Semantic Types for Classes and Mixins

Ugo de'Liguoro and Tzu-chun Chen

University of Turin

ITRS'14, Vienna, July 18, 2014

Motivations

Motivations

Issues:

- Mixins have been proposed in the late 80's to enhance modularity and reusability of code for class based OO programming languages
- definition of mixins still depends on actual implementations, and differs for different languages
- type systems ensuring basic safety properties are quite complex, and do not support specification of semantic properties

Motivations

Issues:

- Mixins have been proposed in the late 80's to enhance modularity and reusability of code for class based OO programming languages
- definition of mixins still depends on actual implementations, and differs for different languages
- type systems ensuring basic safety properties are quite complex, and do not support specification of semantic properties

Goals:

- a simple model for classes and mixins based on λ-calculus with records and Bracha-Cook merge operator
- a type theoretic presentation of the model providing the syntax for semantic specifications of OO modules
- a suitable basis for the "synthesis by inhabitation" program to be applicable to classes and mixins

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:

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

Objects and Kamin's self-application semantics

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**

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

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

From class inheritance ...

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

is equivalent to

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

... 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** $\text{mixin}_{R_i,\vec{x}_i}$

Intersection types

Types:

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

Rules:

$$\frac{\Gamma \vdash M : \sigma \qquad a = M \in R}{\Gamma \vdash R : \langle a : \sigma \rangle} \qquad \frac{\Gamma \vdash M : \langle a : \sigma \rangle}{\Gamma \vdash M.a : \sigma}$$

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 by $\wedge$-introduction:

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

Typing rules for $\oplus$ (merge)

Define

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

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

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:

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

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

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

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

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

Subtyping

We define subtyping as a preorder $\leq$ over types 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

- $\sigma \leq \tau \Longrightarrow \langle a : \sigma \rangle \leq \langle a : \tau \rangle$
- $\langle a : \sigma \rangle \wedge \langle a : \tau \rangle \leq \langle a : \sigma \wedge \tau \rangle$

Typing mixins

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

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

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

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

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

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

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

$$\Gamma \vdash \text{mixin}_{R_2} : \omega \to \langle b : \sigma_2 \rangle$$

so that

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

where $M \circ N \equiv \mathbf{B}\, M\, N =_\beta \lambda x.M(N\, x)$.

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 $\Gamma \not\vdash N_1 : \sigma_3$ we have

$$\Gamma \vdash \text{mixin}_{R_3} : \omega \to \langle a : \sigma_3 \rangle$$

and

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

so that

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

but

$$\Gamma \not\vdash \text{mixin}_{R_1} \circ \text{mixin}_{R_3} : \omega \to \langle a : \sigma_3 \rangle$$

Results

Results

Theorem: subject reduction and expansion

For any $M, N \in \Lambda_R$, such that $M \longrightarrow N$, $\Gamma \vdash M : \sigma \Leftrightarrow \Gamma \vdash N : \sigma$

Results

Theorem: subject reduction and expansion

For any $M, N \in \Lambda_R$, such that $M \longrightarrow N$, $\Gamma \vdash M : \sigma \Leftrightarrow \Gamma \vdash N : \sigma$

Conjecture

The properties of the assignment system w.r.t. the λ-calculus with records and overriding are preserved by adding $\oplus$, in particular:

$$M \longrightarrow^* R \text{ (a record)} \quad \& \quad a \in \ell(R) \quad \Leftrightarrow \quad \exists\, \Gamma, \sigma.\, \Gamma \vdash M : \langle a : \sigma \rangle$$

An extension of FJ with mixins

An extension of FJ with mixins

We extend Featherweight Java by mixins and the compiling relation:

$$(\mathtt{mixin}\ \mathtt{C} = \mathtt{D}\ \mathtt{with}\ \mathtt{R}) \diamond (\mathtt{class}\ \mathtt{E}\ \mathtt{R}') \rightsquigarrow \mathtt{class}\ \mathtt{C}\ (\mathtt{R}' \bullet \mathtt{R})$$

An extension of FJ with mixins

We extend Featherweight Java by mixins and the compiling relation:

$$(\texttt{mixin C} = \texttt{D with R}) \diamond (\texttt{class E R}') \rightsquigarrow \texttt{class C } (\texttt{R}' \bullet \texttt{R})$$

Definition (Mixins interpretation)

$$\begin{array}{rcll} \langle \texttt{class C R} \rangle_{\bar{x}} & = & \langle \texttt{R} \rangle_{\bar{x}} & \\ \langle \texttt{mixin C} = \texttt{D with R} \rangle_{\bar{x}} & = & \lambda z.z \oplus \langle \texttt{R} \rangle_{\bar{x}} & (z \notin \bar{x}) \\ \langle \texttt{T} \diamond \texttt{T}' \rangle_{\bar{x},\bar{y}} & = & (\langle \texttt{T} \rangle_{\bar{x}})\langle \texttt{T}' \rangle_{\bar{y}} & (\bar{x} \cap \bar{y} = \emptyset) \end{array}$$

$$[\![\texttt{T}]\!] = \{\sigma \mid \exists \Gamma.\ \bar{x} \subseteq dom(\Gamma)\ \&\ \Gamma \vdash \langle \texttt{T} \rangle_{\bar{x}} : \sigma\}$$

An extension of FJ with mixins

We extend Featherweight Java by mixins and the compiling relation:

$$(\texttt{mixin C} = \texttt{D with R}) \diamond (\texttt{class E R}') \rightsquigarrow \texttt{class C } (\texttt{R}' \bullet \texttt{R})$$

Definition (Mixins interpretation)

$$\begin{array}{rcll} \langle \texttt{class C R} \rangle_{\bar{x}} & = & \langle \texttt{R} \rangle_{\bar{x}} & \\ \langle \texttt{mixin C} = \texttt{D with R} \rangle_{\bar{x}} & = & \lambda z.z \oplus \langle \texttt{R} \rangle_{\bar{x}} & (z \notin \bar{x}) \\ \langle \texttt{T} \diamond \texttt{T}' \rangle_{\bar{x},\bar{y}} & = & (\langle \texttt{T} \rangle_{\bar{x}}) \langle \texttt{T}' \rangle_{\bar{y}} & (\bar{x} \cap \bar{y} = \emptyset) \end{array}$$

$$[\![\texttt{T}]\!] = \{\sigma \mid \exists \Gamma.\ \bar{x} \subseteq dom(\Gamma)\ \&\ \Gamma \vdash \langle \texttt{T} \rangle_{\bar{x}} : \sigma\}$$

Soundness Theorem

$$\texttt{T} \rightsquigarrow^* \texttt{T}' \implies [\![\texttt{T}]\!] = [\![\texttt{T}']\!]$$

Conclusions

- the calculus $\Lambda_R + \oplus$ is a simple model to represent class inheritance and mixins
- intersection types can be extended to the calculus, while preserving subject reduction and expansion
- intersection types provide a *logical interpretation* for FJ + mixins

Conclusions

- the calculus $\Lambda_R + \oplus$ is a simple model to represent class inheritance and mixins
- intersection types can be extended to the calculus, while preserving subject reduction and expansion
- intersection types provide a *logical interpretation* for FJ + mixins

Future work:

- understand what kind of properties can be expressed in this way
- try to extend the semi algorithm for "synthesis by inhabitation" from combinatory logic to mixins composition.