Hermion - Exploiting Runtime Information in the IDE

David Röthlisberger
Software Composition Group
University of Berne
IDEs focus on static structure

- Message sends?
- Variable accesses?
- Dynamic references?
- Polymorphism?
- Late binding?
Example: Implementors of a Method

OBColumn >> children

^fan children

Static implementors

Dynamic implementors
Dynamic Information I

- Precise knowledge about senders, implementors of methods
- Often just one single candidate

- But we can do even more!
Dynamic Information II

- Precise type information for variables:

- Dynamic references:

- Polymorphism becomes visible:
Integrating the Information I

- Directly embedded in source code:

```ruby
nodeForDropEvent: evt inMorph: pluggableListMorph
| index item label |
index := pluggableListMorph rowAtLocation: evt position.
index = 0 ifTrue: [↑ nil].
item := pluggableListMorph listMorph item: index.
label := item contents asString withBlanksTrimmed.
self children
    detect: [:child | child displayString withBlanksTrimmed = label]
ifNone: [nil]
```
Integrating the Information II

Embed dynamic tools tightly in IDE:
Hermion Demo
How to Gather the Information?

- Message sends, variable accesses
- I.e. sub-method elements
- But: Too much data!
  (up to millions of events)
- Precise selection of desired information crucial
- Reflectivity
Reflectivity

- Precisely select where reifications should occur, eg. only in specific classes
- Selection done in IDE
Defining Reifications

Links for sends and variables:

```ruby
sendLink := GPLink new metaObject: self;
selector: #message:receiver:args:
control: #before
arguments: #(node receiver arguments).
```

```ruby
varLink := GPLink new metaObject: self;
selector: #variable:value:
control: #before
arguments: #(node value).
```

self refers to the collector metaobject
Installing the Links

\[aMethod\] sends do: [[:send | send link: sendLink]].

\[aMethod\] variableReads do: [[:var | var link: varLink]].

- At runtime the information is collected in a database
- The IDE queries this database to display the dynamic information
Hermion - Schema
Hermion - Features

- Analysis of runtime behavior
- Immediate presentation of gathered information
- Embedded in traditional IDE tools, enhancing and enriching them
- No gap between runtime analysis and IDE
Summary

- Dynamic information integrated in the IDE
- Eases navigation and understanding of software systems
- Bridges the gap between analysis and development tools