# Institution Theory

## introduction

Răzvan Diaconescu

Institutul de Matematică “Simion Stoilow”, Romania

UNILOG 2010, Lisbon

# Outline

1. Introduction
2. Definition
3. Logics as Institutions

# Institution theory (INS) and universal logic

INS is a major current trend of universal logic since it has the spirit of universal logic and also fullfills its aims.

In short this means:

Non-substantialist abstract approach to logic phenomenon.

It is major trend in the sense of currently being rather strongly developed, both mathematically and in terms of volume of activity.

# Origins of institution theory

- Mathematical logic, esp. (abstract) model theory.
- Category theory.
- Computer science, esp. algebraic specification.

INS started from within algebraic specification around 1980 with the seminal paper of Goguen and Burstall,

- in response to the explosion in the population of logics used in formal specification,
- achieved uniform abstract development of concepts and results.

Since then it has become the most foundational mathematical structure of formal specification, and also with important applications to other CS areas.

# What is institution theory?

This can be answered on three different levels.

1. A fully abstract categorical model theoretic oriented formalization for the (informal) concept of logical system, including syntax, semantics, and satisfaction between them.
2. A (well developed) body of methods, concepts, results for doing logic and model theory independently of any concrete logical or model theoretic structure.
3. A ‘top-down’ non-substantialist way of thinking about logic and model theory.

# Contributions of INS to logic I

- (Meta-)mathematical clarification of the concept of logic.

### Thesis

**Each “logic” can be formalized as institution!**

- Top-down development methodology, at the most appropriate level of abstraction.
  - *Concepts* come naturally as presumptive features that a “logic” might exhibit or not.
  - *Hypotheses* are kept as general as possible and introduced on a by-need basis.
  - *Results* and *proofs* are modular and easy to track down, despite sometimes very deep content.

# Contributions of INS to logic II

- Deeper understanding of phenomena not suffocated by the (often unimportant) details of the actual logic, but guided by structurally clean causality.
- Uniform general abstract development of logic and model theory, providing substantial model theory for many old or new logics that did not have one.
- Clarification of scope of important concepts and results (e.g. axiomatizability, interpolation, completeness, etc.),
- Redesign of important fundamental logic concepts (e.g. interpolation, definability, etc.),

# Contributions of INS to logic III

- Clarification of some causality relationships between model theoretic phenomena including the demounting of some deep theoretical preconceptions (e.g. some logics not having interpolation, etc.).
- New results even in well studied concrete frameworks (e.g. interpolation, definability, completeness, etc.),
- Easier access to highly non-trivial well established results (e.g. Keisler-Shelah isomorphism Thm.).

# Contributions of INS to logic IV

- A clear and solid mathematical framework for doing logic ‘by translation’ via the so-called ‘institution co-morphisms’. It allows for rigorous exporting of concepts, methods, tools, etc. between logics.
- Approach to logic combination by internalizing logical concepts to abstract institutions.
- Mathematical elegance.

# Contributions of INS to computer science I

- Foundations of algebraic/formal specification theory and practice.
  - It has become standard to base the definition of specification languages upon logic systems captured as institutions such that all the language constructs are reflected rigorously as mathematical entities in the respective institutions (e.g. CASL, CafeOBJ, etc.).
  - Foundations for *heterogeneous* specifications via the so-called *Grothendieck institution* construction.

# Contributions of INS to computer science II

- Uniform general development of concepts and results thus applicable to a wide range of systems based upon various logics (formalized as institutions).
    - This is especially the case of structured (in-the-large) specification/programming.
- General approach to the semantics of declarative programming paradigms,
    - Esp. logic programming (with constraints also).
- Foundational role in other CS areas that involve logic in a significant way, e.g. ontologies, cognitive semantics, concurrency, etc.

# Comparison to other abstract model theoretic approaches

### Conventional ‘abstract’ model theory (Barwise, Feferman, etc.)

Concerned with extensions of conventional logic, abstracts only sentences and satisfaction, leaving signatures and models conventional.

### Categorical model theories (Makkai, Andreka-Nemeti, etc.)

Models are abstract categorical objects, however satisfaction not axiomatized, but rather defined as cone injectivity. Also single-signature framework, aspect leading to severe methodological limitations.

# Current status of development I

- Myriad of logics formalized as institutions.
- Large body of specification theory concepts and results at the level of abstract institutions.
- The most important model theory methods lifted to abstract institutions:
  - method of diagrams,
  - method of ultraproducts,
  - saturated models,
  - forcing.
- Internal logic, capture in abstract institutions
  - atomic sentences, via so-called *basic* sentences,
  - (Boolean) connectives,

# Current status of development II

  - general quantifiers, including first-order via so-called *representable* signature morphism,
  - general substitutions.
- Large body of model theory results at the level of abstract institutions:
  - fundamental ultraproducts theorem (Łos),
    - compactness theorem,
  - Birkhoff-style axiomatizability (Birkhoff institution),
  - existence and uniqueness of saturated models,
    - general version of Keisler-Shelah isomorphism theorem,
  - omitting types,
  - interpolation
    - via axiomatizability
    - via Robinson consistency (solved long standing conjecture in many-sorted classical logic)

# Current status of development III

  - definability
    - via axiomatizability
    - via interpolation
  - (layered) completeness, Birkhoff-style, Godel-style
- Internal possible worlds (Kripke) semantics.
- Grothendieck construction on institutions (for heterogeneous specification).
- Proof theoretic developments.
- Systematic theory of logic (institution) translations, borrowing of logical properties along institution co-morphisms.
- Other developments, e.g. ‘stratified’ institutions, etc.

# Institution I

$\mathscr{I} = (\mathbb{S}ig, Sen, \textsc{Mod}, \models):$

$$\begin{tikzcd}
 & \mathbb{S}et \\
\mathbb{S}ig \arrow[ur, bend left, "Sen"] \arrow[dr, bend right, "\textsc{Mod}"'] & \{\models_\Sigma \subseteq |\textsc{Mod}(\Sigma)| \times Sen(\Sigma)\}_{\Sigma \in |\mathbb{S}ig|} \\
 & \mathbb{C}at^{\mathrm{op}}
\end{tikzcd}$$

# Institution II

## Satisfaction Condition

$$\begin{tikzcd}
\Sigma \arrow[dd, "\varphi"'] & \rho \in \mathit{Sen}(\Sigma) & \textsc{Mod}(\varphi)(M') \models_{\Sigma} \rho \\
 & & \Updownarrow \\
\Sigma' & M' \in |\textsc{Mod}(\Sigma')| & M' \models_{\Sigma'} \mathit{Sen}(\varphi)(\rho)
\end{tikzcd}$$

# Notes on the Satisfaction Condition

Similar to the main axiom of ‘abstract model theory’ (Barwise, Feferman, etc.)

Some CS logics (e.g. behavioural logic, rewriting logic) appeared to satisfy only half of the Satisfaction Condition.

- But in fact in all cases it was a structural mismatch in the respective definition, such as improper models, improper signature morphisms, etc.
- Moreover the fixing of these logics as institutions have led to much more realistic formalizations even from very application oriented viewpoint (e.g. behavioural logic).

Moreover, most of the results (99% or so) depend techincally upon the Satisfaction Condition.

# The category $\mathbb{S}ig$

Signatures, resp. signatures morphisms, are considered fully abstractly as objects, resp. arrows, of an abstract *category*.

$$\begin{tikzcd}
\Sigma \arrow[r, "\varphi"] \arrow[dr, "\varphi;\theta"'] & \Sigma' \arrow[d, "\theta"] \arrow[dr, "\theta;\psi"] & \\
& \Sigma'' \arrow[r, "\psi"'] & \Sigma'''
\end{tikzcd}
\qquad
\begin{tikzcd}
\Sigma \arrow[r, "1_\Sigma"] \arrow[d, "\varphi"'] & \Sigma \arrow[d, "\varphi"] \\
\Sigma' \arrow[r, "1_{\Sigma'}"'] & \Sigma'
\end{tikzcd}$$

# Example: **MSA** signatures

*S-sorted signature* $(S, F)$

- $S$ – set of sort symbols,
- $F = \{F_{w \to s} \mid w \in S^*,\ s \in S\}$ – indexed family of operation symbols.

*Signature morphism* $\varphi : \ (S, F) \to (S', F')$

- $\varphi^{\mathrm{st}} : \ S \to S'$ – mapping on the sort symbols,
- $\varphi^{\mathrm{op}}_{w \to s} : \ F_{w \to s} \to F'_{\varphi(w) \to \varphi(s)}$ – mapping on the operation symbols.

*Composition* $\varphi ; \theta = (S, F) \xrightarrow{\varphi} (S', F') \xrightarrow{\theta} (S'', F'')$:

- $(\varphi ; \theta)^{\mathrm{st}} = \theta^{\mathrm{st}} \circ \varphi^{\mathrm{st}}$,
- $(\varphi ; \theta)^{\mathrm{op}}_{w \to s} = \theta^{\mathrm{op}}_{\varphi(w) \to \varphi(s)} \circ \varphi^{\mathrm{op}}_{w \to s}$.

# The sentence functor *Sen*

$$\begin{tikzcd}
\Sigma \arrow[d, "\varphi"'] \arrow[dd, bend left, "\varphi;\theta"] \\
\Sigma' \arrow[d, "\theta"'] \\
\Sigma''
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
Sen(\Sigma) \arrow[d, "Sen(\varphi)"'] \arrow[dd, bend left, "Sen(\varphi;\theta)"] \\
Sen(\Sigma') \arrow[d, "Sen(\theta)"'] \\
Sen(\Sigma'')
\end{tikzcd}$$

# Example: **MSA** sentences

The set of $(S,F)$-*sentences* is the least set such that:

- Each $(S,F)$-equation $t = t'$ is an $(S,F)$-sentence.
- If $\rho_1$ and $\rho_2$ are $(S,F)$-sentences then
  - $\rho_1 \wedge \rho_2$ (*conjunction*),
  - $\rho_1 \vee \rho_2$ (*disjunction*),
  - $\rho_1 \Rightarrow \rho_2$ (*implication*) and
  - $\neg\rho_1$ (*negation*)

  are also $(S,F)$-sentences.
- If $X$ is a finite set of *variables* for $(S,F)$, then $(\forall X)\rho$ and $(\exists X)\rho$ are $(S,F)$-sentences whenever $\rho$ is an $(S,F\cup X)$-sentence.

# Example: $Sen(\varphi): \; Sen(S,F) \to Sen(S',F')$

Defined by induction of the structure of the $(S,F)$-sentences:

- *Term translation*:
  $\varphi^{\mathrm{tm}}(\sigma(t_1,\ldots,t_n)) = \varphi^{\mathrm{op}}(\sigma)(\varphi^{\mathrm{tm}}(t_1),\ldots,\varphi^{\mathrm{tm}}(t_n))$.
- $Sen(\varphi)(t = t') = (\varphi^{\mathrm{tm}}(t) = \varphi^{\mathrm{tm}}(t'))$,
- $Sen(\varphi)(\neg\rho) = \neg Sen(\varphi)(\rho)$,
- $Sen(\varphi)(\rho_1 \star \rho_2) = Sen(\varphi)(\rho_1) \star Sen(\varphi)(\rho_2)$ for $\star \in \{\wedge, \vee, \Rightarrow\}$,
- $Sen(\varphi)((\forall X)\rho) = (\forall X^{\varphi}) Sen(\varphi_1)(\rho)$

$$\begin{tikzcd}
(S,F) \arrow[r] \arrow[d, "\varphi"'] & (S, F \cup X) \arrow[d, "\varphi_1"] \\
(S',F') \arrow[r] & (S', F' \cup X^{\varphi})
\end{tikzcd}$$

The functoriality of *Sen* needs careful definition of for the variables, a variable being a triple $(x, s, (S,F))$.

# Example: $Sen(\varphi) : \; Sen(S,F) \to Sen(S',F')$

Defined by induction of the structure of the $(S,F)$-sentences:

- *Term translation*:
  $\varphi^{\mathrm{tm}}(\sigma(t_1,\dots,t_n)) = \varphi^{\mathrm{op}}(\sigma)(\varphi^{\mathrm{tm}}(t_1),\dots,\varphi^{\mathrm{tm}}(t_n))$.
- $Sen(\varphi)(t = t') = (\varphi^{\mathrm{tm}}(t) = \varphi^{\mathrm{tm}}(t'))$,
- $Sen(\varphi)(\neg\rho) = \neg Sen(\varphi)(\rho)$,
- $Sen(\varphi)(\rho_1 \star \rho_2) = Sen(\varphi)(\rho_1) \star Sen(\varphi)(\rho_2)$ for $\star \in \{\wedge, \vee, \Rightarrow\}$,
- $Sen(\varphi)((\forall X)\rho) = (\forall X^{\varphi})Sen(\varphi_1)(\rho)$

$$\begin{tikzcd}
(S,F) \arrow[r] \arrow[d, "\varphi"'] & (S,F \cup X) \arrow[d, "\varphi_1"] \\
(S',F') \arrow[r] & (S',F' \cup X^{\varphi})
\end{tikzcd}$$

The functoriality of *Sen* needs careful definition of for the variables, a variable being a triple $(x,s,(S,F))$.

# The model functor Mod

$$\begin{tikzcd}
\Sigma \arrow[d, "\varphi"'] \arrow[dd, bend left, "\varphi;\theta"] \\
\Sigma' \arrow[d, "\theta"'] \\
\Sigma''
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
\textsc{Mod}(\Sigma) \\
\textsc{Mod}(\Sigma') \arrow[u, "\textsc{Mod}(\varphi)"] \\
\textsc{Mod}(\Sigma'') \arrow[u, "\textsc{Mod}(\theta)"] \arrow[uu, bend right, "\textsc{Mod}(\varphi;\theta)"']
\end{tikzcd}$$

# Example: **MSA** models

$(S,F)$-*algebra* $A$ consists of

- a set $A_s$ for each $s \in S$, and
- a function $A_{\sigma:w\rightarrow s}: \ A_w \longrightarrow A_s$ for each $\sigma \in F_{w\rightarrow s}$ (where $A_w = A_{s_1} \times \cdots \times A_{s_n}$, for $w = s_1 \dots s_n$).

$(S,F)$-*homomorphism* $h: \ A \longrightarrow B$ consists of

- $h_s: \ A_s \longrightarrow B_s$ for each $s \in S$,

such that the following *homomorphism condition*

$$h_s(A_\sigma(a)) = B_\sigma(h_w(a)).$$

for each $a \in A_w$.

# Example: $\textsc{Mod}(\varphi) : \textsc{Mod}(S', F') \to \textsc{Mod}(S, F)$

This is called *model reduct*.

For any $(S', F')$-algebra $A'$:

- $\textsc{Mod}(\varphi)(A')_s = A'_{\varphi^{\mathrm{st}}(s)}$, and
- $\textsc{Mod}(\varphi)(A')_\sigma = A'_{\varphi^{\mathrm{op}}(\sigma)}$.

For any $(S', F')$-homomorphism $h' : A' \to B'$:

- $\textsc{Mod}(\varphi)(h')_s = h'_{\varphi^{\mathrm{st}}(s)}$ for each $s \in S$.

# The **MSA** satisfaction relation

Tarskian, defined recursively on the structure of the sentences.

- $A \models t = t'$ if and only if $A_t = A_{t'}$ (where $A_{\sigma(t_1,\ldots,t_n)} = A_\sigma(A_{t_1},\ldots,A_{t_n})$),
- $A \models \rho_1 \wedge \rho_2$ if and only if $A \models \rho_1$ and $A \models \rho_2$,
- $A \models \rho_1 \vee \rho_2$ if and only if $A \models \rho_1$ or $A \models \rho_2$,
- $A \models \rho_1 \Rightarrow \rho_2$ if and only if $A \not\models \rho_1$ or $A \models \rho_2$,
- $A \models \neg \rho_1$ if and only if $A \not\models \rho_1$,
- $A \models_{(S,F)} (\forall X)\rho$ if and only if $A' \models_{(S,F\cup X)} \rho$ for each $(S, F \cup X)$-expansion $A'$ of $A$, and
- $A \models (\exists X)\rho$ if and only if $A \not\models (\forall X)\neg\rho$.

# Proof of **MSA** Satisfaction Condition

$\textsc{Mod}(\varphi)(A') \models_{(S,F)} \rho$ if and only if $A' \models_{(S',F')} Sen(\varphi)(\rho)$.

By induction on the structure of $\rho$:

$\rho = (t = t')$:

$A'_{\varphi^{\mathrm{tm}}(t)} = A'_{\varphi^{\mathrm{tm}}(t')}$ if and only if $\textsc{Mod}(\varphi)(A')_t = \textsc{Mod}(\varphi)(A')_{t'}$
by proving that $A'_{\varphi^{\mathrm{tm}}(t)} = \textsc{Mod}(\varphi)(A')_t$.

$\rho = \rho_1 \wedge \rho_2 \mid \rho_1 \vee \rho_2 \mid \rho_1 \Rightarrow \rho_2 \mid \neg\rho_1$:

Straightforward!

# Proof of **MSA** Satisfaction Condition

$\textsc{Mod}(\varphi)(A') \models_{(S,F)} \rho$ if and only if $A' \models_{(S',F')} \mathit{Sen}(\varphi)(\rho)$.

By induction on the structure of $\rho$:

$\rho = (t = t')$:

$A'_{\varphi^{\mathrm{tm}}(t)} = A'_{\varphi^{\mathrm{tm}}(t')}$ if and only if $\textsc{Mod}(\varphi)(A')_t = \textsc{Mod}(\varphi)(A')_{t'}$ by proving that $A'_{\varphi^{\mathrm{tm}}(t)} = \textsc{Mod}(\varphi)(A')_t$.

$\rho = \rho_1 \wedge \rho_2 \mid \rho_1 \vee \rho_2 \mid \rho_1 \Rightarrow \rho_2 \mid \neg\rho_1$:

Straightforward!

# Proof of **MSA** Satisfaction Condition

$\textsc{Mod}(\varphi)(A') \models_{(S,F)} \rho$ if and only if $A' \models_{(S',F')} \mathit{Sen}(\varphi)(\rho)$.

By induction on the structure of $\rho$:

$\rho = (t = t')$:

$A'_{\varphi^{\mathrm{tm}}(t)} = A'_{\varphi^{\mathrm{tm}}(t')}$ if and only if $\textsc{Mod}(\varphi)(A')_t = \textsc{Mod}(\varphi)(A')_{t'}$ by proving that $A'_{\varphi^{\mathrm{tm}}(t)} = \textsc{Mod}(\varphi)(A')_t$.

$\rho = \rho_1 \wedge \rho_2 \mid \rho_1 \vee \rho_2 \mid \rho_1 \Rightarrow \rho_2 \mid \neg\rho_1$:

Straightforward!

# Proof of **MSA** Satisfaction Condition (continued)

$\rho = (\forall X)\rho'$:

Not so easy... It boils down to the equivalence between

1. $A'_1 \models_{(S',F'\cup X^\varphi)} \varphi_1(\rho')$ for all $(S',F'\cup X^\varphi)$-expansions $A'_1$ of $A'$.
2. $A_1 \models_{(S,F\cup X)} \rho'$ for all $(S,F\cup X)$-exp. $A_1$ of $A = \mathrm{Mod}(\varphi)(A')$.

$$\begin{tikzcd}
A & & & A_1 \\
& (S,F) \arrow[r] \arrow[d, "\varphi"'] & (S,F\cup X) \arrow[d, "\varphi_1"] & \\
& (S',F') \arrow[r] & (S',F'\cup X^{\varphi}) & \\
A' & & & A'_1
\end{tikzcd}$$

by noting the canonical bijection
$\{A'_1 \mid \text{expansion of } A'\} \simeq \{A_1 \mid \text{expansion of } A\}$.

# First order (classical) logic (**FOL**)

Extension of **MSA**.

*Signatures*: $(S,F,P)$ with $P = \{P_w\}_{w \in S^*}$.

*Sentences*: formed also with relational atoms $\pi(t_1,\ldots,t_n)$, $\pi \in P_w$.

*Models*: interpret also each $\pi \in P_w$ as $M_\pi \subseteq M_w$.

*Satisfaction*: extends **MSA** satisfaction to relational atoms

$$M \models \pi(t_1,\ldots,t_n) \text{ if and only if } (M_{t_1},\ldots,M_{t_n}) \in M_\pi$$

# Horn clause (**HCL**), equational (**EQL**), propositional logics (**PL**)

Obtained as 'sub-institution' of **FOL**.

*Horn clause logic* restricts the sentences to Horn clauses $(\forall X)H \Rightarrow C$.

*Equational logic* restricts the signatures to **MSA** ones (i.e. $(S,F)$) and sentences to equations $(\forall X)t = t'$.

*Propositional logic* restricts the signatures to those with $S = \emptyset$, hence a signature is just a set $P$ and a model $M : \; P \to \{0,1\}$.

# Intuitionistic logic (**IPL**)

In the propositional case, the same *signatures* like propositional logic (i.e. sets $P$).

*P-sentences* the same as in propositional logic.

*P-models* $M$ are functions $M \to A$ into Heyting algebras $A$. $M$ extends to $Sen(P)$ by using the operations of the Heyting algebra.

*Satisfaction*: $M \models_P \rho$ if and only if $M(\rho) = 1$.

All these extend also to the first order case.

# First order Modal Logic (**MFOL**)

*Signatures*: tuples $(S, S_0, F, F_0, P, P_0)$ where

- $(S, F, P)$ is a **FOL** signature, and
- $(S_0, F_0, P_0)$ is a sub-signature of $(S, F, P)$ of *rigid* symbols.

*Sentences* are the usual first order modal sentences that include also modal connectives, i.e. $\rho_1 \wedge \rho_2$, $(\forall X)\rho$, but also $\Box\rho$.

$(S, S_0, F, F_0, P, P_0)$-*models* $(W, R)$ are families $W$ of ‘possible worlds’ (i.e. **FOL** $(S, F, P)$-models that share the interpretations of the ‘rigid’ symbols), with an accesiblity relation $R$.

*Satisfaction* is given by *Kripke semantics*,

- first $(W, P) \models^i \rho$ and
- then $(W, R) \models \rho$ if and only if $(W, P) \models^i \rho$ for each ‘possible world’ $i$.

# Higher order logics (**HOL, HNK**)

*Signatures*: tuples $(S,F)$ with $S$ set of *sorts* and $F = \{F_s\}_{s\in types(S)}$.

$(S,F)$-*sentences* built from equations $t = t'$ and quantifications and Boolean connectives,

- where terms $t$ are built by functional applications, i.e. if $t$ is a term of type $s \rightarrow s'$ and $t'$ of type $s$ then $(tt')$ is term of type $s'$.

$(S,F)$-*models* interpret

- sort symbols $s$ as sets $M_s$,
- types $s \rightarrow s'$ as $M_{s\rightarrow s'} = \{f \mid f:\ M_s \rightarrow M_{s'}\}$, and
- $M_\sigma \in M_s$ for each $\sigma \in F_s$.

*Henkin semantics* we relaxes models to $M_{s\rightarrow s'} \subseteq \{f \mid f:\ M_s \rightarrow M_{s'}\}$.

Tarskian satisfaction.

# Many-valued logics (**MVL**)

We fix a residuated lattice $(L, \leq, \otimes)$.

*Signatures*: $(S, C, P)$ with $S$ sort, $C$ constant, $P$ relation symbols.

*Sentences*: pairs $(\rho, k)$ between $\rho$, usual first order but also with the residual connective $\otimes$, and $k \in L$.

$(S, C, P)$-*models* $M$ interpret sorts as sets and $M_\pi : \ M_w \rightarrow L$ for each $\pi \in P_w$.

*Satisfaction*: Tarskian,

- first evaluation $M[\rho]$ in $L$ of sentence $\rho$ by model $M$, and
- then $M \models (\rho, k)$ if and only if $k \leq M[\rho]$.

# Partial algebra (**PA**)

*Signatures*: $(S, TF, PF)$ with $S$ set of sort symbols and $TF/PF$ families of total/partial operation symbols.

*Sentences:* like in **MSA**, constructed with both total and partial operations and only with *total* quantifiers.

$(S, TF, PF)$*-models* interpret operations from $PF$ as *partial* functions.

*Satisfaction* is defined Tarskian starting from $A \models t = t'$ if and only if both $A_t$ and $A_{t'}$ are defined and they are equal.

# Preordered algebra (**POA**)

Like **MSA** but

- besides equality we have also *transition* atoms $t \to t'$,
- models interpret sort symbols as preordered sets rather than simple sets, such that operations are monotone, and
- $A \models t \to t'$ if and only if $A_t \leq A_{t'}$.

# Other logics

- Order sorted logics, membership algebra
- multi-algebras (**MA**),
- other modal, e.g. temporal logics
- Linear logic
- Polymorphic logics
- Process, behavioural, coalgebraic, object-oriented logics
- fuzzy logics
- various combinations of the above

…and many many more …