Notational Variations

Too many syntactic variations of one method idea in most programming languages

- Functional form
  \[ a := \text{sort}(b); \quad e := \text{getElement}(b,i); \quad b := \text{setElement}(b,i,e); \]

- Imperative form
  \[ \text{sort}(b,a); \quad \text{sort}(a,b); \]
  \[ a := b; \quad a := \text{sort}(a); \]

- Imperative, in situ form
  \[ \text{sort}(a); \quad a := \text{sort}(a); \]

- Program form
  \[ \text{unix}\% \text{sort } b > a \]

Solution

- Normalise notation for user
  \[ a := \text{sort}(b); \quad e := b[i]; \quad b[i] := e; \]

- Allow full freedom for implementor
  \[ a := \text{sort}(a); \quad b := \text{getElement}(b,i,e); \quad b := \text{setElement}(b,i,e); \]

STS provides link

- Envelope implemented code in standardised functional/program form
- Rewrite from user invoked form to actually implemented form (*mutification*)
User defined / Domain Specific Optimisation Rules

Allow user to write simple code

// central value ignoring lower and upper quartile
r := ( sort(b)[length(b)/4.0] + sort(b)[3*length(b)/4.0] ) / 2.0;

Transform code using userdefined rewrite rules and mutification
r := ( sort(b)[length(b)/4.0] + sort(b)[3*length(b)/4.0] ) / 2.0;

a1 := partition(b,length(b)/4.0);
a2 := partition(b,3*length(b)/4.0);
r := ( a1+a2 ) / 2.0;

Partition b'.setUpPartition(b);
b'.partition(length(b)/4.0,a1); b'.partition(3*length(b)/4.0,a2);
r := ( a1+a2 ) / 2.0;

User defined rewrite rules
• sort(b)[x] partition(b,x)
• a1 := partition(b,x); a2 := partition(b,y);
  Partition b'.setUpPartition(b); b'.partition(x,a1); b'.partition(y,a2);

Summary

Source Transformation Systems may

• Easily support
  • Multiple notations for method calls
  • Methods as programs

• Exploit software specifications as rewrite rules for
  • Optimisations
  • Views of semantically related methods

• Provide software organisation principles across programming languages
  • ADT and class encapsulation
  • Invasive composition

All this gives substantial software engineering benefits – at low costs?

Can we develop STS as language independent SE support systems?