Dynamic Logic with Non-rigid Functions

A Basis for Object-oriented Program Verification

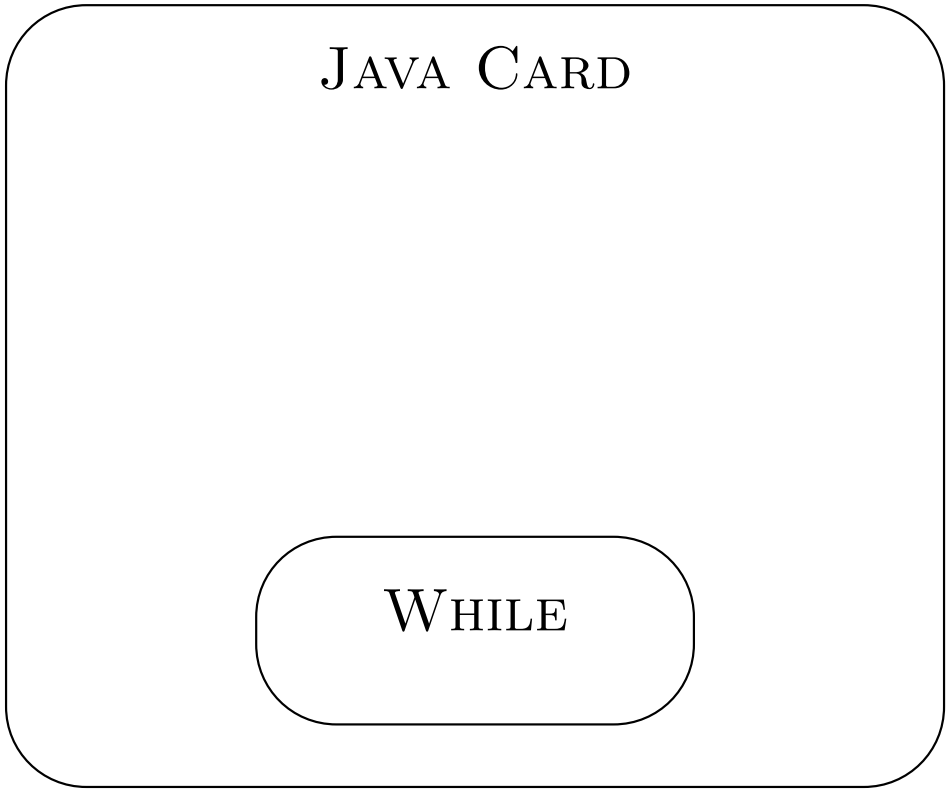
Bernhard Beckert[1] André Platzer[2]

[1]University of Koblenz-Landau, Department of Computer Science
beckert@uni-koblenz.de

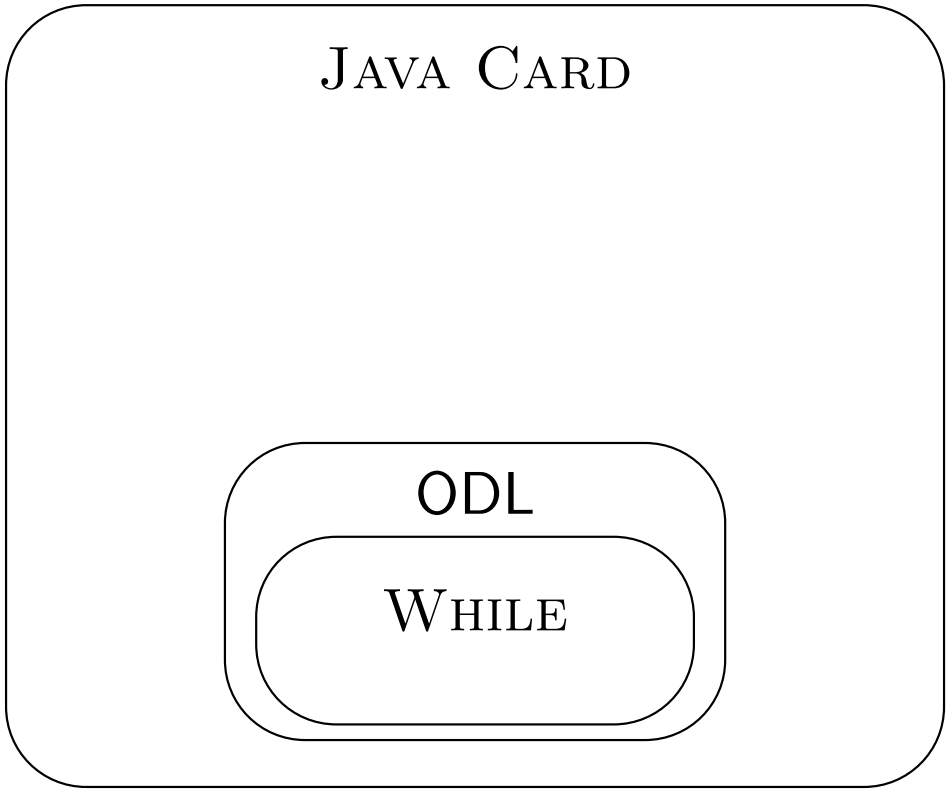
[2]University of Oldenburg, Department of Computing Science
platzer@informatik.uni-oldenburg.de

3rd International Joint Conference on Automated Reasoning

Object-oriented Dynamic Logic

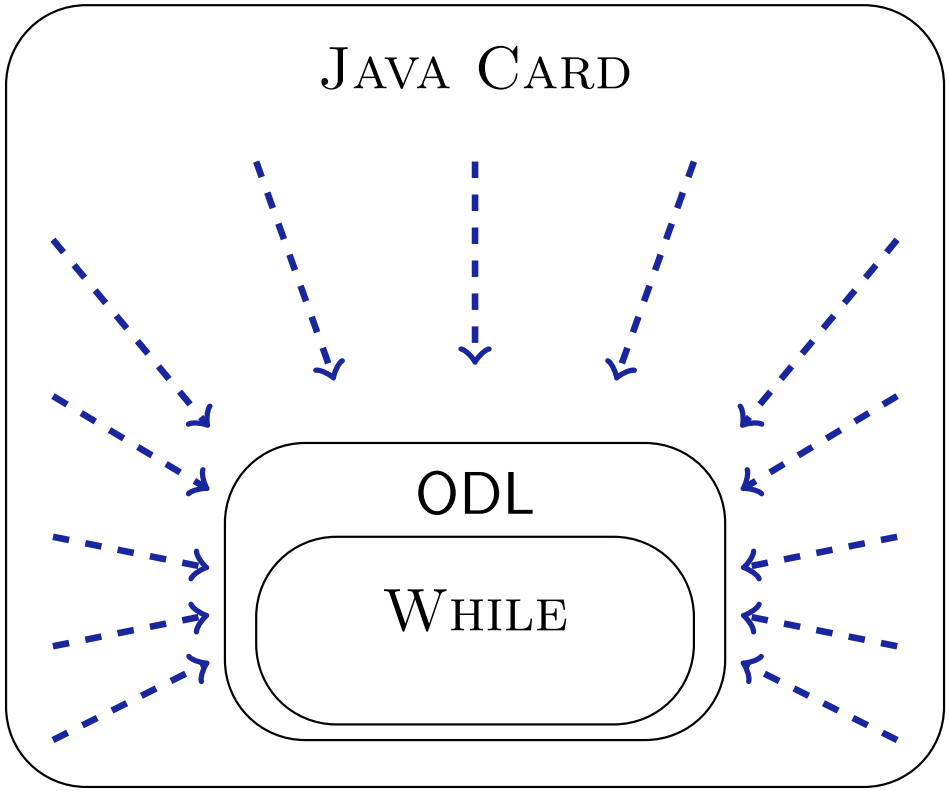


Object-oriented Dynamic Logic



- ODL only contains essentials of object-orientation.

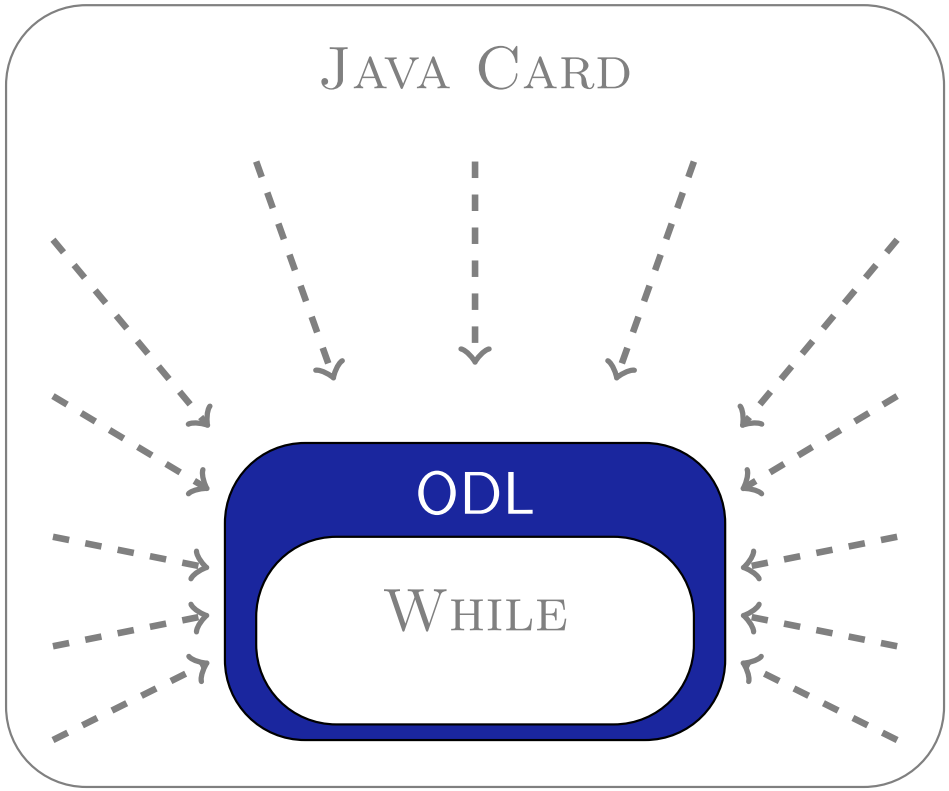
Object-oriented Dynamic Logic



- ✓ Theoretical investigations
- ✓ Program verification

- ODL only contains essentials of object-orientation.
- "Natural" representation of object-orientation.

Object-oriented Dynamic Logic



- ✓ Theoretical investigations
- ✓ Program verification

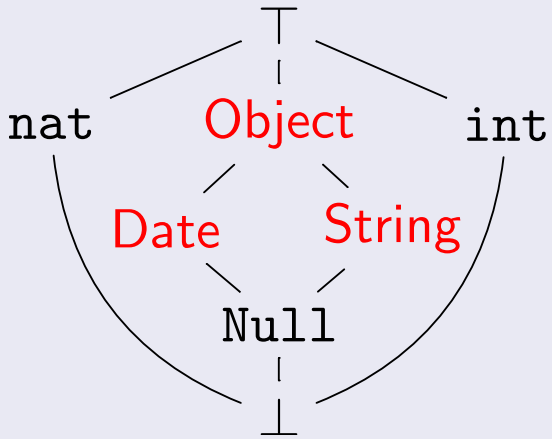
- ODL only contains essentials of object-orientation.
- "Natural" representation of object-orientation.

Outline

Outline

The Logic ODL

Definition (Type system)



The Logic ODL

Definition (Object enumerator symbols)

$\mathtt{obj}_{\mathtt{C}} : \mathtt{nat} \longrightarrow \mathtt{C}$ (disjoint bijections for object-types $\mathtt{C}$)

Example

$\mathtt{obj}_{\mathtt{C}}(3)$

Definition (Non-rigid function symbols)

. . . change value during program execution & represent object attributes.

Example

$\boxed{x.a} \rightsquigarrow \boxed{a(x)}$

The Logic ODL

Definition (Formulas ϕ)

$\neg, \wedge, \vee, \rightarrow, \leftrightarrow, \forall, \exists, \doteq$	(first-order part)
$[\alpha]\phi, \quad \langle\alpha\rangle\phi$	(dynamic part)

Definition (Programs α)

$\alpha;\gamma, \quad \texttt{if}(\phi)\,\alpha\,\texttt{else}\,\gamma, \quad \texttt{while}(\phi)\,\alpha$	(control-structure)
$f(t) := t'$	(state update)

▸ Details

The Logic ODL

Definition (Formulas ϕ)

$\neg, \wedge, \vee, \rightarrow, \leftrightarrow, \forall, \exists, \doteq$ (first-order part)

$[\alpha]\phi, \quad \langle\alpha\rangle\phi$ (dynamic part)

Definition (Programs α)

$\alpha;\gamma, \quad \texttt{if}(\phi)\,\alpha\,\texttt{else}\,\gamma, \quad \texttt{while}(\phi)\,\alpha$ (control-structure)

$f(t) := t', g(r) := r'$ (state update)

▸ Details

Outline

JAVA ⇝ ODL

Simple & natural translation of

- Object creation
- Dynamic dispatch
- Side-effects
- Exception handling
- Inner classes

JAVA $\rightsquigarrow$ ODL: Object Creation

Example (Transformation)

$$\boxed{\mathtt{x} := \mathtt{new}\ \mathtt{C}()} \quad \rightsquigarrow \quad \boxed{\begin{array}{l} \mathtt{x} := \mathtt{obj}_{\mathtt{C}}(\mathit{next}_{\mathtt{C}}), \\ \mathit{next}_{\mathtt{C}} := \mathit{next}_{\mathtt{C}} + 1 \end{array}}$$

JAVA $\rightsquigarrow$ ODL: Object Creation

Example (Transformation)

$$\boxed{\mathtt{x} := \mathtt{new}\ \mathtt{C}()} \quad \rightsquigarrow \quad \boxed{\begin{array}{l}\mathtt{x} := \mathtt{obj}_{\mathtt{C}}(\mathit{next}_{\mathtt{C}}),\\ \mathit{next}_{\mathtt{C}} := \mathit{next}_{\mathtt{C}} + 1\end{array}}$$

- Object identity "new $\neq$ new"

Example

$$\boxed{\begin{array}{l}\mathtt{x} := \mathtt{new}\ \mathtt{C}();\\ \\ \mathtt{y} := \mathtt{new}\ \mathtt{C}();\end{array}} \quad \begin{array}{l}\rightsquigarrow\\ \\ \rightsquigarrow\end{array} \quad \boxed{\begin{array}{l}\mathtt{x} := \mathtt{obj}_{\mathtt{C}}(\mathit{next}_{\mathtt{C}});\\ //\mathit{next}_{\mathtt{C}} := \mathit{next}_{\mathtt{C}} + 1\\ \mathtt{y} := \mathtt{obj}_{\mathtt{C}}(\mathit{next}_{\mathtt{C}} + 1)\\ //\mathtt{x} \neq \mathtt{y}\end{array}}$$

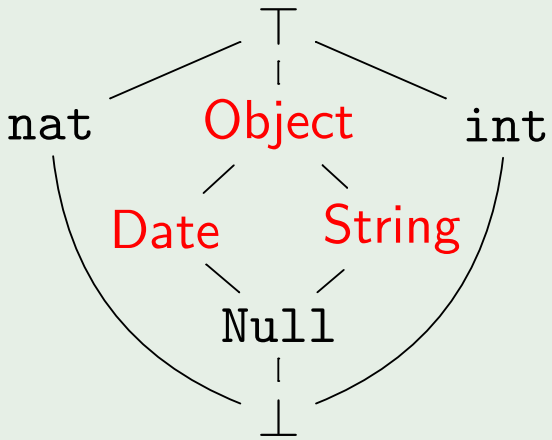
JAVA $\rightsquigarrow$ ODL: Object Creation

Example (Transformation)

$$\boxed{\mathtt{x} := \mathtt{new}\ \mathtt{C}()} \quad \rightsquigarrow \quad \boxed{\begin{array}{l} \mathtt{x} := \mathtt{obj}_{\mathtt{C}}(\mathit{next}_{\mathtt{C}}), \\ \mathit{next}_{\mathtt{C}} := \mathit{next}_{\mathtt{C}} + 1 \end{array}}$$

- Object identity "new $\neq$ new"
- Dynamic type checks

Example



$$\boxed{t\ \mathtt{instanceof}\ \mathtt{String}}$$

$$\Updownarrow$$

$$\boxed{\exists n{:}\mathtt{nat}\ t \doteq \mathtt{obj}_{\mathtt{String}}(n)}$$

Outline

Calculus: State Updates (merge & promote)

- Merge updates

$$\langle \mathcal{U} \rangle \langle f(t) := t' \rangle \phi \quad \rightsquigarrow \quad \left\langle \mathcal{U},\ f(\langle \mathcal{U} \rangle t) := \langle \mathcal{U} \rangle t' \right\rangle \phi$$

"Last change is most recent"

Example

$$\langle g(a) := t \rangle \langle f(g(a)) := h(g(a)) \rangle \phi \quad \rightsquigarrow \quad \langle g(a) := t, f(t) := h(t) \rangle \phi$$

Calculus: State Updates (merge & promote)

- Merge updates

$$\langle \mathcal{U} \rangle \langle f(t) := t' \rangle \phi \quad \rightsquigarrow \quad \left\langle \mathcal{U},\ f(\langle \mathcal{U} \rangle t) := \langle \mathcal{U} \rangle t' \right\rangle \phi$$

"Last change is most recent"

- Promote updates

$$\langle f(t) := t',\ g(t) := c,\ f(s) := s' \rangle\ f(a)$$

Calculus: State Updates (merge & promote)

- Merge updates

$$\langle \mathcal{U} \rangle \langle f(t) := t' \rangle \phi \quad \rightsquigarrow \quad \left\langle \mathcal{U},\ f(\langle \mathcal{U} \rangle t) := \langle \mathcal{U} \rangle t' \right\rangle \phi$$

"Last change is most recent"

- Promote updates

$$\underbrace{\langle f(t) := t',\ g(t) := c,\ f(s) := s' \rangle}_{f}\ f(a)$$

Calculus: State Updates (merge & promote)

- Merge updates

$$\langle \mathcal{U} \rangle \langle f(t) := t' \rangle \phi \quad \rightsquigarrow \quad \left\langle \mathcal{U},\ f(\langle \mathcal{U} \rangle t) := \langle \mathcal{U} \rangle t' \right\rangle \phi$$

"Last change is most recent"

- Promote updates

$$\frac{\langle f(t) := t',\ g(t) := c,\ f(s) := s' \rangle\ f(a)}{f}$$
$$\langle f(t) := t',\ f(s) := s' \rangle\ f(a)$$

Calculus: State Updates (merge & promote)

- Merge updates

$$\langle \mathcal{U} \rangle \langle f(t) := t' \rangle \phi \quad \rightsquigarrow \quad \left\langle \mathcal{U},\ f(\langle \mathcal{U} \rangle t) := \langle \mathcal{U} \rangle t' \right\rangle \phi$$

"Last change is most recent"

- Promote updates

$$\underbrace{\langle f(t) := t',\ g(t) := c,\ f(s) := s' \rangle}_{f}\ f(a)$$

$$\underbrace{\langle f(t) := t',\ f(s) := s' \rangle}\ f(a)$$

$$\textit{if}\ s \doteq a\ \textit{then}\ s'\ \textit{else}\ \ldots\ \textit{fi}$$

Calculus: State Updates (merge & promote)

- Merge updates

$$\langle \mathcal{U} \rangle \langle f(t) := t' \rangle \phi \quad \rightsquigarrow \quad \left\langle \mathcal{U},\ f(\langle \mathcal{U} \rangle t) := \langle \mathcal{U} \rangle t' \right\rangle \phi$$

"Last change is most recent"

- Promote updates

$$\underbrace{\langle f(t) := t',\ g(t) := c,\ f(s) := s' \rangle\ f(a)}_{f}$$

$$\underbrace{\langle f(t) := t',\ f(s) := s' \rangle\ f(a)}$$

$$\mathit{if}\ s \doteq a\ \mathit{then}\ s'\ \mathit{else}\ \mathit{if}\ t \doteq a\ \mathit{then}\ t'\ \mathit{else}\ f(a)\ \mathit{fi}\ \mathit{fi}$$

Calculus: State Updates (merge & promote)

- Merge updates

$$\langle \mathcal{U} \rangle \langle f(t) := t' \rangle \phi \quad \rightsquigarrow \quad \left\langle \mathcal{U},\ f(\langle \mathcal{U} \rangle t) := \langle \mathcal{U} \rangle t' \right\rangle \phi$$

"Last change is most recent"

- Promote updates

$$\overbrace{\langle f(t) := t',\ g(t) := c,\ f(s) := s' \rangle}^{\mathcal{U}}\ f(a)$$

f

$$\langle f(t) := t',\ f(s) := s' \rangle \qquad f(a)$$

$$\mathit{if}\ s \doteq \langle \mathcal{U} \rangle a\ \mathit{then}\ s'\ \mathit{else}\ \mathit{if}\ t \doteq \langle \mathcal{U} \rangle a\ \mathit{then}\ t'\ \mathit{else}\ f(\langle \mathcal{U} \rangle a)\ \mathit{fi}\ \mathit{fi}$$

Calculus: First-order Part

18 rules

$$\frac{\vdash A}{\neg A \vdash} \qquad \frac{A, B \vdash}{A \wedge B \vdash} \qquad \frac{A \vdash \qquad B \vdash}{A \vee B \vdash} \qquad \frac{\vdash A \qquad B \vdash}{A \rightarrow B \vdash}$$

$$\frac{A \vdash}{\vdash \neg A} \qquad \frac{\vdash A \qquad \vdash B}{\vdash A \wedge B} \qquad \frac{\vdash A, B}{\vdash A \vee B} \qquad \frac{A \vdash B}{\vdash A \rightarrow B}$$

$$\frac{\vdash A^X_x}{\vdash \forall x\, A} \qquad \frac{A^t_x, \forall x\, A \vdash}{\forall x\, A \vdash} \qquad \frac{A^X_x \vdash}{\exists x\, A \vdash} \qquad \frac{\vdash A^t_x, \exists x\, A}{\vdash \exists x\, A}$$

$$\frac{}{A \vdash A} \qquad \frac{\Gamma^{t'}_t, t \doteq t' \vdash \Delta^{t'}_t}{\Gamma, t \doteq t' \vdash \Delta} \qquad \frac{}{\vdash t \doteq t}$$

$$\frac{A \vdash \qquad \vdash A}{\vdash} \qquad \frac{\Gamma^{t'}_t, t' \doteq t \vdash \Delta^{t'}_t}{\Gamma, t' \doteq t \vdash \Delta} \qquad \frac{\vdash \phi(0) \qquad \phi(X) \vdash \phi(X{+}1)}{\vdash \forall n\, \phi(n)}$$

Calculus: Program Logic Part

12 rules

$$\frac{\langle\alpha\rangle\langle\gamma\rangle\phi}{\langle\alpha;\gamma\rangle\phi}$$

$$\frac{}{\vdash \mathtt{obj}_{\mathtt{C}}(i) \doteq \mathtt{obj}_{\mathtt{C}}(j) \rightarrow i \doteq j}$$

$$\frac{(e \rightarrow \langle\alpha\rangle\phi) \wedge (\neg e \rightarrow \langle\gamma\rangle\phi)}{\langle \mathtt{if}(e)\,\alpha\,\mathtt{else}\,\gamma\rangle\phi}$$

$$\frac{}{\vdash \neg(\mathtt{obj}_{\mathtt{C}}(i) \doteq \mathtt{obj}_{\mathtt{D}}(j))}$$

$$\frac{(e \rightarrow \phi(t)) \wedge (\neg e \rightarrow \phi(t'))}{\phi(\mathit{if}\, e\, \mathit{then}\, t\, \mathit{else}\, t'\, \mathit{fi})}$$

$$\frac{}{\vdash \forall o : \mathtt{C}\ (o\ \mathtt{instanceof}\ \mathtt{C} \vee o \doteq \mathtt{null})}$$

$$\frac{\langle \mathtt{if}(e)\,\{\alpha; \mathtt{while}(e)\,\alpha\}\rangle\phi}{\langle \mathtt{while}(e)\,\alpha\rangle\phi}$$

$$\frac{\Gamma \vdash \langle\mathcal{U}\rangle p, \Delta \qquad p, e \vdash [\alpha]p \qquad p, \neg e \vdash \phi}{\Gamma \vdash \langle\mathcal{U}\rangle[\mathtt{while}(e)\,\alpha]\phi, \Delta}$$

$$\frac{A \vdash B}{\exists x\, A \vdash \exists x\, B}$$

$$\frac{A \vdash B}{\langle\alpha\rangle A \vdash \langle\alpha\rangle B}$$

$$\langle\mathcal{U}\rangle f(u) \rightsquigarrow \mathit{if}\, s_{i_r} \doteq \langle\mathcal{U}\rangle u\, \mathit{then}\, t_{i_r}\, \mathit{else} \ldots \mathit{if}\, s_{i_1} \doteq \langle\mathcal{U}\rangle u\, \mathit{then}\, t_{i_1}\, \mathit{else}\, f(\langle\mathcal{U}\rangle u)\, \mathit{fi} \ldots \mathit{fi}$$

$$\langle\tilde{\mathcal{U}}\rangle\langle\mathcal{U}\rangle\phi \rightsquigarrow \{\tilde{\mathcal{U}}, f_1(\langle\tilde{\mathcal{U}}\rangle s_1) := \langle\tilde{\mathcal{U}}\rangle t_1, \ldots, f_n(\langle\tilde{\mathcal{U}}\rangle s_n) := \langle\tilde{\mathcal{U}}\rangle t_n\}\phi$$

Verification Example

```
class E { static int g; int id;
    E generate() {E r=new E(); r.id=g; g=g+5; return r;}}
```

$$\forall x{:}\mathrm{E}\,(x.\mathrm{id} < g \rightarrow [\mathsf{generate}]\ (x.\mathrm{id} < r.\mathrm{id}))$$

Verification Example

```
class E { static int g; int id;
    E generate() {E r=new E(); r.id=g; g=g+5; return r;}}
```

$$\forall x{:}\mathrm{E}\,(x.\mathrm{id} < g \rightarrow [\mathsf{generate}]\ (x.\mathrm{id} < r.\mathrm{id}))$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{X.\mathrm{id} < g, \neg o(n) \doteq X \vdash X.\mathrm{id} < g}\,{*} \qquad \dfrac{}{X.\mathrm{id} < g, o(n) \doteq X \vdash g < g}\,{\ldots}}{X.\mathrm{id} < g \vdash (\neg o(n) \doteq X \rightarrow X.\mathrm{id} < g) \wedge (o(n) \doteq X \rightarrow g < g)}}{X.\mathrm{id} < g \vdash (\mathit{if}\, o(n) \doteq X\, \mathit{then}\, g\, \mathit{else}\, X.\mathrm{id}\, \mathit{fi}) < g}}{X.\mathrm{id} < g \vdash \langle r := o(n), n := n{+}1, o(n).\mathrm{id} := g, g := g + 5\rangle\ (X.\mathrm{id} < r.\mathrm{id})}}{X.\mathrm{id} < g \vdash \langle r := o(n), n := n{+}1, o(n).\mathrm{id} := g\rangle[g := g + 5]\ (X.\mathrm{id} < r.\mathrm{id})}}{X.\mathrm{id} < g \vdash \langle r := o(n), n := n{+}1\rangle[r.\mathrm{id} := g][g := g + 5]\ (X.\mathrm{id} < r.\mathrm{id})}}{X.\mathrm{id} < g \vdash \langle r := o(n), n := n{+}1\rangle[r.\mathrm{id} := g; g := g + 5]\ (X.\mathrm{id} < r.\mathrm{id})}}{X.\mathrm{id} < g \vdash [\alpha]\ (X.\mathrm{id} < r.\mathrm{id})}}{\vdash X.\mathrm{id} < g \rightarrow [\alpha]\ (X.\mathrm{id} < r.\mathrm{id})}}{\vdash \forall x{:}\mathrm{E}\,(x.\mathrm{id} < g \rightarrow [\alpha]\ (x.\mathrm{id} < r.\mathrm{id}))}$$

Relative Completeness

Theorem (Relative completeness)

ODL calculus is complete w.r.t. first-order arithmetic:

$$\models \phi \quad \textit{implies} \quad \mathit{Taut}_{\mathit{Arith}} \vdash \phi$$

▸ Proof Outline

Outline

Future Work

- Verification components as add-on rules
- Parametric genericity

Conclusions

- ODL is essentials-only object-oriented dynamic logic:
  1. object type system
  2. object enumerators
  3. non-rigid functions
- Natural translation JAVA $\rightsquigarrow$ ODL.
- Calculus proven sound & relatively complete w.r.t. arithmetic.

Outline

The Logic ODL: Syntax

Definition (Type system)

Type lattice with natural numbers **N** and finite subtypes

The Logic ODL: Semantics

Definition (Semantics of programs)

1. $(s, s') \in \rho_{I,\beta}(f_1(t_1^1, \ldots, t_1^{k_1}) := t_1, \ldots, f_n(t_n^1, \ldots, t_n^{k_n}) := t_n) \iff$
   - $s = s_0, s' = s_n$, and
   - $s_i = s_{i-1}[f_i(val_{I,\beta}(s, t_i^1), \ldots, val_{I,\beta}(s, t_i^{k_i})) \mapsto val_{I,\beta}(s, t_i)]$ $(1 \leq i \leq n)$.
2. $(s, s') \in \rho_{I,\beta}(\alpha; \gamma) \iff (s, u) \in \rho_{I,\beta}(\alpha)$ and $(u, s') \in \rho_{I,\beta}(\gamma)$ for some state u.
3. $(s, s') \in \rho_{I,\beta}(\texttt{if}(\phi)\,\alpha\,\texttt{else}\,\gamma) \iff$
   - $val_{I,\beta}(s, \phi) = \mathit{true}$ and $(s, s') \in \rho_{I,\beta}(\alpha)$, *or*
   - $val_{I,\beta}(s, \phi) = \mathit{false}$ and $(s, s') \in \rho_{I,\beta}(\gamma)$.
4. $(s, s') \in \rho_{I,\beta}(\texttt{while}(\phi)\,\alpha)$ iff there are $n \in \mathbb{N}$ and $s{=}s_0, \ldots, s_n{=}s'$
   - for $0 \leq i < n$, $val_{I,\beta}(s_i, \phi) = \mathit{true}$ and $(s_i, s_{i+1}) \in \rho_{I,\beta}(\alpha)$, and
   - $val_{I,\beta}(s_n, \phi) = \mathit{false}$.

Outline

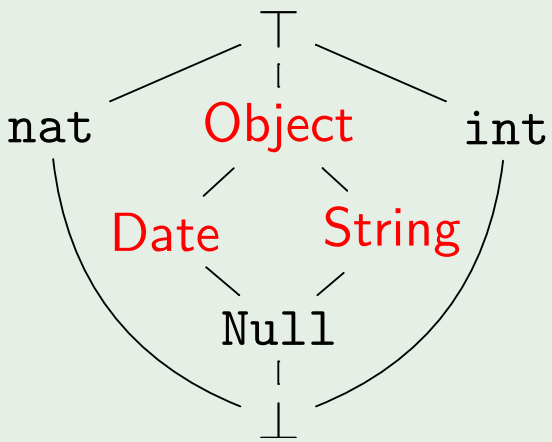
JAVA $\rightsquigarrow$ ODL: Object Creation

Example (Transformation)

$$\boxed{\mathtt{x} := \mathtt{new}\ \mathtt{C}()} \quad \rightsquigarrow \quad \boxed{\begin{array}{l} \mathtt{x} := \mathtt{obj}_{\mathtt{C}}(\mathit{next}_{\mathtt{C}}), \\ \mathit{next}_{\mathtt{C}} := \mathit{next}_{\mathtt{C}} + 1 \end{array}}$$

- Object identity "new $\neq$ new"
- Dynamic type checks

Example



$$\boxed{t\ \mathtt{instanceof}\ \mathtt{Object}}$$

$$\Downarrow$$

$$\boxed{\begin{array}{l} \exists n{:}\mathtt{nat}\ \big(\quad t \doteq \mathtt{obj}_{\mathtt{Date}}(n) \\ \qquad\qquad \vee t \doteq \mathtt{obj}_{\mathtt{String}}(n) \\ \qquad\qquad \vee t \doteq \mathtt{obj}_{\mathtt{Object}}(n)\big) \end{array}}$$

JAVA $\rightsquigarrow$ ODL: Object Creation

Example (Transformation)

$$\boxed{\mathtt{x} := \mathtt{new}\ \mathtt{C}()} \quad \rightsquigarrow \quad \boxed{\begin{array}{l} \mathtt{x} := \mathtt{obj}_{\mathtt{C}}(\mathit{next}_{\mathtt{C}}), \\ \mathit{next}_{\mathtt{C}} := \mathit{next}_{\mathtt{C}} + 1 \end{array}}$$

- Object identity "new $\neq$ new"
- Dynamic type checks

JAVA $\rightsquigarrow$ ODL: Side-effects

```
a [ i++] = b-- + b
```

$$vi := i;\ \ i := i + 1;\ \ vb := b;\ \ b := b - 1;\ \ a(vi) := vb + b$$

$$\wr$$

$$i := i + 1, b := b - 1, a(i) := b + (b - 1)$$

Return

JAVA ⇝ ODL: Exception Handling

```
try { while (d != 0)
        {if (d < 0) {throw new RangeEx(d);} else {d=d-1;}}
      /* do something */
    } catch (RangeEx r) {/* handle range */}
```

```
Exception r = null;
while (r == null && d != 0)
  {if (d < 0) {r = new RangeEx(d);} else {d=d-1;}}
if (r == null) {/* do something */}
else if (r instanceof RangeEx) {/* handle range */}
else {return r; /* pass up the call trace */}
```

Return

JAVA ⇝ ODL: Dynamic Dispatch

```
class Car { int follow(Car d) {...} }
class Van extends Car { int follow(Car d) {...} }
... return b.follow(d);
```

```
class Car { int Car_follow(Car d) {...} }
class Van extends Car { int Van_follow(Car d) {...} }
...  if(b instanceof Van) {return ((Van)b).Van_follow(d);}
else if(b instanceof Car) {return ((Car)b).Car_follow(d);}
else {/* cannot happen when all types are known */}
```

Type-casts expressible as

$$\exists v{:}\mathrm{Van}\, v \doteq b$$

Return

Outline

Calculus: Object Creation

$$t\,\texttt{instanceof}\,\texttt{C} \;:=\; \exists n:\texttt{nat} \bigvee_{\texttt{Null}<\tau\leq\texttt{C}} t \doteq \texttt{obj}_\tau(n)$$

(R31) $$\frac{}{\vdash \forall o:\texttt{C}\,(o\,\texttt{instanceof}\,\texttt{C} \vee o \doteq \texttt{null})}$$

(R32) $$\frac{}{\vdash \texttt{obj}_\texttt{C}(i) \doteq \texttt{obj}_\texttt{C}(j) \rightarrow i \doteq j}$$

Calculus: State Updates (merge & promote)

- Merge updates

$$\langle \mathcal{U} \rangle \langle f(t) := t' \rangle \phi \quad \rightsquigarrow \quad \left\langle \mathcal{U},\ f(\langle \mathcal{U} \rangle t) := \langle \mathcal{U} \rangle t' \right\rangle \phi$$

"Last change is most recent"

- Promote updates

$$\overbrace{\langle f(t) := t',\ g(t) := c,\ f(s) := s' \rangle}^{\mathcal{U}}\ f(a)$$

$$\underbrace{\qquad\qquad\qquad\qquad}_{f}$$

$$\langle f(t) := t',\ f(s) := s' \rangle \qquad f(a)$$

$$\underbrace{\qquad\qquad\qquad\qquad}$$

$$\mathit{if}\ s \doteq \langle \mathcal{U} \rangle a\ \mathit{then}\ s'\ \mathit{else if}\ t \doteq \langle \mathcal{U} \rangle a\ \mathit{then}\ t'\ \mathit{else}\ f(\langle \mathcal{U} \rangle a)\ \mathit{fi fi}$$

Calculus: State Updates (merge & promote)

- Merge updates

$$\langle \mathcal{U} \rangle \langle f(t) := t' \rangle \phi \quad \rightsquigarrow \quad \left\langle \mathcal{U},\ f(\langle \mathcal{U} \rangle t) := \langle \mathcal{U} \rangle t' \right\rangle \phi$$

"Last change is most recent"

- Promote updates

$$\phi\Big(\overbrace{\langle f(t) := t',\ g(t) := c,\ f(s) := s' \rangle}^{\mathcal{U}}\ \underbrace{f(a)}_{f} \Big)$$

$$\phi\Big(\langle f(t) := t',\ f(s) := s' \rangle\ f(a) \Big)$$

$$\phi\Big(\mathit{if}\ s \doteq \langle \mathcal{U} \rangle a\ \mathit{then}\ s'\ \mathit{else}\ \mathit{if}\ t \doteq \langle \mathcal{U} \rangle a\ \mathit{then}\ t'\ \mathit{else}\ f(\langle \mathcal{U} \rangle a)\ \mathit{fi}\ \mathit{fi} \Big)$$

Calculus: State Updates (merge & promote)

- Merge updates

$$\langle \mathcal{U} \rangle \langle f(t) := t' \rangle \phi \quad \rightsquigarrow \quad \left\langle \mathcal{U},\ f(\langle \mathcal{U} \rangle t) := \langle \mathcal{U} \rangle t' \right\rangle \phi$$

"Last change is most recent"

- Promote updates

$$\phi\Big(\overbrace{\langle f(t) := t',\ g(t) := c,\ f(s) := s' \rangle}^{\mathcal{U}}\ f(a) \Big)$$

$$\phi\Big(\underbrace{\langle f(t) := t',\ f(s) := s' \rangle}_{f}\ f(a) \Big)$$

$$\phi\Big(\textit{if } s \doteq \langle \mathcal{U} \rangle a \textit{ then } s' \textit{ else if } t \doteq \langle \mathcal{U} \rangle a \textit{ then } t' \textit{ else } f(\langle \mathcal{U} \rangle a) \textit{ fi fi} \Big)$$

$$\Big(s \doteq \langle \mathcal{U} \rangle a \rightarrow \phi(s') \Big) \ \wedge \ \Big(s \neq \langle \mathcal{U} \rangle a \rightarrow \ldots \phi(f(\langle \mathcal{U} \rangle a)) \Big)$$

Calculus: State Updates

$\langle \mathcal{U} \rangle f(u) \rightsquigarrow$
$\mathit{if}\, s_{i_r} \doteq \langle \mathcal{U} \rangle u\, \mathit{then}\, t_{i_r}\, \mathit{else} \ldots \mathit{if}\, s_{i_1} \doteq \langle \mathcal{U} \rangle u\, \mathit{then}\, t_{i_1}\, \mathit{else}\, f(\langle \mathcal{U} \rangle u)\, \mathit{fi} \ldots \mathit{fi}$
where $i_1 < \cdots < i_r$ are all those indices with $f_{i_j} = f$, for some $r \geq 0$

Example (State updates with aliasing)

$$
\begin{array}{ll}
 & \langle f(s) := t \rangle g(f(r)) \\
\rightsquigarrow & g\big(\langle f(s) := t \rangle f(r)\big) \\
\rightsquigarrow & g\big(\mathit{if}\, s \doteq r\, \mathit{then}\, t\, \mathit{else}\, f(r)\, \mathit{fi}\big) \\
\text{“}\rightsquigarrow\text{”} & \big(s \doteq r \rightarrow g(t)\big) \wedge \\
 & \big(s \neq r \rightarrow g(f(r))\big)
\end{array}
$$

(R33) $\quad \langle \tilde{\mathcal{U}} \rangle \langle \mathcal{U} \rangle \phi \rightsquigarrow \left\langle \tilde{\mathcal{U}}, f_1(\langle \tilde{\mathcal{U}} \rangle s_1) := \langle \tilde{\mathcal{U}} \rangle t_1, \ldots, f_n(\langle \tilde{\mathcal{U}} \rangle s_n) := \langle \tilde{\mathcal{U}} \rangle t_n \right\rangle \phi$

Relative Completeness Proof

$$\models \phi \quad \text{implies} \quad \mathsf{Taut}_{\mathbf{N}} \vdash \phi$$

Proof.

1. propositionally complete
2. first-order complete
3. first-order expressible: $\forall \phi \ \exists F \in \mathrm{FOL} \quad \models \phi \leftrightarrow F$
4. term rewrites are Noetherian
5. (rel.) complete for first-order $F \rightarrow \langle \alpha \rangle G$
6. (rel.) complete for first-order dynamic typing

Return

Use Cases in JAVA CARD DL

- Use ODL object enumerators for object creation.
- Use quicker and rel. complete ODL within KeY in automatic verification scenarios. (Trafo combined with compiler construction technology)
- Import simpler ODL calculus into JAVA CARD DL, for formulas that are "sufficiently" ODL.