# Compositional theories for embedded programming (I)

Margherita Zorzi, Ph.D

A joint work with Davide Trotta, Ph.D
**Department of Computer Science, University of Verona**

IWC 2020 Paris Nord Summer of LoVe 2020

# A minimalist incipit

$$H \leftrightarrows C$$

## The programming perspective

- Embedded programming style allows to split the syntax in two parts, representing a *host language* H and a *core language* C embedded in H respectively. This formally models several situations in which a user writes a code in a main language and delegates some tasks to another ad hoc (sometimes domain specific) language.
- The interaction between H and C is controlled by "mixed" typing rules that specify how to "promote" well-formed terms from the core to the host and (possibly) viceversa.
- Question: how to give a theoretical foundation of host-core programming style?

## Back to the theory ( or Back to the Garden, "Woodstock", 1969)

- A theoretical foundation of a crucial particular case of embedded languages is rooted in the Linear/Non Linear (L/NL) system introduced by Benton (here we mainly follow "Relating Categorical Semantics for Intuitionistic Linear Logic" by Maietti et al.).
- The dichotomy L/NL is interesting but it can not represent "as it is" the basis for a theory for embedded programming.
- One need to define a more flexible system, in which the embedding of an arbitrary (possibly non linear) core into an arbitrary host language is allowed, and where L/NL follows as a particular case.
- Goals: the definition of a systematised type theory able to capture and standardize common properties of embedded languages; to study the relationship between embedded languages and their models.

# Outline

- $HC_0$: syntax
- $HC_0$: semantics
- *Relating syntax and semantics*
- On the *compositionality* of $HC_0$
- WIP: the insight into the PL perspective

## $HC_0$: main ideas

- We start from a simply typed H and a linear C to define a "kernel" type theory, that can be easily extended both in the host and in the core parts.
- We intentionally start from a linear core, since, as we will show, we can easily obtain a non linear core as a particular case.
- $HC_0$ is a minimal language able to capture the good properties common to an ideal class of embedded languages $\{HC_i\}_i$, built out from $HC_0$, in order to derive any more expressive type theory as an extension of the kernel rules, both for the core C and the host H.
- The design of the $HC_0$ type system "privileges" the host language H i.e., mirroring concrete programming, we establish an *hierarchical dependency* of the core part on the control part: the embedded language C is fully described in the host H.

## Judgements and Type constructors

$$\vdash_{\mathsf{H}} X \ : \mathbf{type} \qquad \vdash_{\mathsf{H}} X = Y \ : \mathbf{type} \qquad \Gamma \vdash_{\mathsf{H}} t \ : X \qquad \Gamma \vdash_{\mathsf{H}} t = s \ : X.$$

$$\vdash_{\mathsf{C}} A \ : \mathbf{type} \qquad \vdash_{\mathsf{C}} A = B \ : \mathbf{type} \qquad \Gamma \mid \Omega \vdash_{\mathsf{C}} f \ : A \qquad \Gamma \mid \Omega \vdash_{\mathcal{C}} f = h \ : A$$

$$(t0c) \qquad \vdash_{\mathsf{C}} A \ : \mathbf{type}$$

$$(t1c) \qquad \frac{\vdash_{\mathsf{C}} A \ : \mathbf{type} \qquad \vdash_{\mathsf{C}} B \ : \mathbf{type}}{\vdash_{\mathsf{C}} A \otimes B \ : \mathbf{type}}$$

$$(t2c) \qquad \frac{\vdash_{\mathsf{C}} A \ : \mathbf{type} \qquad \vdash_{\mathsf{C}} B \ : \mathbf{type}}{\vdash_{\mathsf{H}} \mathtt{Proof}(A, B) \ : \mathbf{type}}$$

$$(t0v) \qquad \vdash_{\mathsf{H}} X \ : \mathbf{type}$$

$$(t2c) \qquad \frac{\vdash_{\mathsf{H}} X \ : \mathbf{type} \qquad \vdash_{\mathsf{H}} Y \ : \mathbf{type}}{\vdash_{\mathsf{H}} X \times Y \ : \mathbf{type}}$$

$$(t2v) \qquad \frac{\vdash_{\mathsf{H}} X \ : \mathbf{type} \qquad \vdash_{\mathsf{H}} Y \ : \mathbf{type}}{\vdash_{\mathsf{H}} X \to Y \ : \mathbf{type}}$$


Table: Types constructors

# Well-typing rules (fragment)

$$\text{(av)} \quad \Gamma_1, x : X, \Gamma_2 \vdash_{\mathrm{H}} x \ : X$$

$$(\pi 1v) \ \frac{\Gamma \vdash_{\mathrm{H}} v \ : X \times Y}{\Gamma \vdash_{\mathrm{H}} \mathbf{inl}(v) \ : X}$$

$$\text{(aev)} \ \frac{\Gamma \vdash_{\mathrm{H}} t \ : X \to Y \qquad \Gamma \vdash_{\mathrm{H}} s \ : X}{\Gamma \vdash_{\mathrm{H}} t(s) \ : Y}$$

$$\text{(pv)} \ \frac{\Gamma \vdash_{\mathrm{H}} s \ : X \qquad \Gamma \vdash_{\mathrm{H}} t \ : Y}{\Gamma \vdash_{\mathrm{H}} \langle s, t \rangle \ : X \times Y}$$

$$(\pi 2v) \ \frac{\Gamma \vdash_{\mathrm{H}} v \ : X \times Y}{\Gamma \vdash_{\mathrm{H}} \mathbf{inr}(v) \ : Y}$$

$$\text{(aiv)} \ \frac{\Gamma, x : X \vdash_{\mathrm{H}} t \ : Y}{\Gamma \vdash_{\mathrm{H}} \lambda x : X.t \ : X \to Y}$$

$$\text{(ac)} \quad \Gamma \mid a : A \vdash_{\mathrm{C}} a \ : A$$

$$\text{(tc)} \ \frac{\Gamma \mid \Omega_1 \vdash_{\mathrm{C}} f \ : A \qquad \Gamma \mid \Omega_2 \vdash_{\mathrm{C}} g \ : B}{\Gamma \mid \Omega_1, \Omega_2 \vdash_{\mathrm{C}} f \otimes g \ : A \otimes B}$$

$$\text{(let1c)} \ \frac{\Gamma \mid \Omega_1 \vdash_{\mathrm{C}} f \ : A \otimes B \qquad \Gamma \mid \Omega_2, a : A, b : B \vdash_{\mathrm{C}} h \ : C}{\Gamma \mid \Omega_1, \Omega_2 \vdash_{\mathrm{C}} \mathbf{let}\ f\ \mathbf{be}\ a \otimes b\ \mathbf{in}\ h \ : C}$$

$$\text{(prom)} \ \frac{\Gamma \mid \Omega \vdash_{\mathrm{C}} f \ : A}{\Gamma \vdash_{\mathrm{H}} \mathbf{promote}(f) \ : \mathtt{Proof}(\times_{\Omega}, A)}$$

$$\text{(der)} \ \frac{\Gamma \vdash_{\mathrm{H}} f \ : \mathtt{Proof}(A, B)}{\Gamma \mid a : A \vdash_{\mathrm{C}} \mathbf{derelict}(f) \ : B}$$

## Evaluations

- The $\beta$ and $\eta$ rules for the H-terms are those of simply typed lambda calculus;
- The $\beta$ rules for the $\mathcal{C}$-terms are those of the modality-free fragment of linear logic (without exponents).

# The category $\mathrm{Th}(HC_0)$: objects and morphisms

**Definition ($HC_0$-theory)**

A typed system T is a $HC_0$-**theory** if it is an extension of $HC_0$ with **proper**-T-**axioms**, namely with new ground type symbols, new type equality judgments, new terms symbols and new equality judgments of terms.

**Definition ($HC_0$-translation )**

Given two $HC_0$-theory $T_1$ and $T_2$, an $HC_0$-**translation** $L$ is a function from types and terms of $T_1$ to types and terms of $T_2$ preserving types and terms judgments

## Breaking the dichotomy Linear-Non Linear, Part I

- If we want to extend the language C to a non-linear core, we have to force the tensor product to be a cartesian product. This can done by adding to the core type system the rules for product plus projections (with mixed contexts). Notice that old rules for tensor are easily derivable from the new ones.

## Categorical semantics

- We state a correspondence between the typed calculus $\mathsf{HC}_0$ and its categorical model via the notion of *internal language*.
  $\mathsf{HC}_0$ provides an internal language of the category $\mathrm{Model}(\mathsf{HC}_0)$ of its models if $\mathrm{Model}(\mathsf{HC}_0) \equiv \mathrm{Th}(\mathsf{HC}_0)$.
- Intermediate goal: to define **a category of models** $\mathrm{Model}(\mathsf{HC}_0)$ such that $\mathsf{HC}_0$ **provides an internal language for these models**.
  - Objects of $\mathrm{Model}(\mathsf{HC}_0)$: models of $\mathsf{HC}_0$
  - Morphisms: interesting part of the construction.

# Models of $\mathsf{HC}_0$

We define a model for $\mathsf{HC}_0$ in terms of an instance of an *enriched category*, by interpreting the host part into a category $\mathcal{V}$ and the core part into a suitable category $\mathcal{C}$ enriched in $\mathcal{V}$.

## Definition (Models of $\mathsf{HC}_0$)

A **model** of $\mathsf{HC}_0$ is a pair $(\mathcal{V}, \mathcal{C})$ where $\mathcal{V}$ is a cartesian closed category and $\mathcal{C}$ is a $\mathcal{V}$-monoidal symmetric category.

# Defining the category $\mathrm{Model}(\mathsf{HC}_0)$

Structures and interpretation are defined in the usual way, some cases:

- $[\![\mathtt{Proof}(A, B)]\!] = \mathcal{C}([\![A]\!], [\![B]\!])$
- $[\![\Gamma \vdash_{\mathsf{H}} \mathbf{promote}(f) \ : \mathtt{Proof}(\times_\Omega, A)]\!] := [\![\Gamma \mid \Omega \vdash_{\mathcal{C}} f \ : A]\!]$
- $[\![\Gamma \mid a : A \vdash_{\mathsf{C}} \mathbf{derelict}(g) \ : B]\!] = [\![\Gamma \vdash_{\mathsf{H}} g \ : \mathtt{Proof}(A, B)]\!]$.

**Theorem (Soundess and Completeness)**

*Symmetric, monoidal categories enriched on cartesian closed categories are sound and complete with respect to the calculus* $\mathsf{HC}_0$.

# Defining the category $\mathrm{Model}(HC_0)$

A key notion:

**Definition (Change of base)**

Given a cartesian closed functor $F : \mathcal{V}_1 \longrightarrow \mathcal{V}_2$ , the induced functor $F_* : \mathcal{V}_1\text{-}\mathbf{Cat} \longrightarrow \mathcal{V}_2\text{-}\mathbf{Cat}$ is called **change of base**.

- TT-programming "reading key": if we have a translation between two host language $H_1$ and $H_2$ and C is an embedded language in $H_1$, then we can use the translation to *change the host language* for C, obtaining an embedding in $H_2$.

# The category $\mathrm{Model}(\mathsf{HC}_0)$

**Definition (Category of $\mathsf{HC}_0$ models)**

- Objects of $\mathrm{Model}(\mathsf{HC}_0)$: models $(\mathcal{V}, \mathcal{C})$;
- An arrow in $\mathrm{Model}(\mathsf{HC}_0)$ between two models $(\mathcal{V}_1, \mathcal{C}_1)$ and $(\mathcal{V}_2, \mathcal{C}_2)$ is given by a pair $(F, f)$ where $F \colon \mathcal{V}_1 \longrightarrow \mathcal{V}_2$ if a strict cartesian closed functor, and $f \colon F_*(\mathcal{C}_1) \longrightarrow \mathcal{C}_2$ is a $\mathcal{V}_2$-functor, preserving strictly all the structures.
- Given two arrows $(F, f) \colon (\mathcal{V}_1, \mathcal{C}_1) \longrightarrow (\mathcal{V}_2, \mathcal{C}_2)$ and $(G, g) \colon (\mathcal{V}_2, \mathcal{C}_2) \longrightarrow (\mathcal{V}_3, \mathcal{C}_3)$ the composition of these is given by the pair $(G, g)(F, f) := (GF, \overline{gf})$ where

$$\overline{gf} := gG_*(f).$$

# $\mathrm{Th}(\mathsf{HC}_0) \equiv \mathrm{Model}(\mathsf{HC}_0)$

**Theorem**

*The typed calculus* $\mathsf{HC}_0$ *provides an internal language for* $\mathrm{Model}(\mathsf{HC}_0)$*, i.e.* $\mathrm{Th}(\mathsf{HC}_0)$ *is equivalent to* $\mathrm{Model}(\mathsf{HC}_0)$*.*

Proof (sketch)

- $C : \mathrm{Th}(\mathsf{HC}_0) \longrightarrow \mathrm{Model}(\mathsf{HC}_0)$ by mapping a $\mathsf{HC}_0$-theory to its syntactic category defined as in Theorem Soundness and Completeness.
- $L : \mathrm{Model}(\mathsf{HC}_0) \longrightarrow \mathrm{Th}(\mathsf{HC}_0)$ is defined by mapping an object $(\mathcal{V}, \mathcal{C})$ of $\mathrm{Model}(\mathsf{HC}_0)$ to the $\mathsf{HC}_0$-theory obtained by extending $\mathsf{HC}_0$. For example:
  - we add new C-types $A$ with the corresponding axiom $\vdash_{\mathsf{C}} A$ : **type** for each element of $\mathbf{ob}(\mathcal{C})$ (we are naming the objects of $\mathcal{C}$ in the calculus and we extend the interpretation of the C-types of $\mathsf{HC}_0$ by interpreting the new names with the corresponding objects);
  - we add new C-terms $f$ and $\Gamma \mid \times_\Omega \vdash_{\mathsf{C}} f : A$ for each morphism $f : \Gamma \longrightarrow \mathtt{Proof}(\times_\Omega, A)$ of $\mathcal{V}$ having the interpretation of $\Gamma$ as domain and of $\mathtt{Proof}(\times_\Omega, A)$ has codomain.

## Breaking the linearity of the core, Part II- Semantics

- If we want to extend C to a non-linear core, we have to force the tensor product of the enriched category $\mathcal{C}$ to be a $\mathcal{V}$-cartesian product. Notice that the new product is a particular case of the $\mathcal{V}$-tensor product.

## On the compositionality of $HC_0$-I

- The kernel calculus $HC_0$ has been designed to be "compositional" (both from the syntactical and the semantics viewpoints).
- In our host-core type theory we establish a hierarchical dependency of the core on the host. Notwithstanding, H and C are independent calculi, with their own syntax and denotation. They both can work as *standalone* languages, with independent operational semantics and categorical interpretation. In the moment one decide to embed C in H, the promotion mechanism allows to import syntax from C into H (to have a"mixed" syntax is mandatory).
- Both H and C can be extended to include more complex features easily mirroring the extension at the semantical level.

## On the compositionality of $HC_0$- II: "tuning" the computational paradigm

- The core C can be "specialized" to a computational paradigm.
- As a particular case, we can set the C to a circuit description languages and we can "tune" it to quantum, probabilistic or reversible computation.
- Relation with Staton's EWire and the embedded language QWire (Zdancewic et al.).

# In progress: the PL perspective, informally...

- We are now able to provide a formal definition of host-core languages: Given $\mathcal{L}_1$ and $\mathcal{L}_2$ standalone languages with their respective syntactic denotations $D_1$ and $D_2$, we say that $\mathcal{L}_2$ is embedded in $\mathcal{L}_1$ if the pair $(D_1, D_2)$ belongs to $\mathrm{Model}(\mathsf{HC}_0)$. This means that there exists an host-core language $\mathcal{L}_3$ such that $\mathcal{L}_1$ hosts $\mathcal{L}_2$.
- From the constructive proof of the correspondence between syntax and semantics we extract an algorithm that, given two models $D_1$ and $D_2$ *builds* a "candidate" type theory $\mathcal{L}_3$.
- Working on concrete instances of the family $\mathsf{HC}_i$ and specializations to non classical computational paradigm: we are studying instances of (versions of) $\mathsf{HC}_0$ as higher-order quantum/reversible/probabilistic circuit description languages.

# Mini Biblio


- D. Trotta, M. Zorzi. Compositional theories for embedded languages, journal paper, submitted, 2020. https://arxiv.org/abs/2006.10604
- P. N. Benton. 1995. A mixed linear and non-linear logic: Proofs, terms and models. In Computer Science Logic, Leszek Pacholski and Jerzy Tiuryn (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 121-135
- M. E. Maietti, P. Maneggia, V. de Paiva, and E. Ritter. 2005. Relating Categorical Semantics for Intuitionistic Linear Logic. Appl. Categ. Structures 13, 1 (2005), 1-36.
- Mathys Rennela and Sam Staton. 2018. Classical Control and Quantum Circuits in Enriched Category Theory. Electronic Notes in Theoretical Computer Science 336 (2018), 257-279. https://doi.org/10.1016/j.entcs.2018.03.027. MFPS XXXIII.
- Jennifer Paykin, Robert Rand, and Steve Zdancewic. 2017. QWIRE: A Core Language for Quantum Circuits. In Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages (Paris, France) (POPL 2017). ACM, New York, NY, USA, 846-858. https://doi.org/10.1145/3009837.3009894


thanks, merci, gracias, grazie, vic-tour e smursma, peace&love