Indexed vs. Fibred Semantics in view of Metamodeling

Uwe Wolter, University of Bergen, Norway

Collaboration with Zinovy Diskin, Adrian Rutle, Alessandro Rossini, Yngve Lamo

ACCAT 2010, March 21st 2010, Paphos, Cyprus

Outline

Motivation for the Talk

- Looking back I realized that there is always a problem with my talks: I rather point out problems and present interesting observations and questions, to initiate scientific discussions, then to show "knackige Resultate".
- The motivation for this talk goes 15 years back when we have been forced in Berlin to look at and to work with UML. My viewpoint was that UML is only syntax and that there is no semantics. Today I present a kind of revision of this viewpoint.
- Everything is triggered by the development and the applications of the Diagram Predicate Framework (DPF), formerly also called Generalized Sketch Framework.
- There are many other interesting things to work at and to talk about but let's try to concentrate today at ...

Motivation behind DPF

- In Software Engineering we find a variety of "diagrammatic modeling/specification techniques" like UML- or ER-, or ORM-diagrams, database schema, ontologies, XML, Petri Nets, State Charts, ...
- Recently, there is a movement (MDE, MDD, MDA,...) aimed at making models rather than code the primary artifacts of software development. To achieve this goal, having a precise formal semantics for diagrammatic notations is not only desired but an absolute must.
- The development of DPF is motivated by those practical needs and requirements, but also by some simple pure scientific questions.

Aims of our Research

Development of a modeling/specification/logical framework:

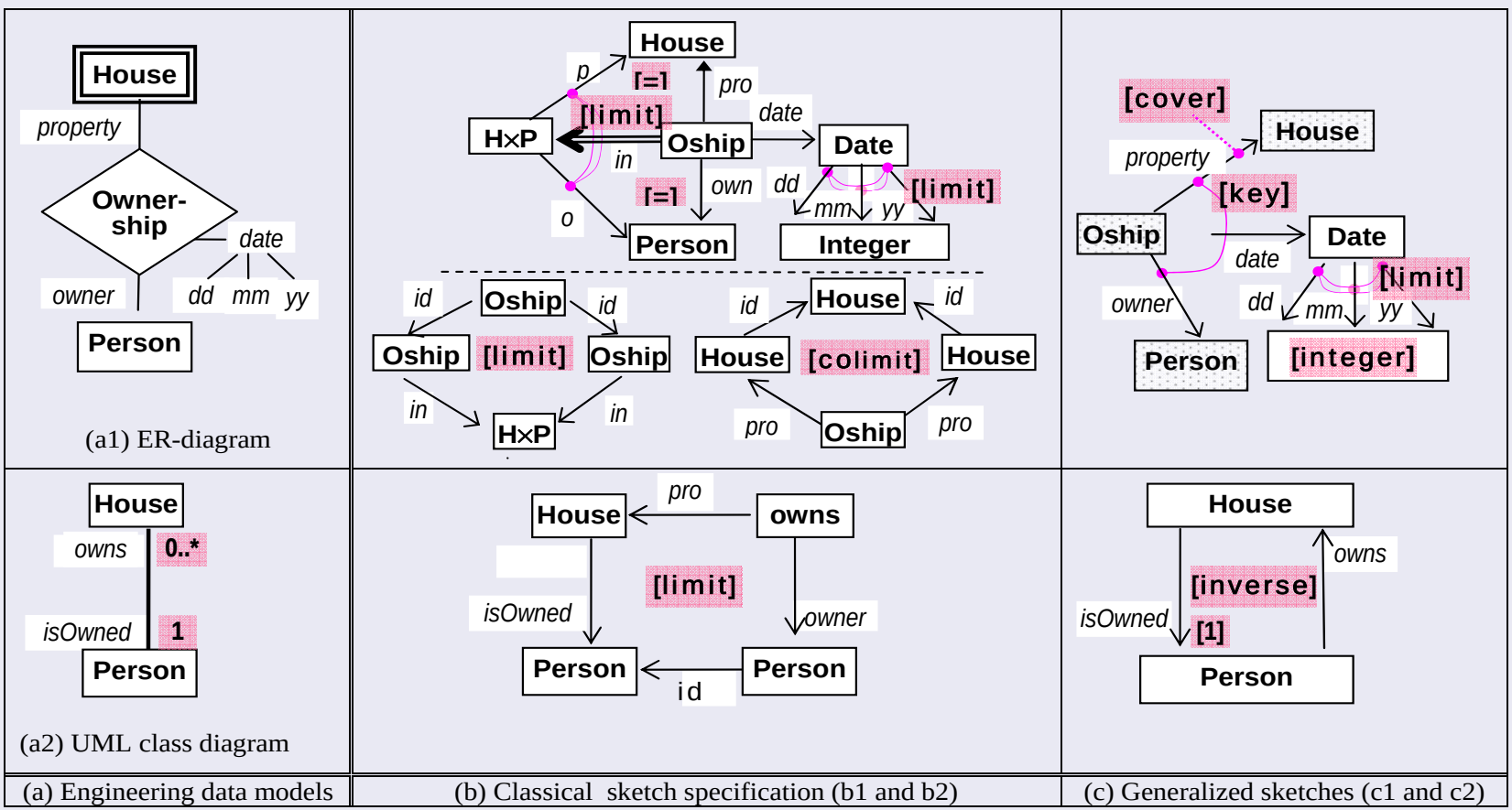
- Conceptually close to and covering the needs in "diagrammatic modeling", especially in MDE.
- Precise formal and compositional syntax and semantics.
- Generic w.r.t. underlying graph structures (graphs, typed graphs, attributed graphs, ...).
  - In the talk we consider only directed multi-graphs.
- Graph based formulas, instead of string-based, and a fully fledged graph-based logic.
- Extends as many results, constructions and techniques, as possible, from Graph Transformations to DPF.

DPF as an Extension of GT

DPF can be seen as a natural extension of Graph Transformations:

- Instead of graph structures we consider specifications (models), i.e., graph structures together with "atomic" constraints. That is, we extend graphs by a logical component.
  - The components of transformation rules become specifications.
  - The "old" (injective) graph transformation rules are still present, but now as "operation declarations".
  - Graph constraints turn into "universal constraints" (called sketch axioms/entailments by Makkai 1997).
  - ...

A sample of sketching a diagrammatic notation



(a1) ER-diagram

(a2) UML class diagram

(a) Engineering data models

(b) Classical sketch specification (b1 and b2)

(c) Generalized sketches (c1 and c2)

DPF - Syntax

Diagrammatic Notation $\cong$ Signature

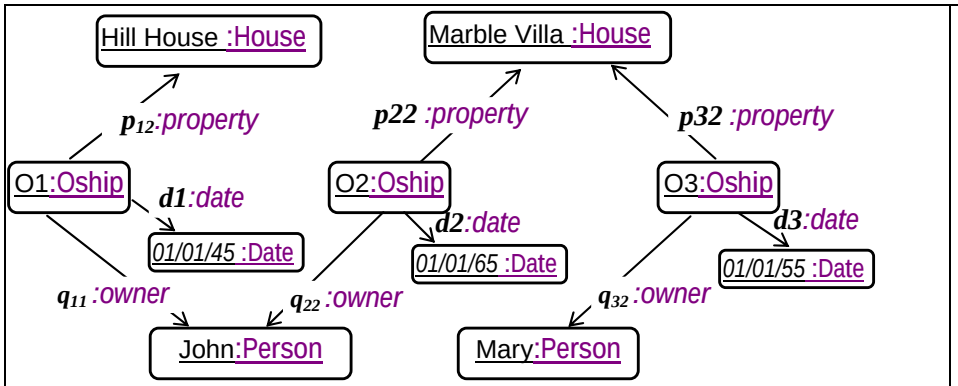
A Signature $\Sigma = (\Pi, \alpha)$ is given by a set Π of **predicate labels (symbols)** and a map α assigning to each label $P \in \Pi$ its **arity (graph)** $\alpha(P)$.

Specification/Model $\cong$ Graph + Constraints

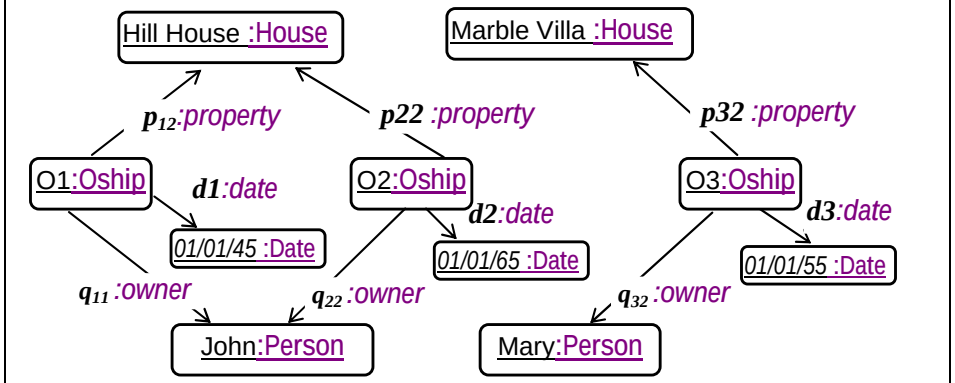
A Σ-**specification (model)** $\mathcal{S} = (S, C^{\mathcal{S}})$ consists of a graph S and a set $C^{\mathcal{S}}$ of **constraints** (diagrams) (P, d) with $P \in \Pi$ a label and $d : \alpha(P) \to G$ a graph homomorphism.

$$\Sigma \overset{C^{\mathcal{S}}}{\dashrightarrow} S$$

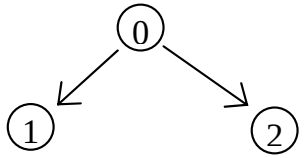
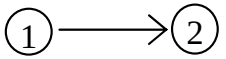
Example



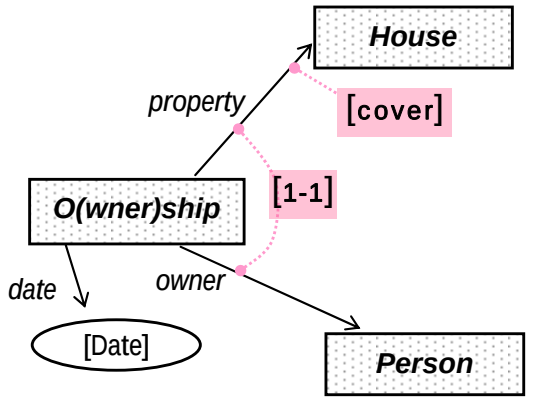
(c1) Valid instance of schema $\mathcal{S}$



(c2) Invalid instance of schema $\mathcal{S}$

Predicate symbol	Arity
[1-1]	
[cover]	

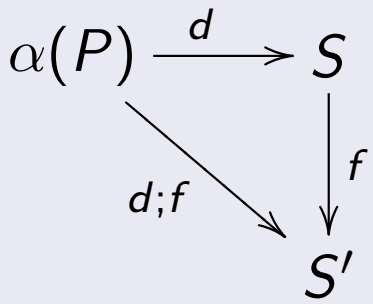
(a) Signature, Π



(b) Schema/ specification/model/sketch, $\mathcal{S}$

Specification Morphisms

A Σ-**specification morphism** $f : \mathcal{S} \to \mathcal{S}'$ between two Σ-specifications $\mathcal{S} = (S, C^{\mathcal{S}})$ and $\mathcal{S}' = (S', C^{\mathcal{S}'})$ is a graph homomorphism $f : S \to S'$ preserving constraints, i.e., for all constraints $(P, d) \in C^{\mathcal{S}}$ we have that $(P, d; f) \in C^{\mathcal{S}'}$.



(Co)Completeness

For any signature Σ the category $Spec(\Sigma)$ of Σ-specifications and Σ-specification morphisms is finitely complete and finitely cocomplete.

Indexed vs. Fibred Semantics

Indexed Semantics

- Tradition in Algebraic Specifications, Denotational and Functorial Semantics, Institutions ...
- A **model** is an interpretation of a **specification** $\mathcal{S}$ in a **"semantic universe"** $\mathcal{U}$, i.e., an arrow $m : \mathcal{S} \to \mathcal{U}$ in a **"meta-universe"** Meta.

Fibred Semantics

- Preferred by "practitioners" (and some theoreticians).
- Specifications are called **"models"** (note the "cultural clash"!). A model is an object $\mathcal{M}$ in a category Fibr and an **instance** of $\mathcal{M}$ is given by another object I in Fibr together with a morphism $\iota : I \to \mathcal{M}$.

Example - Many-Sortedness

Indexed

A many-sorted set is a family of sets $\mathcal{A} = (A_s \mid s \in S)$ indexed by a given set S of sort symbols, i.e., a map (functor) $\mathcal{A} : S \to \mathrm{Set}$.

Fibred

A many-sorted set is given by a set A and a "sorting" map $t_A : A \to S$ into a given set S of sort symbols, i.e., by a morphism in the category Set.

Transitions

From indexed to fibred by disjoint union construction and vice versa by inverse image construction. Gives an equivalence of categories.

Example - Arity (of Operation Symbols)

Indexed

Operation symbols are defined as an indexed set $\mathcal{O} = (O_{(w,s)} \mid (w,s) \in S^* \times S)$ for a given set S of sort symbols, i.e., a map (functor) $\mathcal{O} : S^* \times S \to \mathrm{Set}$.
$\Rightarrow$ Allows for "overloading".

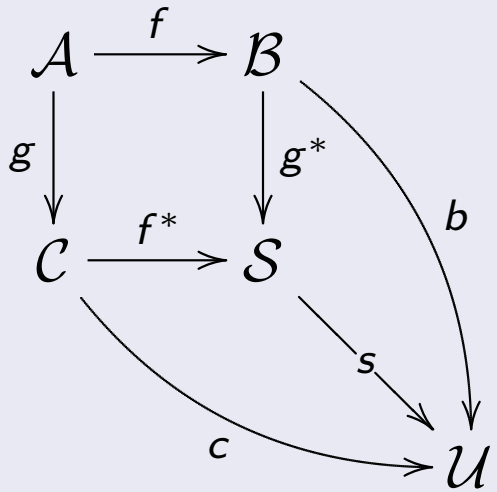
Fibred

Operation symbols are given by a set O and an "arity" map $t_O : O \to S^* \times S$ for a given set S of sort symbols.
$\Rightarrow$ No "overloading" due to disjointness.

Example - Amalgamation (WADT'08)

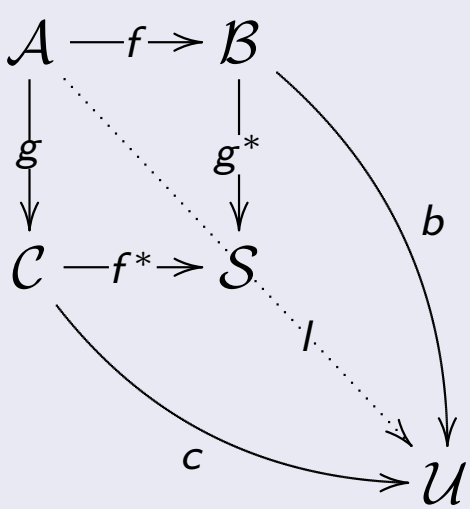
Indexed Amalgamation $\cong$ Pushout Property

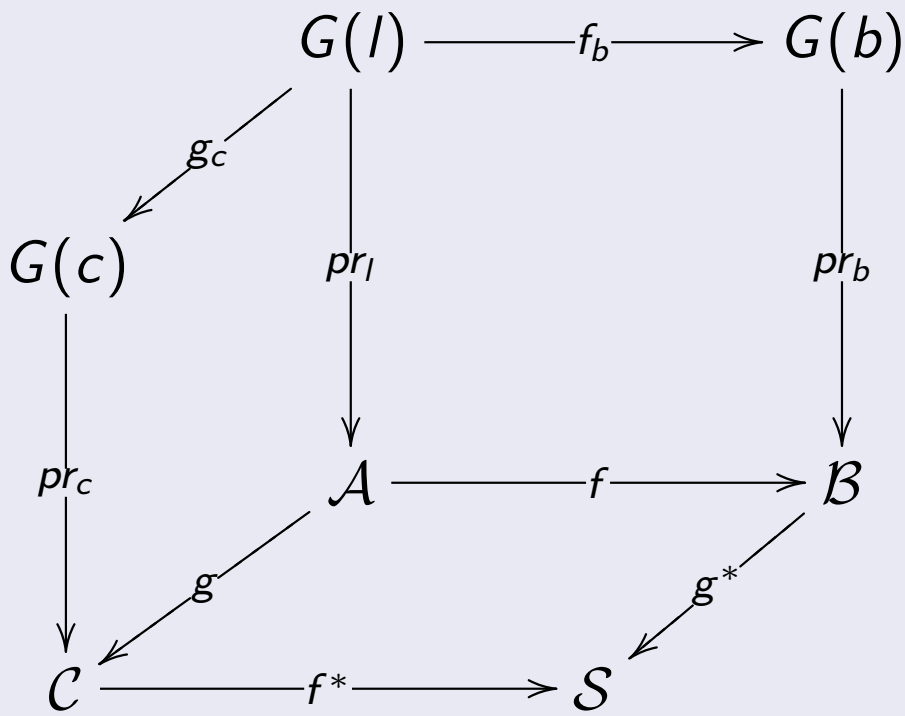
Let be given a pushout of specifications. Then there exists for any *coherent pair* of interpretations, i.e., for any $b : \mathcal{B} \to \mathcal{U}$ and $c : \mathcal{C} \to \mathcal{U}$ with $f; b = g; c$, a unique interpretation $s : \mathcal{S} \to \mathcal{U}$ such that $f^*; s = c$ and $g^*; s = b$ called the *amalgamated sum* of b and c.



Fibred Amalgamation $\cong$???

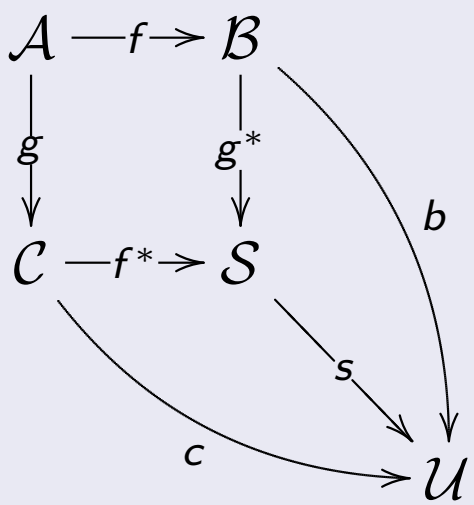
A Grothendieck construction turns composition (commutative triangles) into pullback diagrams thus any coherent pair of interpretations gives rise to a **pullback-pushout half cube**.

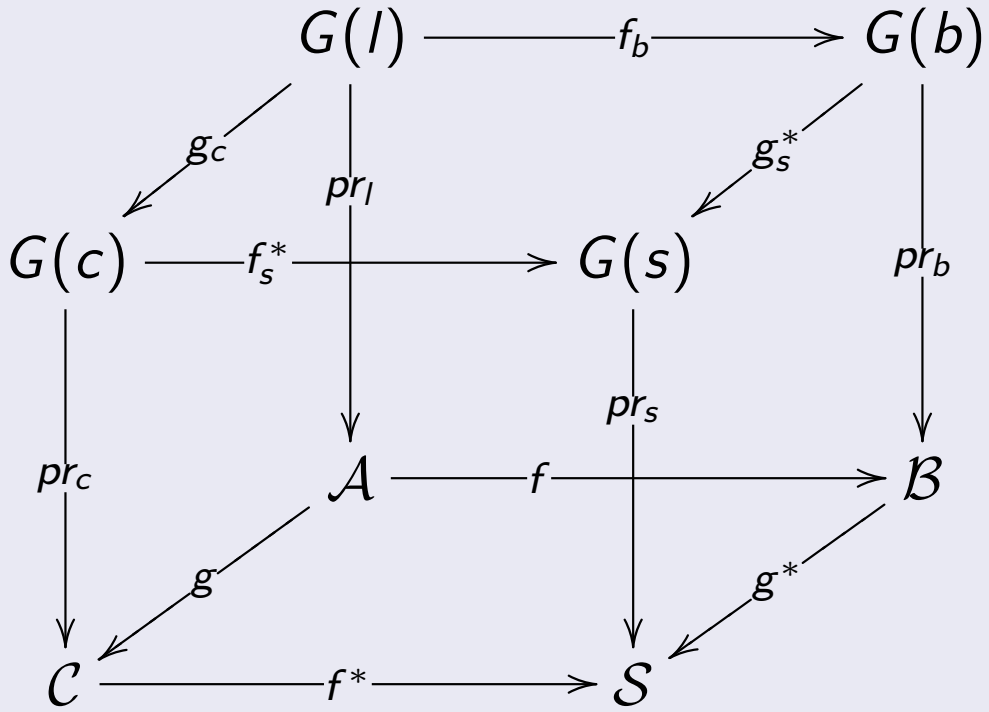




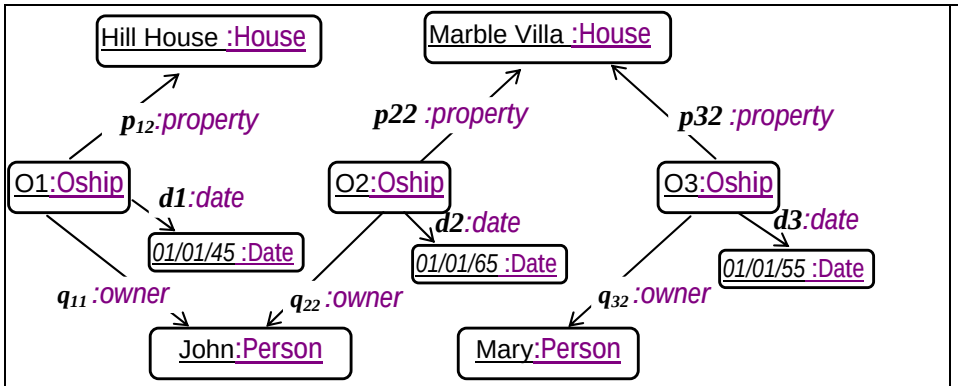
Fibred Amalgamation $\cong$ van-Kampen Square?

The existence of a unique mediating morphism s is turned, by the Grothendieck construction, into the existence of a unique pullback completion of the pullback-pushout half cube (WADT'08).

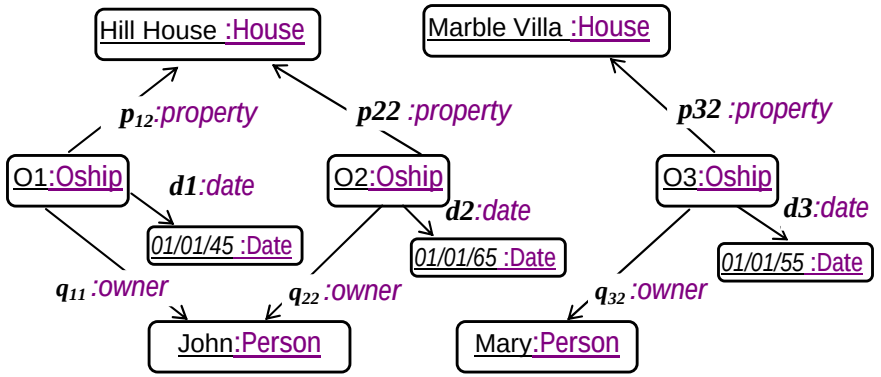




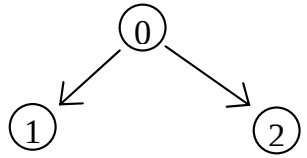
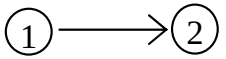
Fibred Semantics - Example



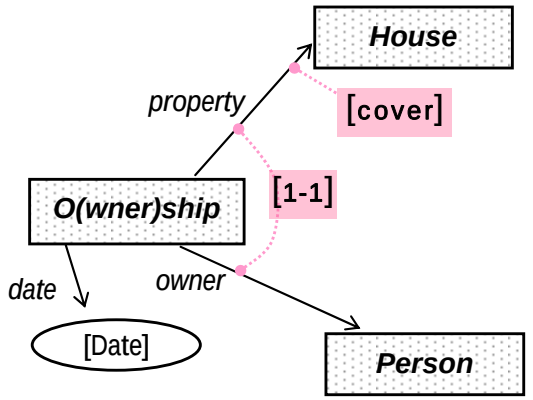
(c1) Valid instance of schema $\mathcal{S}$



(c2) Invalid instance of schema $\mathcal{S}$

Predicate symbol	Arity
[1-1]	
[cover]	

(a) Signature, Π



(b) Schema/ specification/model/sketch, $\mathcal{S}$

Fibred Semantics of DPF

To describe things like "an object diagram for a given class diagram" and "a state of a database" we have to answer the question: What is an "instance" of a specification? And before we can do this we have to fix the meaning of the constraints of our modeling technique.

Semantics of Signatures

A **semantic interpretation** of a signature $\Sigma = (\Pi, \alpha)$ is a mapping $[\![..]\!]$, which assigns to each predicate symbol $P \in \Pi$ a set $[\![P]\!]$ of graph homomorphisms $\iota : I \to \alpha(P)$ called the **valid instances of** P.

In practical applications we have to program a "validator" for each new predicate we coin.

Fibred Semantics of DPF

Instances

An **instance** of a Σ-specification $\mathcal{S} = (S, C^{\mathcal{S}})$ is given by a graph O and a graph homomorphism $\tau : O \to S$ such that $\tau|_d \in [\![P]\!]$ for all constraints (P, d) in $C^{\mathcal{S}}$, where $\tau|_d$ is given by the following pullback

$$\begin{array}{ccc} O|_d & \xrightarrow{d^*} & O \\ {\scriptstyle \tau|_d}\downarrow & & \downarrow{\scriptstyle \tau} \\ \alpha(P) & \xrightarrow{d} & S \end{array} \qquad \begin{array}{ccc} O & & \\ \downarrow{\scriptstyle \tau} & & \\ \Sigma & \xrightarrow{C^{\mathcal{S}}} & S \end{array}$$

By $Inst(\mathcal{S})$ we denote the full subcategory of the comma category $(\mathrm{Graph} \downarrow S)$ given by all instances of $\mathcal{S}$.

Fibred Semantics of DPF

Reduction of Instances

Every Σ-specification morphism $f : \mathcal{S} \to \mathcal{S}'$ induces a functor

$$Inst(f) : Inst(\mathcal{S}') \to Inst(\mathcal{S}) \quad \text{with} \quad Inst(f)(\tau) \stackrel{def}{=} \tau|_f$$

for all instances $\tau : O \to S'$ of $\mathcal{S}'$.

$$\begin{array}{ccc} O|_f & \xrightarrow{f^*} & O \\ {\scriptstyle \tau|_f}\downarrow & & \downarrow{\scriptstyle \tau} \\ S & \xrightarrow{f} & S' \end{array}$$

Remark: Since pullbacks are only determined up to isomorphism all these maps provide globally only a pseudo "instance functor" $Inst : Spec(\Sigma) \to$ Cat (ACCAT'07).

Meta-modeling

What is Meta-modeling?

- The syntax (of a collection) of models on a certain modeling level is described as the semantics of a model on a more abstract modeling level (meta-level).
- In other words: Specifications on a meta-level are used to define (restrictions on) the syntax of models.

Steps in fibred Meta-Modeling

Step 1: Typing

The underlying graph of a specification $\mathcal{S}$ is required to be "typed" over a graph T. (We can also require that the arities of predicates are typed over T (JLAP 2010)).

Steps in fibred Meta-Modeling

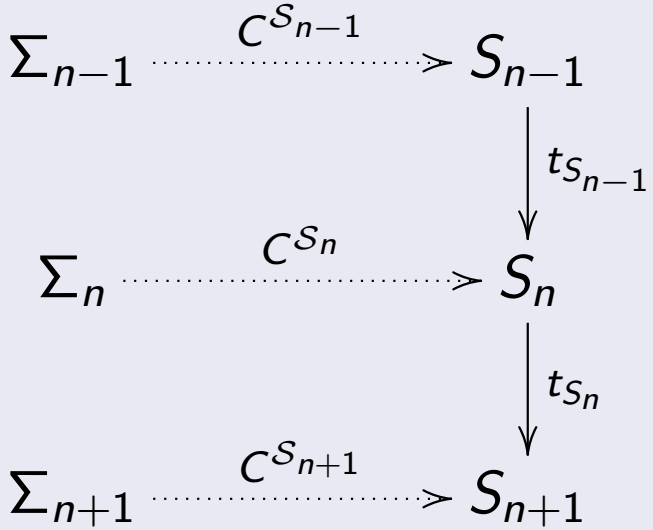
Step 2: Constraints

We chose a possibly different signature Σ_T to formulate constraints which the typing morphisms have to satisfy. That is, we define a Σ_T-specification $\mathcal{T} = (T, C^{\mathcal{T}})$ on the meta-level

$$\begin{array}{ccc} \Sigma & \overset{C^{\mathcal{S}}}{\dashrightarrow} & S \\ & & \downarrow{\scriptstyle t_S} \\ \Sigma_T & \overset{C^{\mathcal{T}}}{\dashrightarrow} & T \end{array}$$

Iteration of fibred Meta-Modeling

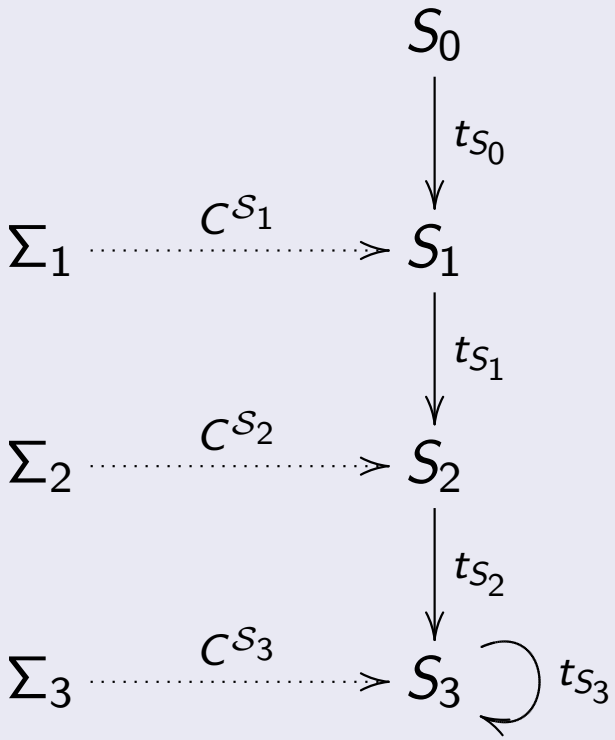
We can naturally iterate the idea of fibred meta-modeling - as well backward (abstraction) as forward (instances).



In practice we may also have inclusions
$\ldots \subseteq \Sigma_{n+1} \subseteq \Sigma_n \subseteq \Sigma_{n-1} \subseteq \ldots$

Iteration of fibred Meta-Modeling

Backward chaining (abstraction) ends up, naturally, in a reflexive specification, where forward chaining stops at (one step before) "reality".



Outline of some theory concerning fibred Meta-Modeling

Fibred semantics and Institutions

If we consider now only graphs as models and as instances then we have the following situation at one single modeling level:

- The category of specifications is the category of graphs: $\mathrm{Spec} = \mathrm{Graph}$
- For any specification S the corresponding category of instances is given by a slice category: $Inst(S) = (\mathrm{Graph} \downarrow S)$
- Any specification morphism $f : S \to S'$ defines, by pullback construction, a functor $Inst(f) : Inst(S') \to Inst(S)$.
- All these functors constitute a pseudo instance functor $Inst : \mathrm{Graph}^{op} \to \mathrm{Cat}$.

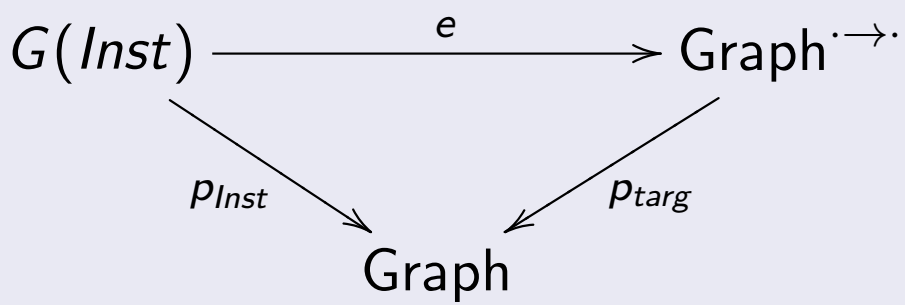
Attention: We have now a mixture: Internally we use the fibred concept of a slice category and globally we use indexed categories. **Fully fibred description of this situation?**

Outline of some theory concerning fibred Meta-Modeling

Grothendieck vs. fibrations

A generalized Grothendieck construction provides for the pseudo functor $Inst : \mathrm{Graph}^{op} \to \mathrm{Cat}$ a category $G(Inst)$ together with a projection functor $p_{Inst} : G(Inst) \to \mathrm{Graph}$.

Proposition: The Grothendieck category $G(Inst)$ is equivalent to the arrow category $\mathrm{Graph}^{\cdot\to\cdot}$.



Hypothesis: Arrow categories provide the adequate concept to describe and to reason about multilevel fibred semantics.

Outline of some theory concerning fibred Meta-Modeling

Pullbacks vs. Fibrations

Proposition: For any category C with pullbacks the arrow category $C^{\cdot\to\cdot}$ together with the target projection functor $p_{targ} : C^{\cdot\to\cdot} \to C$ is a fibration, where the cartesian arrows correspond exactly to the pullbacks in C.

$\Rightarrow$ These constructions and results can be iterated over arbitrary levels and the corresponding categories for fibred semantics are the categories $C^{\cdot\to\cdot\to\cdots\to\cdot}$ of commutative multi-squares, where cartesian arrows are given by multi-pullbacks.

Open Program for Basic Research - Fibred vs. Indexed

1. Comprehensive theory of comma categories, arrow categories, and fibrations. Not available at the moment!
2. Full understanding and mastering of the transitions between the fibred and the indexed world.
   - $\Rightarrow$ Slice category functor is the fibred version of the Yoneda Embedding.
   - $\Rightarrow$ What are the fibred versions of pre-sheaf topoi?
   - $\Rightarrow$ A more proper equivalence between fibrations and indexed categories.
3. Abstract vs. concrete syntax?
4. Fibred semantics for behavioural models (Petri nets, State Charts, Sequence Charts, ...).
5. ...

Thanks

Thanks for your attention! Questions?

Indexed Meta-Modeling?

- Given a category $Spec(\Sigma)$ of specifications and a semantic category $\mathcal{U}$ we assign to any specification $\mathcal{S}$ (a subcategory of) the functor category $[\mathcal{S} \to \mathcal{U}]$.
- Forward chaining would mean to interpret models $m : \mathcal{S} \to \mathcal{U}$ as specifications and to interpret them in another(?) semantic category $\mathcal{U}'$.
  - $\Rightarrow$ It seems that nobody has realized such a strange looking idea?
- Backward chaining (abstraction) means to find a semantic category $\mathcal{V}$ and a meta-specification $\mathcal{M}$ such that $Spec(\Sigma)$ can be described as (subcategory of) $[\mathcal{M} \to \mathcal{V}]$.
  - $\Rightarrow$ This idea is "in the air" and people focus mostly on reflective formalisms , i.e., $\mathcal{V} = \mathcal{U}$ and $\mathcal{M} \in Spec(\Sigma)$.
- There seems to be no elegant formalization of iterated meta-modeling as we have it for fibred semantics.

Indexed Semantics

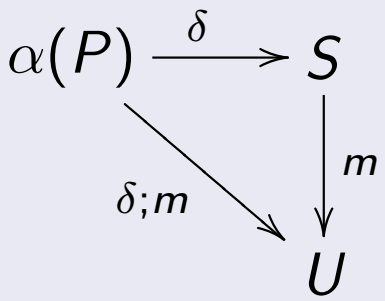
Semantics of Signatures

Following the "culture of theoreticians" we can define the semantics of Σ-specifications in an indexed way. First, we have to decide for a **"semantic universe"**, i.e., we have to chose a "big" graph U constituted by appropriate semantic objects and morphisms between them. Most commonly the semantic universe will be given by one of the categories Set, Par, Rel, Graph, or Cat, respectively. In a second step we have to convert the graph U into a "semantic Σ-specification" $\mathcal{U} = (U, \mathcal{U}(\Pi))$, i.e., for all labels $P \in \Pi$ we have to give a mathematical exact definition of the "property P", i.e., of the corresponding subset of $\mathrm{GRAPH}(\alpha(P), U)$ of all "semantic valid diagrams" of arity $\alpha(P)$, where GRAPH is the category of "big" graphs. Then we can define models within this chosen "semantic universe".

Indexed Semantics

$\mathcal{S}$-model

An $\mathcal{S}$-**model** of a Σ-specification $\mathcal{S}$ in a semantic Σ-universe $\mathcal{U}$ is a Σ-specification morphism $m : \mathcal{S} \to \mathcal{U}$, i.e.,
$(P, \delta : \alpha(P) \to S) \in \mathcal{S}(P)$ implies $(P, \delta; m : \alpha(P) \to U) \in \mathcal{U}(P)$ for all $P \in \Pi$.

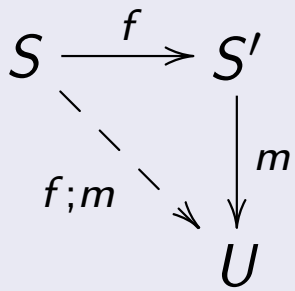


By $Mod(\mathcal{S}, \mathcal{U})$ we denote the collection of all $\mathcal{S}$-models in $\mathcal{U}$.

Indexed Semantics

Forgetful Map

For any Σ-specification morphism $f : \mathcal{S} \to \mathcal{S}'$ we obtain a **forgetful map** $f_{\mathcal{U}} : Mod(\mathcal{S}', \mathcal{U}) \to Mod(\mathcal{S}, \mathcal{U})$ where $f_{\mathcal{U}}(m) \stackrel{def}{=} f; m$ for any $\mathcal{S}'$-model $m : \mathcal{S}' \to \mathcal{U}$.



Remark: Since composition is associative all these maps provide globally a so-called model functor for each signature Σ!

(Variants of) The Grothendieck Constructioncs

- Provide the transition from indexed to fibred semantics.
- Given the category Spec of specifications and a semantic universe $\mathcal{U}$ (and thus Meta) we have to find an appropriate category Fibr together with an embedding $em : \text{Spec} \to \text{Fibr}$ such that for each specification $\mathcal{S}$ there is a "model" $em(\mathcal{S})$ and for each arrow $m : \mathcal{S} \to \mathcal{U}$ in Meta there is an instance $pr_m : Fl(m) \to em(\mathcal{S})$.
- **Here now:**

$$\begin{aligned} \text{Spec} &= \text{Graph} \\ \mathcal{U} &= \text{Set} \\ \text{Meta} &= \text{GRAPH} \\ \text{Fibr} &= \text{Graph} \end{aligned}$$

Grothendieck Construction: A simple version

For any graph H and any graph morphism $m : H \to \text{Set}$ we can construct a graph $G(m)$ as follows:

- **nodes:** $\langle i, x \rangle$ with i a node in H and x a node in $m(i)$
- **arrows:** $\langle \sigma, x \rangle : \langle i, x \rangle \to \langle j, m(\sigma)(x) \rangle$ with $\sigma : i \to j$ an arrow in H and x a node in $m(i)$ (and thus $m(\sigma)(x)$ a node in $m(j)$).

$$\begin{array}{ccc} i & m(i) & x \\ \downarrow{\scriptstyle \sigma} & \downarrow{\scriptstyle m(\sigma)} & \downarrow \\ j & m(j) & m(\sigma)(x) \end{array}$$

We obtain a graph morphism $pr_m : G(m) \to H$ with $pr_m(\langle i, x \rangle) = i$ for any node $\langle i, x \rangle$ in $G(m)$ and $pr_m(\langle \sigma, x \rangle) = \sigma$ for any arrow $\langle \sigma, x \rangle$ in $G(m)$.

Commutative Triangles to Pullbacks

Proposition

For any graph morphism $\varphi : H \to K$ and any graph morphism $m : K \to \mathrm{Set}$ the assignments

- $\varphi_m(\langle i, x\rangle) = \langle \varphi(i), x\rangle$ for any node $\langle i, x\rangle$ in $G(\varphi, m)$,
- $\varphi_m(\langle \sigma, x\rangle) = \langle \varphi(\sigma), x\rangle$ for any arrow $\langle \sigma, x\rangle$ in $G(\varphi, m)$.

define a graph morphism $\varphi_m : G(\varphi, m) \to G(m)$ such that the following right diagram is a pullback diagram in Graph