Onspect: Ontology Based Aspects

Parisa Rashidi,
Roger T. Alexander

Summary

- Onspect: Ontology based aspects
  - Based on semantic vs. syntax
    - Ideas from semantic web

Background: AOP & shortcomings …

- **Based on syntax:** name patterns
  - call(void Point.make*(..))
- Problems:
  - Naming Conflicts
  - Fragile pointcuts
  - Semantics dependent on naming
  - Concerns are **semantic**, not syntactic
- Limited expressiveness power
- compare to databases queries
  - Procedural: how to traverse records
  - Declarative: what data I want
- Pointcuts
  - Syntax based: how to trigger a joinpoint
  - Semantic based: what behavior I want to trigger

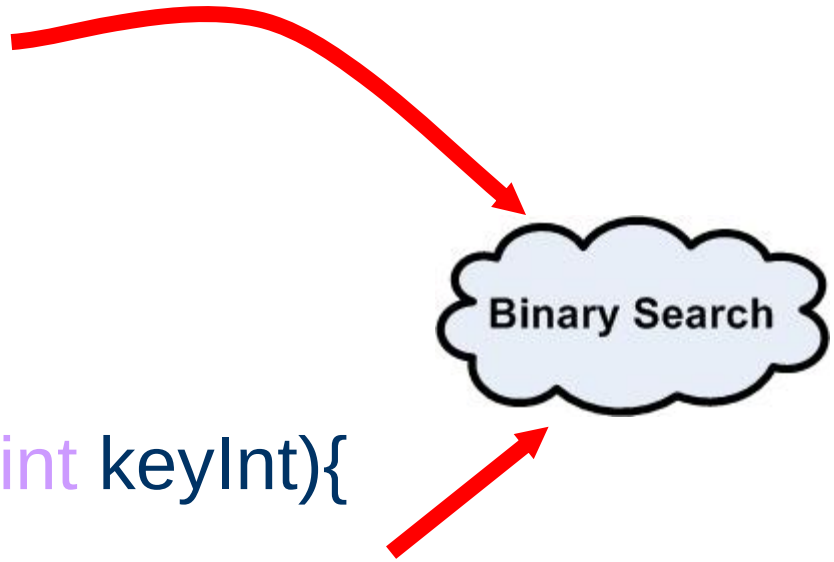
Background: Ontology - Definition

- Define concepts in a universal way to share knowledge:
  - You say: Car
  - Frenchman: Voiture
  - The same concept!
- Started in philosophy and AI
- Became popular with idea of semantic web
- Many languages
  - OWL, OIL+DMAL,RDF,..

The same for program!

```
bool bsearch2(int k){
…
}

bool myBinarySearch(int keyInt){
…
}
```



Basic Motivation

- Semantic adaptive pointcuts
  - Providing semantic level interoperability
  - Easier to understand in a single program
  - Suitable for defining distributed pointcuts in
    - Heterogeneous environment

Solution - Brief

- Formal model for programming ontology
- Mapping to annotation templates
  - Mapping to OWL
- Defining semantic pointcut
- Inference and deployment

Formal Model – Basic Elements

- Concepts
- Attributes
- Relationships
- Hierarchies
- Axioms & Constraints

$$O = \langle C, A^C, R, H, X \rangle$$

Mapping to Annotation

- Using annotation facilities in Java 1.5
- Three basic templates
  - Behavior Descriptor
  - Agent Descriptor
  - Subject Descriptor

```
@BehaviorDescriptor(
        name            = "Login to Server X",
        complexity      = "log(n)",
        Targets         = {"ID","Password"},
        Inputs          = {"PrivateKey"},
        Outputs         = {"Login Info"},
        is_a            = "Login Method")
public int login(String usr, String psswrd)
 {
```

Example

```
@BehaviorDescriptor(
        name          = "Login to Server X",
        complexity    = "log(n)",
        Targets       = {"ID","Password"},
        Inputs        = {"PrivateKey"},
        Outputs       = {"Login Info"},
        is_a          = "Login Method")
public int login(String usr, String psswrd)
 {
```

Semantic Annotation

```
connector LoginConnector {

  LoginHook h =
      new LoginHook(
              (
               behavior.isA == "Login Method") &&
               behavior.name("Login to Server X"));
}
```

Connector Definition

```
class LoginAspect {

    hook LoginHook {

        LoginHook(method(..args)) {
            execution(method);
        }

        around() {
            System.out.println("signing in ..");
        }
    }
}
```

Onspect Definition

Mapping to OWL – why OWL?

- Standard language for ontology modeling
- In XML, easy to exchange
- Reasoning facilities
  - IODT

Inference

- Convert each semantic quantifier into IODT query
- Evaluate for each agents
  - First on(group)
  - Other basic quantifications
- Find set of agents that pointcut applies to

Future Direction

- Extend Formal template to provide further semantics
- Standard content definition for formal ontology
- Robust, easy to use toolkit
- Performance enhancement
- Providing facilities to detect changes in source code and suggest correction for annotations