A combinatorial view of module composition for OO programming languages

Ugo de'Liguoro

University of Turin

Dagstuhl, June 2014

Components in OO programming

- **Object based**: a library is just collection of objects that can be cloned and updated at run time
- **Class based**: a class hierarchy which is either a tree (single hineritance) or a DAG (multiple inheritance) of class declarations
- **Mixin based**: mixins are parametric sub-classes; classes are obtained by *linearization* of mixins composition
- **Trait addition**: traits are stateless records of methods that can be added to a class *without overriding* methods in the class with the same name

Modeling components

- Objects and classes represented by means of typed λ-calculus with records (Cardelli-Wegner, Bruce, Fisher-Honsel-Mitchell)
- A calculus of objects (Abadi-Cardelli)
- Classes and mixins via typed and typefree λ-calculus (Bracha-Cook)
- Formalization in idealized Java (Igarashi-Pierce-Wadler)
- ...

Advantages: safety properties (exclusion of "method not understood" errors), discovery and clarification of subtle ambiguities end problems (self reference and mytype, inheritance vs. subtyping, binary methods)

Drowbacks: very complex type systems, using recursive types, second order universal and existential quantifiers, F-bounded polymorphism, ...

An alternative direction

- represent OO entities by a suitable extension of type free λ-calculus
- use intersection types as a finitary representation of semantic properties
- find a combinatory factorization of OO module composition

dL01 U. de'Liguoro: Characterizing Convergent Terms in Object Calculi via Intersection Types. LNCS 2044: 315-328(2001)

dL02 U. de'Liguoro: Subtyping in Logical Form. ENTCS 70 (2002)

vBdL08 S. van Bakel, U. de'Liguoro: Logical Equivalence for Subtyping Object and Recursive Types. Theory Comput. Syst. 42(3): 306-348 (2008)

dLC14 U. de'Liguoro, T-C. Chen: Semantic Types for Classes and Mixins, ITRS 2014 to appear

and related works

RvB14 R. Rowe, S. van Bakel: Semantic Types and Approximation for Featherweight Java, TCS, 517 (2014) 3474

BDDM14 J. Bessai, B. Düdder, A. Dudenhefner, M. Martens: Delegation-based Mixin Composition Synthesis, ITRS 2014 to appear

The calculus Λ_R

Terms:

$$\begin{array}{rcll} M, N & ::= & x \mid \lambda x.M \mid MN \mid R \mid M.a \mid M \oplus R & \text{(term)} \\ R & ::= & \langle a = M_a \mid a \in A \rangle & \text{(record)} \end{array}$$

If $A = \emptyset$ then we get the empty record $\langle\rangle$

Reduction:

$$\begin{array}{lll} (\beta) & (\lambda x.M)N \longrightarrow M\{N/x\} & \\ (R_1) & \langle a = M_a \mid a \in A \rangle.b \longrightarrow M_b & \text{if } b \in A \\ (R_2) & \langle a = M_a \mid a \in A \rangle \ \oplus \ \langle b = N_b \mid b \in B \rangle & \\ & \longrightarrow \ \langle a = M_a, \ b = N_b \mid a \in A \setminus B, b \in B \rangle & \end{array}$$

Intersection types

$$\sigma, \tau ::= \alpha \mid \omega \mid \sigma \to \tau \mid \sigma \wedge \tau \mid \diamondsuit \mid \langle a : \sigma \rangle$$

with denumerably many type variables $\alpha \in \mathsf{TypeVar}$.
We define a preorder $\leq$ s.t.

- ω is top, $\wedge$ meet
- $\sigma \to \omega \leq \omega \to \omega$
- $(\sigma \to \tau) \wedge (\sigma \to \rho) \leq \sigma \to (\tau \wedge \rho)$
- $\sigma' \leq \sigma, \tau \leq \tau' \Longrightarrow \sigma \to \tau \leq \sigma' \to \tau'$

plus

- $\sigma \leq \tau \Longrightarrow \langle a : \sigma \rangle \leq \langle a : \tau \rangle$
- $\langle a : \sigma \rangle \wedge \langle a : \tau \rangle \leq \langle a : \sigma \wedge \tau \rangle$
- $\langle a : \sigma \rangle \leq \diamondsuit$ (from [BDDM14])

Semantics

If $\eta : \mathrm{TypeVar} \to 2^{\Lambda_R^0}$, where Λ_R^0 is the set of closed Λ_R-terms

$$
\begin{array}{lcl}
[\![\alpha]\!]_\eta & = & \eta(\alpha) \\
[\![\omega]\!]_\eta & = & \Lambda_R^0 \\
[\![\sigma \to \tau]\!]_\eta & = & \{M \in \Lambda_R^0 \mid \exists M'.\ M \longrightarrow^* \lambda x.M' \ \ \& \\
 & & \qquad\qquad \forall N \in [\![\sigma]\!]_\eta.\ M'\{N/x\} \in [\![\tau]\!]_\eta\} \\
[\![\langle a : \sigma \rangle]\!]_\eta & = & \{M \in \Lambda_R^0 \mid \exists R.\ M \longrightarrow^* R \ \ \& \ \ R.a \in [\![\sigma]\!]_\eta\} \\
[\![\sigma \wedge \tau]\!]_\eta & = & [\![\sigma]\!]_\eta \cap [\![\tau]\!]_\eta
\end{array}
$$

Semantics

If $\eta : \mathsf{TypeVar} \to 2^{\Lambda^0_R}$, where Λ^0_R is the set of closed Λ_R-terms

$$
\begin{array}{lcl}
[\![\alpha]\!]_\eta & = & \eta(\alpha) \\
[\![\omega]\!]_\eta & = & \Lambda^0_R \\
[\![\sigma \to \tau]\!]_\eta & = & \{M \in \Lambda^0_R \mid \exists M'.\ M \longrightarrow^* \lambda x.M' \ \ \& \\
 & & \qquad \forall N \in [\![\sigma]\!]_\eta.\ M'\{N/x\} \in [\![\tau]\!]_\eta\} \\
[\![\langle a : \sigma \rangle]\!]_\eta & = & \{M \in \Lambda^0_R \mid \exists R.\ M \longrightarrow^* R \ \ \& \ \ R.a \in [\![\sigma]\!]_\eta\} \\
[\![\sigma \wedge \tau]\!]_\eta & = & [\![\sigma]\!]_\eta \cap [\![\tau]\!]_\eta
\end{array}
$$

Semantics

If $\eta : \mathsf{TypeVar} \to 2^{\Lambda^0_R}$, where Λ^0_R is the set of closed Λ_R-terms

$$
\begin{array}{lcl}
[\![\alpha]\!]_\eta & = & \eta(\alpha) \\
[\![\omega]\!]_\eta & = & \Lambda^0_R \\
[\![\sigma \to \tau]\!]_\eta & = & \{M \in \Lambda^0_R \mid \exists M'.\ M \longrightarrow^* \lambda x.M' \ \ \& \\
 & & \qquad \forall N \in [\![\sigma]\!]_\eta.\ M'\{N/x\} \in [\![\tau]\!]_\eta\} \\
[\![\langle a : \sigma \rangle]\!]_\eta & = & \{M \in \Lambda^0_R \mid \exists R.\ M \longrightarrow^* R \ \ \& \ \ R.a \in [\![\sigma]\!]_\eta\} \\
[\![\sigma \wedge \tau]\!]_\eta & = & [\![\sigma]\!]_\eta \cap [\![\tau]\!]_\eta
\end{array}
$$

Semantics

If $\eta : \mathsf{TypeVar} \to 2^{\Lambda_R^0}$, where Λ_R^0 is the set of closed Λ_R-terms

$$
\begin{array}{lcl}
[\![\alpha]\!]_\eta & = & \eta(\alpha) \\
[\![\omega]\!]_\eta & = & \Lambda_R^0 \\
[\![\sigma \to \tau]\!]_\eta & = & \{M \in \Lambda_R^0 \mid \exists M'.\ M \longrightarrow^* \lambda x.M' \ \ \& \\
 & & \qquad\qquad \forall N \in [\![\sigma]\!]_\eta.\ M'\{N/x\} \in [\![\tau]\!]_\eta\} \\
[\![\langle a : \sigma \rangle]\!]_\eta & = & \{M \in \Lambda_R^0 \mid \exists R.\ M \longrightarrow^* R \ \ \& \ \ R.a \in [\![\sigma]\!]_\eta\} \\
[\![\sigma \wedge \tau]\!]_\eta & = & [\![\sigma]\!]_\eta \cap [\![\tau]\!]_\eta
\end{array}
$$

then

$$\sigma \leq \tau \Longrightarrow \forall \eta.\ [\![\sigma]\!]_\eta \subseteq [\![\tau]\!]_\eta$$

Type assignment

To BCD rules

$$\frac{}{\Gamma, x : \sigma \vdash x : \sigma}\,(Ax) \qquad \frac{}{\Gamma \vdash M : \omega}\,(\omega)$$

$$\frac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash \lambda x.M : \sigma \to \tau}\,(\to I) \qquad \frac{\Gamma \vdash M : \sigma \to \tau \quad \Gamma \vdash N : \sigma}{\Gamma \vdash MN : \tau}\,(\to E)$$

$$\frac{\Gamma \vdash M : \sigma \quad \Gamma \vdash M : \tau}{\Gamma \vdash M : \sigma \wedge \tau}\,(\wedge) \qquad \frac{\Gamma \vdash M : \sigma \quad \sigma \leq \tau}{\Gamma \vdash M : \tau}\,(\leq)$$

where $\Gamma = \{x_1 : \sigma_1, \ldots, x_k : \sigma_k\}$ and $\Gamma, x : \sigma = \Gamma \cup \{x : \sigma\}$

Type assignment

... we add

$$\frac{\Gamma \vdash M : \langle a : \sigma \rangle}{\Gamma \vdash M.a : \sigma} \qquad \frac{\Gamma \vdash M : \sigma \qquad a \notin B}{\Gamma \vdash \langle a = M, b = N_b \mid b \in B \rangle : \langle a : \sigma \rangle}$$

If we abbreviate

$$\langle a : \sigma_a \mid a \in A \rangle \equiv \bigwedge_{a \in A} \langle a : \sigma_a \rangle$$

then we get

$$\frac{\sigma_b \leq \tau_b \qquad \forall b \in B \subseteq A}{\langle a : \sigma_a \mid a \in A \rangle \leq \langle b : \tau_b \mid b \in B \rangle}$$

hence

$$\frac{\Gamma \vdash M_b : \sigma_b \qquad \forall b \in B \subseteq A}{\Gamma \vdash \langle a = M_a \mid a \in A \rangle : \langle b : \sigma_b \mid b \in B \rangle}$$

Type invariance

1. $\Gamma \vdash M : \sigma \ \ \& \ \ M \longrightarrow N \Longrightarrow \Gamma \vdash N : \sigma$ (subject reduction)
2. $\Gamma \vdash N : \sigma \ \ \& \ \ M \longrightarrow N \Longrightarrow \Gamma \vdash M : \sigma$ (subject expansion)

Type invariance

1. $\Gamma \vdash M : \sigma \ \ \& \ \ M \longrightarrow N \Longrightarrow \Gamma \vdash N : \sigma$ (subject reduction)
2. $\Gamma \vdash N : \sigma \ \ \& \ \ M \longrightarrow N \Longrightarrow \Gamma \vdash M : \sigma$ (subject expansion)

Soundness theorem

A *closed substitution* is a map $\theta : \mathsf{TermVar} \to \Lambda_R$

$$\theta, \eta \models \Gamma = \{x_1 : \sigma_1, \ldots, x_k : \sigma_k\} \overset{\text{def}}{\iff} \theta(x_i) \in [\![\sigma_i]\!]_\eta \quad i = 1, \ldots, k$$

Then

$$\Gamma \vdash M : \sigma \;\; \& \;\; \theta, \eta \models \Gamma \Longrightarrow M\theta \in [\![\sigma]\!]_\eta$$

where $M\theta \equiv M\{\theta(x_1)/x_1\} \ldots \{\theta(x_k)/x_k\}$

Objects and the typing of self (this)

According to the self-application representation, an **object** is a record

$$O \equiv \langle a = \lambda t.M_a \mid a \in A \rangle$$

where $a = \lambda t.M_a$ is the **method** labelled by a

Define

$$(O \Leftarrow a) \equiv (O.a)\, O$$

then

$$(O \Leftarrow a) \longrightarrow (\lambda t.M_a)\, O \longrightarrow M_a\{O/t\}$$

hence t is the self (this) variable.

If $t \notin fv(M_a)$ then $M_a\{O/t\} \equiv M_a$, and we say that $a = \lambda t.M_a$ is a **field**

Objects and the typing of self (this)

For any method $a = \lambda t.M \in O$ we have the typing

$$\frac{\frac{\Gamma, t : \sigma \vdash M : \tau}{\Gamma \vdash \lambda t.M : \sigma \to \tau}}{\Gamma \vdash O : \langle a : \sigma \to \tau \rangle}$$

In particular if $a = \lambda t.M \in O$ is a field then $t \notin fv(M)$ and

$$\frac{\frac{\Gamma, t : \omega \vdash M : \tau}{\Gamma \vdash \lambda t.M : \omega \to \tau}}{\Gamma \vdash O : \langle a : \omega \to \tau \rangle}$$

Note that $\langle a : \omega \to \tau \rangle \leq \langle a : \sigma \to \tau \rangle$ but in general $\not\geq$

Objects and the typing of self (this)

Recalling that $(O \Leftarrow a) \equiv (O.a)\, O$ we have the derived rule

$$\frac{\Gamma \vdash O : \langle a : \sigma \to \tau \rangle \qquad \Gamma \vdash O : \sigma}{\Gamma \vdash O \Leftarrow a : \tau}$$

that also implies $\Gamma \vdash O : \langle a : \sigma \to \tau \rangle \wedge \sigma$. However if

$$O \equiv \langle a = \lambda t.1,\ b = \lambda t.(t \Leftarrow a) + 1 \rangle$$

then

$$\vdash O : \langle a : \omega \to \text{odd}, b : \langle a : \omega \to \text{even} \rangle \to \text{odd} \rangle$$

even if

$$\not\vdash O : \langle a : \omega \to \text{even} \rangle$$

i.e. the typing of the self (this) is just conditional

Objects and the typing of self (this)

Recalling that $(O \Leftarrow a) \equiv (O.a)\, O$ we have the derived rule

$$\frac{\Gamma \vdash O : \langle a : \sigma \to \tau \rangle \qquad \Gamma \vdash O : \sigma}{\Gamma \vdash O \Leftarrow a : \tau}$$

that also implies $\Gamma \vdash O : \langle a : \sigma \to \tau \rangle \wedge \sigma$. However if

$$O \equiv \langle a = \lambda t.1,\ b = \lambda t.(t \Leftarrow a) + 1 \rangle$$

then

$$\vdash O : \langle a : \omega \to \text{odd}, b : \langle a : \omega \to \text{even} \rangle \to \text{odd} \rangle$$

even if

$$\not\vdash O : \langle a : \omega \to \text{even} \rangle$$

i.e. the typing of the self (this) is just conditional

Typing rules for $\oplus$ (merge)

Define

$$\ell(\langle a = M_a \mid a \in A \rangle) = A$$

$$\frac{\Gamma \vdash M : \diamondsuit \quad \Gamma \vdash R : \langle a : \sigma \rangle}{\Gamma \vdash M \ \oplus \ R : \langle a : \sigma \rangle} (\oplus_1) \qquad \frac{\Gamma \vdash M : \langle a : \sigma \rangle \qquad a \notin \ell(R)}{\Gamma \vdash M \ \oplus \ R : \langle a : \sigma \rangle} (\oplus_2)$$

so that the following rule is admissible

$$\frac{\Gamma \vdash M : \langle a : \sigma_a \mid a \in A \rangle \qquad \Gamma \vdash R : \langle b : \tau_b \mid b \in B \rangle}{\Gamma \vdash M \ \oplus \ R : \langle a : \sigma_a, b : \tau_b \mid a \in A \setminus B, b \in B \rangle}$$

Remark

The following are **not** well formed terms:

$$R \oplus x \notin \Lambda_R \qquad \text{and hence} \qquad \lambda x.(R \oplus x) \notin \Lambda_R$$

Otherwise set $\ell(M) = \emptyset$ if M is not a record, then

$$\frac{\Gamma, x : \sigma \vdash R : \langle a : \tau \rangle \qquad a \notin \ell(x) = \emptyset}{\dfrac{\Gamma, x : \sigma \vdash R \oplus x : \langle a : \tau \rangle}{\Gamma \vdash \lambda x.\ R \oplus x : \sigma \to \langle a : \tau \rangle}}$$

Say that $x \notin fv(R)$ and take $\sigma \equiv \langle a : \rho \rangle$ and N s.t.

$$\Gamma \vdash N : \rho \quad \text{and} \quad \Gamma \not\vdash N : \tau$$

then

$$\Gamma \vdash (\lambda x.\ R \oplus x)\langle a = N \rangle : \langle a : \tau \rangle \ \text{ but } \ \Gamma \not\vdash R \oplus \langle a = N \rangle : \langle a : \tau \rangle$$

Extension and overriding

For any record R define

$$(R.a := N) \;\equiv\; R \oplus \langle a = N \rangle$$

then we have the rules

$$\frac{\Gamma \vdash M : \sigma}{\Gamma \vdash R.a := M : \langle a : \sigma \rangle}\,(upd_1) \qquad \frac{\Gamma \vdash R : \langle a : \sigma \rangle \quad a \neq b}{\Gamma \vdash R.b := M : \langle a : \sigma \rangle}\,(upd_2)$$

and therefore

$$\frac{\Gamma \vdash \langle a = N_a \mid a \in A \rangle : \langle b : \sigma_b \mid b \in B \rangle \qquad B \subseteq A \qquad \Gamma \vdash N : \tau}{\Gamma \vdash \langle a = N_a \mid a \in A \rangle.c := N : \langle c : \tau, b : \sigma_b \mid b \in B \setminus \{c\} \rangle}$$

Classes

A **class** is a generator of objects with parameters:

class Person
    field: name;
    method: display() { print(name); }

which is formally

$$\text{Person} \equiv \lambda n.\, \langle \text{name} = \lambda t.\, n,\ \text{display} = \lambda t.\, \text{print}(t \Leftarrow \text{name}) \rangle$$

Inheritance

A class can extend some fixed superclass(es) by **inheritance**:

class Graduate
superclass: Person;
field: degree;
method: display() {**super**.display(); print(degree);}

that is

$$\begin{aligned}\text{Graduate} \equiv \lambda n.\, \lambda d.\, & \\ & \langle\, \text{name} = \lambda t.\, n, \\ & \quad \text{degree} = \lambda t.\, d, \\ & \quad \text{display} = \lambda t.\, (\text{Person}\ n) \Leftarrow \text{display}; \text{print}(t \Leftarrow \text{degree}) \rangle\end{aligned}$$

From class inheritance ...

$$\begin{aligned}\text{Graduate} \equiv\ & \lambda n.\, \lambda d. \\ & \langle\ \text{name} = \lambda t.\, n, \\ & \ \ \text{degree} = \lambda t.\, d, \\ & \ \ \text{display} = \lambda t.\, (\text{Person}\ n) \Leftarrow \text{display}; \text{print}(t \Leftarrow \text{degree})\rangle\end{aligned}$$

is equivalent to

$$\begin{aligned}&\lambda n.\ \ \lambda d. \\ &\quad (\text{Person}\ n)\ \ \oplus \\ &\qquad\qquad \langle\ \text{degree} = \lambda t.\, d, \\ &\qquad\qquad\ \ \text{display} = \lambda t.\, (\text{Person}\ n) \Leftarrow \text{display}; \text{print}(t \Leftarrow \text{degree})\rangle\end{aligned}$$

... to mixin inheritance

$$\begin{array}{l}\lambda n.\ \ \lambda d. \\ \quad (\text{Person}\ n)\ \ \oplus \\ \qquad\qquad \langle\ \text{degree} = \lambda t.\, d, \\ \qquad\qquad\ \ \text{display} = \lambda t.\, (\text{Person}\ n) \Leftarrow \text{display}; \text{print}(t \Leftarrow \text{degree})\rangle\end{array}$$

that is equivalent to

$$\begin{array}{l}\lambda n.\ \ \ \lambda d. \\ \quad (\lambda s.\ (s\ n)\ \ \oplus \\ \qquad\qquad \langle\ \text{degree} = \lambda t.\, d, \\ \qquad\qquad\ \ \text{display} = \lambda t.\, (s\ n) \Leftarrow \text{display}; \text{print}(t \Leftarrow \text{degree})\rangle) \\ \quad \text{Person}\end{array}$$

Mixins (after Cook and Bracha)

A **mixin** is a parametric heir of a super-class:

Definition

$$\text{mixin}_{R,\vec{x}} \stackrel{\text{def}}{=} \lambda\vec{x}\,\lambda s.(s \oplus R) \qquad fv(R) \subseteq \{s, \vec{x}\}$$

where $s = \text{super}$

Since any record R can be factored into $R_1 \oplus \cdots \oplus R_k$, where k and the R_i are **not** unique, we have that for any class C:

$$C\,\vec{d_1} \cdots \vec{d_k} = ((\text{mixin}_{R_1,\vec{d_1}}) \circ \cdots \circ (\text{mixin}_{R_k,\vec{d_k}}))\langle\rangle$$

where $\text{mixin}_{R_i,\vec{d_i}} \equiv (\text{mixin}_{R_i,\vec{x_i}})\ \vec{d_i}$

for several choices of preexisting **modules** $\text{mixin}_{R_i,\vec{x_i}}$

Typing mixins

Set $\mathsf{mixin}_{R_i} \equiv \lambda x.(x \ \oplus \ R_i)$ where

$$R_1 \equiv \langle a = N_1 \rangle, \quad R_2 \equiv \langle b = N_2 \rangle, \quad R_3 \equiv \langle a = N_3 \rangle$$

and assume $x \notin fv(R_i)$ then

$$\dfrac{\dfrac{\Gamma, x : \diamondsuit \vdash x : \diamondsuit \qquad \dfrac{\Gamma, x : \diamondsuit \vdash N_1 : \sigma_1}{\Gamma, x : \diamondsuit \vdash \langle a = N_1 \rangle : \langle a : \sigma_1 \rangle}}{\Gamma, x : \diamondsuit \vdash x \ \oplus \ \langle a = N_1 \rangle : \langle a : \sigma_1 \rangle}}{\Gamma \vdash \mathsf{mixin}_{R_1} \equiv \lambda x.(x \ \oplus \ \langle a = N_1 \rangle) : \diamondsuit \to \langle a : \sigma_1 \rangle}$$

Typing mixins

$$R_1 \equiv \langle a = N_1 \rangle, \quad R_2 \equiv \langle b = N_2 \rangle, \quad R_3 \equiv \langle a = N_3 \rangle \qquad x \notin fv(R_i)$$

and also

$$\frac{\dfrac{\Gamma, x : \langle b : \sigma_2 \rangle \vdash x : \langle b : \sigma_2 \rangle \qquad b \notin \ell(\langle a = N_1 \rangle)}{\Gamma, x : \langle b : \sigma_2 \rangle \vdash x \ \oplus \ \langle a = N_1 \rangle : \langle b : \sigma_2 \rangle}}{\Gamma \vdash \mathrm{mixin}_{R_1} \equiv \lambda x.(x \ \oplus \ \langle a = N_1 \rangle) : \langle b : \sigma_2 \rangle \to \langle b : \sigma_2 \rangle}$$

therefore since $\diamondsuit \to \langle a : \sigma_1 \rangle \leq \langle b : \sigma_2 \rangle \to \langle a : \sigma_1 \rangle$

$$\Gamma \vdash \mathrm{mixin}_{R_1} : \langle b : \sigma_2 \rangle \to \langle a : \sigma_1 \rangle \wedge \langle b : \sigma_2 \rangle \to \langle b : \sigma_2 \rangle$$

that is

$$\Gamma \vdash \mathrm{mixin}_{R_1} : \langle b : \sigma_2 \rangle \to \langle a : \sigma_1, b : \sigma_2 \rangle$$

Typing mixins

$$R_1 \equiv \langle a = N_1 \rangle, \quad R_2 \equiv \langle b = N_2 \rangle, \quad R_3 \equiv \langle a = N_3 \rangle \qquad x \notin fv(R_i)$$

Similarly for $\mathsf{mixin}_{R_2} \equiv \lambda x.(x \oplus R_2)$ assuming that $\Gamma \vdash N_2 : \sigma_2$ we have

$$\Gamma \vdash \mathsf{mixin}_{R_2} : \diamondsuit \to \langle b : \sigma_2 \rangle$$

so that

$$\frac{\Gamma \vdash \mathsf{mixin}_{R_2} : \diamondsuit \to \langle b : \sigma_2 \rangle \qquad \Gamma \vdash \mathsf{mixin}_{R_1} : \langle b : \sigma_2 \rangle \to \langle a : \sigma_1, b : \sigma_2 \rangle}{\Gamma \vdash \mathsf{mixin}_{R_1} \circ \mathsf{mixin}_{R_2} : \diamondsuit \to \langle a : \sigma_1, b : \sigma_2 \rangle}$$

Typing mixins

$$R_1 \equiv \langle a = N_1 \rangle, \quad R_2 \equiv \langle b = N_2 \rangle, \quad R_3 \equiv \langle a = N_3 \rangle \qquad x \notin fv(R_i)$$

If $\Gamma \vdash N_1 : \sigma_1$, $\Gamma \vdash N_3 : \sigma_3$ but $\Gamma \nvdash N_1 : \sigma_3$ we have

$$\Gamma \vdash \mathrm{mixin}_{R_3} : \diamondsuit \to \langle a : \sigma_3 \rangle$$

and

$$\Gamma \vdash \mathrm{mixin}_{R_1} : \diamondsuit \to \langle a : \sigma_1 \rangle \leq \langle a : \sigma_3 \rangle \to \langle a : \sigma_1 \rangle$$

so that

$$\Gamma \vdash \mathrm{mixin}_{R_1} \circ \mathrm{mixin}_{R_3} : \diamondsuit \to \langle a : \sigma_1 \rangle$$

but

$$\Gamma \nvdash \mathrm{mixin}_{R_1} \circ \mathrm{mixin}_{R_3} : \diamondsuit \to \langle a : \sigma_3 \rangle$$

Traits (after Ducasse and others)

A **trait** is a record of methods (not of fields) without any super-call

Traits satisfy

- Methods defined in a class itself take precedence over methods provided by a trait
- A non-overridden method in a trait has the same semantics as if it were implemented directly in the class (*flattening property*)
- *Composition order is irrelevant*; all the traits have the same precedence, and hence conflicting trait methods must be explicitly disambiguated

Exclusive merge $\otimes$

$$R, S \quad ::= \quad \langle a = M_a \mid a \in A \rangle \mid R \otimes S \quad \text{(record)}$$

$$\begin{aligned} \langle a = M_a \mid a \in A \rangle \quad & \otimes \quad \langle b = N_b \mid b \in B \rangle \\ & \longrightarrow \quad \langle a = M_a,\ b = N_b \mid a \in A \setminus B, b \in B \setminus A \rangle \end{aligned}$$

$$\frac{\Gamma \vdash R : \langle a : \sigma \rangle \qquad a \notin \ell(S)}{\Gamma \vdash R \otimes S : \langle a : \sigma \rangle} (\otimes_1) \qquad \frac{\Gamma \vdash S : \langle a : \sigma \rangle \qquad a \notin \ell(R)}{\Gamma \vdash R \otimes S : \langle a : \sigma \rangle} (\otimes_2)$$

Classes from traits

Any class C (forgetting about parameters) can be factored into

$$C = (T_1 \otimes \cdots \otimes T_k) \oplus R = \mathrm{mixin}_R \left(\bigotimes_{i=1}^{k} T_i \right)$$

where the T_i are traits

More radically we could say that $C = \langle a_1 = M_1, \ldots, a_n = M_n \rangle$

$$C = \left(\bigotimes_{i=1}^{n} T_i \right)$$

where $T_i \equiv \langle a_i = M_i \rangle$, namely **unit traits** made of just one method!

The synthesis problem

A tentative formulation of the "synthesis by inhabitation" problem:

$$T_1 : \sigma_1, \ldots, T_k : \sigma_k \vdash ? : \tau$$

How complex could be a solution?

$$T_1 \equiv \langle a = 1 \rangle : \langle a : \mathrm{odd} \rangle,$$

$$T_2 \equiv \langle b = \lambda t.(t.a + 1) \rangle : \langle b : \langle a : \mathrm{odd} \rangle \to \mathrm{even} \wedge \langle a : \mathrm{even} \rangle \to \mathrm{odd} \rangle$$

Problem: $? : \langle a : \mathrm{even} \rangle$
Solution: $\langle a = (T_1 \otimes T_2) \Leftarrow b \rangle$

Problem: $? : \langle b : \langle a : \mathrm{odd} \rangle \to \mathrm{odd} \rangle$
Solution: $\langle b = \lambda t.((t \ \oplus \ \langle a = (t \Leftarrow b) \rangle).a + 1) \rangle$

Challenges

- find a set of combinators including $\oplus$, $\otimes$, $\circ$, mixin_R, generating a "combinatory algebra of modules" s.t. given any set of traits $T_1, \ldots, T_k$, if $C \in \Lambda_R(T_1, \ldots, T_k)$ is a class term then $C\vec{d}$ is equivalent (possibly the normal form of) to a term built only by the combinators in the algebra and the traits T_i and d_j
- extend the "synthesis by inhabitation" method to the algebra of modules
- try to apply the method to existing programming languages like Scala and Python, say via staged combinatory synthesis