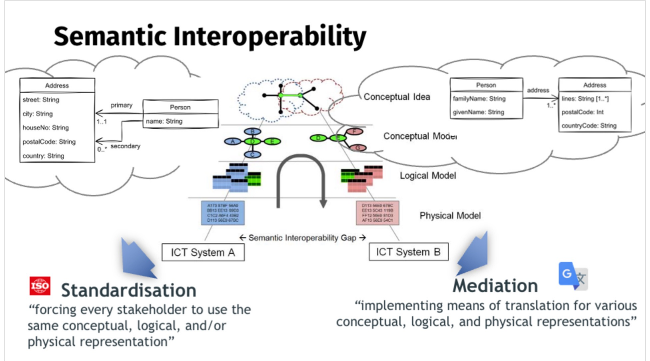
Background



Interoperability

> "the ability of two or more systems or components to exchange information and to use the information that has been exchanged"
> [IEEE Standard Computer Glossary]

Legal Level	Laws
Organisational Level	Contracts, Guidelines etc.
Semantic Level	**Domain Entities**
Syntactic Level	Serialization Formats & Communication Architectures
Foundational Level	"OSI Layers 1-5"

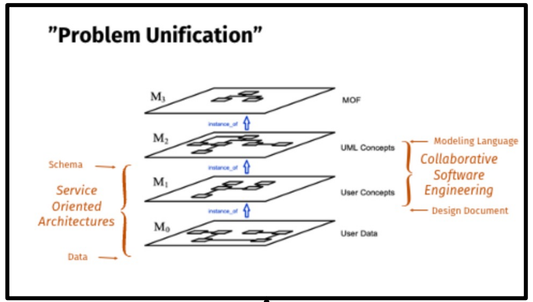


Semantic Interoperability

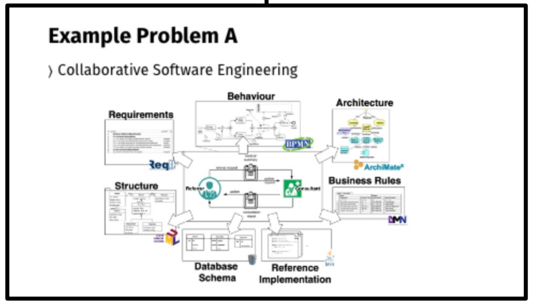
Standardisation
"forcing every stakeholder to use the same conceptual, logical, and/or physical representation"

Mediation
"implementing means of translation for various conceptual, logical, and physical representations"

Example

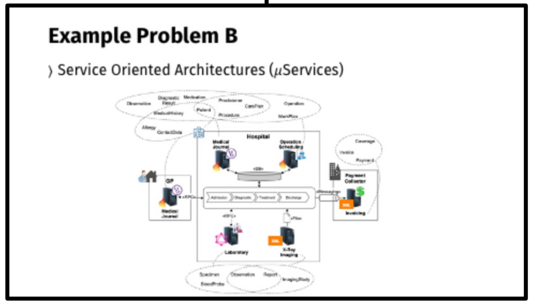


"Problem Unification"



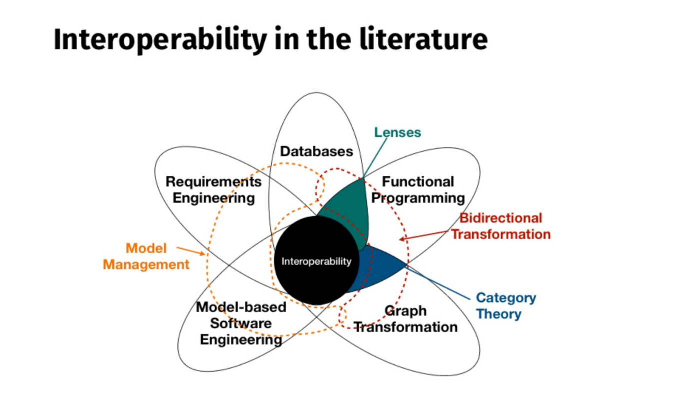
Example Problem A

- Collaborative Software Engineering

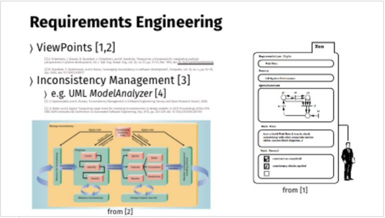


Example Problem B

- Service Oriented Architectures (μServices)



Interoperability in the literature

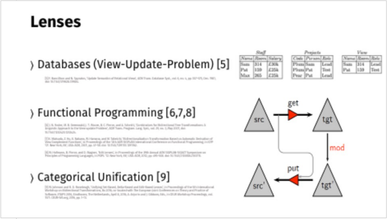


Requirements Engineering

- ViewPoints [1,2]
- Inconsistency Management [3]
  - e.g. UML *ModelAnalyzer* [4]

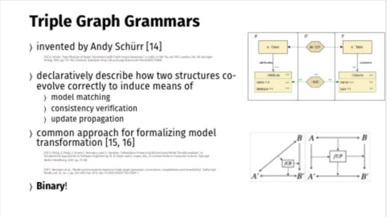
from [2]

from [1]



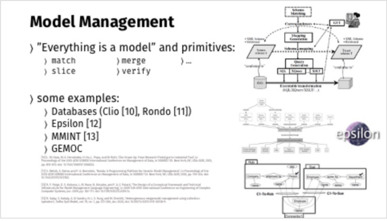
Lenses

- Databases (View-Update-Problem) [5]
- Functional Programming [6,7,8]
- Categorical Unification [9]



Triple Graph Grammars

- invented by Andy Schürr [14]
- declaratively describe how two structures co-evolve correctly to induce means of
  - model matching
  - consistency verification
  - update propagation
- common approach for formalizing model transformation [15, 16]
- **Binary!**

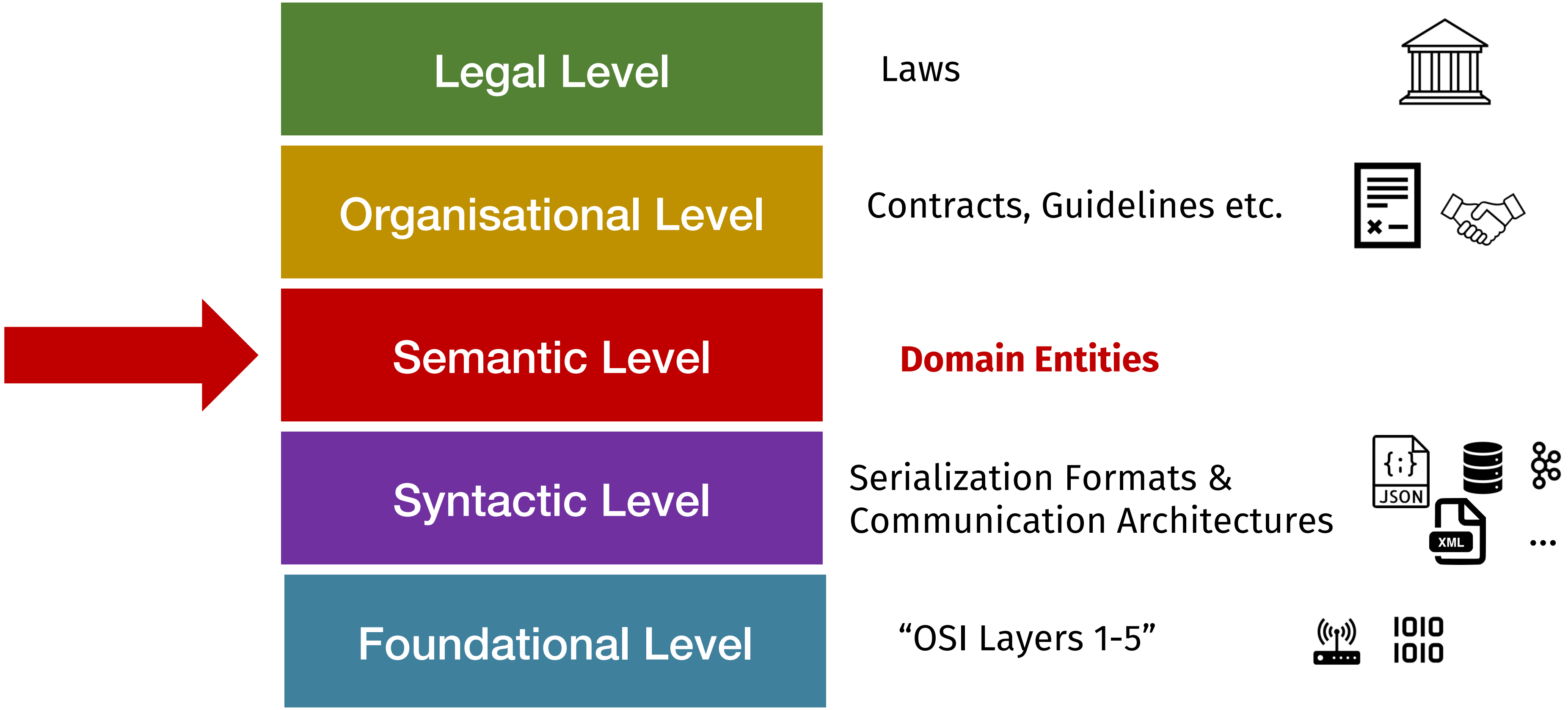


Model Management

- "Everything is a model" and primitives:
  - `match`
  - `slice`
  - `merge`
  - `verify`
  - ...
- some examples:
  - Databases (Clio [10], Rondo [11])
  - Epsilon [12]
  - MMINT [13]
  - GEMOC

Interoperability

> ”the ability of two or more systems or components to exchange information and to use the information that has been exchanged”
>
> [IEEE Standard Computer Glossary]



Legal Level — Laws

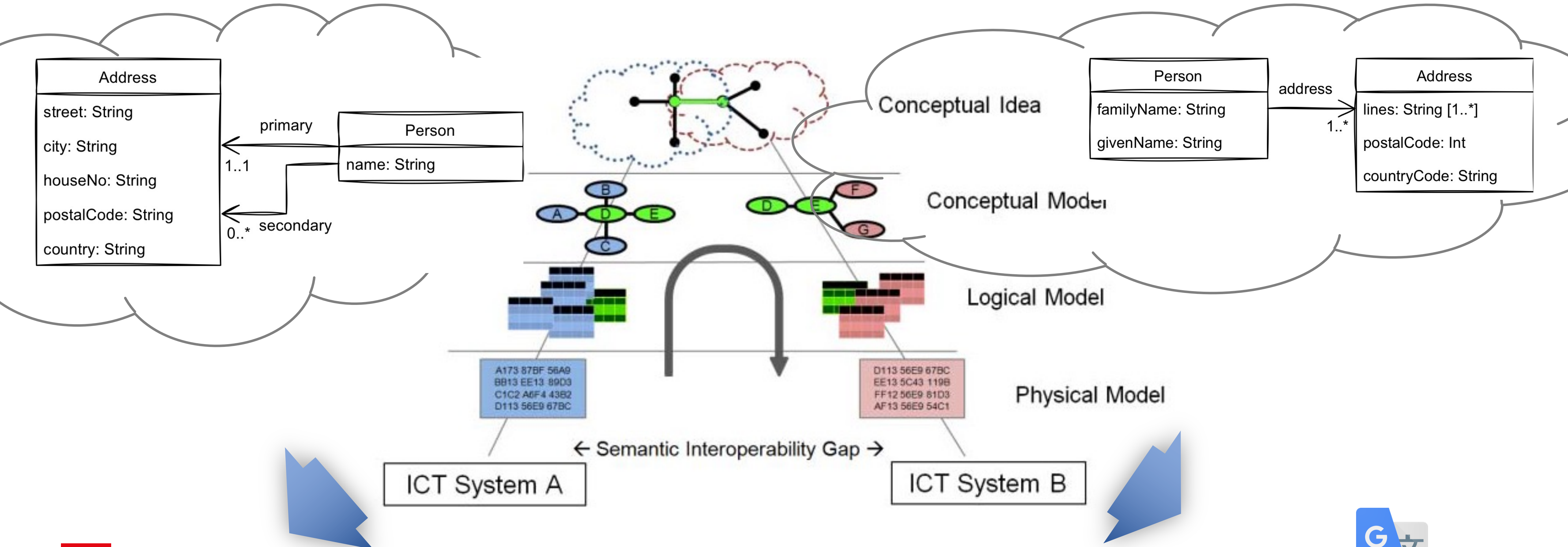
Organisational Level — Contracts, Guidelines etc.

Semantic Level — **Domain Entities**

Syntactic Level — Serialization Formats & Communication Architectures …

Foundational Level — “OSI Layers 1-5”

Semantic Interoperability



Standardisation

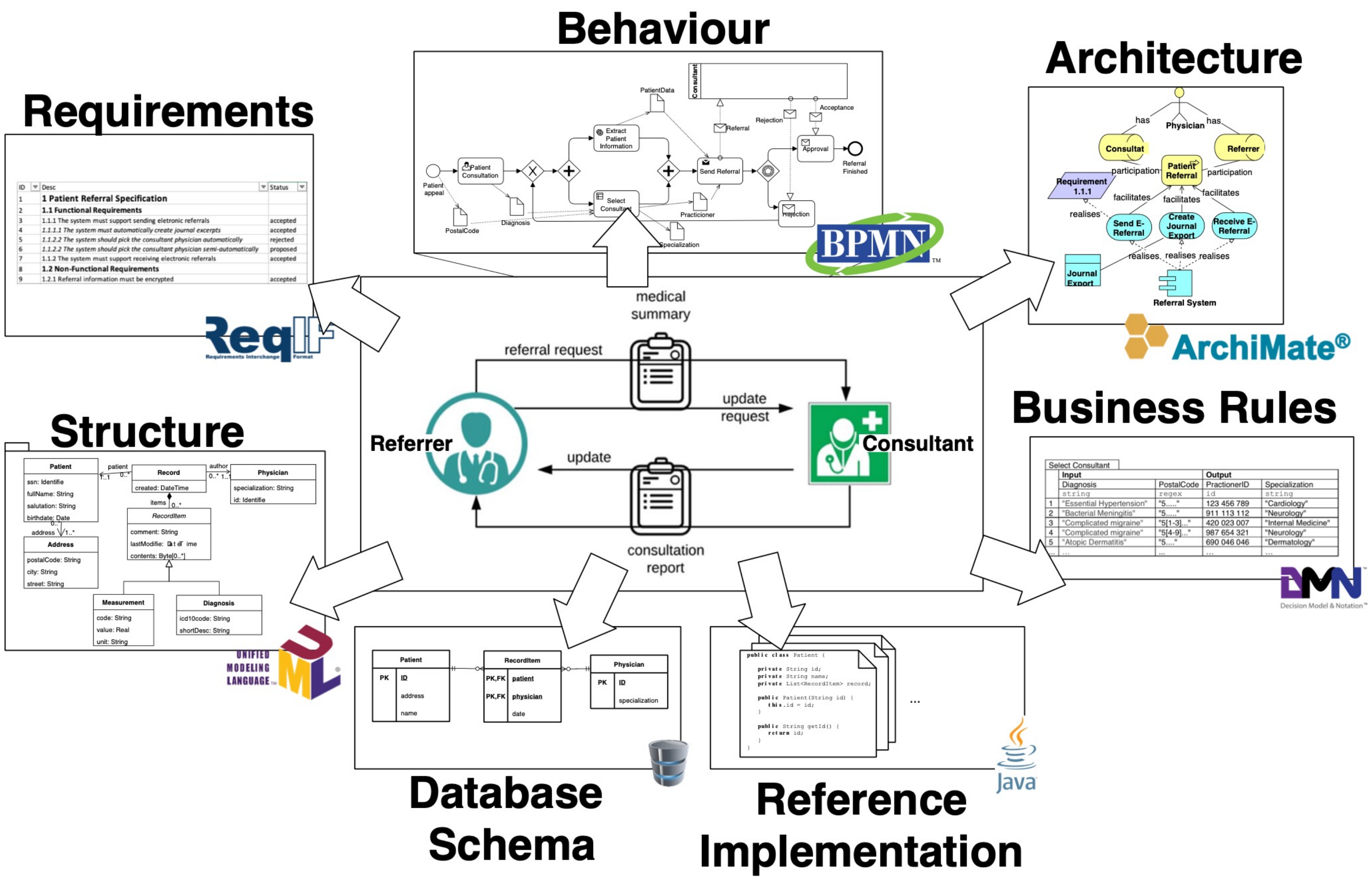
“forcing every stakeholder to use the same conceptual, logical, and/or physical representation”

Mediation

“implementing means of translation for various conceptual, logical, and physical representations”

Example Problem A

› Collaborative Software Engineering



ID	Desc	Status
1	**1 Patient Referral Specification**	
2	**1.1 Functional Requirements**	
3	1.1.1 The system must support sending eletronic referrals	accepted
4	*1.1.1.1 The system must automatically create journal excerpts*	accepted
5	*1.1.2.2 The system should pick the consultant physician automatically*	rejected
6	*1.1.2.2 The system should pick the consultant physician semi-automatically*	proposed
7	1.1.2 The system must support receiving electronic referrals	accepted
8	**1.2 Non-Functional Requirements**	
9	1.2.1 Referral information must be encrypted	accepted

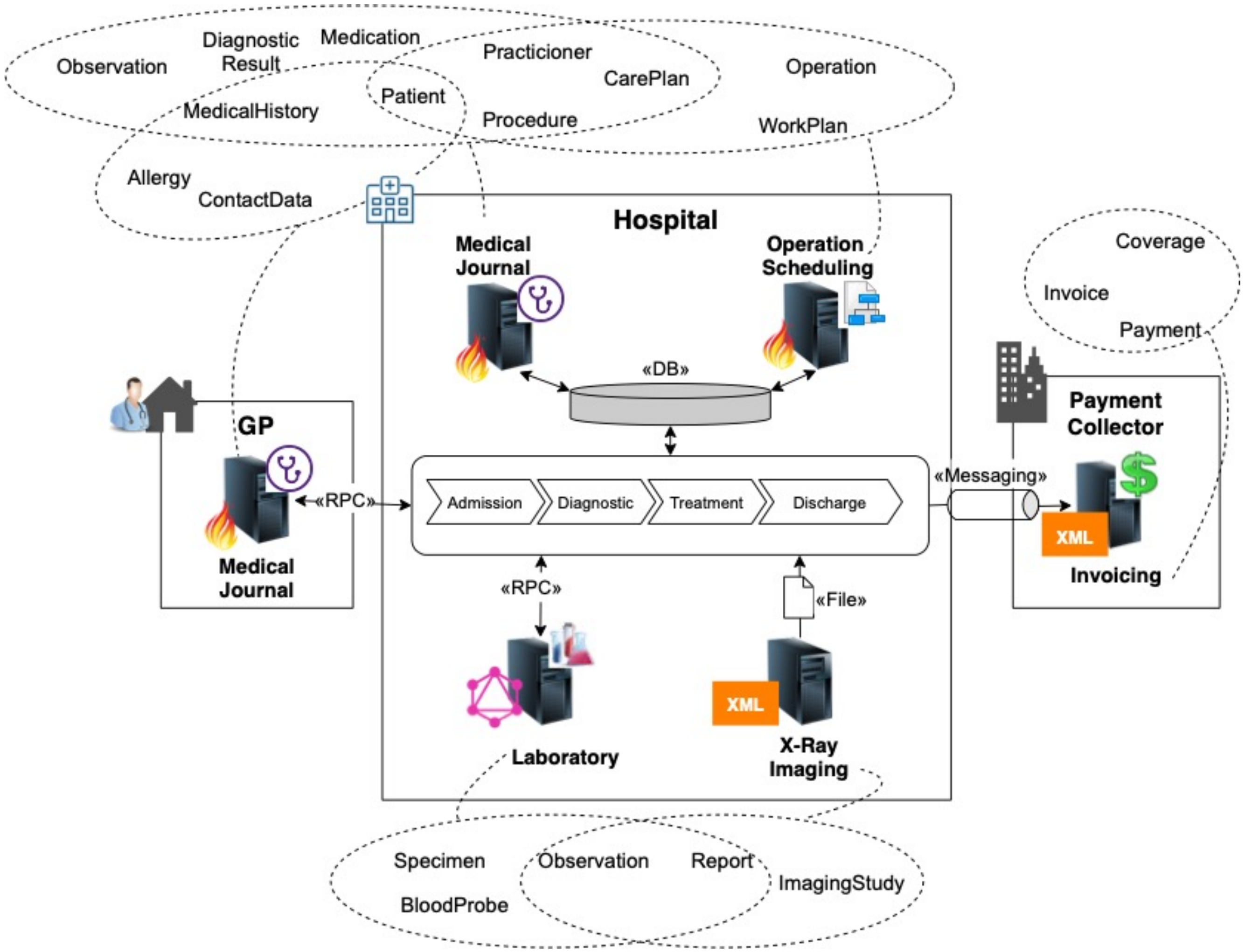
Select Consultant

	Input		Output	
	Diagnosis	PostalCode	PractionerID	Specialization
	string	regex	id	string
1	"Essential Hypertension"	"5....."	123 456 789	"Cardiology"
2	"Bacterial Meningitis"	"5....."	911 113 112	"Neurology"
3	"Complicated migraine"	"5[1-3]..."	420 023 007	"Internal Medicine"
4	"Complicated migraine"	"5[4-9]..."	987 654 321	"Neurology"
5	"Atopic Dermatitis"	"5...."	690 046 046	"Dermatology"
...	...	...	...	...

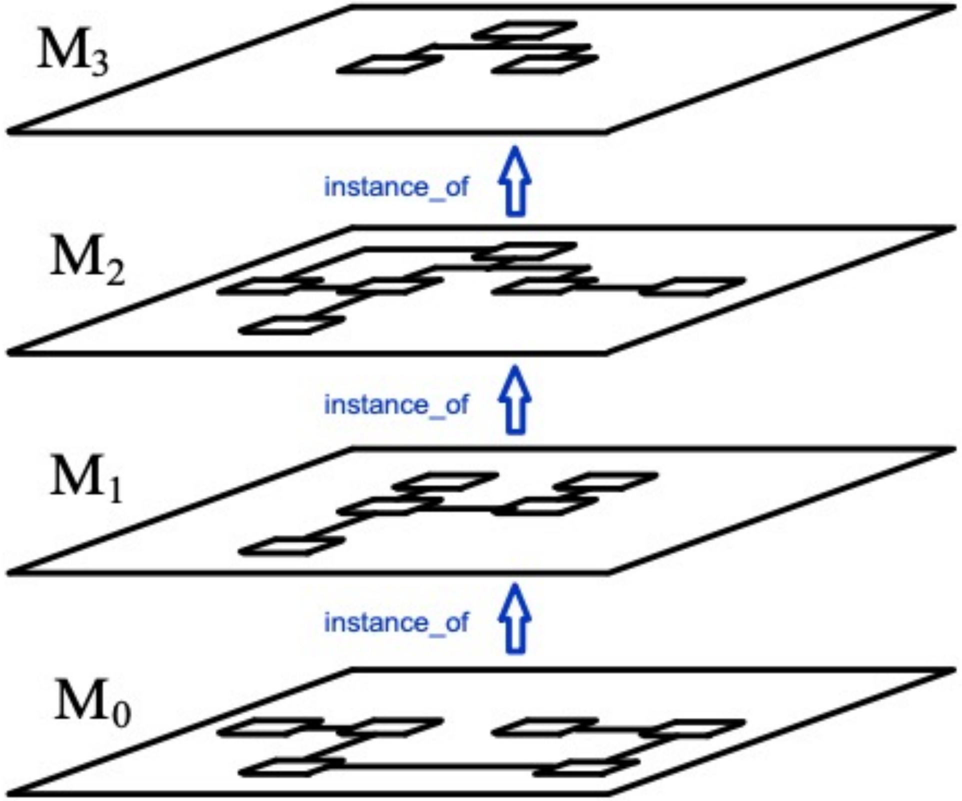
```
public class Patient {

  private String id;
  private String name;
  private List<RecordItem> record;

  public Patient(String id) {
    this.id = id;
  }

  public String getId() {
    return id;
  }
}
```

Example Problem B

〉 Service Oriented Architectures (μServices)



”Problem Unification”



MOF

UML Concepts

User Concepts

User Data

Modeling Language

Collaborative Software Engineering

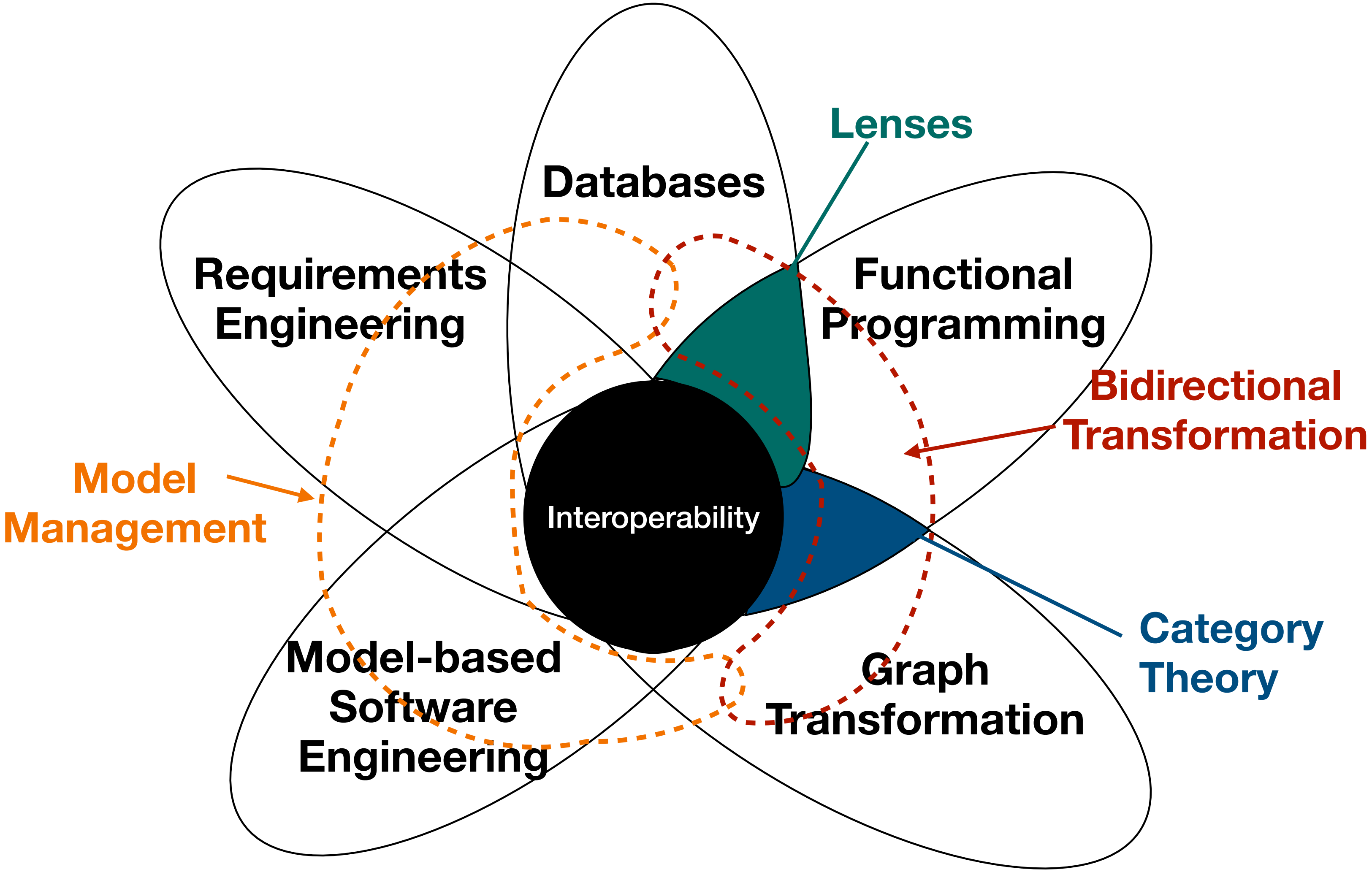
Design Document

Schema

Service Oriented Architectures

Data

Interoperability in the literature



Requirements Engineering

- ViewPoints [1,2]

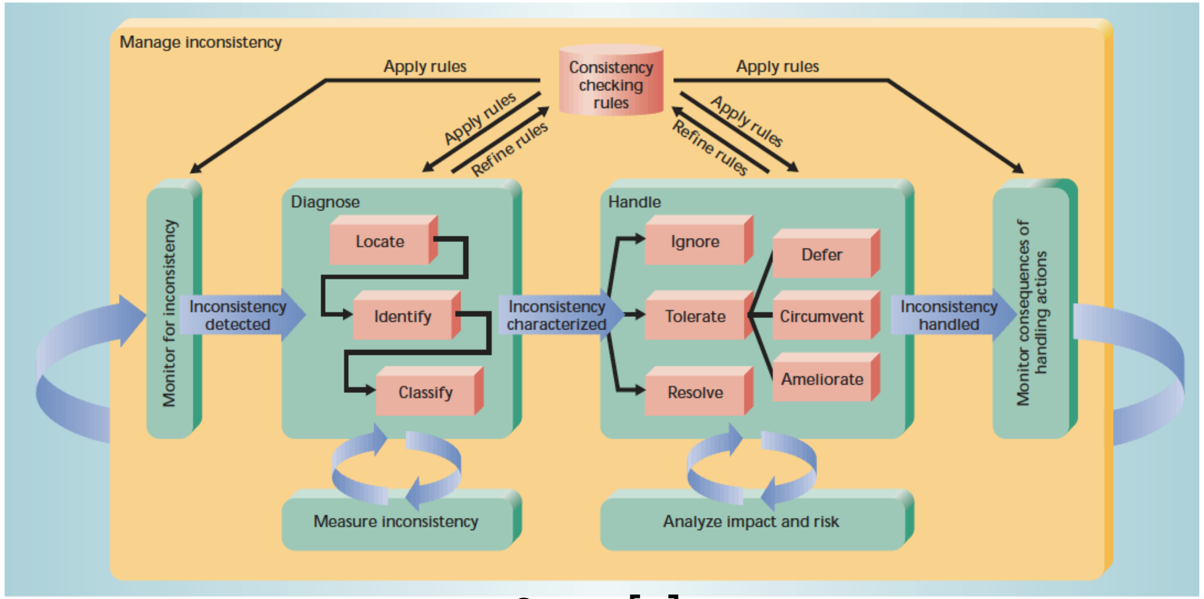
[1] A. Finkelstein, J. Kramer, B. Nuseibeh, L. Finkelstein, and M. Goedicke, ‘Viewpoints: a framework for integrating multiple perspectives in system development’, *Int. J. Soft. Eng. Knowl. Eng.*, vol. 02, no. 01, pp. 31–57, Mar. 1992, doi: 10.1142/S0218194092000038.

[2] B. Nuseibeh, S. Easterbrook, and A. Russo, ‘Leveraging inconsistency in software development’, Computer, vol. 33, no. 4, pp. 24–29, Apr. 2000, doi: 10.1109/2.839317.

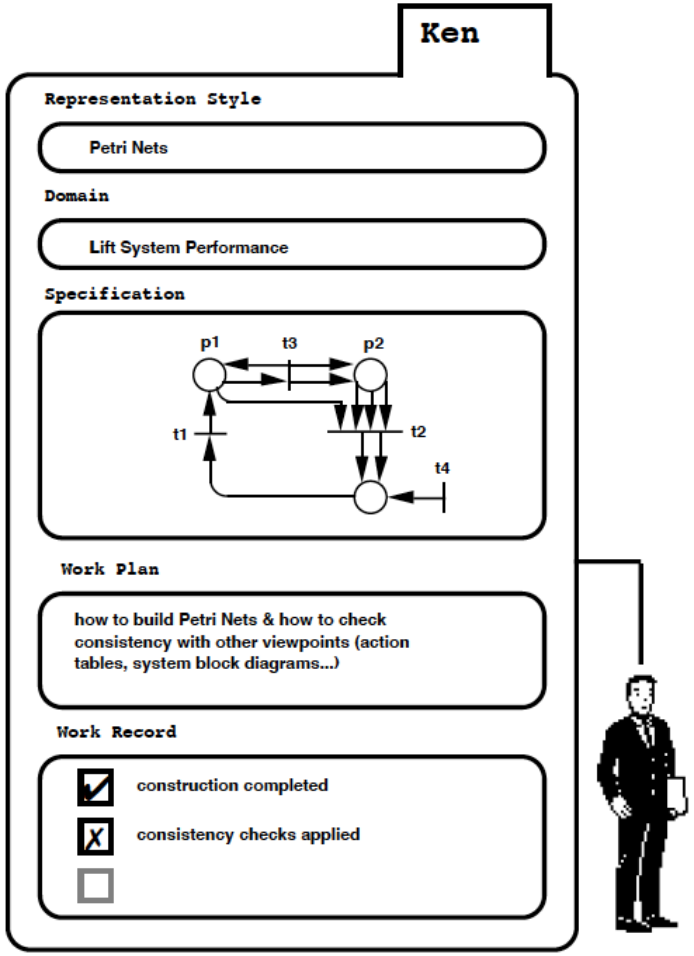
- Inconsistency Management [3]
    - e.g. UML *ModelAnalyzer* [4]

[3] G. Spanoudakis and A. Zisman, ‘Inconsistency Management in Software Engineering: Survey and Open Research Issues’, 2000.

[4] A. Reder and A. Egyed, ‘Computing repair trees for resolving inconsistencies in design models’, in 2012 Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering, Sep. 2012, pp. 220–229. doi: 10.1145/2351676.2351707.



from [2]



Ken

Representation Style

Petri Nets

Domain

Lift System Performance

Specification

Work Plan

how to build Petri Nets & how to check consistency with other viewpoints (action tables, system block diagrams...)

Work Record

☑ construction completed

☒ consistency checks applied

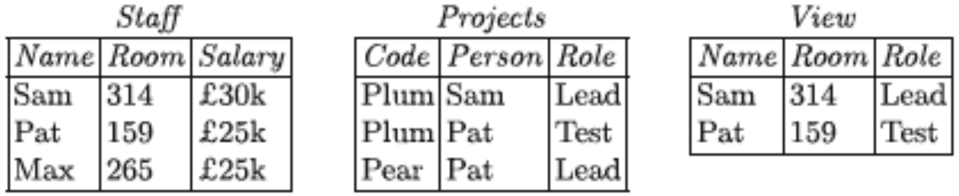
□

from [1]

Lenses

- Databases (View-Update-Problem) [5]

[5] F. Bancilhon and N. Spyratos, ‘Update Semantics of Relational Views’, ACM Trans. Database Syst., vol. 6, no. 4, pp. 557–575, Dec. 1981, doi: 10.1145/319628.319634.



Staff

Name	*Room*	*Salary*
Sam	314	£30k
Pat	159	£25k
Max	265	£25k

Projects

Code	*Person*	*Role*
Plum	Sam	Lead
Plum	Pat	Test
Pear	Pat	Lead

View

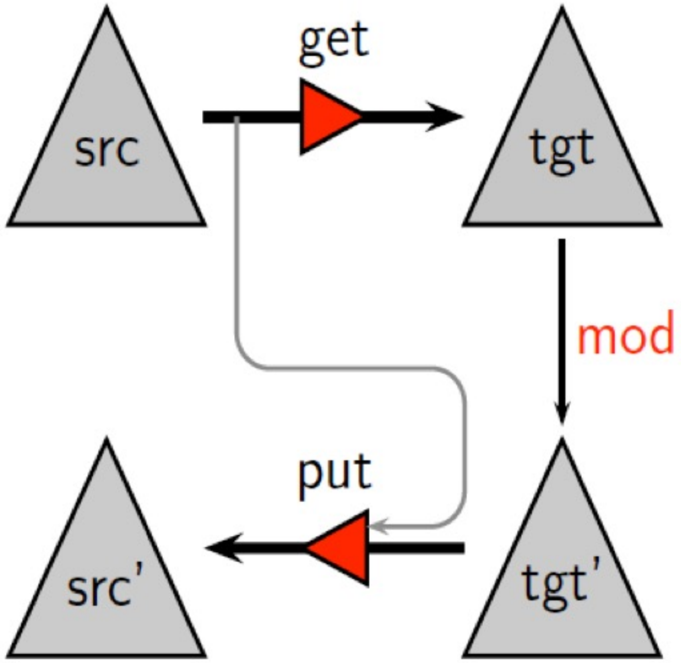
Name	*Room*	*Role*
Sam	314	Lead
Pat	159	Test

- Functional Programming [6,7,8]

[6] J. N. Foster, M. B. Greenwald, J. T. Moore, B. C. Pierce, and A. Schmitt, ‘Combinators for Bidirectional Tree Transformations: A Linguistic Approach to the View-update Problem’, ACM Trans. Program. Lang. Syst., vol. 29, no. 3, May 2007, doi: 10.1145/1232420.1232424.

[7] K. Matsuda, Z. Hu, K. Nakano, M. Hamana, and M. Takeichi, ‘Bidirectionalization Transformation Based on Automatic Derivation of View Complement Functions’, in Proceedings of the 12th ACM SIGPLAN International Conference on Functional Programming, in ICFP ’07. New York, NY, USA: ACM, 2007, pp. 47–58. doi: 10.1145/1291151.1291162.

[8] M. Hofmann, B. Pierce, and D. Wagner, ‘Edit Lenses’, in Proceedings of the 39th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, in POPL ’12. New York, NY, USA: ACM, 2012, pp. 495–508. doi: 10.1145/2103656.2103715.



- Categorical Unification [9]

[9] M. Johnson and R. D. Rosebrugh, ‘Unifying Set-Based, Delta-Based and Edit-Based Lenses’, in Proceedings of the 5th International Workshop on Bidirectional Transformations, Bx 2016, co-located with The European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 8, 2016, A. Anjorin and J. Gibbons, Eds., in CEUR Workshop Proceedings, vol. 1571. CEUR-WS.org, 2016, pp. 1–13.

Model Management

- ”Everything is a model” and primitives:
  - match
  - slice
  - merge
  - verify
  - ...

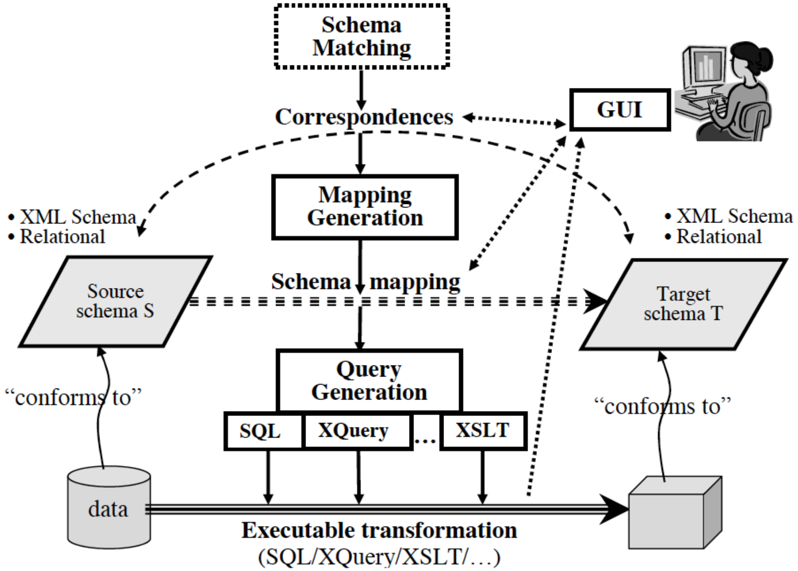
- some examples:
  - Databases (Clio [10], Rondo [11])
  - Epsilon [12]
  - MMINT [13]
  - GEMOC

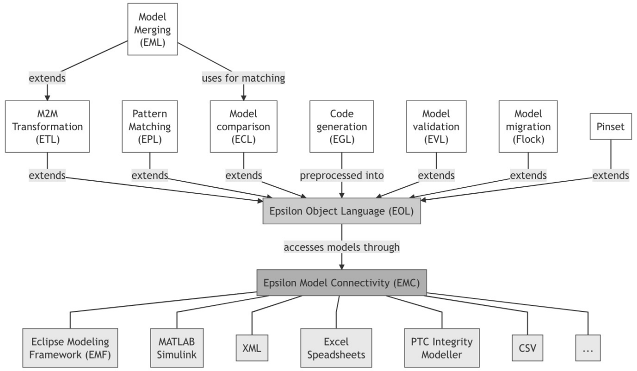
[10] L. M. Haas, M. A. Hernández, H. Ho, L. Popa, and M. Roth, ‘Clio Grows Up: From Research Prototype to Industrial Tool’, in Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, in SIGMOD ’05. New York, NY, USA: ACM, 2005, pp. 805–810. doi: 10.1145/1066157.1066252.

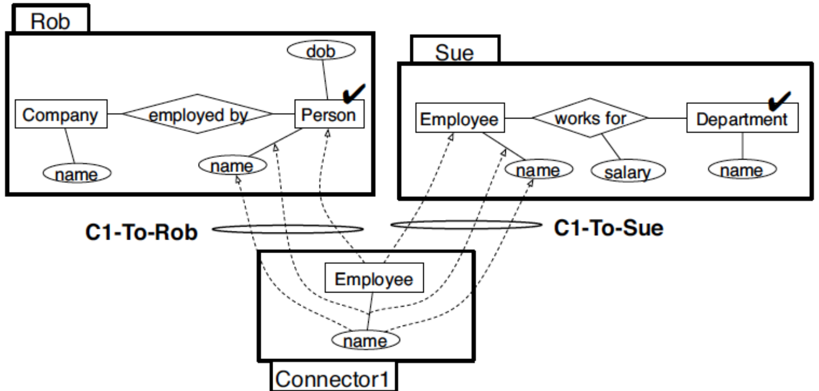
[11] S. Melnik, E. Rahm, and P. A. Bernstein, ‘Rondo: A Programming Platform for Generic Model Management’, in Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, in SIGMOD ’03. New York, NY, USA: ACM, 2003, pp. 193–204. doi: 10.1145/872757.872782.

[12] R. F. Paige, D. S. Kolovos, L. M. Rose, N. Drivalos, and F. A. C. Polack, ‘The Design of a Conceptual Framework and Technical Infrastructure for Model Management Language Engineering’, in 2009 14th IEEE International Conference on Engineering of Complex Computer Systems, Jun. 2009, pp. 162–171. doi: 10.1109/ICECCS.2009.14.

[13] R. Salay, S. Kokaly, A. Di Sandro, N. L. S. Fung, and M. Chechik, ‘Heterogeneous megamodel management using collection operators’, Softw Syst Model, vol. 19, no. 1, pp. 231–260, Jan. 2020, doi: 10.1007/s10270-019-00738-9.







Triple Graph Grammars

- invented by Andy Schürr [14]

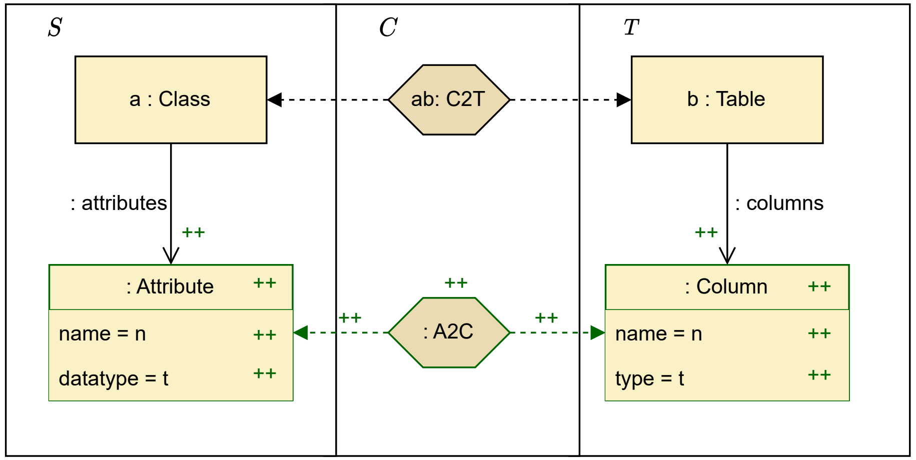
  [14] A. Schürr, ‘Specification of Graph Translators with Triple Graph Grammars’, in LNCS, in WG ’94, vol. 903. London, UK, UK: Springer-Verlag, 1994, pp. 151–163. [Online]. Available: http://dl.acm.org/citation.cfm?id=647675.731658

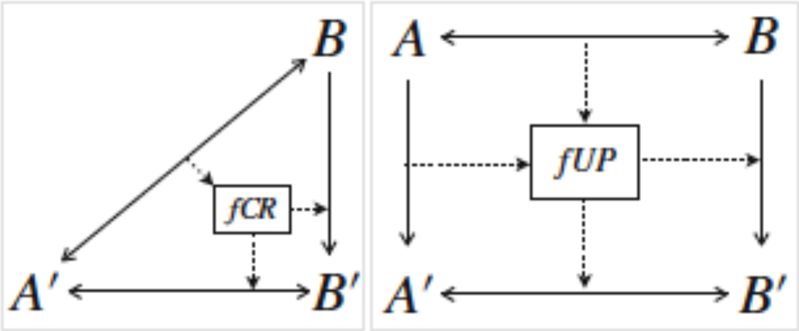
- declaratively describe how two structures co-evolve correctly to induce means of
  - model matching
  - consistency verification
  - update propagation
- common approach for formalizing model transformation [15, 16]

  [15] H. Ehrig, K. Ehrig, C. Ermel, F. Hermann, and G. Taentzer, ‘Information Preserving Bidirectional Model Transformations’, in Fundamental Approaches to Software Engineering, M. B. Dwyer and A. Lopes, Eds., in Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, pp. 72–86.

  [16] F. Hermann et al., ‘Model synchronization based on triple graph grammars: correctness, completeness and invertibility’, Softw Syst Model, vol. 14, no. 1, pp. 241–269, Feb. 2015, doi: 10.1007/s10270-012-0309-1.

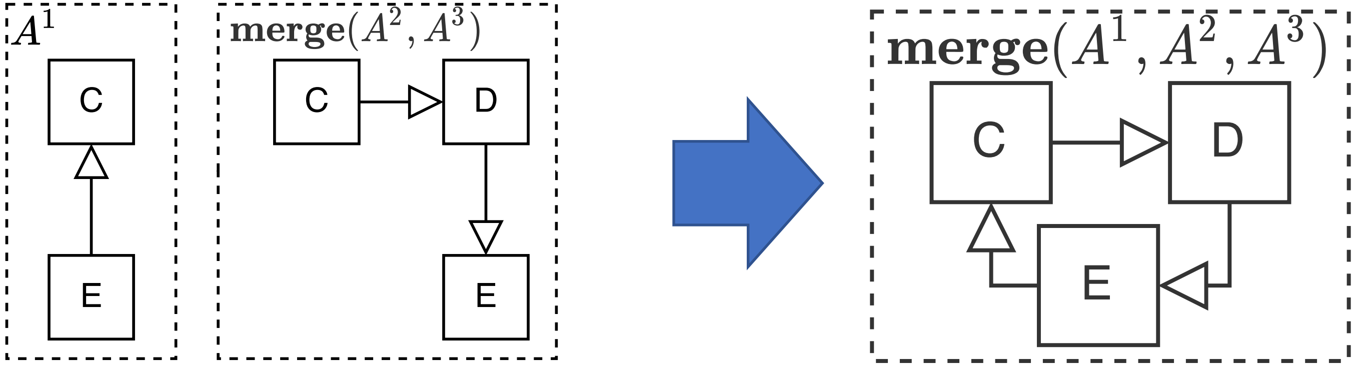
- **Binary**!





Bin-ary vs. Multi-ary ?!

Example (© Z.Diskin): Remember: ”*Inheritance must be acyclic!*”

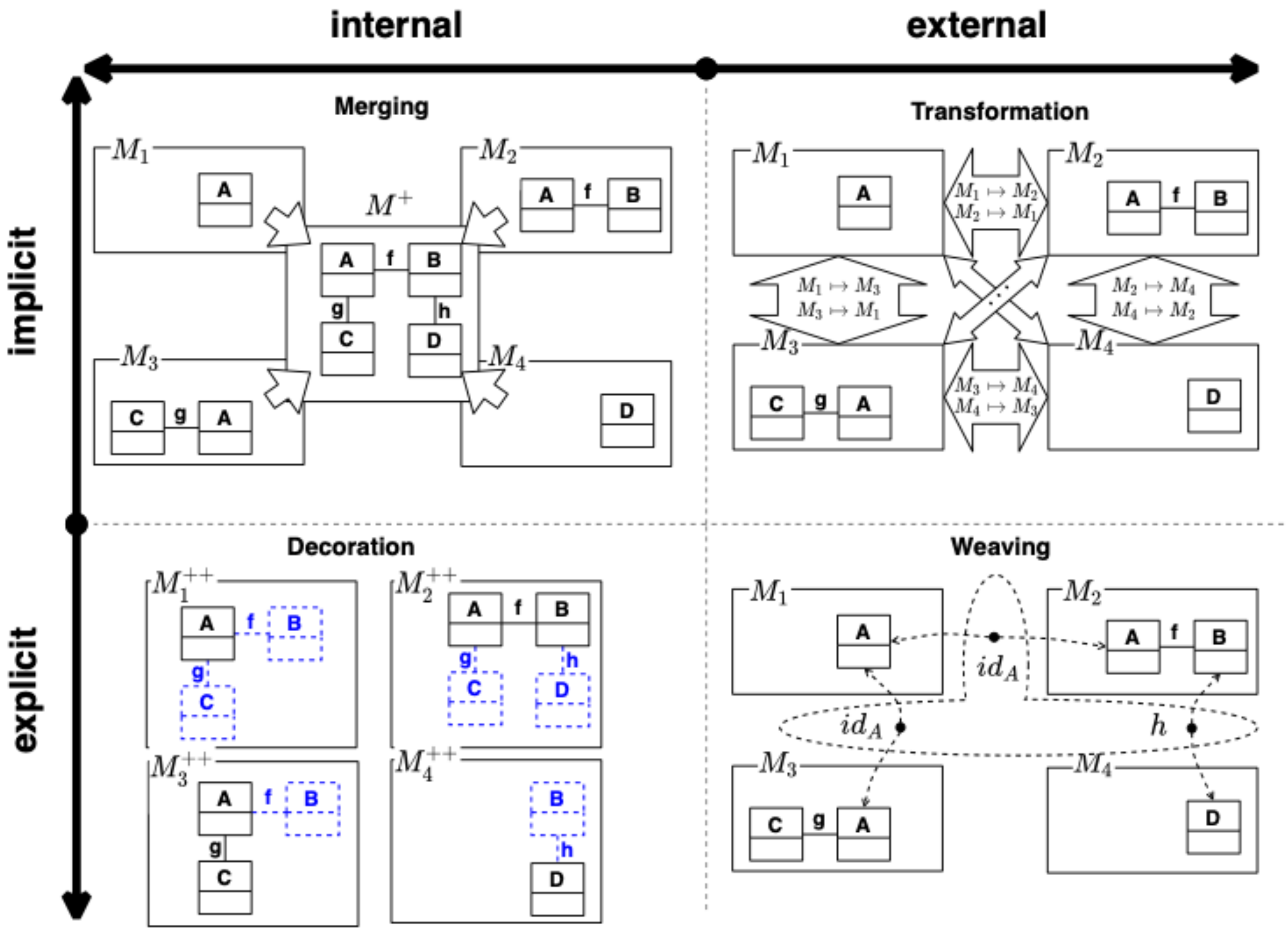


also **Peirce’s Reduction Thesis**: ”all relations can be generated from the ensemble of unary, binary and ternary relations, but that at least some ternary relations cannot be reduced to relations of lower arity” [17]

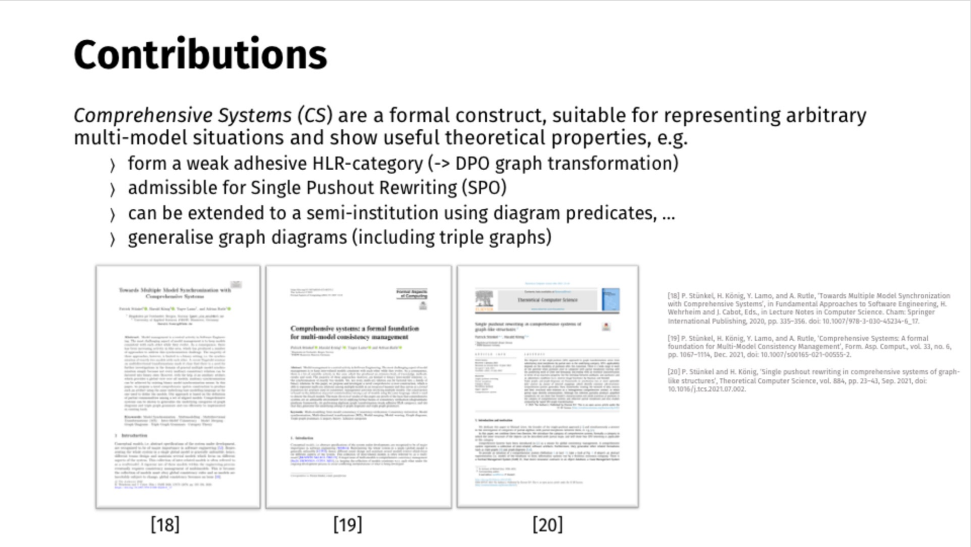
[17] J. Hereth and R. Pöschel, ‘Peircean Algebraic Logic and Peirce’s Reduction Thesis’, Semiotica, vol. 2011, no. 186, pp. 141–167, 2011, doi: 10.1515/semi.2011.050.

=> We need **multi-ary** consistency management!

Classification Attempt



Theory



Contributions

Comprehensive Systems (CS) are a formal construct, suitable for representing arbitrary multi-model situations and show useful theoretical properties, e.g.

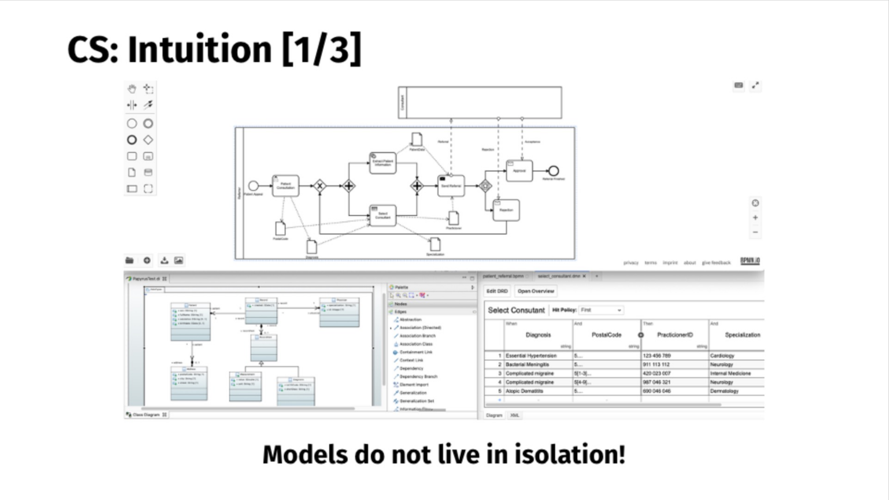
- form a weak adhesive HLR-category (-> DPO graph transformation)
- admissible for Single Pushout Rewriting (SPO)
- can be extended to a semi-institution using diagram predicates, ...
- generalise graph diagrams (including triple graphs)

[18] [19] [20]

[18] P. Stünkel, H. König, Y. Lamo, and A. Rutle, 'Towards Multiple Model Synchronization with Comprehensive Systems', in Fundamental Approaches to Software Engineering, H. Wehrheim and J. Cabot, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 335–356. doi: 10.1007/978-3-030-45234-6_17.

[19] P. Stünkel, H. König, Y. Lamo, and A. Rutle, 'Comprehensive Systems: A formal foundation for Multi-Model Consistency Management', Form. Asp. Comput., vol. 33, no. 6, pp. 1067–1114, Dec. 2021, doi: 10.1007/s00165-021-00555-2.

[20] P. Stünkel and H. König, 'Single pushout rewriting in comprehensive systems of graph-like structures', Theoretical Computer Science, vol. 884, pp. 23–43, Sep. 2021, doi: 10.1016/j.tcs.2021.07.002.

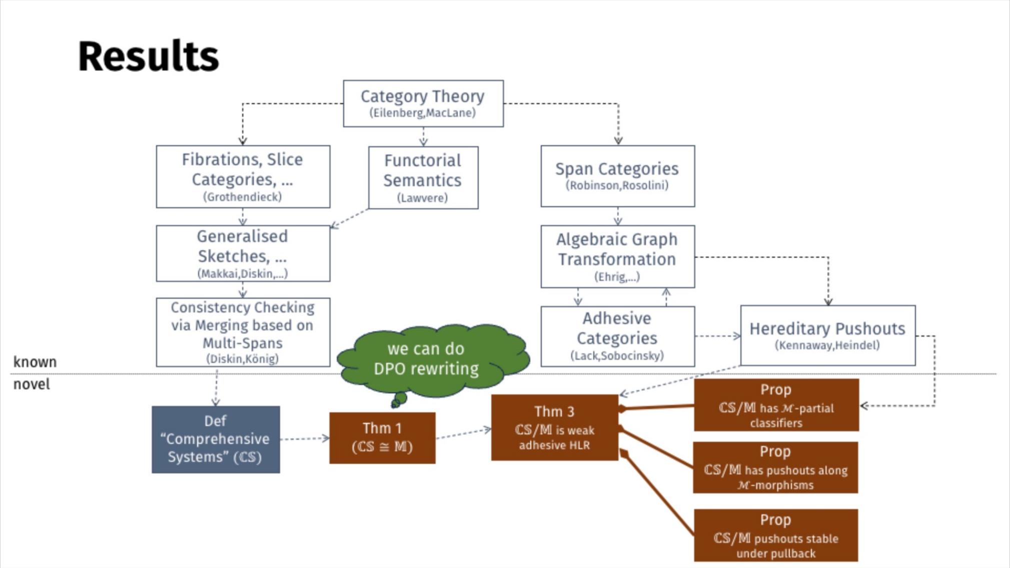


CS: Intuition [1/3]

Select Consutant | Hit Policy: First

	When Diagnosis	And PostalCode	Then PracticionerID	And Specialization
1	Essential Hypertension	5....	123 456 789	Cardiology
2	Bacterial Meningitis	5....	911 113 112	Neurology
3	Complicated migraine	5[1-3]...	420 023 007	Internal Medicione
4	Complicated migraine	5[4-9]...	987 046 321	Neurology
5	Atopic Dematitis	5....	690 046 046	Dermatology

Models do not live in isolation!



Results

Category Theory (Eilenberg,MacLane)

Fibrations, Slice Categories, ... (Grothendieck)

Functorial Semantics (Lawvere)

Span Categories (Robinson,Rosolini)

Generalised Sketches, ... (Makkai,Diskin,...)

Algebraic Graph Transformation (Ehrig,...)

Consistency Checking via Merging based on Multi-Spans (Diskin,König)

Adhesive Categories (Lack,Sobocinsky)

Hereditary Pushouts (Kennaway,Heindel)

known

novel

we can do DPO rewriting

Def "Comprehensive Systems" ($\mathbb{CS}$)

Thm 1 ($\mathbb{CS} \cong \mathbb{M}$)

Thm 3 $\mathbb{CS}/\mathbb{M}$ is weak adhesive HLR

Prop $\mathbb{CS}/\mathbb{M}$ has $\mathcal{M}$-partial classifiers

Prop $\mathbb{CS}/\mathbb{M}$ has pushouts along $\mathcal{M}$-morphisms

Prop $\mathbb{CS}/\mathbb{M}$ pushouts stable under pullback

Contributions

Comprehensive Systems (*CS*) are a formal construct, suitable for representing arbitrary multi-model situations and show useful theoretical properties, e.g.

- form a weak adhesive HLR-category (-> DPO graph transformation)
- admissible for Single Pushout Rewriting (SPO)
- can be extended to a semi-institution using diagram predicates, …
- generalise graph diagrams (including triple graphs)

Towards Multiple Model Synchronization with Comprehensive Systems

Patrick Stünkel[1], Harald König[2], Yngve Lamo[1], and Adrian Rutle[1]

[1] Høgskulen på Vestlandet, Bergen, Norway {past,yla,aru}@hvl.no
[2] University of Applied Sciences, FHDW, Hannover, Germany
Harald.Koenig@fhdw.de

Abstract. Model management is a central activity in Software Engineering. The most challenging aspect of model management is to keep models consistent with each other while they evolve. As a consequence, there has been increasing activity in this area, which has produced a number of approaches to address this synchronization challenge. The majority of these approaches, however, is limited to a binary setting; i.e. the synchronization of exactly two models with each other. A recent Dagstuhl seminar on multidirectional transformations made it clear that there is a need for further investigations in the domain of general multiple model synchronization simply because not every multiary consistency relation can be factored into binary ones. However, with the help of an auxiliary artifact, which provides a global view over all models, multiary synchronization can be achieved by existing binary model synchronization means. In this paper, we propose a novel *comprehensive system* construction to produce such an artifact using the same underlying base modelling language as the one used to define the models. Our approach is based on the definition of partial commonalities among a set of aligned models. Comprehensive systems can be shown to generalize the underlying categories of graph diagrams and triple graph grammars and can efficiently be implemented in existing tools.

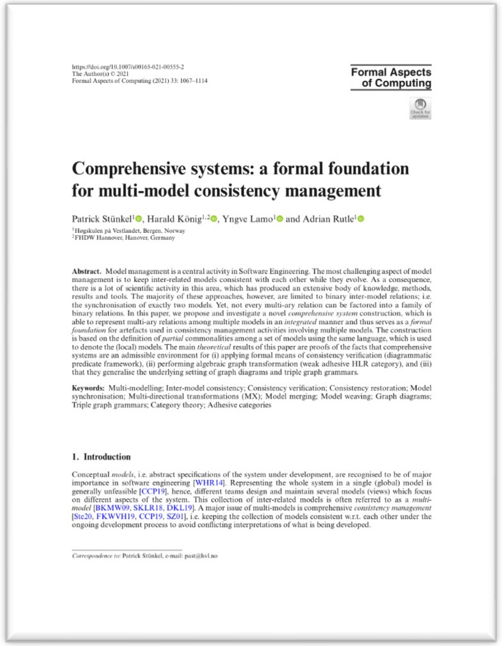
Keywords: Model Synchronization · Multimodelling · Multidirectional Transformations (MX) · Inter-Model Consistency · Model Merging · Graph Diagrams · Triple Graph Grammars · Category Theory

1 Introduction

Conceptual *models*, i.e. abstract specifications of the system under development, are recognized to be of major importance in software engineering [52]. Representing the whole system in a single global model is generally unfeasible, hence, different teams design and maintain several models which focus on different aspects of the system. This collection of inter-related models is often referred to as a *multimodel*. A rigorous use of these models within the engineering process eventually requires consistency management of multimodels. This is because the collection of models must obey global consistency rules and as models are inevitably subject to change, global consistency becomes an issue [16].

© The Author(s) 2020
H. Wehrheim and J. Cabot (Eds.): FASE 2020, LNCS 12076, pp. 335–356, 2020.
https://doi.org/10.1007/978-3-030-45234-6_17

[18]



https://doi.org/10.1007/s00165-021-00555-2
The Author(s) © 2021
Formal Aspects of Computing (2021) 33: 1067–1114

Formal Aspects of Computing

Comprehensive systems: a formal foundation for multi-model consistency management

Patrick Stünkel[1], Harald König[1,2], Yngve Lamo[1] and Adrian Rutle[1]
[1]Høgskulen på Vestlandet, Bergen, Norway
[2]FHDW Hannover, Hanover, Germany

Abstract. Model management is a central activity in Software Engineering. The most challenging aspect of model management is to keep inter-related models consistent with each other while they evolve. As a consequence, there is a lot of scientific activity in this area, which has produced an extensive body of knowledge, methods, results and tools. The majority of these approaches, however, are limited to binary inter-model relations; i.e. the synchronisation of exactly two models. Yet, not every multi-ary relation can be factored into a family of binary relations. In this paper, we propose and investigate a novel *comprehensive system* construction, which is able to represent multi-ary relations among multiple models in an *integrated* manner and thus serves as a *formal foundation* for artefacts used in consistency management activities involving multiple models. The construction is based on the definition of *partial* commonalities among a set of models using the same language, which is used to denote the (local) models. The main *theoretical* results of this paper are proofs of the facts that comprehensive systems are an admissible environment for (i) applying formal means of consistency verification (diagrammatic predicate framework), (ii) performing algebraic graph transformation (weak adhesive HLR category), and (iii) that they generalise the underlying setting of graph diagrams and triple graph grammars.

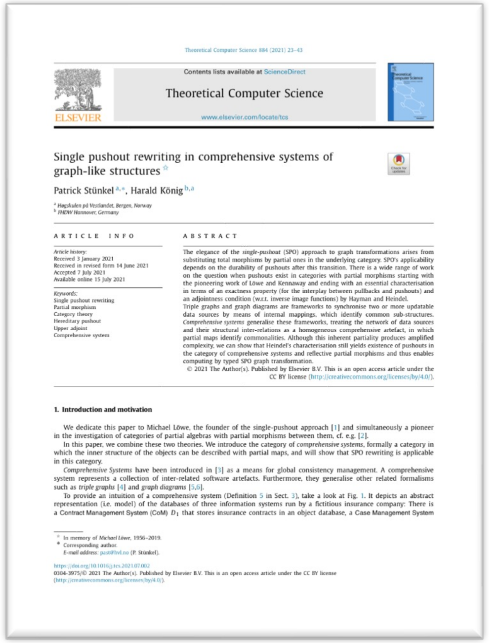
Keywords: Multi-modelling; Inter-model consistency; Consistency verification; Consistency restoration; Model synchronisation; Multi-directional transformations (MX); Model merging; Model weaving; Graph diagrams; Triple graph grammars; Category theory; Adhesive categories

1. Introduction

Conceptual *models*, i.e. abstract specifications of the system under development, are recognised to be of major importance in software engineering [WHR14]. Representing the whole system in a single (global) model is generally unfeasible [CCP19], hence, different teams design and maintain several models (views) which focus on different aspects of the system. This collection of inter-related models is often referred to as a *multi-model* [BKMW09, SKLR18, DKL19]. A major issue of multi-models is comprehensive *consistency management* [Stc20, FKWVH19, CCP19, SZ01], i.e. keeping the collection of models consistent w.r.t. each other under the ongoing development process to avoid conflicting interpretations of what is being developed.

Correspondence to: Patrick Stünkel, e-mail: past@hvl.no

[19]



Theoretical Computer Science 884 (2021) 23–43

Contents lists available at ScienceDirect
Theoretical Computer Science
www.elsevier.com/locate/tcs
ELSEVIER

Single pushout rewriting in comprehensive systems of graph-like structures ☆

Patrick Stünkel [a,*], Harald König [b,a]

[a] Høgskulen på Vestlandet, Bergen, Norway
[b] FHDW Hannover, Germany

ARTICLE INFO

Article history:
Received 3 January 2021
Received in revised form 14 June 2021
Accepted 7 July 2021
Available online 15 July 2021

Keywords:
Single pushout rewriting
Partial morphism
Category theory
Hereditary pushout
Upper adjoint
Comprehensive system

ABSTRACT

The elegance of the *single-pushout* (SPO) approach to graph transformations arises from substituting total morphisms by partial ones in the underlying category. SPO's applicability depends on the durability of pushouts after this transition. There is a wide range of work on the question when pushouts exist in categories with partial morphisms starting with the pioneering work of Löwe and Kennaway and ending with an essential characterisation in terms of an exactness property (for the interplay between pullbacks and pushouts) and an adjointness condition (w.r.t. inverse image functions) by Hayman and Heindel.
Triple graphs and graph diagrams are frameworks to synchronise two or more updatable data sources by means of internal mappings, which identify common sub-structures. *Comprehensive systems* generalise these frameworks, treating the network of data sources and their structural inter-relations as a homogeneous comprehensive artefact, in which partial maps identify commonalities. Although this inherent partiality produces amplified complexity, we can show that Heindel's characterisation still yields existence of pushouts in the category of comprehensive systems and reflective partial morphisms and thus enables computing by typed SPO graph transformation.

1. Introduction and motivation

We dedicate this paper to Michael Löwe, the founder of the single-pushout approach [1] and simultaneously a pioneer in the investigation of categories of partial algebras with partial morphisms between them, cf. e.g. [2].

In this paper, we combine these two theories. We introduce the category of *comprehensive systems*, formally a category in which the inner structure of the objects can be described with partial maps, and will show that SPO rewriting is applicable in this category.

Comprehensive Systems have been introduced in [3] as a means for global consistency management. A comprehensive system represents a collection of inter-related software artefacts. Furthermore, they generalise other related formalisms such as *triple graphs* [4] and *graph diagrams* [5,6].

To provide an intuition of a comprehensive system (Definition 5 in Sect. 3), take a look at Fig. 1. It depicts an abstract representation (i.e. model) of the databases of three information systems run by a fictitious insurance company: There is a Contract Management System (CoM) D_1 that stores insurance contracts in an object database, a Case Management System

☆ In memory of *Michael Löwe*, 1956–2019.
* Corresponding author.
E-mail address: past@hvl.no (P. Stünkel).

https://doi.org/10.1016/j.tcs.2021.07.002
0304-3975/© 2021 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

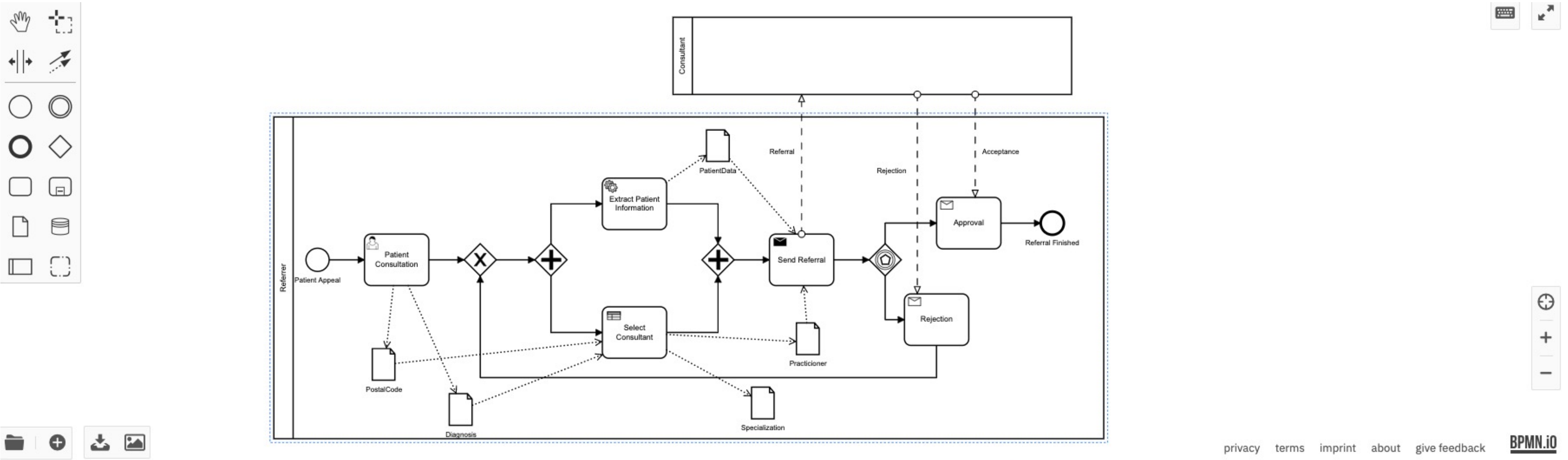
[20]

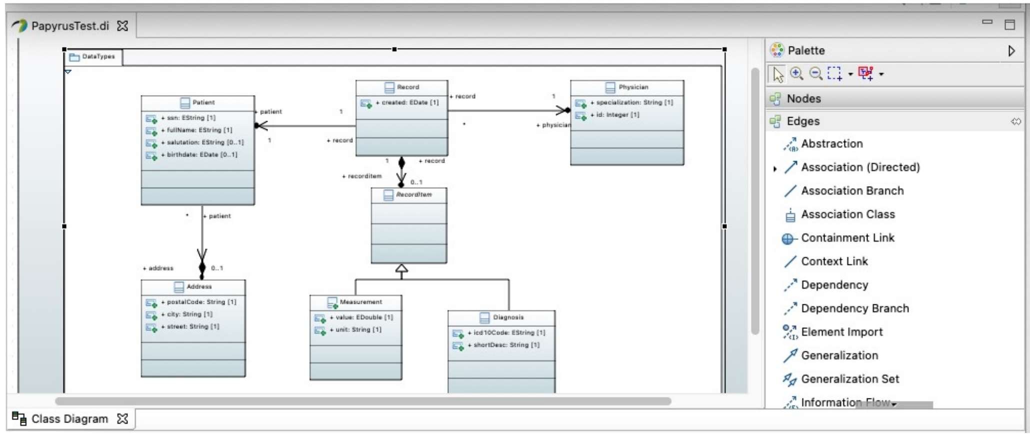
[18] P. Stünkel, H. König, Y. Lamo, and A. Rutle, 'Towards Multiple Model Synchronization with Comprehensive Systems', in Fundamental Approaches to Software Engineering, H. Wehrheim and J. Cabot, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 335–356. doi: 10.1007/978-3-030-45234-6_17.

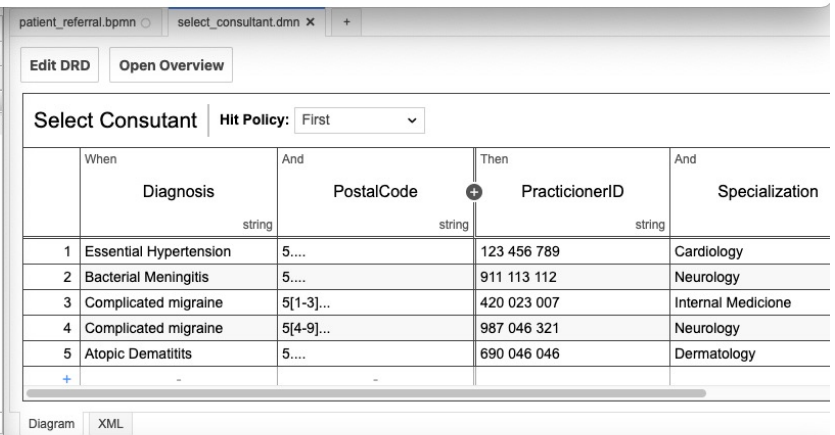
[19] P. Stünkel, H. König, Y. Lamo, and A. Rutle, 'Comprehensive Systems: A formal foundation for Multi-Model Consistency Management', Form. Asp. Comput., vol. 33, no. 6, pp. 1067–1114, Dec. 2021, doi: 10.1007/s00165-021-00555-2.

[20] P. Stünkel and H. König, 'Single pushout rewriting in comprehensive systems of graph-like structures', Theoretical Computer Science, vol. 884, pp. 23–43, Sep. 2021, doi: 10.1016/j.tcs.2021.07.002.

CS: Intuition [1/3]





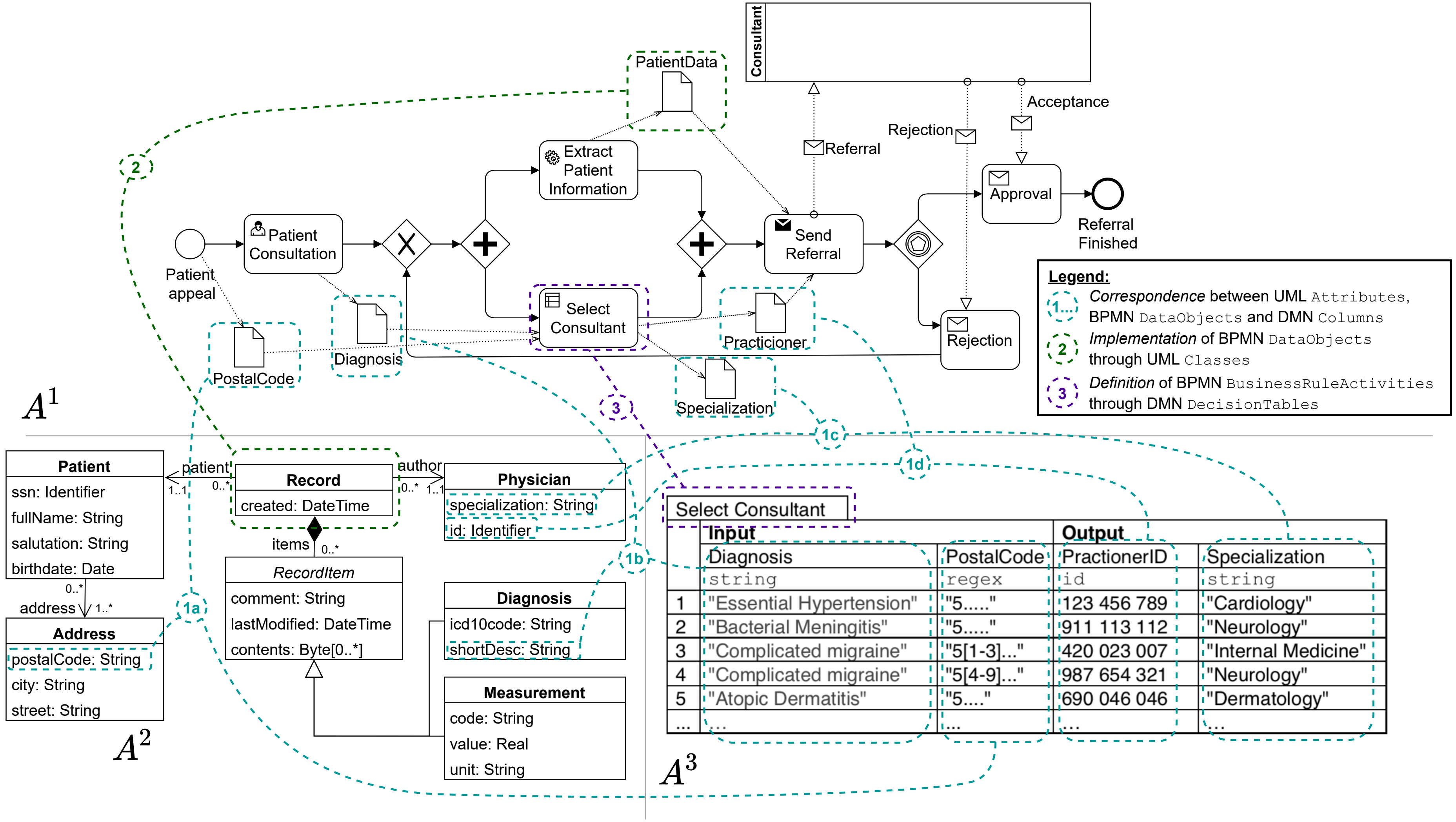


Select Consutant **Hit Policy:** First

	When Diagnosis string	And PostalCode string	Then PracticionerID string	And Specialization
1	Essential Hypertension	5....	123 456 789	Cardiology
2	Bacterial Meningitis	5....	911 113 112	Neurology
3	Complicated migraine	5[1-3]...	420 023 007	Internal Medicione
4	Complicated migraine	5[4-9]...	987 046 321	Neurology
5	Atopic Dematitits	5....	690 046 046	Dermatology
+	-	-		

Models do not live in isolation!

CS: Intuition [2/3]



Legend:

1... *Correspondence* between UML `Attributes`, BPMN `DataObjects` and DMN `Columns`

2 *Implementation* of BPMN `DataObjects` through UML `Classes`

3 *Definition* of BPMN `BusinessRuleActivities` through DMN `DecisionTables`

Select Consultant

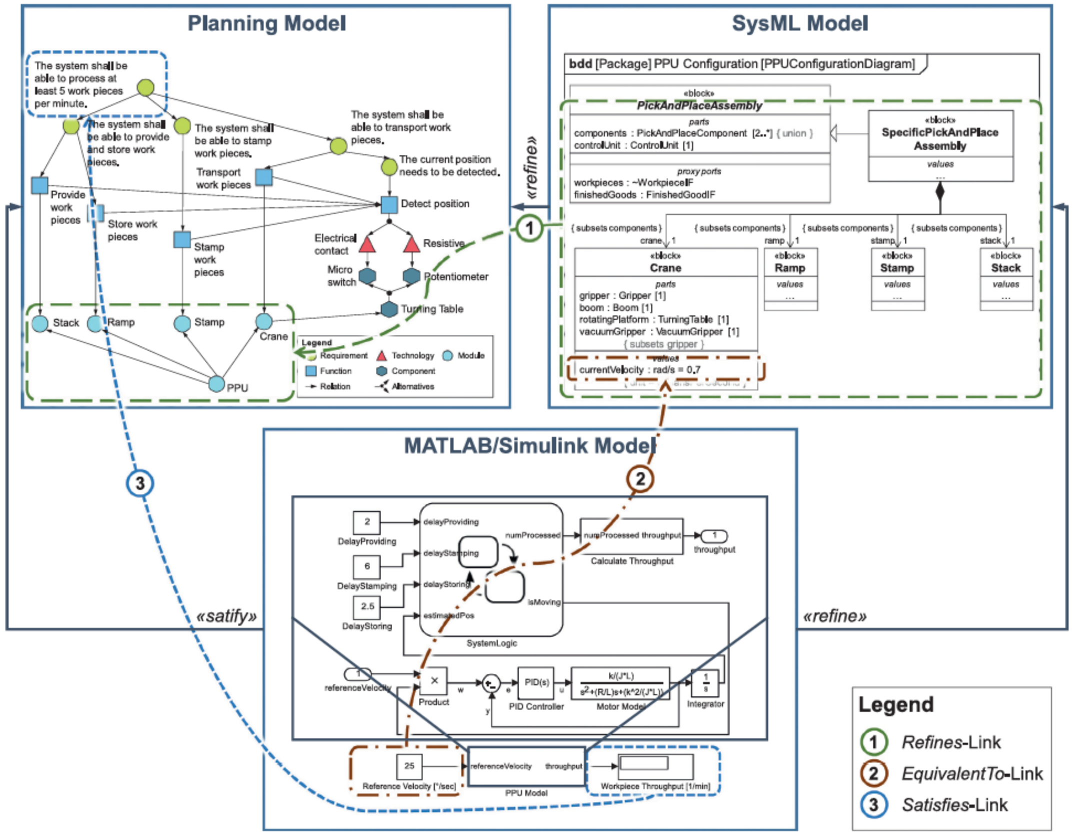
	Input		Output	
	Diagnosis	PostalCode	PractionerID	Specialization
	`string`	`regex`	`id`	`string`
1	"Essential Hypertension"	"5....."	123 456 789	"Cardiology"
2	"Bacterial Meningitis"	"5....."	911 113 112	"Neurology"
3	"Complicated migraine"	"5[1-3]..."	420 023 007	"Internal Medicine"
4	"Complicated migraine"	"5[4-9]..."	987 654 321	"Neurology"
5	"Atopic Dermatitis"	"5...."	690 046 046	"Dermatology"
...	...	...	...	...

Think: *"Drawing links on a white-board"*

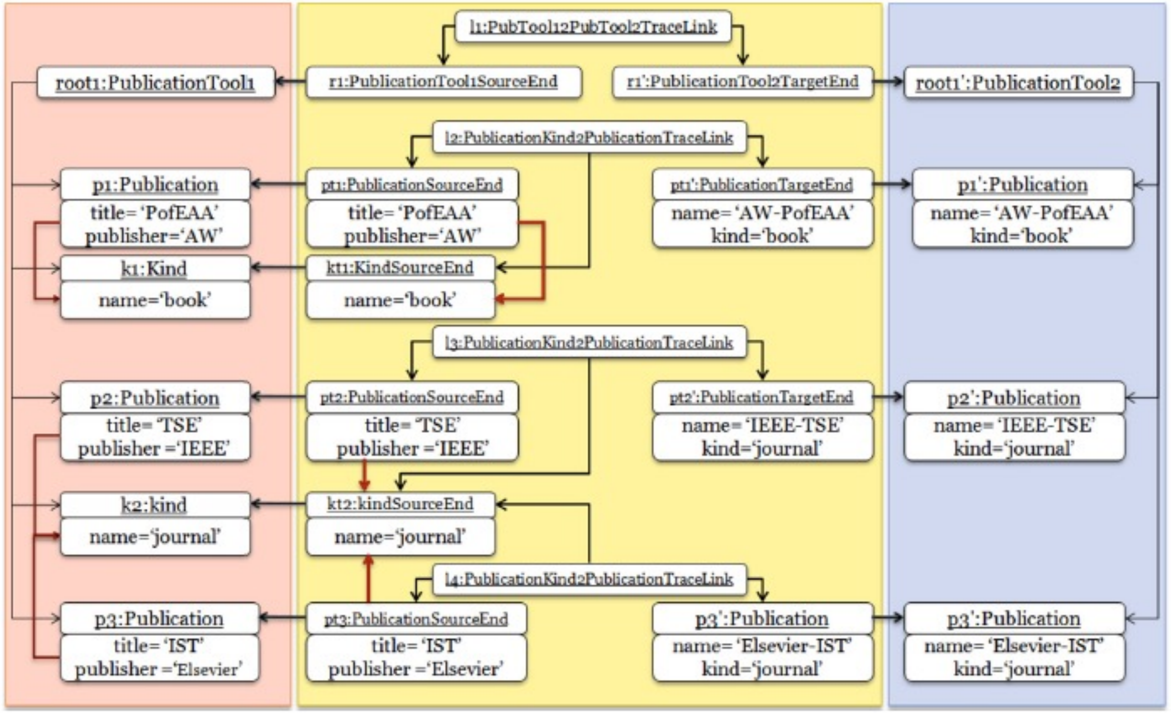
Digression: Model Weaving

[21] S. Feldmann, K. Kernschmidt, M. Wimmer, and B. Vogel-Heuser, ‘Managing inter-model inconsistencies in model-based systems engineering: Application in automated production systems engineering’, Journal of Systems and Software, vol. 153, pp. 105–134, Jul. 2019, doi: 10.1016/j.jss.2019.03.060.

[22] L. Samimi-Dehkordi, B. Zamani, and S. Kolahdouz-Rahimi, ‘EVL+Strace: a novel bidirectional model transformation approach’, Information and Software Technology, vol. 100, pp. 47–72, Aug. 2018, doi: 10.1016/j.infsof.2018.03.011.



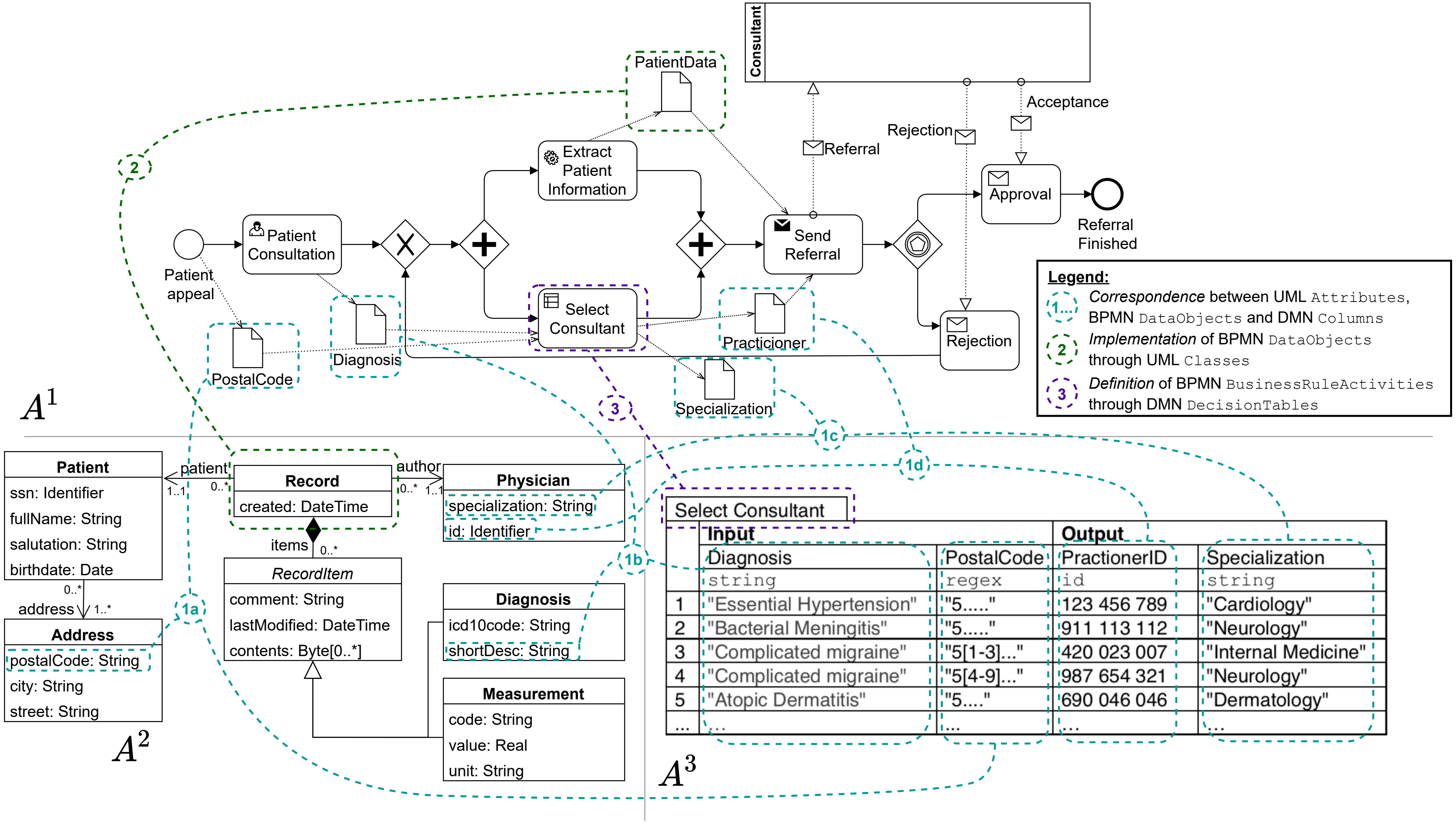
Feldmann et. al. [21]



Samimi-Dehkordi et. al. [22]

I am not alone with this approach 😉

CS: Intuition [2/3]



Select Consultant

	Input		Output	
	Diagnosis	PostalCode	PractionerID	Specialization
	string	regex	id	string
1	"Essential Hypertension"	"5....."	123 456 789	"Cardiology"
2	"Bacterial Meningitis"	"5....."	911 113 112	"Neurology"
3	"Complicated migraine"	"5[1-3]..."	420 023 007	"Internal Medicine"
4	"Complicated migraine"	"5[4-9]..."	987 654 321	"Neurology"
5	"Atopic Dermatitis"	"5...."	690 046 046	"Dermatology"
...	...	...	...	...

Legend:

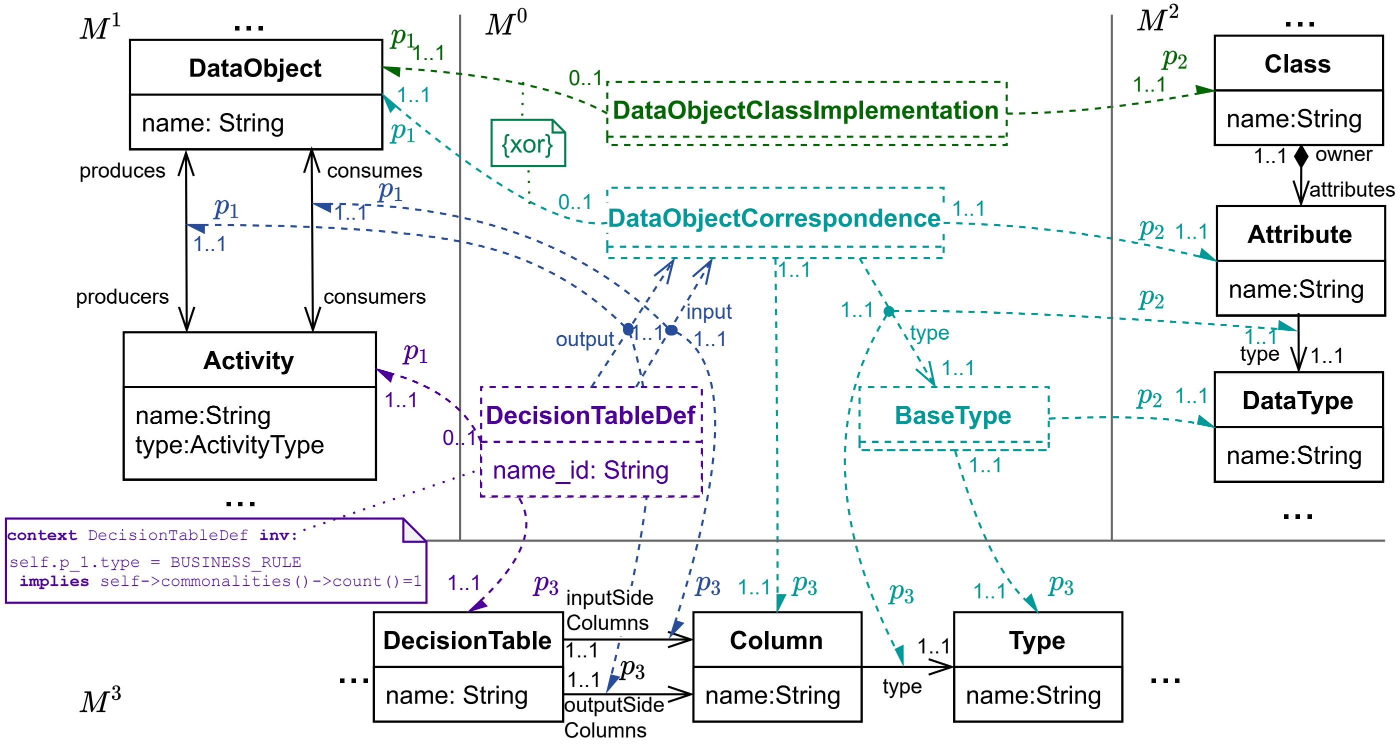
1... *Correspondence* between UML `Attributes`, BPMN `DataObjects` and DMN `Columns`

2 *Implementation* of BPMN `DataObjects` through UML `Classes`

3 *Definition* of BPMN `BusinessRuleActivities` through DMN `DecisionTables`

Think: *"Drawing links on a white-board"*

CS: Intuition [3/3]

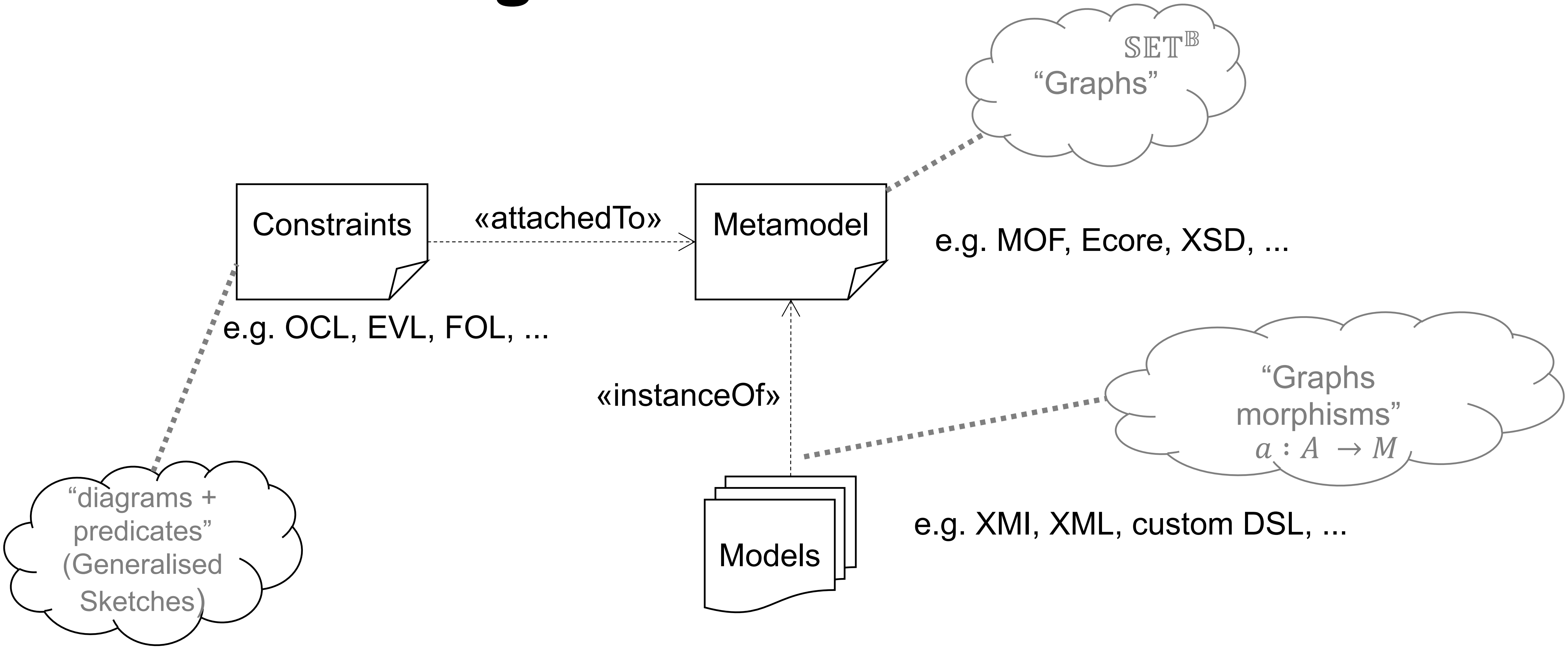


```
context DecisionTableDef inv:
self.p_1.type = BUSINESS_RULE
 implies self->commonalities()->count()=1
```

What happened here?

- 〉 We removed the concrete syntax (=> graphs)!
- 〉 We added (partial) mappings (=> graph morphisms)!

Local Modelling



$\mathbb{SET}^{\mathbb{B}}$

“Graphs”

Constraints

«attachedTo»

Metamodel

e.g. MOF, Ecore, XSD, ...

e.g. OCL, EVL, FOL, ...

«instanceOf»

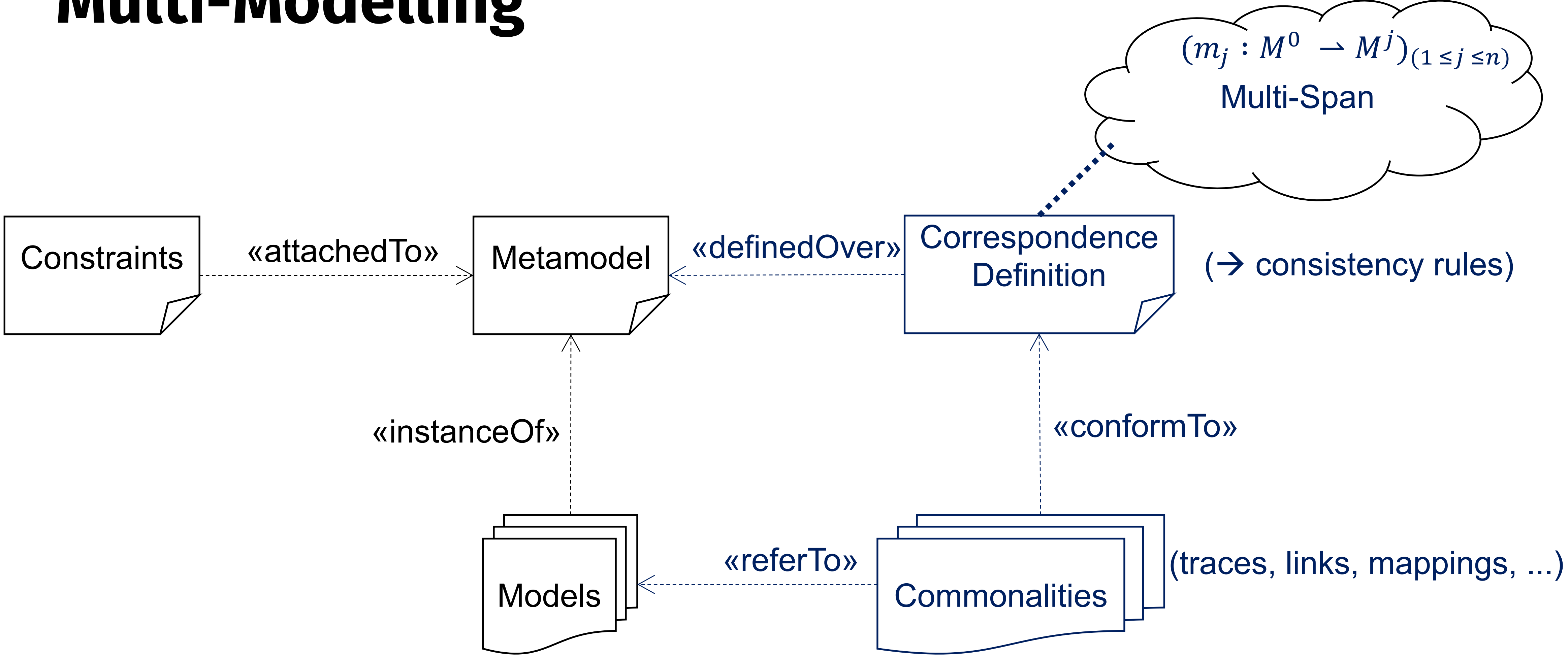
“Graphs morphisms” $a : A \to M$

“diagrams + predicates” (Generalised Sketches)

Models

e.g. XMI, XML, custom DSL, ...

Multi-Modelling



$(m_j : M^0 \rightharpoonup M^j)_{(1 \leq j \leq n)}$

Multi-Span

Constraints

«attachedTo»

Metamodel

«definedOver»

Correspondence Definition

(→ consistency rules)

«instanceOf»

«conformTo»

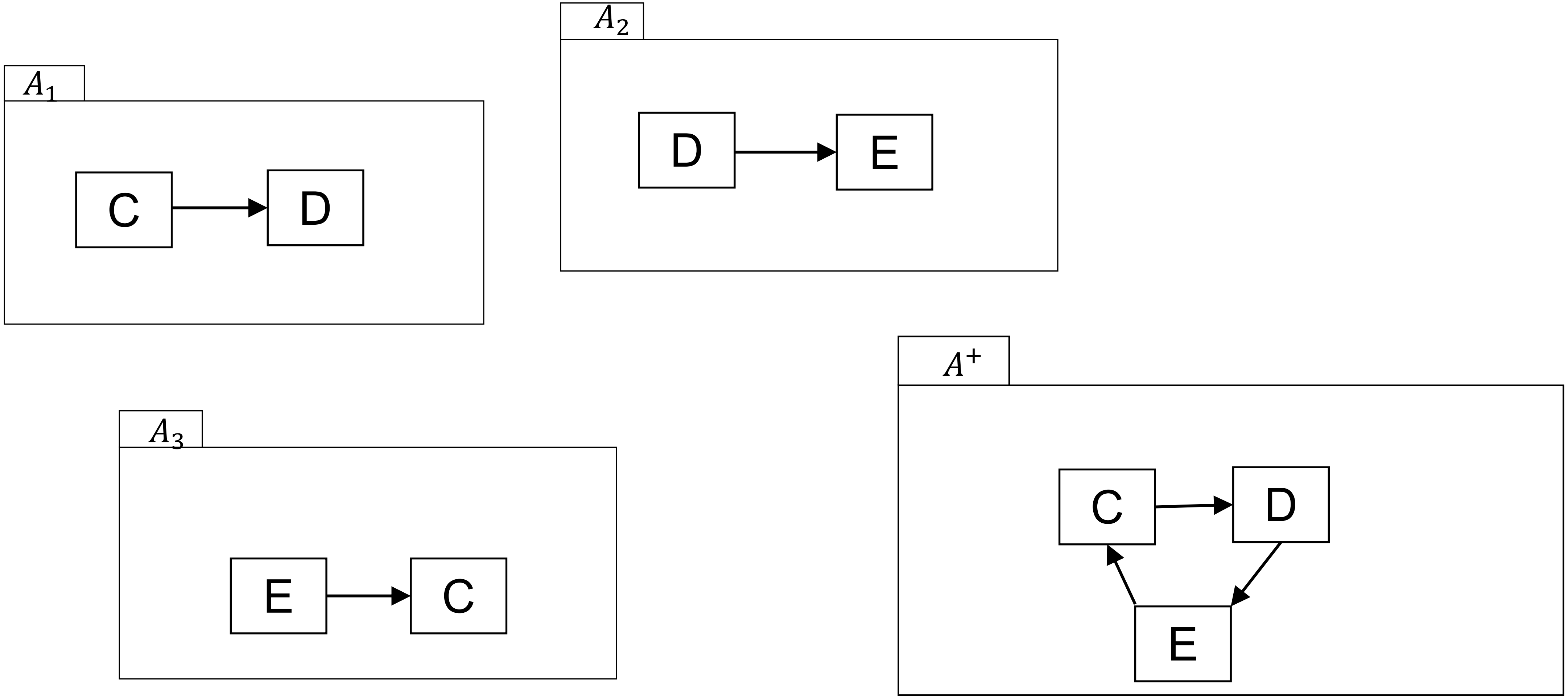
Models

«referTo»

Commonalities

(traces, links, mappings, ...)

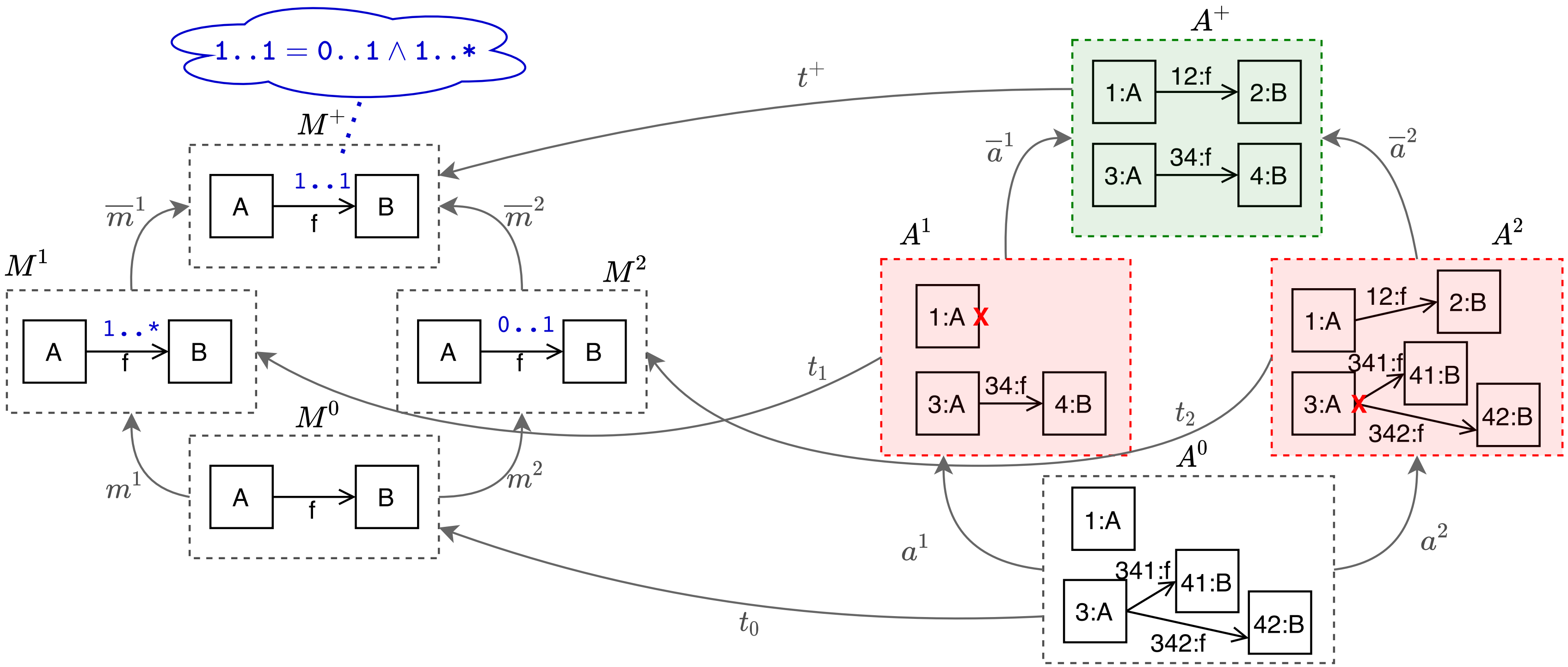
Model Merging / Colimit



Issue 1: Loosing the origin!
Issue 2: Only truly ”fitting” for identity-relations!

Merging Issue:

- Locally inconsistent models can be globally consistent



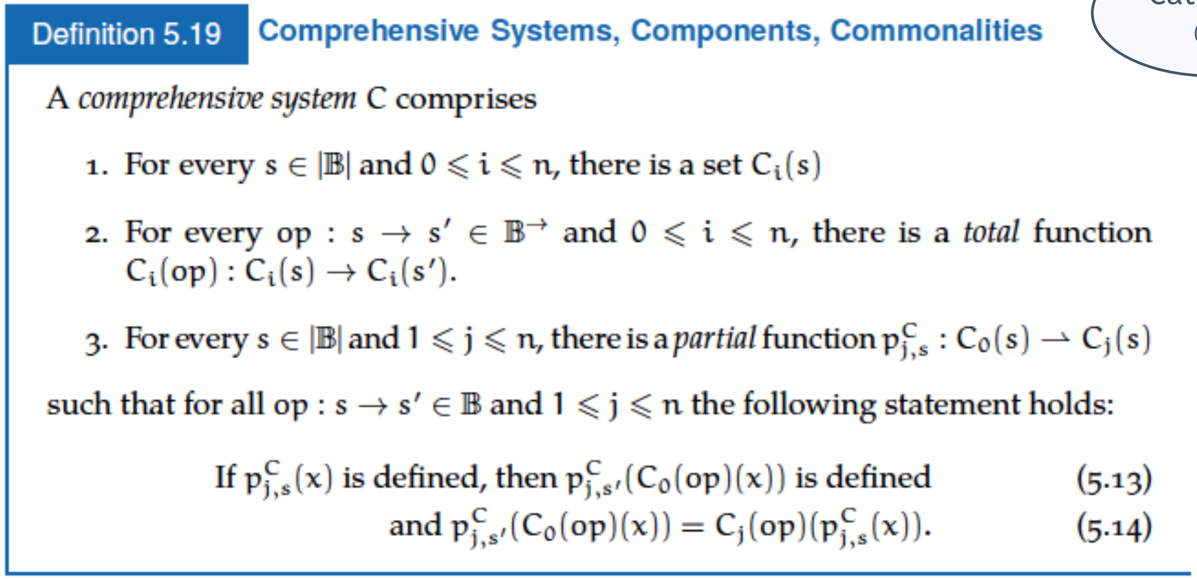
CS: Objective

- Handle the multi-ary case!
- Formalize Model Weaving
- Address shortcomings of Model Merging
- Achieve a foundation for multi-model management

- **Assumptions**
  - Software Models are (graph-like) structures
  - Model relationships are morphisms (mappings)
  - Local Modelling comes with means for consistency verification and restoration

CS: Definition

Let $\mathbb{B}$ *be an algebraic signature with unary operations only*:



Definition 5.19 **Comprehensive Systems, Components, Commonalities**

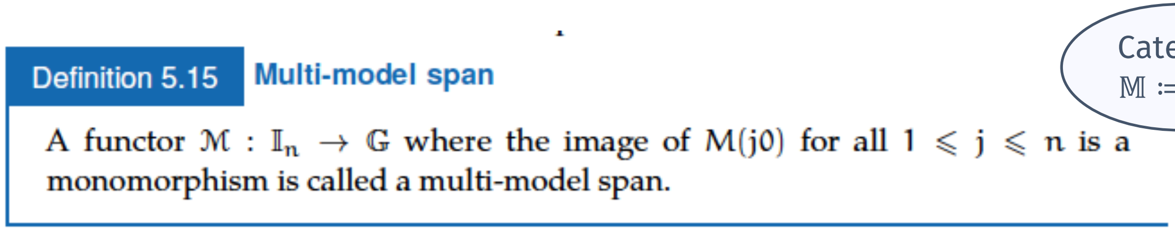
A *comprehensive system* C comprises

1. For every $s \in |\mathbb{B}|$ and $0 \leqslant i \leqslant n$, there is a set $C_i(s)$
2. For every $op : s \to s' \in \mathbb{B}^{\to}$ and $0 \leqslant i \leqslant n$, there is a *total* function $C_i(op) : C_i(s) \to C_i(s')$.
3. For every $s \in |\mathbb{B}|$ and $1 \leqslant j \leqslant n$, there is a *partial* function $p^C_{j,s} : C_0(s) \rightharpoonup C_j(s)$

such that for all $op : s \to s' \in \mathbb{B}$ and $1 \leqslant j \leqslant n$ the following statement holds:

$$\text{If } p^C_{j,s}(x) \text{ is defined, then } p^C_{j,s'}(C_0(op)(x)) \text{ is defined} \qquad (5.13)$$

$$\text{and } p^C_{j,s'}(C_0(op)(x)) = C_j(op)(p^C_{j,s}(x)). \qquad (5.14)$$



Definition 5.15 **Multi-model span**

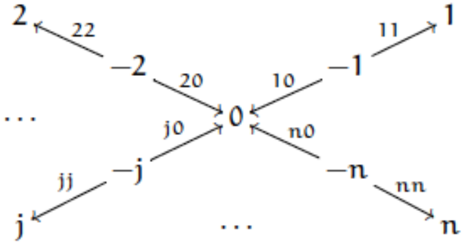
A functor $\mathcal{M} : \mathbb{I}_n \to \mathbb{G}$ where the image of $\mathcal{M}(j0)$ for all $1 \leqslant j \leqslant n$ is a monomorphism is called a multi-model span.

Category $\mathbb{CS}$

$\cong$ (Theorem 1)

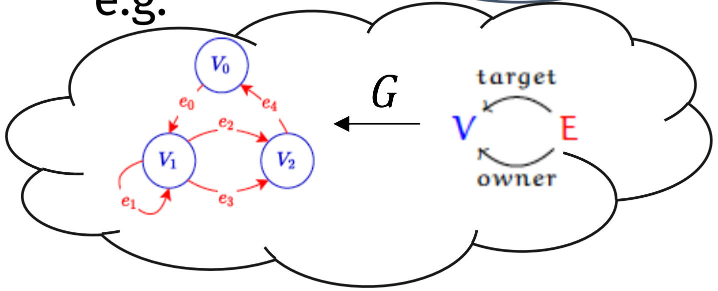
Category $\mathbb{M} := \mathbb{G}^{\mathbb{I}_n}$

Category $\mathbb{I}_n$



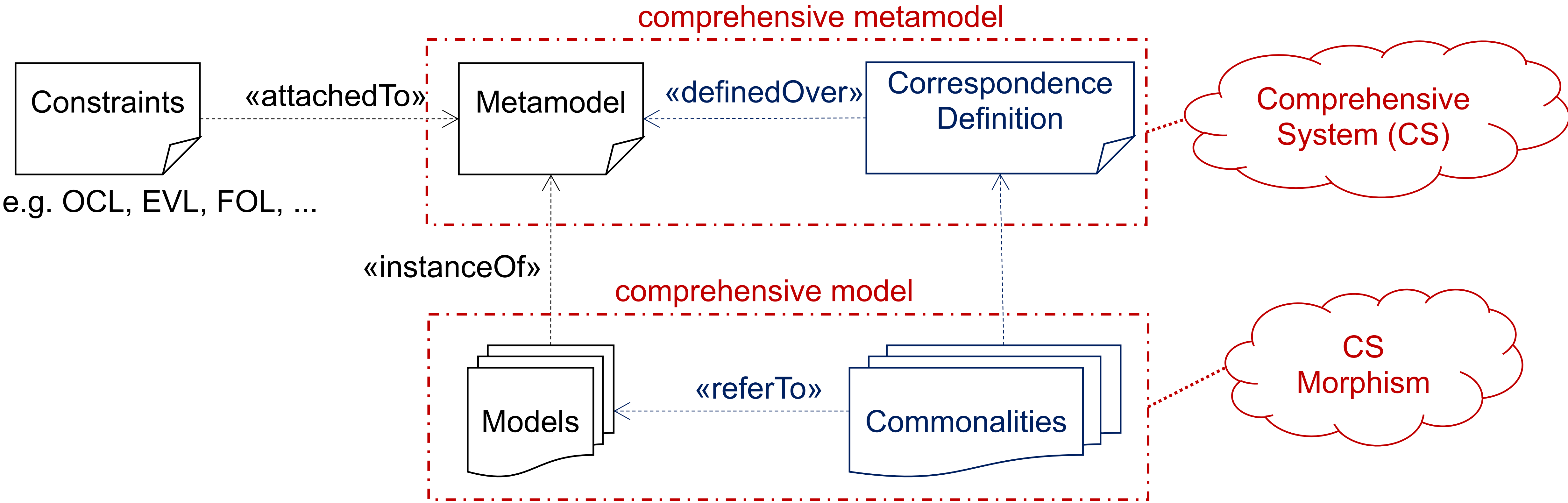
$\mathbb{G} := \boldsymbol{Set}^{\mathbb{B}}$

e.g.



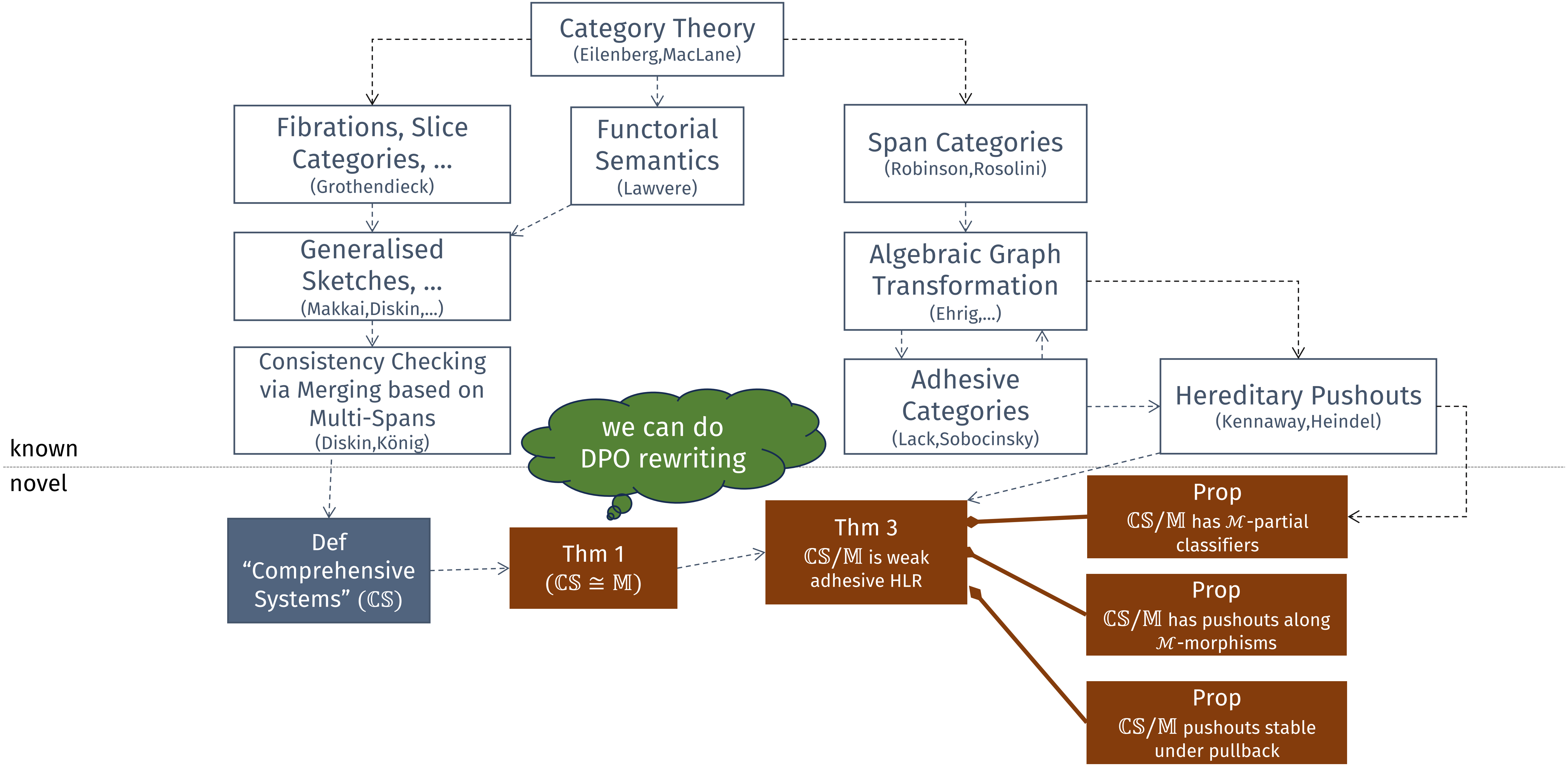
$$\boxed{Set^{(\mathbb{B}\times\mathbb{I}_n)} \cong Set^{\mathbb{B}^{\mathbb{I}_n}}}$$

Theorem 1 implication



- A comprehensive system is single a global artefact with a "graph-like" structure that internalises the inter-relations (Theorem 1)

Results



Category Theory
(Eilenberg,MacLane)

Fibrations, Slice Categories, ...
(Grothendieck)

Functorial Semantics
(Lawvere)

Span Categories
(Robinson,Rosolini)

Generalised Sketches, ...
(Makkai,Diskin,...)

Algebraic Graph Transformation
(Ehrig,...)

Consistency Checking via Merging based on Multi-Spans
(Diskin,König)

Adhesive Categories
(Lack,Sobocinsky)

Hereditary Pushouts
(Kennaway,Heindel)

known

novel

we can do DPO rewriting

Def
“Comprehensive Systems” ($\mathbb{CS}$)

Thm 1
($\mathbb{CS} \cong \mathbb{M}$)

Thm 3
$\mathbb{CS}/\mathbb{M}$ is weak adhesive HLR

Prop
$\mathbb{CS}/\mathbb{M}$ has $\mathcal{M}$-partial classifiers

Prop
$\mathbb{CS}/\mathbb{M}$ has pushouts along $\mathcal{M}$-morphisms

Prop
$\mathbb{CS}/\mathbb{M}$ pushouts stable under pullback

Proof technique

› Diagram chasing

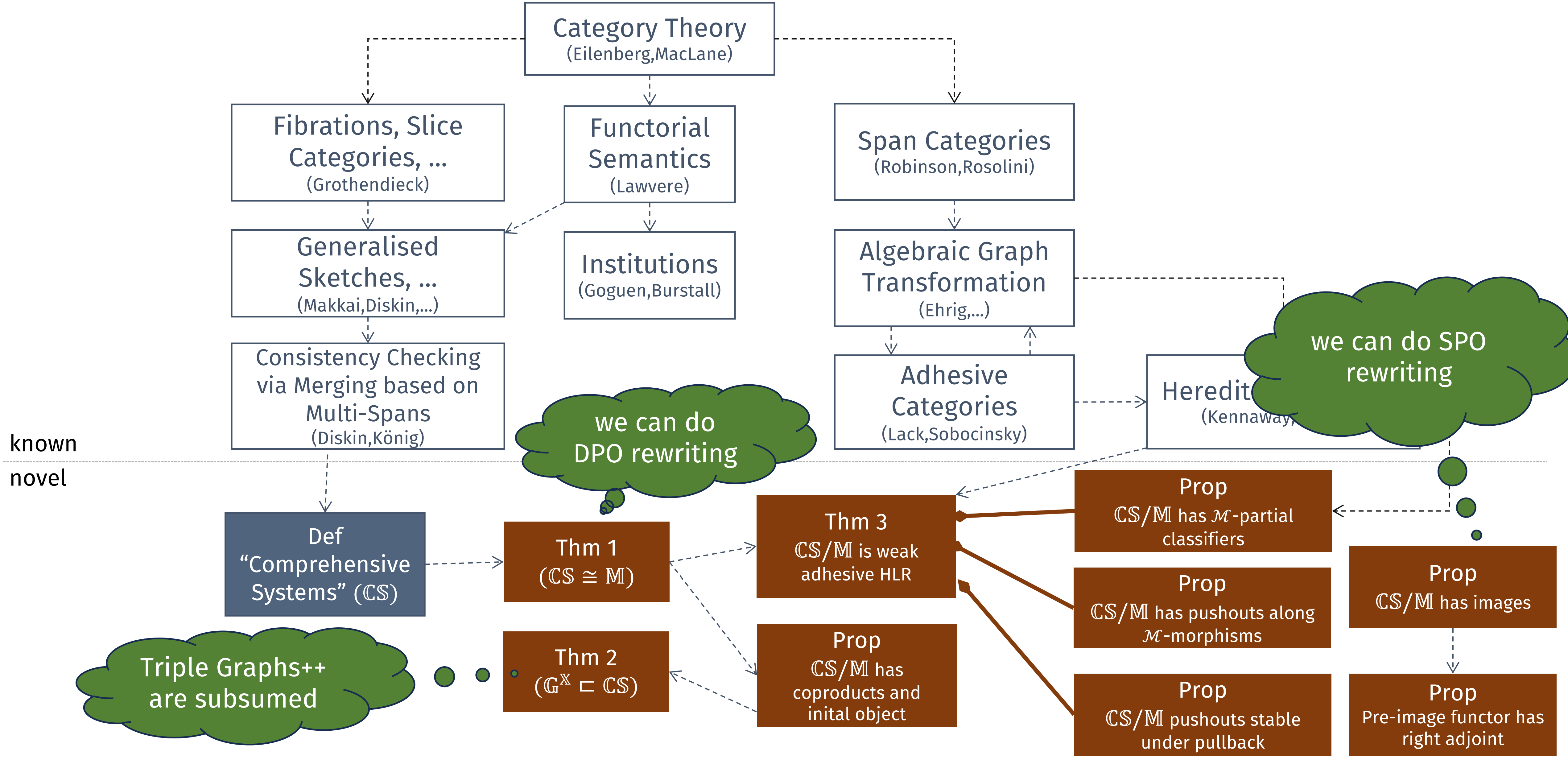
$$\begin{tikzcd}
 & & A^0 \arrow[rr, "n_0"] \arrow[dl, "g_0"'] & & C^0 \arrow[dl, "f_0"] \\
0 & B^0 \arrow[rr, "m_0"] & & D^0 & \\
 & & \mathrm{dom}(p_j^A) \arrow[uu, hook, dashed, "\subseteq_j^A"] \arrow[rr, "n_{-j}"] \arrow[dl, "g_{-j}"'] \arrow[dd, dashed, "\underline{p}_j^A"] & & \mathrm{dom}(p_j^C) \arrow[uu, hook, "\subseteq_j^C"'] \arrow[dl, "f_{-j}"] \arrow[dd, "\underline{p}_j^C"] \\
-j \arrow[uu, "j0"] \arrow[dd, "jj"'] & \mathrm{dom}(p_j^B) \arrow[uu, hook, "\subseteq_j^B"] \arrow[rr, "m_{-j}"] \arrow[dd, "\underline{p}_j^B"] & & \mathrm{dom}(p_j^D) \arrow[uu, hook, "\subseteq_j^D"] \arrow[dd, "\underline{p}_j^D"] & \\
 & & A^j \arrow[rr, "n_j"] \arrow[dl, "g_j"'] & & C^j \arrow[dl, "f_j"] \\
j & B^j \arrow[rr, "m_j"] & & D^j &
\end{tikzcd}$$

$$\begin{tikzcd}
 & M^0 + N^0 & \\
M^0 \arrow[ur, "\lambda_0"] & \mathrm{dom}(p_j^M) + \mathrm{dom}(p_j^N) \arrow[u, dotted, "u_j"] \arrow[d, dotted, "\nu_j"] & N^0 \arrow[ul, "\kappa_0"'] \\
\mathrm{dom}(p_j^M) \arrow[u, hook, "\subseteq_j^M"] \arrow[ur, "\lambda_{-j}"] \arrow[d, "\underline{p}_j^M"'] & M^j + N^j & \mathrm{dom}(p_j^N) \arrow[u, hook, "\subseteq_j^N"'] \arrow[ul, "\kappa_{-j}"'] \arrow[d, "\underline{p}_j^M"] \\
M^j \arrow[ur, "\lambda_j"] & & N^j \arrow[ul, "\kappa_j"']
\end{tikzcd}$$

$$\begin{tikzcd}
 & X^0 \arrow[rr, tail, "m_0"] \arrow[dl, "f_0"'] & & A^0 \arrow[dl, dashed, "\overline{[m_0, f_0\rangle}"] \\
B^0 \arrow[rr, tail, "\eta_{B_0}"'] & & \mathcal{L}B^0 & \\
 & \mathrm{dom}(p_j^X) \arrow[uu, hook, "\subseteq_j^X"'] \arrow[rr, tail, "m_{-j}"] \arrow[dl, "f_{-j}"'] & & \mathrm{dom}(p_j^A) \arrow[uu, hook, "\subseteq_j^A"'] \arrow[dl, dashed, "\overline{[m_{-j}, f_{-j}\rangle}"] \\
\mathrm{dom}(p_j^B) \arrow[uu, hook, "\subseteq_j^B"] \arrow[rr, tail, "\eta_{\mathrm{dom}(p_j^B)}"'] & & \mathcal{L}\mathrm{dom}(p_j^B) \arrow[uu, dotted, "\overline{[\eta_{\mathrm{dom}(p_j^B)}, \subseteq_j^B\rangle}"'] &
\end{tikzcd}$$

› More details in the papers 😜

Theoretical Results



Category Theory
(Eilenberg,MacLane)

Fibrations, Slice Categories, ...
(Grothendieck)

Functorial Semantics
(Lawvere)

Span Categories
(Robinson,Rosolini)

Generalised Sketches, ...
(Makkai,Diskin,...)

Institutions
(Goguen,Burstall)

Algebraic Graph Transformation
(Ehrig,...)

Consistency Checking via Merging based on Multi-Spans
(Diskin,König)

Adhesive Categories
(Lack,Sobocinsky)

Heredit
(Kennaway,

we can do SPO rewriting

we can do DPO rewriting

known

novel

Def
“Comprehensive Systems” ($\mathbb{CS}$)

Thm 1
($\mathbb{CS} \cong \mathbb{M}$)

Thm 3
$\mathbb{CS}/\mathbb{M}$ is weak adhesive HLR

Prop
$\mathbb{CS}/\mathbb{M}$ has $\mathcal{M}$-partial classifiers

Prop
$\mathbb{CS}/\mathbb{M}$ has pushouts along $\mathcal{M}$-morphisms

Prop
$\mathbb{CS}/\mathbb{M}$ has images

Triple Graphs++ are subsumed

Thm 2
($\mathbb{G}^{\mathbb{X}} \sqsubset \mathbb{CS}$)

Prop
$\mathbb{CS}/\mathbb{M}$ has coproducts and inital object

Prop
$\mathbb{CS}/\mathbb{M}$ pushouts stable under pullback

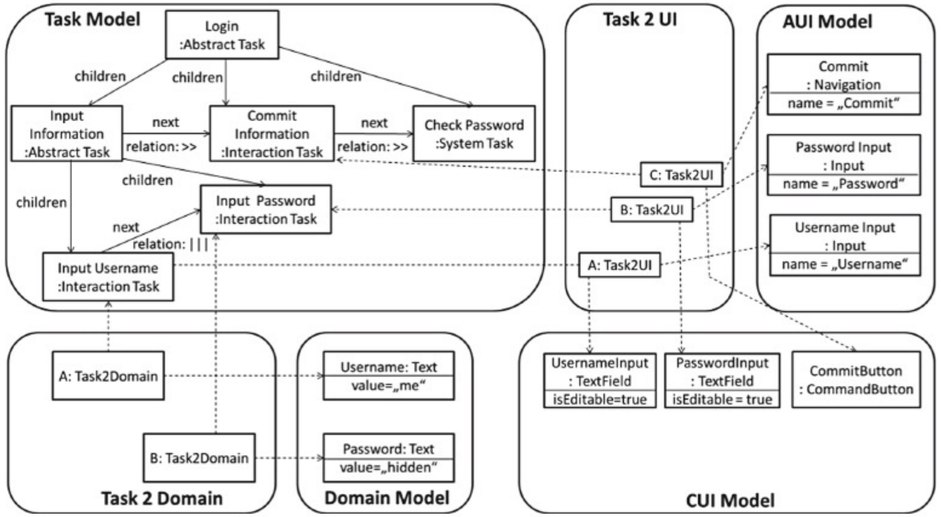
Prop
Pre-image functor has right adjoint

Relationship with TG(G)s

- Triple graphs are a special case of CS with $n = 2$ and all projections being *total*

- There is a multi-ary generalization of triple graphs called *graph diagrams* [23]
  - also subsumed by CS

[23] F. Trollmann and S. Albayrak, ‘Extending Model Synchronization Results from Triple Graph Grammars to Multiple Models’, in Theory and Practice of Model Transformations, P. Van Gorp and G. Engels, Eds., in Lecture Notes in Computer Science. Springer International Publishing, 2016, pp. 91–106.



- Since CS are weak adhesive HLR, one may define *Comprehensive System Grammars (CSG)*

$\mathcal{M}$ = Reflective Monomorphisms

$\mathbb{CS}$

$$p^{C}_{j,s}(x) \text{ is defined } \Leftrightarrow p^{D}_{j,s}(f_{0,s}(x)) \text{ is defined and } p^{D}_{j,s}(f_{0,s}(x)) = f_{j,s}(p^{C}_{j,s}(x))$$

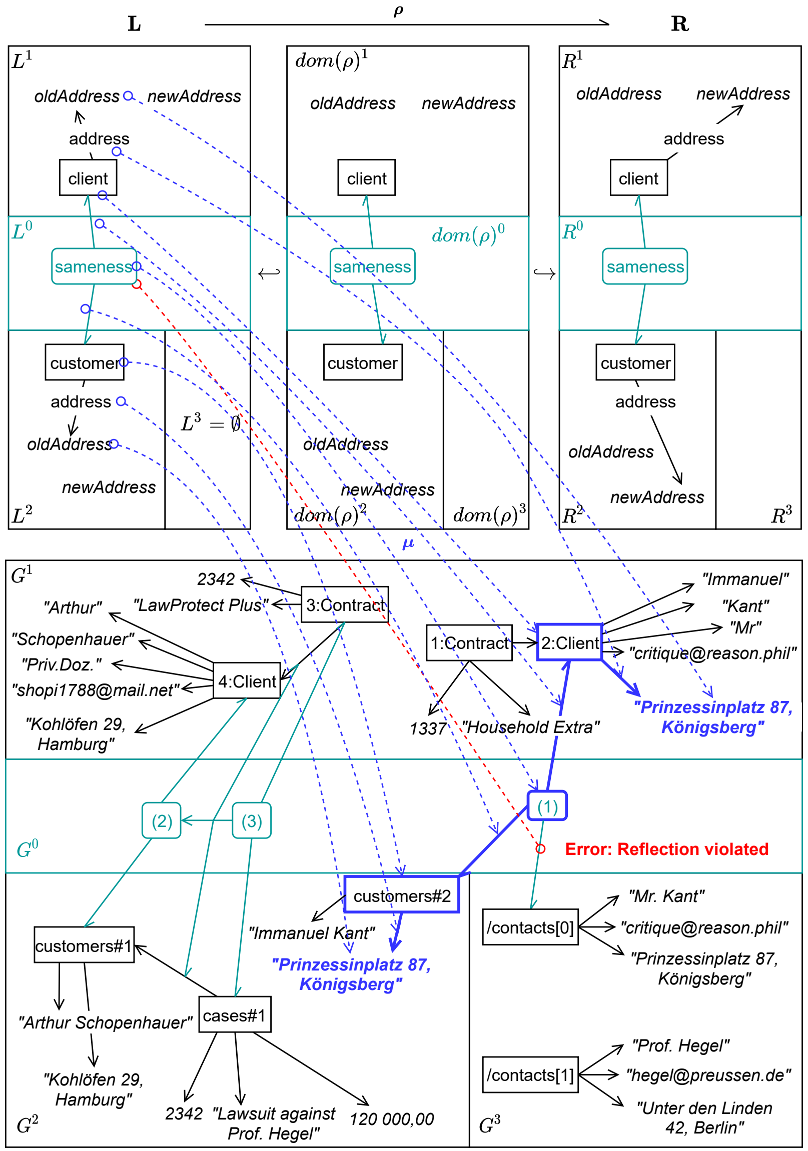
$\mathbb{M}$

$$p^{N}_{j} \circ [id_{M^0}, m_0\rangle = [id_{M^j}, m_j\rangle \circ p^{M}_{j}$$

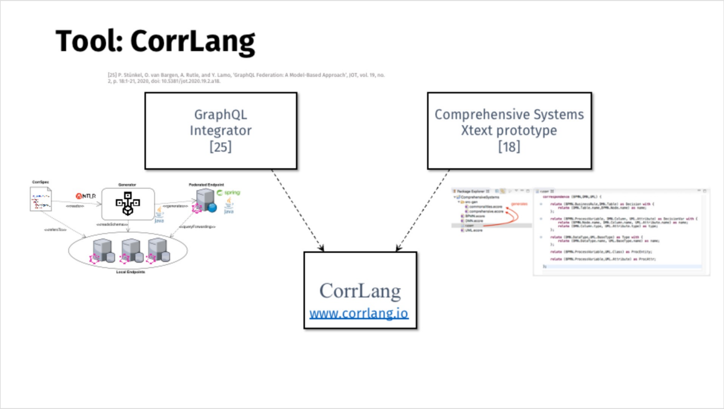
$$\begin{tikzcd}
M^0 \arrow[r, equal, "id_{M^0}"] & M^0 \arrow[r, tail, "m_0"] & N^0 \\
dom(p^M_j) \arrow[u, hook, "\subseteq^M_j"] \arrow[d, "p^M_j"'] \arrow[r, equal, "id_{dom(p^M_j)}"] & dom(p^M_j) \arrow[u, hook, "\subseteq^M_j"] \arrow[d, "p^M_j"'] \arrow[r, dashed, "m_{-j}"] & dom(p^N_j) \arrow[u, hook, "\subseteq^N_j"'] \arrow[d, "p^N_j"'] \\
M^j \arrow[r, equal, "id_{M^j}"'] & M^j \arrow[r, tail, "m_j"'] & N^j
\end{tikzcd}$$

=> related to [24]

[24] J. Kosiol, L. Fritsche, A. Schürr, and G. Taentzer, 'Adhesive Subcategories of Functor Categories with Instantiation to Partial Triple Graphs', in Graph Transformation, E. Guerra and F. Orejas, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 38–54. doi: 10.1007/978-3-030-23611-3_3.



Application



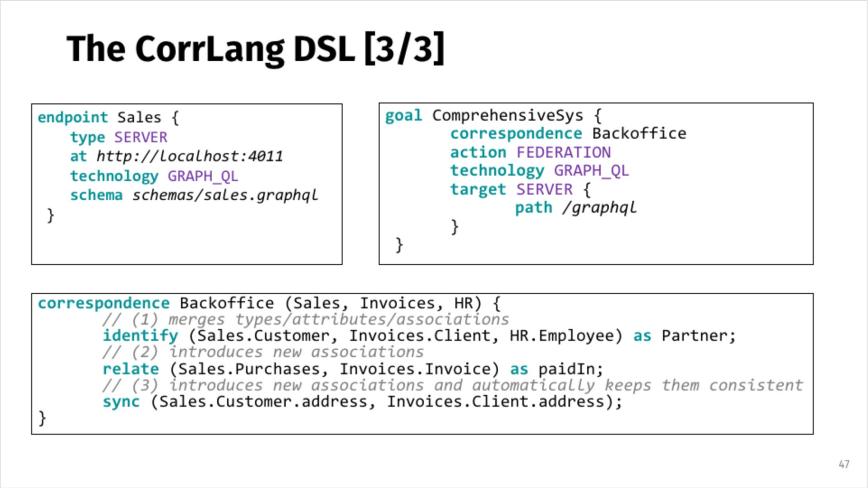
Tool: CorrLang

[25] P. Stünkel, O. van Bargen, A. Rutle, and Y. Lamo, 'GraphQL Federation: A Model-Based Approach', JOT, vol. 19, no. 2, p. 18:1-21, 2020, doi: 10.5381/jot.2020.19.2.a18.

GraphQL Integrator [25]

Comprehensive Systems Xtext prototype [18]

CorrLang
www.corrlang.io

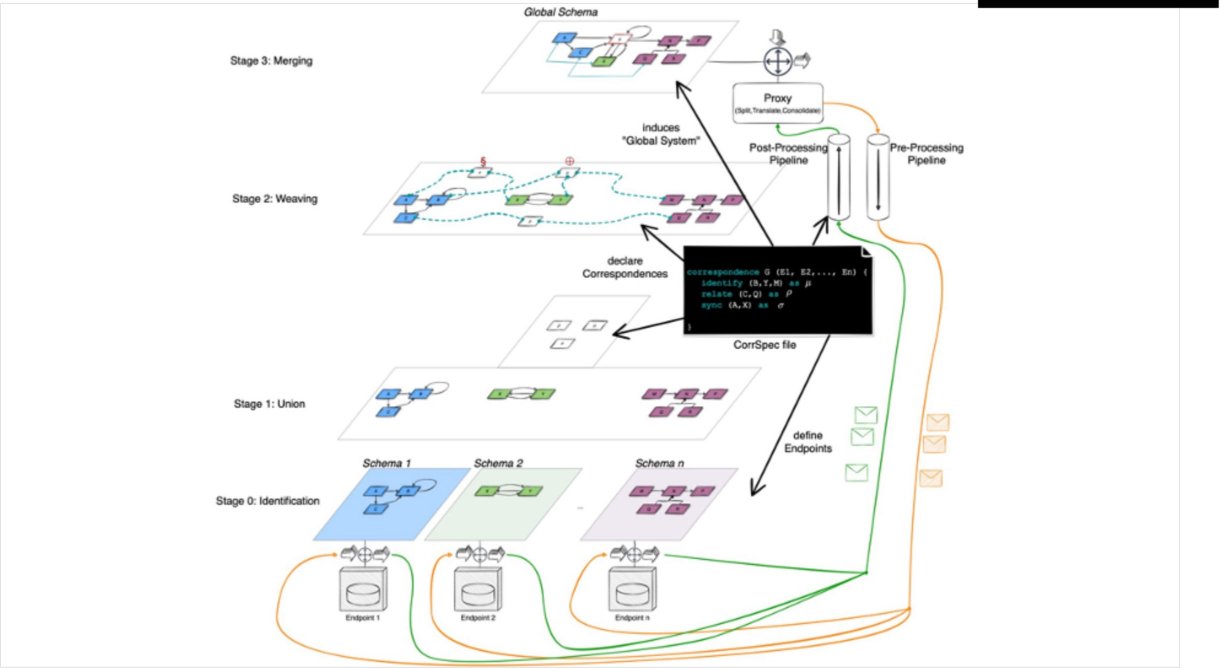


The CorrLang DSL [3/3]

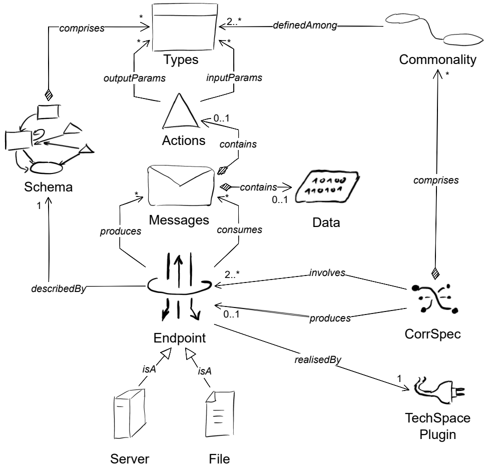
```
endpoint Sales {
    type SERVER
    at http://localhost:4011
    technology GRAPH_QL
    schema schemas/sales.graphql
 }
```

```
goal ComprehensiveSys {
        correspondence Backoffice
        action FEDERATION
        technology GRAPH_QL
        target SERVER {
            path /graphql
        }
 }
```

```
correspondence Backoffice (Sales, Invoices, HR) {
        // (1) merges types/attributes/associations
        identify (Sales.Customer, Invoices.Client, HR.Employee) as Partner;
        // (2) introduces new associations
        relate (Sales.Purchases, Invoices.Invoice) as paidIn;
        // (3) introduces new associations and automatically keeps them consistent
        sync (Sales.Customer.address, Invoices.Client.address);
}
```



```
correspondence G (E1, E2,..., En) {
   identify (B,Y,M) as μ
   relate (C,Q) as ρ
   sync (A,X) as σ
}
```

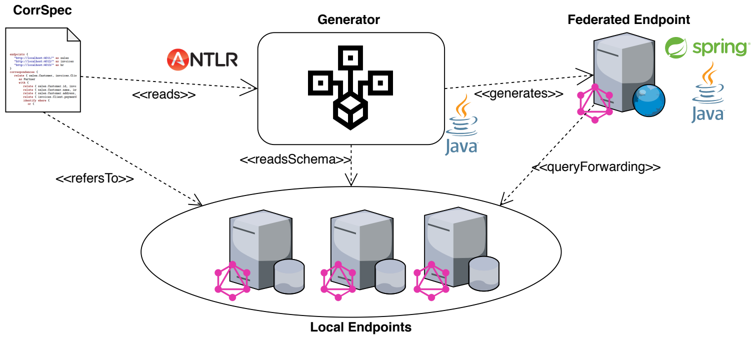


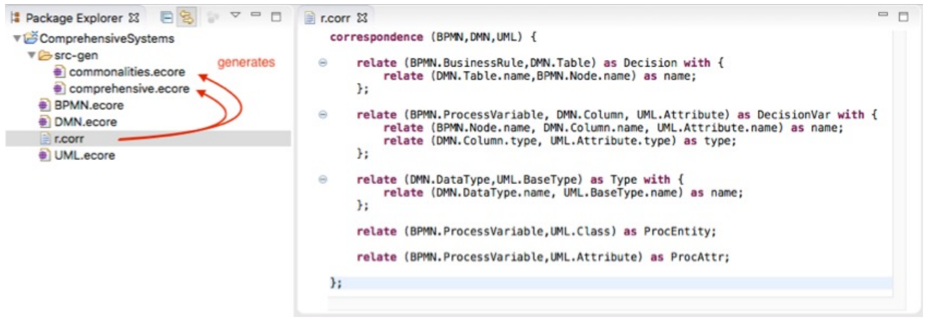
Tool: CorrLang

[25] P. Stünkel, O. van Bargen, A. Rutle, and Y. Lamo, ‘GraphQL Federation: A Model-Based Approach’, JOT, vol. 19, no. 2, p. 18:1-21, 2020, doi: 10.5381/jot.2020.19.2.a18.

GraphQL Integrator [25]

Comprehensive Systems Xtext prototype [18]





```
correspondence (BPMN,DMN,UML) {

    relate (BPMN.BusinessRule,DMN.Table) as Decision with {
        relate (DMN.Table.name,BPMN.Node.name) as name;
    };

    relate (BPMN.ProcessVariable, DMN.Column, UML.Attribute) as DecisionVar with {
        relate (BPMN.Node.name, DMN.Column.name, UML.Attribute.name) as name;
        relate (DMN.Column.type, UML.Attribute.type) as type;
    };

    relate (DMN.DataType,UML.BaseType) as Type with {
        relate (DMN.DataType.name, UML.BaseType.name) as name;
    };

    relate (BPMN.ProcessVariable,UML.Class) as ProcEntity;

    relate (BPMN.ProcessVariable,UML.Attribute) as ProcAttr;

};
```

CorrLang

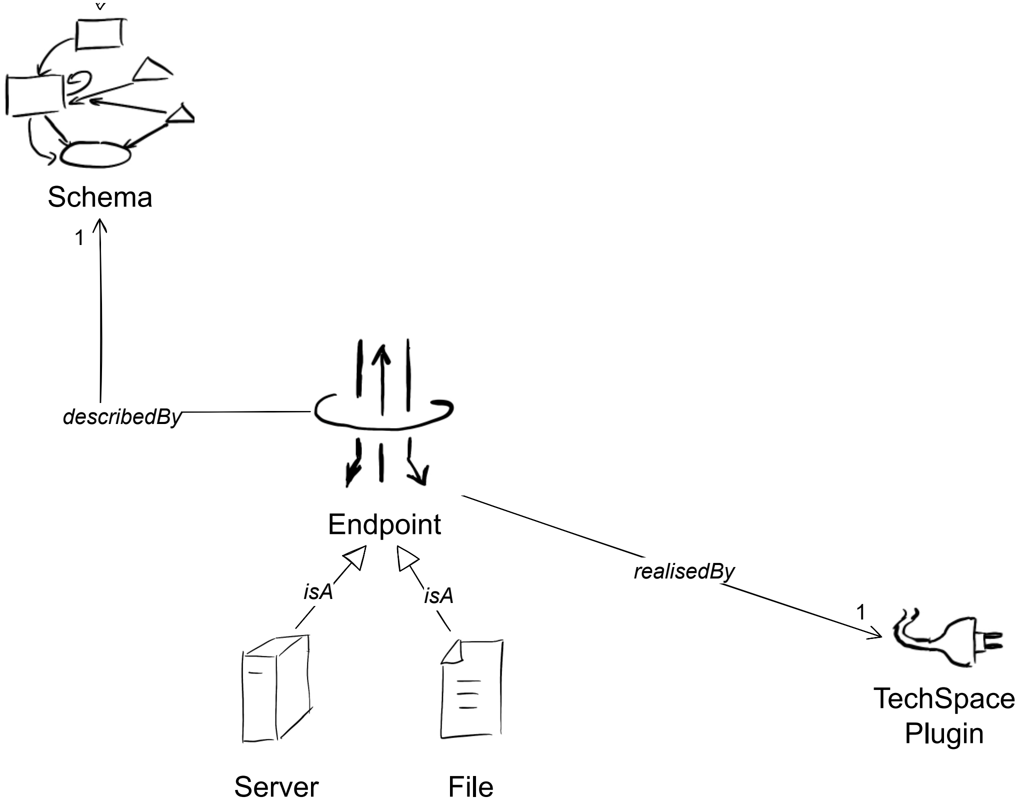
www.corrlang.io

CorrLang: Summary

- A *DSL* to describe inter-model-relationships...
- ...generates mediation via *Message translation*
- Conceptual Abstraction: Servers and Files are ”*Endpoints*”
- Formal Abstraction: Schemas, Messages and Data are ”*Graphs*”

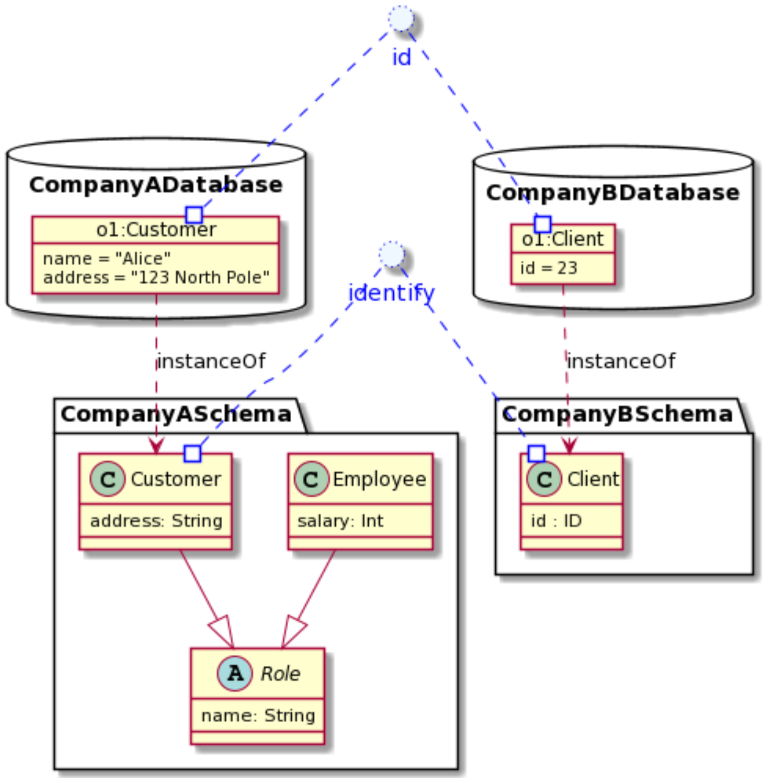
CorrLang: Endpoints

```
endpoint Sales {
    type SERVER
    at http://localhost:4011
    technology GRAPH_QL
    schema schemas/sales.graphql
 }
```

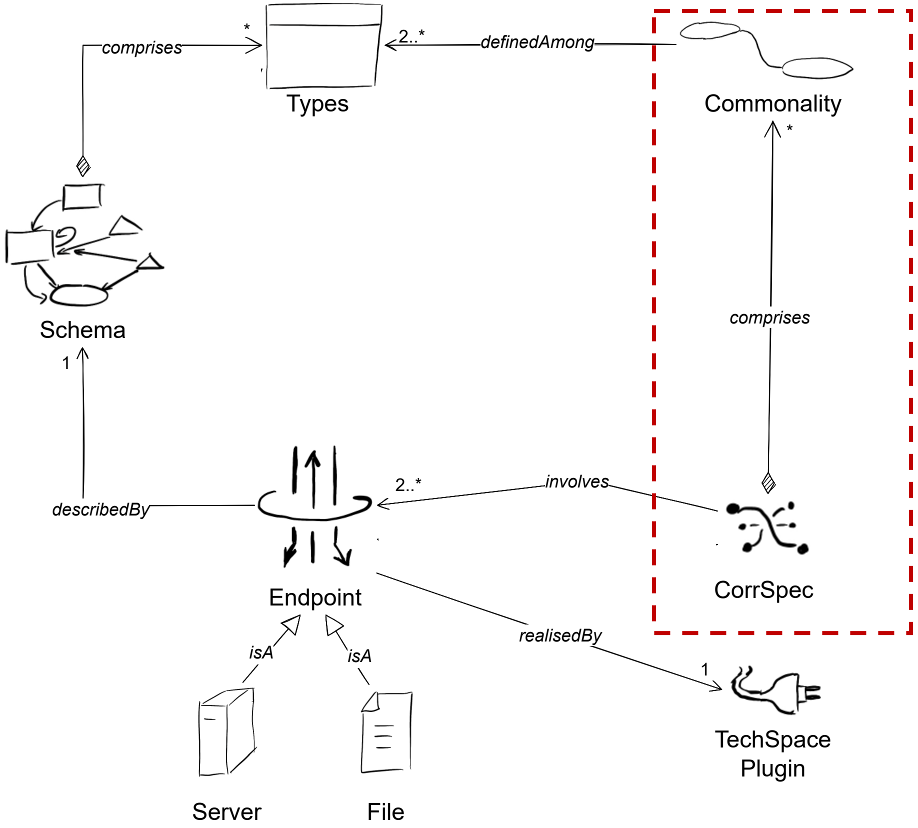


Formal Representation

- Schemas are graphs
  - Nodes = Entities, Data Types, Classes...
  - Edges = Associations, Attributes, ...
- Data are graphs
  - Nodes = Objects, Values, ...
  - Edges = Links, Attribute valuations, ...
- Traceability-links are multi-spans



CorrLang: Correspondences



i.e. the apex of the
span + projections

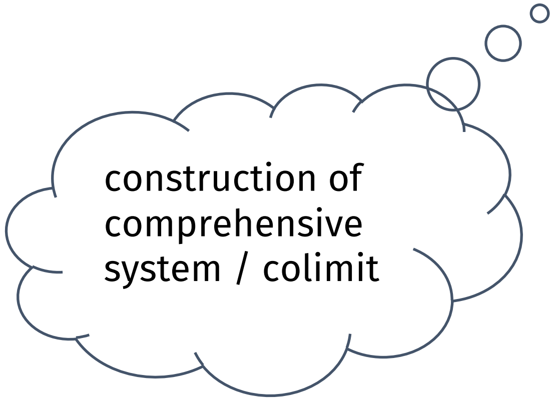
```
correspondence Backoffice (Sales, Invoices, HR) {
       // (1) merges types/attributes/associations
       identify (Sales.Customer, Invoices.Client, HR.Employee) as Partner;
       // (2) introduces new associations
       relate (Sales.Purchases, Invoices.Invoice) as paidIn;
       // (3) introduces new associations and automatically keeps them consistent
       sync (Sales.Customer.address, Invoices.Client.address);
}
```

The CorrLang DSL [2/3]

```
endpoint Sales {
    type SERVER
    at http://localhost:4011
    technology GRAPH_QL
    schema schemas/sales.graphql
 }
```

```
correspondence Backoffice (Sales, Invoices, HR) {
       // (1) merges types/attributes/associations
       identify (Sales.Customer, Invoices.Client, HR.Employee) as Partner;
       // (2) introduces new associations
       relate (Sales.Purchases, Invoices.Invoice) as paidIn;
       // (3) introduces new associations and automatically keeps them consistent
       sync (Sales.Customer.address, Invoices.Client.address);
}
```

CorrLang: Goals

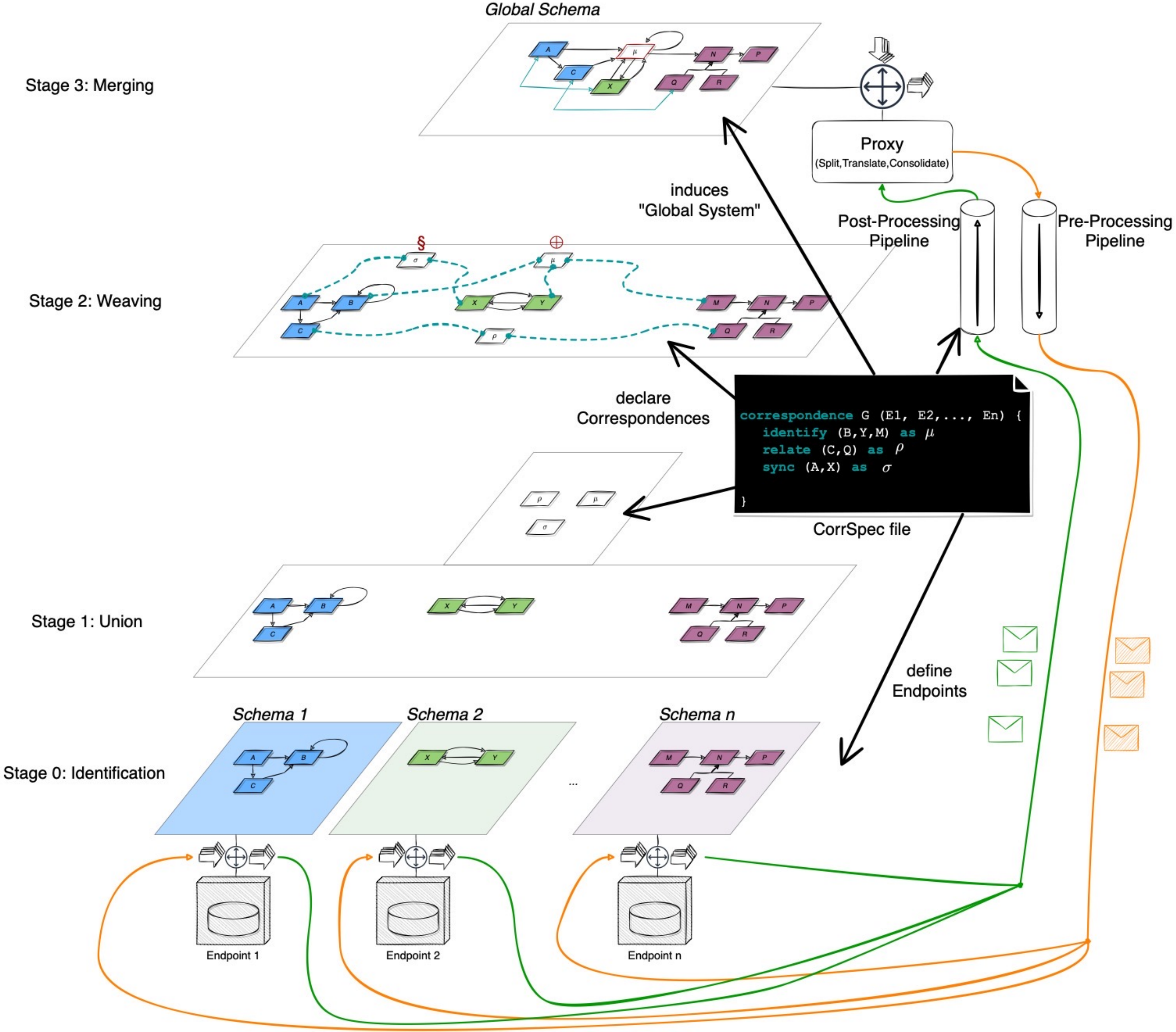


construction of comprehensive system / colimit

```
goal ComprehensiveSys {
        correspondence Backoffice
        action FEDERATION
        technology GRAPH_QL
        target SERVER {
                path /graphql
        }
 }
```

〉 Supported actions (currently):

SCHEMA	**global view** of the comprehensive *schema*
FEDERATION	**global view** of the comprehensive *data*
VERIFY	check the ”consistency” of the **global view**



```
correspondence G (E1, E2,..., En) {
  identify (B,Y,M) as μ
  relate (C,Q) as ρ
  sync (A,X) as  σ

}
```

The CorrLang DSL [3/3]

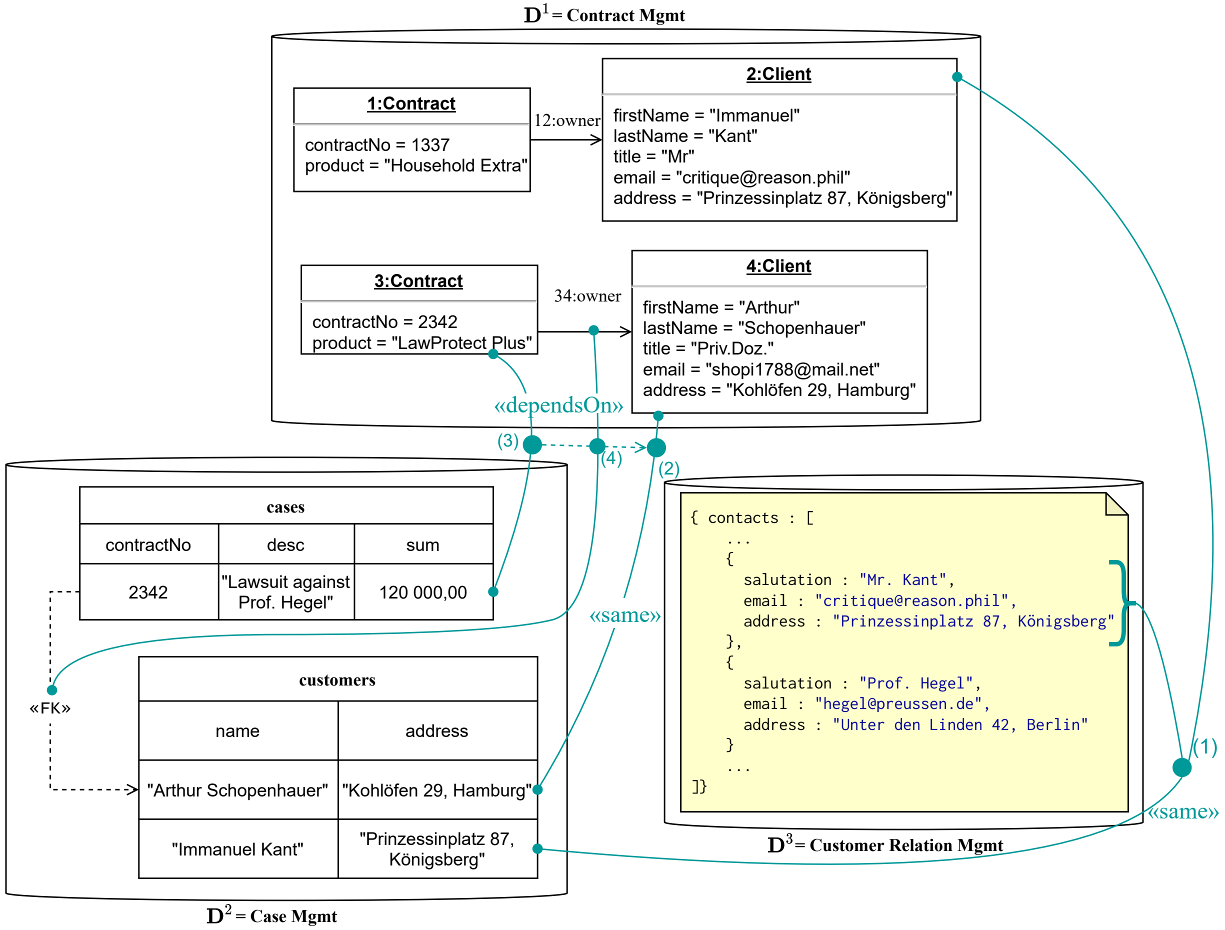
```
endpoint Sales {
    type SERVER
    at http://localhost:4011
    technology GRAPH_QL
    schema schemas/sales.graphql
 }
```

```
goal ComprehensiveSys {
       correspondence Backoffice
       action FEDERATION
       technology GRAPH_QL
       target SERVER {
              path /graphql
       }
 }
```

```
correspondence Backoffice (Sales, Invoices, HR) {
       // (1) merges types/attributes/associations
       identify (Sales.Customer, Invoices.Client, HR.Employee) as Partner;
       // (2) introduces new associations
       relate (Sales.Purchases, Invoices.Invoice) as paidIn;
       // (3) introduces new associations and automatically keeps them consistent
       sync (Sales.Customer.address, Invoices.Client.address);
}
```

CorrLang: Keys

```
correspondence Compr (CoM,CaM,CRM) {
  identify (CoM.Customer,CaM.Client,CRM.Contact)
        as Partner
        when (CoM.Customer.name ==
                CaM.Client.firstName ++ “ “ ++
                CaM.Client.lastName ||
                CaM.Client.title ++ “ “ ++
                CaM.Client.lastName ==
                CRM.Contact.salutation );


}
```



D^1 = Contract Mgmt

1:Contract
contractNo = 1337
product = "Household Extra"

12:owner

2:Client
firstName = "Immanuel"
lastName = "Kant"
title = "Mr"
email = "critique@reason.phil"
address = "Prinzessinplatz 87, Königsberg"

3:Contract
contractNo = 2342
product = "LawProtect Plus"

34:owner

4:Client
firstName = "Arthur"
lastName = "Schopenhauer"
title = "Priv.Doz."
email = "shopi1788@mail.net"
address = "Kohlöfen 29, Hamburg"

D^2 = Case Mgmt

cases

contractNo	desc	sum
2342	"Lawsuit against Prof. Hegel"	120 000,00

customers

name	address
"Arthur Schopenhauer"	"Kohlöfen 29, Hamburg"
"Immanuel Kant"	"Prinzessinplatz 87, Königsberg"

D^3 = Customer Relation Mgmt

```
{ contacts : [
    ...
    {
      salutation : "Mr. Kant",
      email : "critique@reason.phil",
      address : "Prinzessinplatz 87, Königsberg"
    },
    {
      salutation : "Prof. Hegel",
      email : "hegel@preussen.de",
      address : "Unter den Linden 42, Berlin"
    }
    ...
]}
```

CorrLang: Rules

```
correspondence Big (BPMN,DMN,UML) {
         identify (CoM.Customer,CaM.Client,CRM.Contact)
                  as Partner
                  when (...)
                  check CaseHasConstract;

         rule CaseHasConstract {
         using EVL """
                  Case.all().select(c|c.customer = self) implies
                           Contract.all().exists(c|c.owner = self);
         ""”
         }
}
```

CorrLang: Demo

https://youtu.be/98z64J3mPiQ

Related Work

- Semantic Web
  - similar representation (knowledge graph)
  - concrete technology ties vs. technology agnostic (tech space plugins)
  - focus on reasoning vs. focus on translation
- ETL/ELT
  - similar objectives
  - many-to-one vs. many-to-many
  - operational2analytical vs. operational2operational
- Categorical Databases (CQL)
  - similar objectives & related theoretical background
  - focus on databases vs. focus on system interfaces

Future Work

- Theory
  - Comprehensive System Grammars (automatic rule derivations)
  - Thorough unified treatment of attributes, inheritance, and commonalities (profunctors)
  - comprehensive systems of comprehensive systems

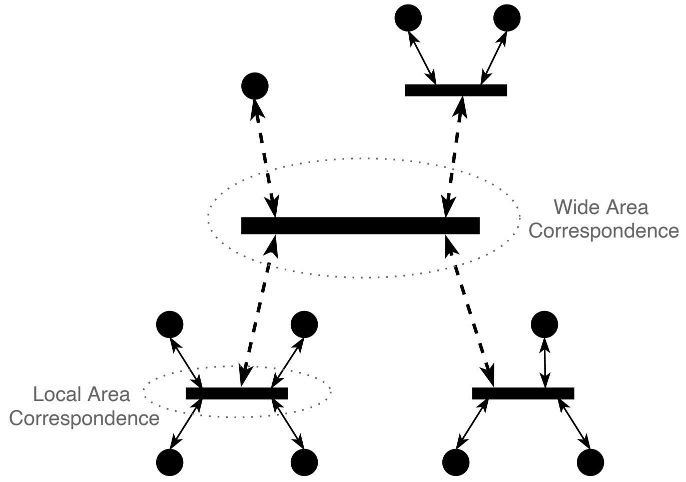
$$\begin{tikzcd}
G_0 \arrow[r, Rightarrow, "r_{s_1}"] \arrow[dr, Rightarrow, "r_1"] & G_{01} \arrow[r, Rightarrow, "r_{s_2}"] \arrow[d, Rightarrow, "\bar{r}_{s_1}"] & G_{02}\dots \arrow[r, Rightarrow, "r_{s_n}"] \arrow[d, Rightarrow, "\bar{r}_{s_1}"] & G_{0n} \arrow[d, Rightarrow, "\bar{r}_{s_1}"] \\
 & G_1 \arrow[r, Rightarrow, "r_{s_2}"] \arrow[dr, Rightarrow, "r_2"] & G_{12}\dots \arrow[r, Rightarrow, "r_{s_n}"] \arrow[d, Rightarrow, "\bar{r}_{s_2}"] & G_{1n} \arrow[d, Rightarrow, "\bar{r}_{s_2}"] \\
 & & G_2\dots \arrow[dr, Rightarrow, "r_n"] & \dots \arrow[d, Rightarrow, "\bar{r}_{s_n}"] \\
 & & & G_n
\end{tikzcd}$$

$AG : \mathbb{B}\times\mathbb{D} \to \boldsymbol{Set}$ (Attributed Graphs)

$CS : \mathbb{B}\times\mathbb{I} \to \boldsymbol{Set}$ (Comprehensive Systems

$IG: \mathbb{B}\times\mathbb{B} \to \boldsymbol{Set}$ (Graph with Inheritance)

- Practice
  - TechSpace Adapters: OpenAPI, gRPC, SQL, Avro
  - Use Cases **[open for suggestions!]**
  - improved tooling / IDE support (LSP)



References

- [1] A. Finkelstein, J. Kramer, B. Nuseibeh, L. Finkelstein, and M. Goedicke, ‘Viewpoints: a framework for integrating multiple perspectives in system development’, Int. J. Soft. Eng. Knowl. Eng., vol. 02, no. 01, pp. 31–57, Mar. 1992, doi: 10.1142/S0218194092000038.
- [2] B. Nuseibeh, S. Easterbrook, and A. Russo, ‘Leveraging inconsistency in software development’, Computer, vol. 33, no. 4, pp. 24–29, Apr. 2000, doi: 10.1109/2.839317
- [3] G. Spanoudakis and A. Zisman, ‘Inconsistency Management in Software Engineering: Survey and Open Research Issues’, 2000.
- [4] A. Reder and A. Egyed, ‘Computing repair trees for resolving inconsistencies in design models’, in 2012 Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering, Sep. 2012, pp. 220–229. doi: 10.1145/2351676.2351707.
- [5] F. Bancilhon and N. Spyratos, ‘Update Semantics of Relational Views’, ACM Trans. Database Syst., vol. 6, no. 4, pp. 557–575, Dec. 1981, doi: 10.1145/319628.319634.
- [6] J. N. Foster, M. B. Greenwald, J. T. Moore, B. C. Pierce, and A. Schmitt, ‘Combinators for Bidirectional Tree Transformations: A Linguistic Approach to the View-update Problem’, ACM Trans. Program. Lang. Syst., vol. 29, no. 3, May 2007, doi: 10.1145/1232420.1232424.
- [7] K. Matsuda, Z. Hu, K. Nakano, M. Hamana, and M. Takeichi, ‘Bidirectionalization Transformation Based on Automatic Derivation of View Complement Functions’, in Proceedings of the 12th ACM SIGPLAN International Conference on Functional Programming, in ICFP ’07. New York, NY, USA: ACM, 2007, pp. 47–58. doi: 10.1145/1291151.1291162.
- [8] M. Hofmann, B. Pierce, and D. Wagner, ‘Edit Lenses’, in Proceedings of the 39th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, in POPL ’12. New York, NY, USA: ACM, 2012, pp. 495–508. doi: 10.1145/2103656.2103715

References

- [8] M. Hofmann, B. Pierce, and D. Wagner, ‘Edit Lenses’, in Proceedings of the 39th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, in POPL ’12. New York, NY, USA: ACM, 2012, pp. 495–508. doi: 10.1145/2103656.2103715.
- [9] M. Johnson and R. D. Rosebrugh, ‘Unifying Set-Based, Delta-Based and Edit-Based Lenses’, in Proceedings of the 5th International Workshop on Bidirectional Transformations, Bx 2016, co-located with The European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 8, 2016, A. Anjorin and J. Gibbons, Eds., in CEUR Workshop Proceedings, vol. 1571. CEUR-WS.org, 2016, pp. 1–13.
- [10] L. M. Haas, M. A. Hernández, H. Ho, L. Popa, and M. Roth, ‘Clio Grows Up: From Research Prototype to Industrial Tool’, in Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, in SIGMOD ’05. New York, NY, USA: ACM, 2005, pp. 805–810. doi: 10.1145/1066157.1066252.
- [11] S. Melnik, E. Rahm, and P. A. Bernstein, ‘Rondo: A Programming Platform for Generic Model Management’, in Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, in SIGMOD ’03. New York, NY, USA: ACM, 2003, pp. 193–204. doi: 10.1145/872757.872782.
- [12] R. F. Paige, D. S. Kolovos, L. M. Rose, N. Drivalos, and F. A. C. Polack, ‘The Design of a Conceptual Framework and Technical Infrastructure for Model Management Language Engineering’, in 2009 14th IEEE International Conference on Engineering of Complex Computer Systems, Jun. 2009, pp. 162–171. doi: 10.1109/ICECCS.2009.14.
- [13] R. Salay, S. Kokaly, A. Di Sandro, N. L. S. Fung, and M. Chechik, ‘Heterogeneous megamodel management using collection operators’, Softw Syst Model, vol. 19, no. 1, pp. 231–260, Jan. 2020, doi: 10.1007/s10270-019-00738-9.

References

- [14] A. Schürr, ‘Specification of Graph Translators with Triple Graph Grammars’, in LNCS, in WG ’94, vol. 903. London, UK, UK: Springer-Verlag, 1994, pp. 151–163. [Online]. Available: http://dl.acm.org/citation.cfm?id=647675.731658
- [15] H. Ehrig, K. Ehrig, C. Ermel, F. Hermann, and G. Taentzer, ‘Information Preserving Bidirectional Model Transformations’, in Fundamental Approaches to Software Engineering, M. B. Dwyer and A. Lopes, Eds., in Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, pp. 72–86.
- [16] F. Hermann et al., ‘Model synchronization based on triple graph grammars: correctness, completeness and invertibility’, Softw Syst Model, vol. 14, no. 1, pp. 241–269, Feb. 2015, doi: 10.1007/s10270-012-0309-1.
- [17] J. Hereth and R. Pöschel, ‘Peircean Algebraic Logic and Peirce’s Reduction Thesis’, Semiotica, vol. 2011, no. 186, pp. 141–167, 2011, doi: 10.1515/semi.2011.050.
- [18] P. Stünkel, H. König, Y. Lamo, and A. Rutle, ‘Towards Multiple Model Synchronization with Comprehensive Systems’, in Fundamental Approaches to Software Engineering, H. Wehrheim and J. Cabot, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 335–356. doi: 10.1007/978-3-030-45234-6_17.
- [19] P. Stünkel, H. König, Y. Lamo, and A. Rutle, ‘Comprehensive Systems: A formal foundation for Multi-Model Consistency Management’, Form. Asp. Comput., vol. 33, no. 6, pp. 1067–1114, Dec. 2021, doi: 10.1007/s00165-021-00555-2.
- [20] P. Stünkel and H. König, ‘Single pushout rewriting in comprehensive systems of graph-like structures’, Theoretical Computer Science, vol. 884, pp. 23–43, Sep. 2021, doi: 10.1016/j.tcs.2021.07.002.

References

- [21] S. Feldmann, K. Kernschmidt, M. Wimmer, and B. Vogel-Heuser, ‘Managing inter-model inconsistencies in model-based systems engineering: Application in automated production systems engineering’, Journal of Systems and Software, vol. 153, pp. 105–134, Jul. 2019, doi: 10.1016/j.jss.2019.03.060.
- [22] L. Samimi-Dehkordi, B. Zamani, and S. Kolahdouz-Rahimi, ‘EVL+Strace: a novel bidirectional model transformation approach’, Information and Software Technology, vol. 100, pp. 47–72, Aug. 2018, doi: 10.1016/j.infsof.2018.03.011.
- [23] F. Trollmann and S. Albayrak, ‘Extending Model Synchronization Results from Triple Graph Grammars to Multiple Models’, in Theory and Practice of Model Transformations, P. Van Gorp and G. Engels, Eds., in Lecture Notes in Computer Science. Springer International Publishing, 2016, pp. 91–106.
- [24] J. Kosiol, L. Fritsche, A. Schürr, and G. Taentzer, ‘Adhesive Subcategories of Functor Categories with Instantiation to Partial Triple Graphs’, in Graph Transformation, E. Guerra and F. Orejas, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 38–54. doi: 10.1007/978-3-030-23611-3_3.
- [25] P. Stünkel, O. van Bargen, A. Rutle, and Y. Lamo, ‘GraphQL Federation: A Model-Based Approach’, JOT, vol. 19, no. 2, p. 18:1-21, 2020, doi: 10.5381/jot.2020.19.2.a18.