Multi-view Consistency in UML

Alexander Knapp[1] Till Mossakowski[2]

[1]University of Augsburg, Germany
[2]University of Magdeburg, Germany

IFIP WG 1.3, 10 Jan 2017

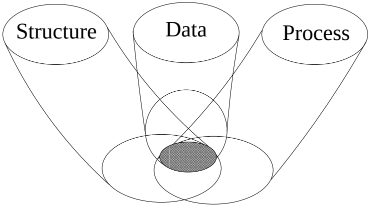
Multi-View(point) Modelling

"...modularisation concepts are required, which allow to compose a complete and consistent specification out of possibly overlapping pieces.

...one way to obtain this modularisation is the concept of views and view integration.

The basic idea is to monitor the relationships between different viewpoints, to detect inconsistencies and to resolve them by interactive support of the user."

Gregor Engels, Reiko Heckel, Gabriele Taentzer, and Hartmut Ehrig. *A Combined Reference Model- and View-Based Approach to System Specification*. In: Intl. J. Softw. Eng. Knowl. Eng. 7.4 (1997), pp. 457–477



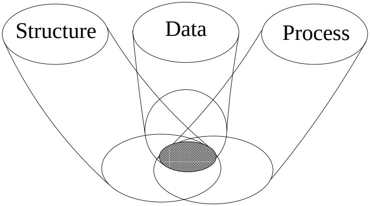
The Unified Modeling Language (UML)

UML provides different diagram type for different viewpoints

Structure		Class Diagram		static structure (generic/snapshot)
		Composite Structure Diagram		logical system structure
		Component Diagram		physical system structure
		Deployment Diagram		computing infrastructure / deployment
		Package Diagram		containment hierarchy
Behavior		Use Case Diagram		abstract functionality
		Activity Diagram		controlflow and dataflow
	Interaction	Sequence Diagram	*interactions by message exchange*	message exchange over time
		Communication Diagram		structure of interacting elements
		Timing Diagram		coordinated state change over time
		Interaction Overview Diagram		flows of interactions
		State Machine Diagram		event-triggered state change

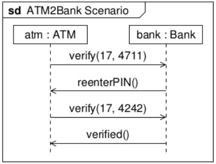
Multi-viewpoint model = family of UML models

Central Question

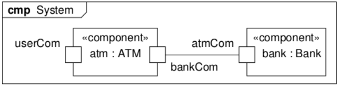


Is a UML multi-viewpoint model consistent?
I.e. is it possible to realise the model?
Early detection of inconsistency avoids costly redesign!

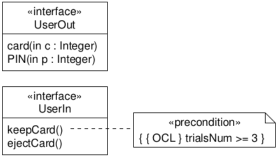
Multi-view consistency example: an ATM



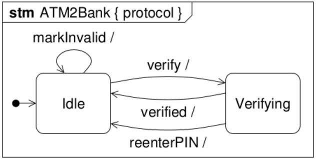
(a) Interaction



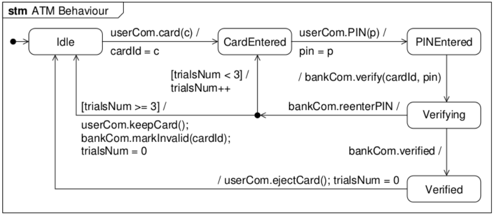
(b) Composite structure



(c) Interfaces

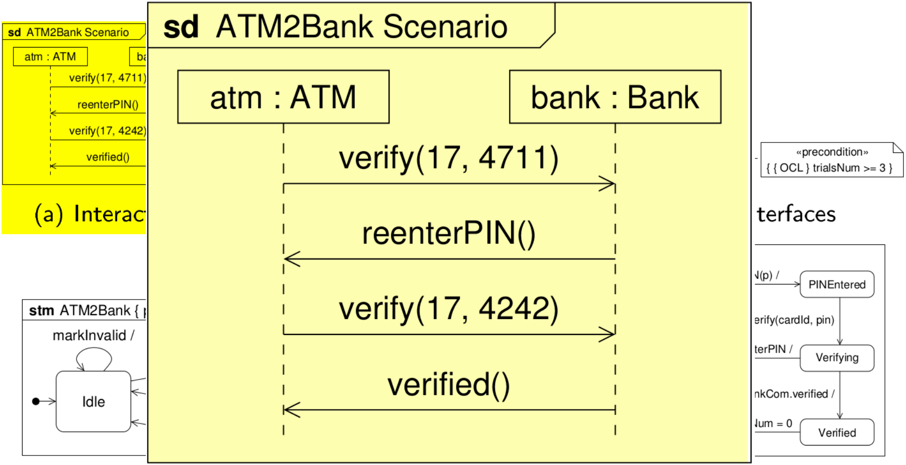


(d) Protocol state machine



(e) State machine

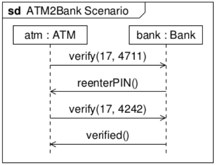
Multi-view consistency example: an ATM



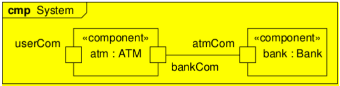
(d) Protocol state machine

(e) State machine

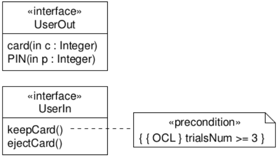
Multi-view consistency example: an ATM



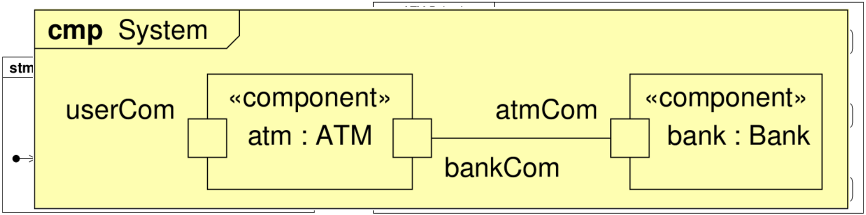
(a) Interaction



(b) Composite structure



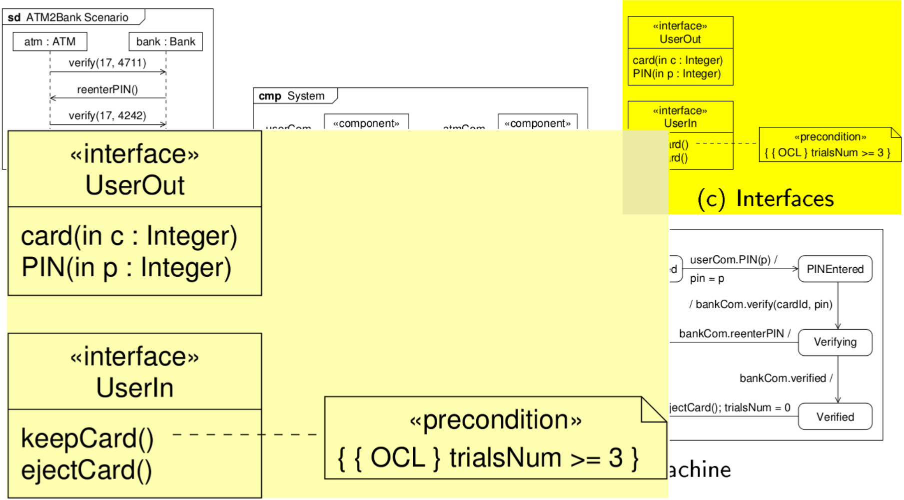
(c) Interfaces



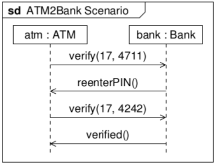
(d) Protocol state machine

(e) State machine

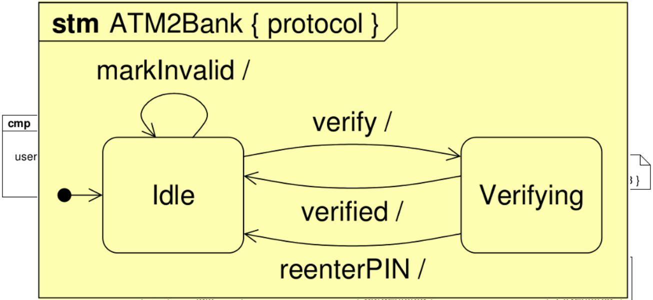
Multi-view consistency example: an ATM

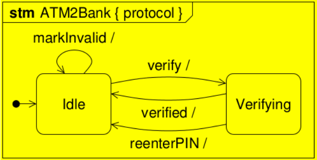


Multi-view consistency example: an ATM

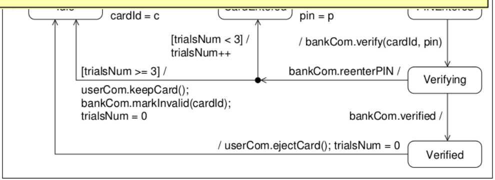


(a) Interaction



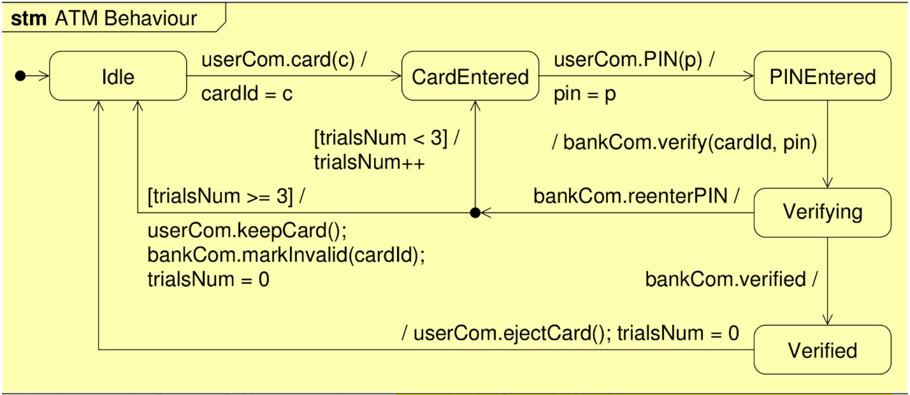


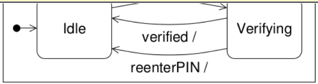
(d) Protocol state machine



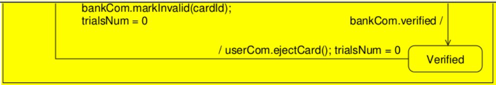
(e) State machine

Multi-view consistency example: an ATM





(d) Protocol state machine



(e) State machine

Approaches to Multi-View Consistency

Scope:

- s structural, static consistency checks
- b behavioural, dynamic consistency checks

Method:

- S system model
- D dynamic meta-modelling
- U universal logic
- T heterogeneous transformation

Coverage:

- ● supports at least a substantial subset of the diagram/sub-language type
- ◑ only supports a limited subset

Classification of UML1/OCL1 Approaches

Reference	CD	OD	SM	ID	AD	OCL	cons.	class.	Form./Tool
* Egyed [18, 19]	◑	◑	◑	◑			T	s	VIEWINTEGRA
* Große-Rhode [32, 33]	◑		◑	◑			S	b	transf. syst.
Reggio et al. [66]	◑		◑				U	(s/b)	CASL-LTL
McUmber, Cheng [56]	◑		◑				U	b	SPIN
* krtUML [14]	◑	◑	◑				S	s/b	symb. trans. syst.
Bernardi et al. [7]			◑	◑			U	(b)	Petri nets
* xUML [57]	●		●	◑	◑	◑	S	(s)	Exec. UML
* Küster et al. [23, 26]	◑	◑	◑	◑			U	b	CSP/FDR
* Hausmann et al. [25]			◑	◑			D	b	Graph transf.
Spanoudakis, Kim [70]	◑			◑			U	s	Dempster-Shafer
Litvak et al. [51]			●	◑			U	b	BVUML
Rasch, Wehrheim [64]	◑		◑				U	s/b	Z, CSP/FDR
* Wirsing, Knapp [75]	◑		◑	◑			T	s/b	univ. alg.
Kyas et al. [47]	◑		◑			◑	U	s/b	PVS
van der Straeten [71, 72]	◑		◑	◑			U	s	Desc. Logic
Amálio et al. [2]	◑	◑	◑				U	s	Z
Kim, Carrington [41]	◑		◑				U	s	Object-Z
Diethers, Huhn [16]			◑	◑			U	b	UPPAAL
Yang et al. [76]	◑			●			U	s	rCOS
Yeung [78]	◑		◑				U	b	CSP, B

Classification of UML1/OCL1 Approaches

Reference	CD	OD	SM	ID	AD	OCL	cons.	class.	Form./Tool
* Egyed [18, 19]	◑	◑	◑	◑			T	s	VIEWINTEGRA
* Große-Rhode [32, 33]	◑		◑	◑			S	b	transf. syst.
Reggio et al. [66]	◑		◑				U	(s/b)	CASL-LTL
McUmber, Cheng [56]	◑		◑				U	b	SPIN
* krtUML [14]	◑	◑	◑				S	s/b	symb. trans. syst.
Bernardi et al. [7]			◑	◑			U	(b)	Petri nets
* xUML [57]	●		●	◑	◑	◑	S	(s)	Exec. UML
* Küster et al. [23, 26]	◑	◑	◑	◑			U	b	CSP/FDR
* Hausmann et al. [25]			◑	◑			D	b	Graph transf.
Spanoudakis, Kim [70]	◑			◑			U	s	Dempster-Shafer
Litvak et al. [51]			●	◑			U	b	BVUML
Rasch, Wehrheim [64]	◑		◑				U	s/b	Z, CSP/FDR
* Wirsing, Knapp [75]	◑		◑	◑			T	s/b	univ. alg.
Kyas et al. [47]	◑		◑			◑	U	s/b	PVS
van der Straeten [71, 72]	◑		◑	◑			U	s	Desc. Logic
Amálio et al. [2]	◑	◑	◑				U	s	Z
Kim, Carrington [41]	◑		◑				U	s	Object-Z
Diethers, Huhn [16]			◑	◑			U	b	UPPAAL
Yang et al. [76]	◑			●			U	s	rCOS
Yeung [78]	◑		◑				U	b	CSP, B

Classification of UML2/OCL2 Approaches

Reference	CD	OD	CMP	CSD	SM	ID	AD	OCL	cons.	class.	Form./Tool
Lam, Padget [49]					◑	◑			U	b	π-calculus
Long et al. [54]	◑					◑			U	s	rCOS
Lucas et al. [55]	◑					◑			U	s	Maude
Okalas et al. [61]	◑				◑				U	s/b	B
Rasch, Wehrheim [65]	◑		◑		◑	◑			U	s/b	Z, CSP/FDR
Wang et al. [74]					◑	◑			U	b	LTSA
Bellur, Vallieswaran [6]	◑		◑		◑	◑			U	s	meta-model
Li et al. [50, 53]	◑	●			◑	●			U	s/(b)	UTP
O'Keefe [62]	◑				◑	◑			U	b	Dynamic Logic
Shinkawa [68]	◑				◑	◑	◑		U	b	CPN
Yao, Shatz [77]					◑	◑			U	b	Petri nets
Zhao et al. [79]					◑	◑			U	b	SPIN
Anastasakis et al. [3]	●							◑	U	s/(b)	Alloy
* Gogolla et al. [29]	●	●			◑	◑		●	U	s/(b)	USE
Knapp, Wuttke [43]	◑				●	●			U	b	Hugo/RT
Sapna, Mohanty [67]	◑				◑	◑	◑		U	s	SQL
Brændshøi [8]					◑	◑			U	b	impl.
Banerjee et al. [4, 5]	◑				●				U	b	Rhapsody/LTL

Classification of UML2/OCL2 Approaches

Reference	CD	OD	CMP	CSD	SM	ID	AD	OCL	cons.	class.	Form./Tool
Lam, Padget [49]					◑	◑			U	b	π-calculus
Long et al. [54]	◑					◑			U	s	rCOS
Lucas et al. [55]	◑					◑			U	s	Maude
Okalas et al. [61]	◑				◑				U	s/b	B
Rasch, Wehrheim [65]	◑		◑		◑	◑			U	s/b	Z, CSP/FDR
Wang et al. [74]					◑	◑			U	b	LTSA
Bellur, Vallieswaran [6]	◑		◑		◑	◑			U	s	meta-model
Li et al. [50, 53]	◑	●			◑	●			U	s/(b)	UTP
O'Keefe [62]	◑				◑	◑			U	b	Dynamic Logic
Shinkawa [68]	◑				◑	◑	◑		U	b	CPN
Yao, Shatz [77]					◑	◑			U	b	Petri nets
Zhao et al. [79]					◑	◑			U	b	SPIN
Anastasakis et al. [3]	●							◑	U	s/(b)	Alloy
* Gogolla et al. [29]	●	●			◑	◑		●	U	s/(b)	USE
Knapp, Wuttke [43]	◑				●	●			U	b	Hugo/RT
Sapna, Mohanty [67]	◑				◑	◑	◑		U	s	SQL
Brændshøi [8]					◑	◑			U	b	impl.
Banerjee et al. [4, 5]	◑				●				U	b	Rhapsody/LTL

Classification of UML2/OCL2 Approaches, cont'd

Reference	CD	OD	CMP	CSD	SM	ID	AD	OCL	cons.	class.	Form./Tool
* Cengarle et al. [17]	●					◑		◑	T	s/b	institutions
* Alanazi [1]					◑	◑			U	(b)	impl.
Hammal [49]					◑	◑			U	(b)	Petri nets
Laleau, Polack [72]	◑				◑	◑			U	s	meta-model
* Broy et al. [14, 15]	●	●			●	●			S	s/b	set theory
* Kuske et al. [67]	◑	◑			◑	◑			U	(s/b)	graph. transf.
* Grönniger [46]	◑	◑			◑	◑		◑	S	s/b	Isabelle/HOL
Nimiya et al. [89]					◑	◑			U	b	Alloy
Khai [60]	◑					◑			U	s	Prolog
Ober, Dragomir [90]			●	●	◑				U	s/b	OMEGA2
Puczynski [99]	◑				◑	◑			U	s/b	impl.
Gerlinger et al. [41]	●			●			●		U	s	Common Logic
El Miloudi et al. [30, 31]	◑				◑	◑			U	s	Z
Khan, Porres [61]	●	●			◑			◑	U	s	Desc. Logic
* fUML [92]	●						●		S	s/b	Common Logic

Classification of UML2/OCL2 Approaches, cont'd

	Reference	CD	OD	CMP	CSD	SM	ID	AD	OCL	cons.	class.	Form./Tool
*	Cengarle et al. [17]	●					◑		◑	T	s/b	institutions
*	Alanazi [1]					◑	◑			U	(b)	impl.
	Hammal [49]					◑	◑			U	(b)	Petri nets
	Laleau, Polack [72]	◑				◑	◑			U	s	meta-model
*	Broy et al. [14, 15]	●	●			●	●			S	s/b	set theory
*	Kuske et al. [67]	◑	◑			◑	◑			U	(s/b)	graph. transf.
*	Grönniger [46]	◑	◑			◑	◑		◑	S	s/b	Isabelle/HOL
	Nimiya et al. [89]					◑	◑			U	b	Alloy
	Khai [60]	◑					◑			U	s	Prolog
	Ober, Dragomir [90]			●	●	◑				U	s/b	OMEGA2
	Puczynski [99]	◑				◑	◑			U	s/b	impl.
	Gerlinger et al. [41]	●			●			●		U	s	Common Logic
	El Miloudi et al. [30, 31]	◑				◑	◑			U	s	Z
	Khan, Porres [61]	●	●			◑			◑	U	s	Desc. Logic
*	fUML [92]	●						●		S	s/b	Common Logic

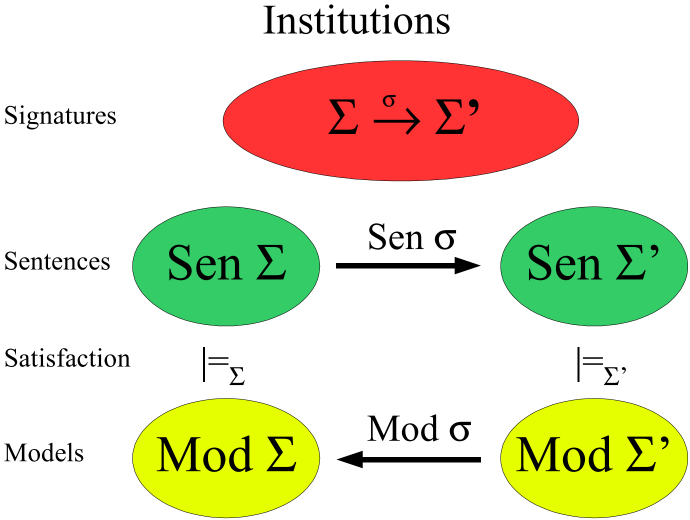
Problems with Current Approaches to Multi-View Consistency

- only part of the UML diagram types are covered (5 of 14)
- "universal logic" approach is predominant, but fails to handle complexity
  - each new feature requires a substantial extension of the model
- "heterogeneous transformation approaches" fall short in leveraging the translations to build up a distributed system of viewpoints

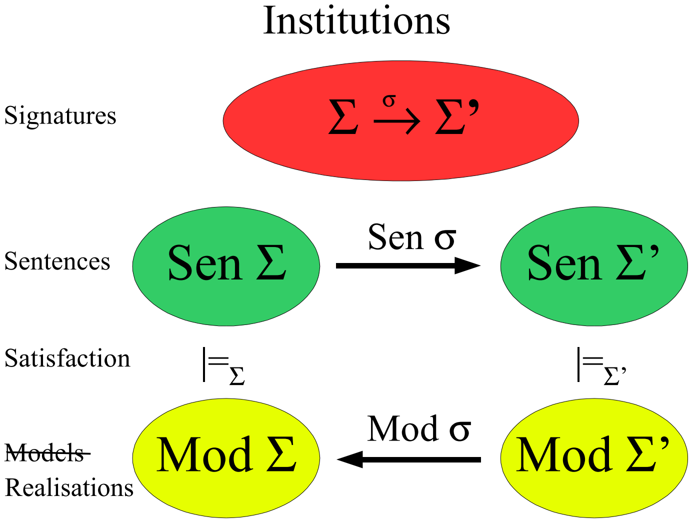
A New Approach to Multi-View Consistency

- follows the heterogeneous transformation paradigm
- use of institutions in order to capture semantics of different diagram types "as-is"
- use of institution (co)morphisms for transformations
- use of distributed heterogeneous specifications for capturing the multi-viewpoint nature
  - distributed specifications
    = diagrams in the category of heterogeneous specifications
    = networks in DOL
- goal: capture all semantically relevant UML diagram types with static and dynamic consistency

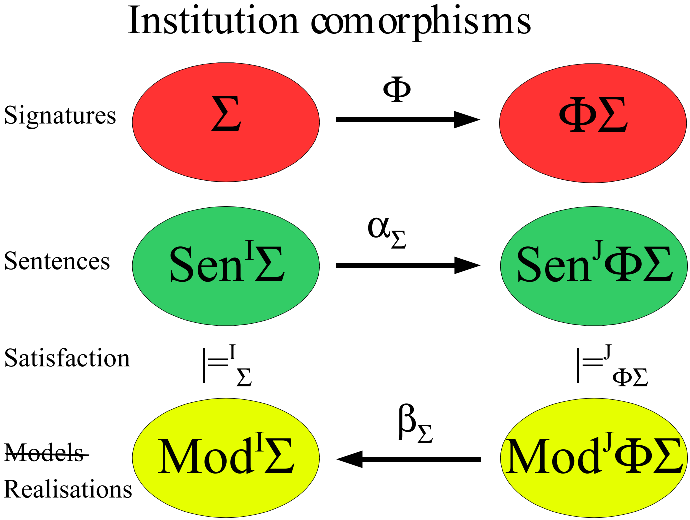
DOL Semantic Foundations: Institutions



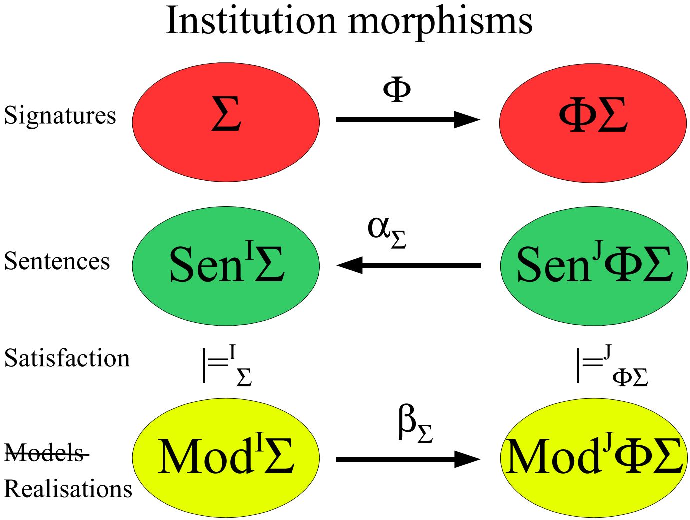
DOL Semantic Foundations: Institutions



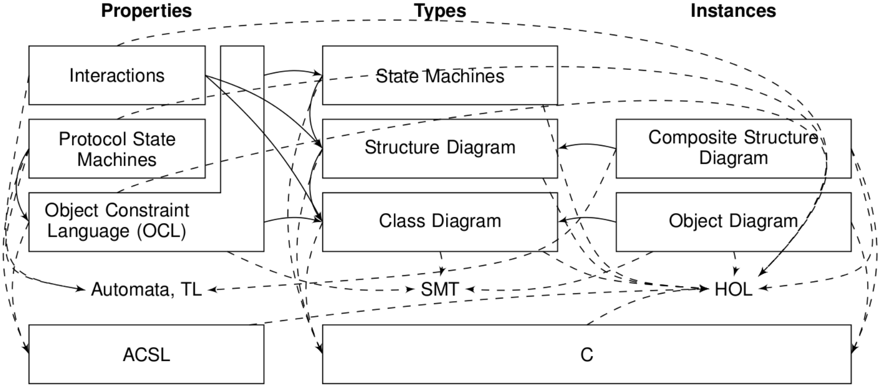
Institution comorphisms (embeddings, encodings)



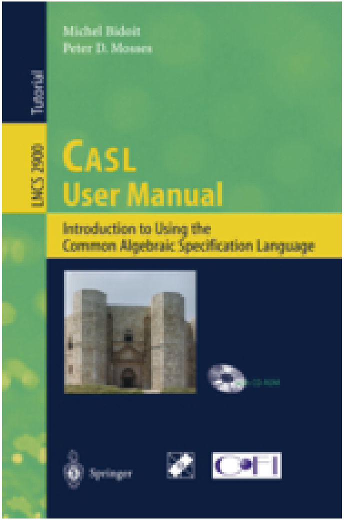
Institution morphisms (projections)

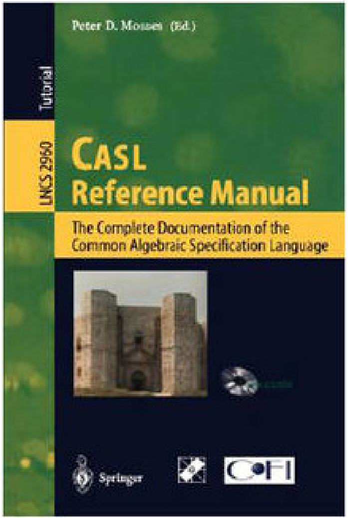


An Initial Graph of Institutions for UML2/OCL2



Structuring of Specifications in an Arbitrary Institution





CASL arose from the IFIP WG 1.3

The IFIP WG1.3 Referees' Report on CASL reviewed the initial design proposal for CASL (version 0.97, May 1997); the CASL Designers' final response to the referees indicated how the points raised in the report had influenced the final design (version 1.0.1-DRAFT, June 2000, approved and released as version 1.0.1 in March 2001).
The IFIP WG1.3 reviewers consisted of Hartmut Ehrig (Coordinator), José Meseguer, Ugo Montanari, Fernando Orejas, Peter Padawitz, Francesco Parisi-Presicce, Martin Wirsing, and Uwe Wolter.

CASL reference manual, p.4

DOL – An OMG standard

- DOL = Distributed Ontology, Model, and Specification Language
- OMG Specification, Beta 1 released
- Has been approved by OMG
- Now in finalization process
- DOL has a fully formal semantics
- OMG documents are freely available
- DOL is open for your ideas, so join us!

OBJECT MANAGEMENT GROUP®

Origins of DOL:

- Ontology modularisation, alignment, distributed descr. logics
- Multi-viewpoint models
- structured specification over an arbitrary institution: CLEAR, ASL, CASL, ...

Overview of DOL: Toolkit in Summary

1. OMS (ontologies, models, specifications)
    - basic OMS, written as-is (flattenable)
    - references to named OMS (by URL)
    - extensions, unions, translations (flattenable)
    - reductions, minimization, maximization (elusive)
    - approximations, module extractions, filterings (flattenable)
    - combinations of networks (flattenable)
2. OMS mappings (between OMS)
    - interpretations, refinements, alignments, . . .
3. OMS networks (based on OMS and mappings)
4. OMS libraries (based on OMS, mappings, networks)
    - OMS definitions (giving a name to an OMS)
    - definitions of interpretations, refinements, alignments
    - definitions of networks, entailments, equivalences, . . .

OMS in DOL

```
OMS ::= ⟨I, Σ, Γ⟩ %% basic OMS in institution I
      | OMS then ⟨I, Σ, Γ⟩ %% extension of OMS
      | OMS and OMS  %% intersection of realisation classes
      | OMS with σ   %% σ : signature morphism
      | OMS with translation ρ %% ρ : institution comorphism
      | OMS hide Σ     | OMS reveal Σ
      | OMS hide along μ %% μ : institution morphism
      | OMS remove Σ   | OMS extract Σ
      | OMS forget Σ   | OMS keep Σ
      | OMS keep I
      | OMS reject Σ   | OMS select Σ
      | free { OMS }         %% initial semantics
      | minimize { OMS }     %% McCarthy's circumscription
      | logic l : { OMS }
      | combine Network      %% colimit of diagram
```

UML multi-view consistency through DOL networks: sequence diagrams and class diagrams

```
model ATM_Bank_Interaction_cd =
  ATM_Bank_Interaction hide along sd2cd
end
```

Semantics of **hide along** sd2cd:
Projection along institution morphism

```
refinement r1 =
   { User_Interface reveal ATM_Bank_Interaction_cd }
      refined to ATM_Bank_Interaction_cd
end
```

Semantics of **refined to**: Specification morphism

State machines

```
model ATM_stm =
  User_Interface with translation cd2stm
then
  ATM_stm_definition
end

model Bank_stm =
  User_Interface with translation cd2stm
then
  Bank_stm_definition
end
```

Semantics of **with translation** cd2stm:
Translation along institution comorphism

Composite Structure Diagram

```
model System =
  ATM_stm with translation stm2cmp with cid |-> atm
and
  Bank_stm with translation stm2cmp with cid |-> bank
then
  cmp
end
```

Semantics of **and**:
Union of signatures, intersection of classes of realisations

State machines vs. sequence diagram

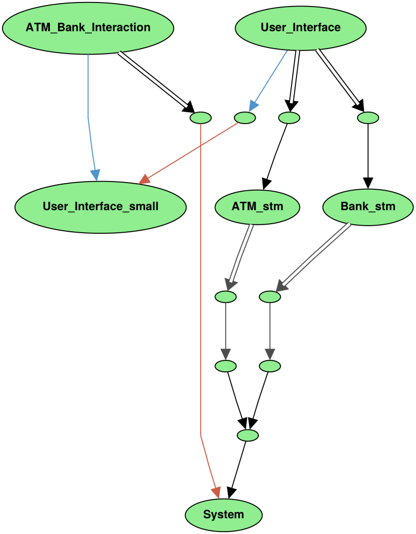
```
%% the sequence diagram can be realised by
%% the two state machines
%% as combined by the composite structure diagram
refinement r2 =
  ATM_Bank_Interaction refined to
    { System hide along cmp2sd }
end
```

The network

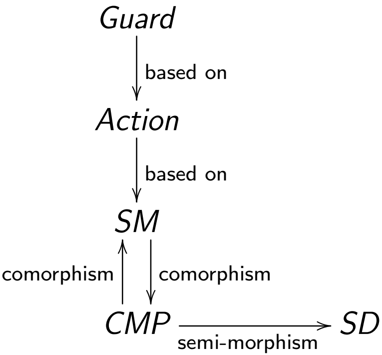
```
%% multi-view consistency
network N = %consistent
  User_Interface, ATM_stm, Bank_stm, System,
  ATM_Bank_Interaction, r1, r2
end
```

Realisation of a network = family of realisations, one for each node, that is compatible along the edges

The Network in the Heterogeneous Tool Set (Hets)



Overview of institutions and (co)morphisms



SM: state machines
CMP: composite structure diagrams
SD: sequence diagrams

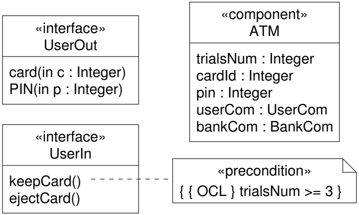
Institution of Guards

Capturing mild requirements on guard language

- Signatures: Sets V (variables/attributes) and functions υ
  - variables abstracting attributes and association ends
- Structures: valuations $\omega : V \to \mathrm{Val}$, reducts just composition
  - Val unspecified domain of values
- Sentences functor G and satisfaction relation $\models_V$ unspecified
- Satisfaction condition $\omega' \circ \upsilon \models_V g \iff \omega' \models_{V'} \upsilon(g)$

Example: For ATM

- Variables: trialsNum, ...
- Sentences: true, trialsNum $< n$, trialsNum $== n$, ...

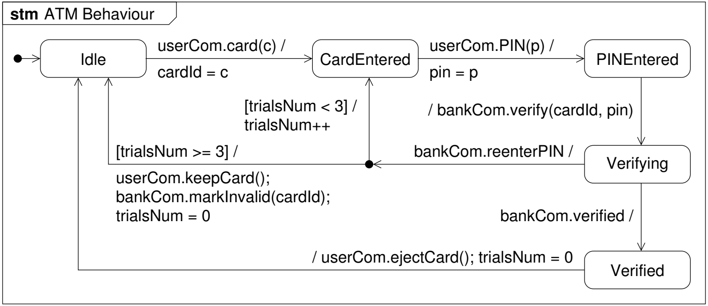


Institution of Actions

Parametric in institution of guards

- Signatures $H = (A, M, V)$ with actions A, messages M, variables/attributes V
- Structures $\Omega \subseteq (V \to \mathrm{Val}) \times (A \times \wp(M)) \times (V \to \mathrm{Val})$
  - $\omega \xrightarrow[\Omega]{a,\overline{m}} \omega'$ for "action a leads from state ω to state ω' with messages $\overline{m}$"
  - reduct along $\eta : H \to H'$:
    $\{\omega_1|_{\eta_V} \xrightarrow[\Omega'|_\eta]{a,\eta_M^{-1}(\overline{m})} \omega_2|_{\eta_V} \mid \omega_1 \xrightarrow[\Omega']{\eta_A(a),\overline{m}} \omega_2\}$
- Sentences $g_{\mathrm{pre}} \to [a]\overline{m} \rhd g_{\mathrm{post}}$
- Satisfaction relation $\Omega \models^{\mathrm{Act}}_H g_{\mathrm{pre}} \to [a]\overline{m} \rhd g_{\mathrm{post}}$ iff for all $\omega \in (V \to \mathrm{Val})$
  - if $\omega \models g_{\mathrm{pre}}$ and $\omega \xrightarrow[\Omega]{a,\overline{m}'} \omega'$, then $\omega' \models g_{\mathrm{post}}$ and $\overline{m} \subseteq \overline{m}'$

Institution of Actions: Example



- true $\rightarrow$
  [userCom.ejectCard(); trialsNum = 0]{userCom.ejectCard()} $\triangleright$ trialsNum == 0
- trialsNum == n $\rightarrow$ [trialsNum++]$\emptyset$ $\triangleright$ trialsNum == n+1

Institution SM of Behavioural State Machines (1)

Built over institution of actions

- action signature $H = (A, M, V)$, action structure Ω over H
- Signatures $\Sigma = (E, F, S)$ with external events E, completion events F, control states S
  - morphisms injective renamings
- Structures $\Theta = (I, \Delta)$ with

  initial configurations $I \subseteq \wp(V \to \mathrm{Val}) \times S$

  transition relation $\Delta \subseteq C \times \underbrace{\wp(M)}_{\text{messages}} \times C$

  $$\text{configurations } C = \underbrace{(V \to \mathrm{Val})}_{\text{data state}} \times \underbrace{\wp(E \cup F)}_{\text{event pool}} \times \underbrace{S}_{\text{control state}}$$

  - reducts deleting events from event pool not present in pre-image of signature morphism

Institution SM of Behavioural State Machines (2)

- Sentences $\varphi = (s_0, T)$

  start control state $s_0 \in S$

  transitions $T \subseteq S \times (E \cup F) \times (G(V) \times A \times \wp(F)) \times S$

- $s \xrightarrow[T]{p[g]/a,\overline{f}} s'$ for "transition from state s with trigger p, guard g, action a, completions $\overline{f}$ to state s'"

Example: For ATM

$$(\mathsf{Idle}, \{\mathsf{Idle} \xrightarrow[\mathsf{ATM}]{\mathsf{card[true]/cardId = c},\emptyset} \mathsf{CardEntered},$$

$$\mathsf{CardEntered} \xrightarrow[\mathsf{ATM}]{\mathsf{PIN[true]/pin = p,PINEntered}} \mathsf{PINEntered},$$

$$\mathsf{PINEntered} \xrightarrow[\mathsf{ATM}]{\mathsf{PINEntered[true]/bank.verify(cardId,\ pin)},\emptyset} \mathsf{Verifying},$$

$$\mathsf{Verifying} \xrightarrow[\mathsf{ATM}]{\mathsf{reenterPIN[trialsNum < 3]/trialsNum{+}{+}},\emptyset} \mathsf{CardEntered}, \ldots\})$$

Institution SM of Behavioural State Machines (3)

- Satisfaction relation $(I, \Delta) \models^{\mathrm{SM}(H,\Omega)}_{\Sigma} (s_0, T)$ iff $\pi_2(I) = s_0$ and

$$(\omega, p :: \overline{p}, s) \xrightarrow[\Delta]{\overline{m} \setminus E} (\omega', \overline{p} \lhd ((\overline{m} \cap E) \cup \overline{f}), s') \quad \text{if}$$

$$\exists s \xrightarrow[T]{p[g]/a, \overline{f}} s' \,.\, \omega \models g \wedge \omega \xrightarrow[\Omega]{a, \overline{m}} \omega'$$

"transition with event p enabled in control state s, produces messages $\overline{m}$, some targeted to current machine, completions $\overline{f}$"

$$(\omega, p :: \overline{p}, s) \xrightarrow[\Delta]{\emptyset} (\omega, \overline{p}, s) \quad \text{if} \quad \forall s \xrightarrow[T]{p'[g]/a, \overline{f}} s' \,.\, p \neq p' \vee \omega \not\models g$$

"otherwise event discarded"

Flat State Machine Institution

Given an institution of guards, we flatten the institutions $\mathsf{SM}(H, \Omega)$ into a single institution SM:

Signatures: $\langle H, \Sigma \rangle$: action signature H, state machine signature Σ

Sentences: control transition relations (over H and Σ); dynamic logic formulas (over H)

Structures: $\langle \Omega, \Theta \rangle$ for $\Omega \in \mathrm{Str}_{Act}(H)$, $\Theta \in \mathrm{Str}_{SM}{}^{(H,\Omega)}(\Sigma)$

Reducts: $\langle \Omega', \Theta' \rangle|_{(\eta,\sigma)} = \langle \Omega'|_\eta, \Theta'|_\sigma|_\eta \rangle$ where $\Theta''|_\eta = (I_{\Theta''}, \{c''_1, \eta_M^{-1}(\overline{m}''), c''_2) \mid (c''_1, \overline{m}'', c''_2) \in \Delta_{\Theta''}\})$.

Satisfaction: is inherited

Institution CMP of Composite Structure Diagrams

Signatures $(C, S, P, Conn)$ with

- set of *components* C,
- *state machine assignment* $S : C \to |\mathrm{Sig}_{SMFlat}|$,
- *ports* $(c, pn) \in P$ ($c \in C$, pn a name) and
- *connectors* $(p_1, p_2) \in Conn$, where $p_1, p_2 \in P$.

Morphisms $\sigma = (\sigma_C, \sigma_S, \sigma_P) : (C, S, P, Conn) \to (C', S', P', Conn')$ with $\sigma_C : C \to C'$, $\sigma_S(c) : S(c) \to S'(\sigma_C(c))$ $(c \in C)$, $\sigma_P : P \to P'$, such that $\sigma_P \times \sigma_P(Conn) \subseteq Conn'$.

Sentences (c, φ) with $c \in C, \varphi \in \mathrm{Sen}_{SM}(S(c))$

Realisations $(c : C) \stackrel{R}{\to} \mathrm{Str}_{SM}(S(c))$

Reduct $(c' : C') \stackrel{R'}{\to} \mathrm{Str}_{SM}(S'(c'))$ is reduced to $(c : C) \stackrel{R}{\mapsto} R'(\sigma_c(c))|_{\sigma_{SM}(c)}$

Satisfaction $R \models (c, \varphi)$ iff $R(c) \models \varphi$

Comorphism SM $\mapsto$ CMP

Signatures $\Sigma \mapsto (\{cid\}, S, \emptyset, \emptyset)$ with $S(cid) = \Sigma$

Sentences $\varphi \mapsto (cid, \varphi)$

Realisations $R \mapsto R(cid)$

Comorphism CMP $\mapsto$ SM

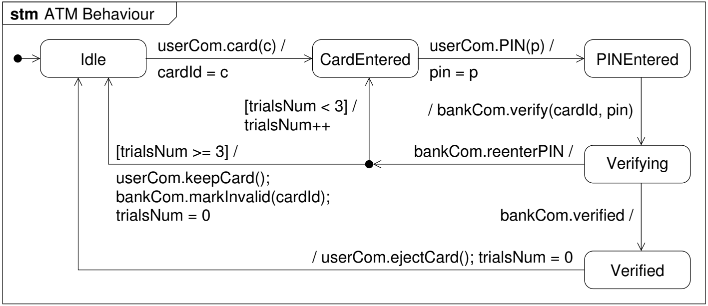
Signatures $(C, S, P, Conn) \mapsto$
$(\coprod_{c \in C} A_{S(c)}, \coprod_{c \in C} M_{S(c)}, \bigcup_{c \in C} V_{S(c)}, \coprod_{c \in C} E_{S(c)},$
$\coprod_{c \in C} F_{S(c)}, \prod_{c \in C} S_{S(c)})$

Sentences $(c, \varphi) \mapsto$ "make moves according to φ in component c"

Realisations Family R of transition relations $\mapsto$
Interleaved transition relation

$$(\omega, p :: (\overline{p}_{c_1} \cup \cdots \cup \overline{p}_{c_n}), (s_{c_1}, \ldots, s_{c_n})) \xrightarrow[\Delta_\Theta]{\overline{m} \setminus E_{conn}}$$
$$(\omega', (\overline{p}_{c_1} \cup \cdots \cup \overline{p}_{c_n} \lhd ((\overline{p}' \cup p_{i_{1-k}}.\overline{m}) \cap E_{conn}), (s'_{c_1}, \ldots, s'_{c_n}))$$
iff $conn = ((c_{i_1}, p_{i_1}), (c_{i_2}, p_{i_2})) \in Conn\,.\exists k \in \{1, 2\}$
$$(\omega|_{V_{H_{c_{i_k}}}}, p :: \overline{p}_{c_{i_k}}, s_{c_{i_k}}) \xrightarrow[\Delta_{\Theta_{R(c_{i_k})}}]{p_{i_k}.\overline{m}} (\omega'|_{V_{H_{c_{i_k}}}}, \overline{p}_{c_{i_k}} \lhd \overline{p}', s'_{c_{i_k}}) \wedge$$
$$\forall j \in \{1, \ldots, n\} \setminus \{i_k\}\,.\,(\omega|_{V_{H_{c_j}}}, \overline{p}_{c_j}, s_{c_j}) = (\omega'|_{V_{H_{c_j}}}, \overline{p}_{c_j}, s'_{c_j})$$

Comorphism CMP $\mapsto$ SM: example



$$(\omega, \{PINEntered\} \cup \emptyset, (PINEntered, s_{Bank}))$$

$$\emptyset \Big\downarrow \Delta_\Theta$$

$$(\omega, \emptyset \cup \{bankCom.verify(17, 4711)\}, (Verifying, s_{Bank}))$$

where $\omega = \{cardId \mapsto 17, pin \mapsto 4711\}$

Institution SD of Sequence Diagrams

Signatures (L, M) with *lifelines* L and *messages* M.

Morphisms $\sigma = (\sigma_L, \sigma_M) : (L, M) \to (L', M')$ with $\sigma_L : L \to L'$ and $\sigma_M : M \to M'$.

Sentences

$$F ::= \mathrm{skip} \mid \mathrm{snd}(s, r, m) \mid \mathrm{rcv}(s, r, m) \mid \mathrm{strict}(F_1, F_2) \mid \mathrm{seq}(F_1, F_2) \mid \mathrm{par}(F_1, F_2) \mid \mathrm{alt}(F_1, F_2)$$

Realisations sets of event occurrence traces $T \subseteq \mathscr{E}(L, M)^*$

Events $\mathscr{E}(L, M)$:

- $snd(o_s, o_r, n)$ ("object $o_s \in L$ sends invocation $n \in M$ to $o_r \in L$")
- $rcv(o_s, o_r, n)$ ("object o_r receives invocation n from o_s").

Reduct along $\sigma : (L, M) \to (L', M')$: taking pre-image

Auxiliary notions

Objects *active* in an event occurrence:

$$\alpha(snd(o_s, o_r, c)) = \{o_s\}$$
$$\alpha(rcv(o_s, o_r, c)) = \{o_r\}$$

Conflict:

$$e_1 \asymp e_2 \iff \alpha(e_1) \cap \alpha(e_2) \neq \emptyset$$

Operations on traces:

$$\langle\rangle \mathbin{;} t_2 = \{t_2\}$$
$$(e :: t_1) \mathbin{;} t_2 = \{e :: t \mid t \in t_1 \mathbin{;} t_2\}$$

$$\langle\rangle \mathbin{;_\asymp} t_2 = \{t_2\}$$
$$t_1 \mathbin{;_\asymp} \langle\rangle = \{t_1\}$$
$$(e_1 :: t_1) \mathbin{;_\asymp} (e_2 :: t_2) = \{e_1 :: t \mid t \in t_1 \mathbin{;_\asymp} (e_2 :: t_2)\} \cup \{e_2 :: t \mid t \in (e_1 :: t_1) \mathbin{;_\asymp} t_2,\ \neg(e_1 \asymp e_2)\}$$

Auxiliary notions (cont'd)

$$\begin{aligned}
\langle\rangle \parallel t_2 &= \{t_2\} \\
t_1 \parallel \langle\rangle &= \{t_1\} \\
(e_1 :: t_1) \parallel (e_2 :: t_2) &= \{e_1 :: t \mid t \in t_1 \parallel (e_2 :: t_2)\} \cup \\
&\quad\ \{e_2 :: t \mid t \in (e_1 :: t_1) \parallel t_2\}
\end{aligned}$$

Lifting to sets of traces

$$T_1 \diamond T_2 = \bigcup\{t_1 \diamond t_2 \mid t_1 \in T_1,\ t_2 \in T_2\}$$

Satisfaction relation

Traces of a formula

$$\mathscr{P}(\mathsf{skip}) = \{\langle\rangle\}$$
$$\mathscr{P}(\mathsf{snd}(l_s, l_r, m) = \{\langle snd(l_s, l_r, m)\rangle\}$$
$$\mathscr{P}(\mathsf{rcv}(l_s, l_r, m) = \{\langle rcv(l_s, l_r, m)\rangle\}$$
$$\mathscr{P}(\mathsf{strict}(F_1, F_2)) = \mathscr{P}(F_1)\ ;\ \mathscr{P}(F_2)$$
$$\mathscr{P}(\mathsf{seq}(F_1, F_2)) = \mathscr{P}(F_1)\ ;_{\asymp}\ \mathscr{P}(F_2)$$
$$\mathscr{P}(\mathsf{par}(F_1, F_2)) = \mathscr{P}(F_1) \parallel \mathscr{P}(F_2)$$
$$\mathscr{P}(\mathsf{alt}(F_1, F_2)) = \mathscr{P}(F_1) \cup \mathscr{P}(F_2)$$

Satisfaction relation $T \models_{\Sigma} F \iff \mathscr{P}(F) \cap T \neq \emptyset$

Semi-morphism CMP $\mapsto$ SD

Signatures $(C, S, P, Conn) \mapsto (C, M)$ (components $\mapsto$ lifelines)
where $M = \bigcup_{((c_1,p_1),(c_2,p_2)) \in Conn} M(p_1, p_2)$

Realisations family of transition systems $\mapsto$ traces over the interleaved product

Approaches to Consistency Checking

- encoding into some “universal” logic $\Rightarrow$ realisation finders (in logic speak: model finders)
- incremental constructions of realisations
- use of CASL architectural specifications
  - decompose large consistency problems into smaller ones

DOL Resources

- `http://dol-omg.org` Central page for DOL
- `http://hets.eu` Analysis and Proof Tool Hets, speaking DOL
- `http://ontohub.org` Ontohub web platform, speaking DOL
- `http://ontohub.org/dol-examples` DOL examples
- `https://ontohub.org/esslli-2016`
  ESSLLI repository of DOL examples
- `http://ontoiop.org` Initial standardization initiative

Conclusions and Future Work

- UML2/OCL2 is a language for multi-viewpoint models
- detection of consistency is important for avoiding costly redesign
- classified 53 existing approaches
  - best approaches partly cover 5 diagram types
- institutions and DOL networks provide a new approach
  - goal is to cover all semantically relevant diagram types

Future work:

- formalise more UML diagram types as institutions
- formalise transformations as institution (co)morphisms
- integration into Heterogeneous Tool Set (Hets)
  - interfacing with suitable proof and model finding tools
- development of consistency strategies

Paper available at `http://arxiv.org/abs/1610.03960`