Indirect Anaphors in a Programming Language: Anchoring, Coreference, Referential Ambiguity

Sebastian Lohmeier
sl@monochromata.de

TaCoS 22, 06/01/2012

Overview

Naturalistic Programming

- "the primitive abstractions in programming languages should be drawn from the study of Natural Languages, rather than from Computer Engineering or Mathematics or ad-hoc metaphors such as Objects."
- is not for "'natural language programming,' [...]. We don't advocate implementing English! The languages we are proposing are naturalistic, but not natural."
- "reads like English"

Lopes et al. (2003)

Naturalistic Programming: Pegasus

```
Take the matrix ([1, 2, 3], [4, 5, 6],
  [7, 8, 9]).
Print the number of rows and print the
   number of columns. [...]
Print the matrix.
Fill it with random entries from 1 to 3.
```

(Knöll and Mezini, 2006)

```
a beautiful modern house (
  (which) is located next to (the river) and
    where (some window) is open )
```

(Knöll et al., 2011)

Naturalistic Programming vs. CNL

```
A man drives a car along a road for 1 hour.
The speed of the car is 30 km/h.
```

Clark et al. (2010)

```
A person drives a vehicle.
The path of the driving is a road.
The duration of the driving is 1 hour.
The speed of the driving is 30 km/h.
```

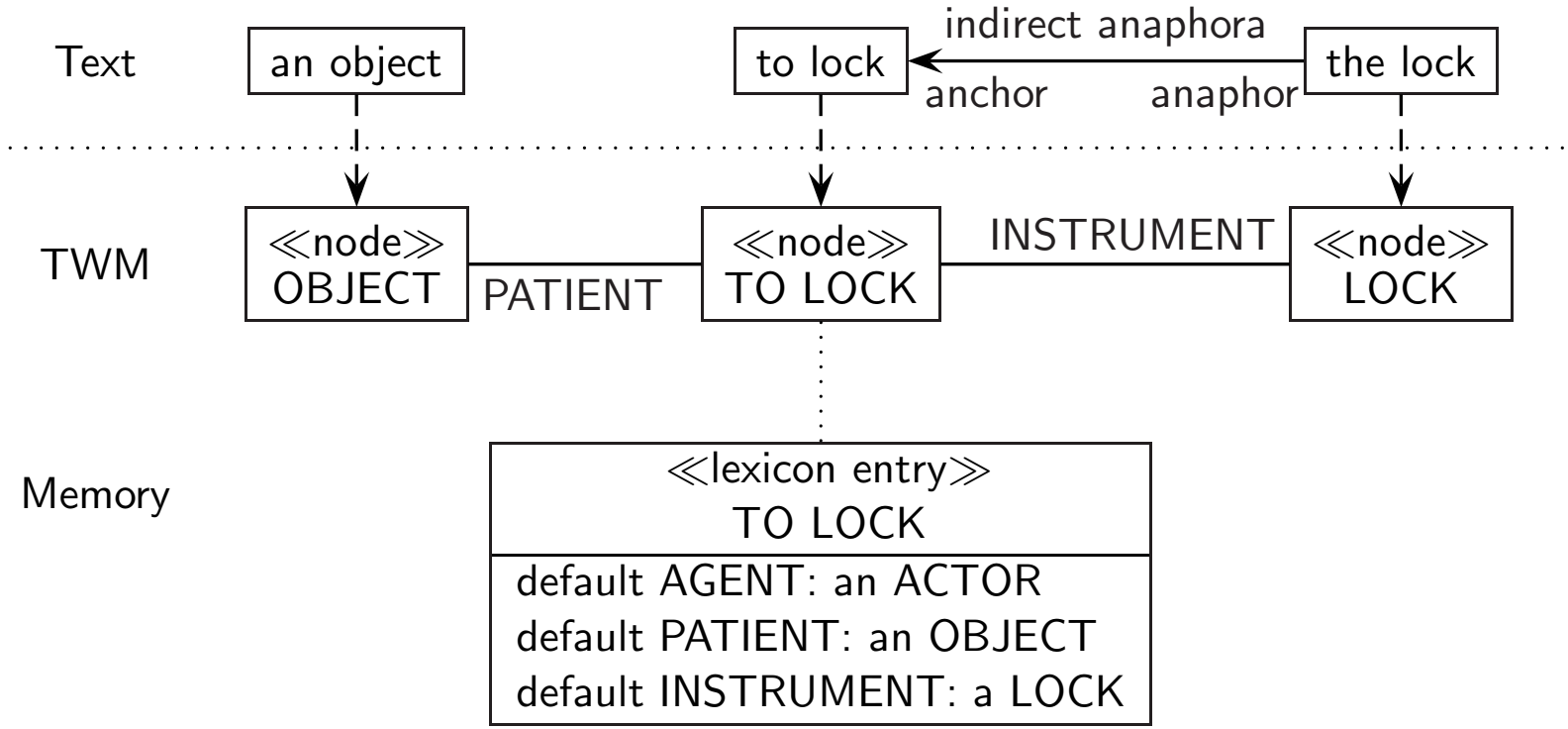
Clark et al. (2010)

Naturalistic Programming Through a Cognitive Linguistics of Programming

- Modify an existing PL (Java) to work more like English instead of reading like it
- Cognitive Linguistics (Schwarz, 2000; Schwarz-Fiesel, 2007) as a blueprint
- OOP encodes the knowledge that CogLing needs
- Metaphor: AST=TWM, classes=frames/schemata (Lohmeier, 2011)
- Potentially model individual differences

Indirect Anaphors in English

If the method m is synchronized, then an object must be locked before the transfer of control. No further progress can be made until the current thread can obtain the lock. (Gosling et al., 2005, 478)



Anchoring Indirect Anaphors in Source Code

```
getDiscourseElementNode(id)                    .Node
```

1. Identify all anchors suitable for an IA
   - 1.1 find anchor in same text=method/constructor/initializer
   - 1.2 anchoring cognitively plausible, i.e. at least one of 3.x possible
   - 1.3 Later: theme of anchor is focused
2. Reduce co-referential anchors
3. Anchor referentially unambiguous IAs in
   - 3.1 a matching method invocation or instance creation (kind 1), or
   - 3.2 an expression whose type has a suitable field/getter (kind 2), or
   - 3.3 an expression permitting inference (kind 3)

adapted from Schwarz (2000)

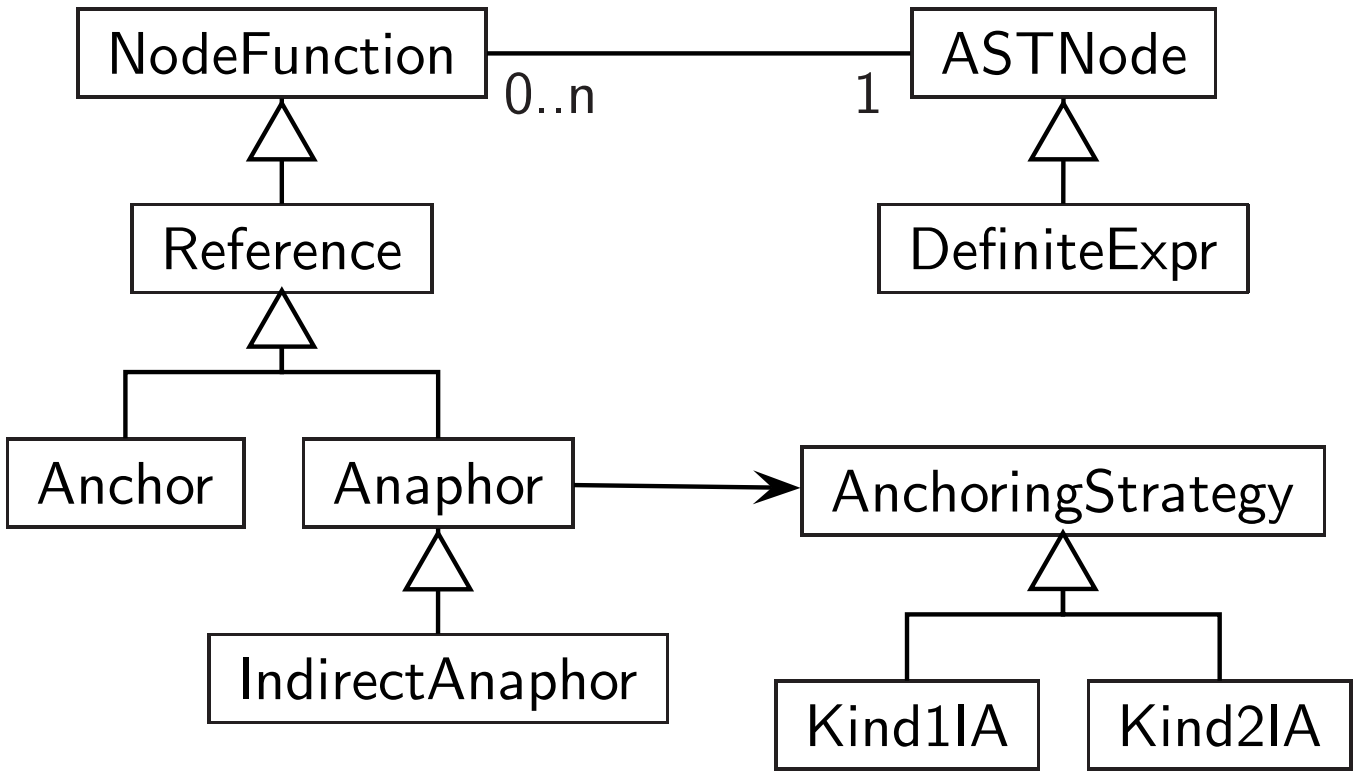
Anchoring: Syntax of DefiniteExpr

```
DefiniteExpr def_expr =
    DOT name;
Access name =
    simple_name
  | qualified_name;
Expr primary_no_new_array :=
    [...]
  | def_expr LPAREN argument_list? RPAREN;
```

using JastAddJ (Ekman and Hedin, 2007)

Anchoring: Types used in the compiler

```
getDiscourseElementNode(id)                    .Node
```



Anchoring: Code generated

```
getDiscourseElementNode(id);
.Node.getFirstChild().getNodeValue();
.Node.getAttributes().getNamedItem ("id");
```

```
Node $anchor$0 =
   getDiscourseElementNode(id);
$anchor$0.getFirstChild().getNodeValue();
$anchor$0.getAttributes().getNamedItem("id");
```

Anchoring

```
public final MMAX2DiscourseElement
    getDiscourseElementByID(String id) {
MMAX2DiscourseElement result = null;
Node temp = getDiscourseElementNode(id);
result = new MMAX2DiscourseElement(
 temp.getFirstChild().getNodeValue(),
 temp.getAttributes().getNamedItem("id")
  .getNodeValue(),
 this.getDiscoursePositionFromDiscourseElementID(
  temp.getAttributes().getNamedItem("id")
      .getNodeValue()),
 MMAX2Utils.convertNodeMapToHashMap(
  temp.getAttributes()));
return result;
}
```

Based on http://mmax2.sourceforge.net/ sources

Anchoring in class instance creation expression (kind 1)

```
public final MMAX2DiscourseElement
    getDiscourseElementByID(String id) {

Node temp = getDiscourseElementNode(id);
new MMAX2DiscourseElement(
 temp.getFirstChild().getNodeValue(),
 temp.getAttributes().getNamedItem("id")
  .getNodeValue(),
 this.getDiscoursePositionFromDiscourseElementID(
  temp.getAttributes().getNamedItem("id")
   .getNodeValue()),
 MMAX2Utils.convertNodeMapToHashMap(
  temp.getAttributes()));
return .MMAX2DiscourseElement;
}
```

Based on http://mmax2.sourceforge.net/ sources

Anchoring in class instance creation expression (kind 1)

```
public final MMAX2DiscourseElement
    getDiscourseElementByID(String id) {

Node temp = getDiscourseElementNode(id);
new MMAX2DiscourseElement(
 temp.getFirstChild().getNodeValue(),
 temp.getAttributes().getNamedItem("id")
  .getNodeValue(),
 this.getDiscoursePositionFromDiscourseElementID(
  temp.getAttributes().getNamedItem("id")
   .getNodeValue()),
 MMAX2Utils.convertNodeMapToHashMap(
  temp.getAttributes()));
return .Element;
}
```

Based on http://mmax2.sourceforge.net/ sources

Anchoring and control flow

```
public final MMAX2DiscourseElement
    getDiscourseElementByID(String id) {

Node temp = getDiscourseElementNode(id);
if (temp != null) {
new MMAX2DiscourseElement(
 temp.getFirstChild().getNodeValue(),
 temp.getAttributes().getNamedItem("id")
  .getNodeValue(),
 this.getDiscoursePositionFromDiscourseElementID(
  temp.getAttributes().getNamedItem("id")
   .getNodeValue()),
 MMAX2Utils.convertNodeMapToHashMap(
  temp.getAttributes()));
}
return .MMAX2DiscourseElement;
}
```

Anchoring in method invocation (kind 1)

```
public final MMAX2DiscourseElement
    getDiscourseElementByID(String id) {

getDiscourseElementNode(id);
new MMAX2DiscourseElement(
 .Node.getFirstChild().getNodeValue(),
 .Node.getAttributes().getNamedItem("id")
  .getNodeValue(),
 this.getDiscoursePositionFromDiscourseElementID(
  .Node.getAttributes().getNamedItem("id")
   .getNodeValue()),
 MMAX2Utils.convertNodeMapToHashMap(
  .Node.getAttributes()));
return .Element;
}
```

Based on http://mmax2.sourceforge.net/ sources

Anchoring in method invocation, getter (kind 2)

```
public final MMAX2DiscourseElement
    getDiscourseElementByID(String id) {

getDiscourseElementNode(id);
new MMAX2DiscourseElement(
 .Node.getFirstChild().getNodeValue(),
 .NamedNodeMap.getNamedItem("id")
  .getNodeValue(),
 this.getDiscoursePositionFromDiscourseElementID(
  .NamedNodeMap.getNamedItem("id")
   .getNodeValue()),
 MMAX2Utils.convertNodeMapToHashMap(
  .NamedNodeMap));
return .Element;
}
```

Based on http://mmax2.sourceforge.net/ sources

Anchoring in method invocation, getter (kind 2)

```
public final MMAX2DiscourseElement
    getDiscourseElementByID(String id) {

getDiscourseElementNode(id);
new MMAX2DiscourseElement(
 .Node.getFirstChild().getNodeValue(),
 .Attributes.getNamedItem("id")
  .getNodeValue(),
 this.getDiscoursePositionFromDiscourseElementID(
  .Attributes.getNamedItem("id")
   .getNodeValue()),
 MMAX2Utils.convertNodeMapToHashMap(
  .Attributes));
return .Element;
}
```

Based on http://mmax2.sourceforge.net/ sources

Anchoring to underspecify façade methods (kind 3)

```
class MMAX2AnnotationScheme {
private MMAX2AttributePanel attributepanel;
public MMAX2AttributePanel
    getAttributePanel() {
  return attributepanel;
}
public void setAttributePanelContainer(
    MMAX2AttributePanelContainer _container){
  attributepanel.setAttributePanelContainer(
      _container);
}}
```

```
MMAX2AnnotationScheme tempScheme =
    affectedLevel.updateAnnotationScheme();
tempScheme.setAttributePanelContainer(this);
```

Based on http://mmax2.sourceforge.net/ sources

Anchoring to underspecify façade methods (kind 3)

```
class MMAX2AnnotationScheme {
private MMAX2AttributePanel attributepanel;
public MMAX2AttributePanel
    getAttributePanel() {
  return attributepanel;
}}
```

```
public void addAttributePanel(
    MMAX2AttributePanel _panel) {
    .AttributePanelContainer = this;
}

public void addAttributePanel(
    MMAX2AttributePanel _panel) {
    _panel.getAttributePanel()
        .setAttributePanelContainer(this);
}
```

Coreference on the surface: (in)direct anaphors

```
class NodeOwner {
  private Node node;
  public void setNode(Node node) {
    this.node = node;
    System.out.println(.Node+" set");
    this.node.reset();
    System.out.println(.Node+" reset");
  }
}
```

```
Object temp = getDiscourseElementNode(id);
System.out.println(.Object);
```

Deep coreference: only indirect anaphors

```
class SeeminglyAmbiguous {
  public Child c = new Child();
  public Child getChild() { return c; }
}
new SeemlinglyAmbiguous();
System.out.println(.Child);
```

Coreference resolution vs. alias analysis

```
public void foo(Node parent, Node child) {
  .1.Node.appendChild(.2.Node);
}
createNode();
foo(.Node,.Node);
```

Referential Ambiguity

```
outerBox = Box.createVerticalBox();
innerBox = Box.createHorizontalBox();
add(outerBox);
rightBox = Box.createVerticalBox();
middleBox = Box.createVerticalBox();
leftBox = Box.createVerticalBox();
innerBox.add(leftBox);
innerBox.add(Box.createHorizontalStrut(6));
innerBox.add(middleBox);
innerBox.add(Box.createHorizontalStrut(6));
innerBox.add(rightBox);
outerBox.add(innerBox);
```

Based on http://mmax2.sourceforge.net/ sources

Referential Ambiguity and Order

```
new JPanel();
Box.createHorizontalBox();
Box.createHorizontalBox();
.1.Box.add(level.getUpdateButton());
.JPanel.add(.1.Box);
.2.Box.add(level.getMoveUpButton());
.JPanel.add(.2.Box);
```

Based on http://mmax2.sourceforge.net/ sources

Referential Ambiguity and Association Names

```
Box.getVerticalBox();
Box.getHorizontalBox();
add(.VerticalBox);
.VerticalBox.add(.HorizontalBox);
```

Based on http://mmax2.sourceforge.net/ sources

Referential Ambiguity and Focus

```
new JPanel();
Box.createHorizontalBox();
.Box.add(level.getUpdateButton());
.Box.add(Box.createHorizontalStrut(3));
.JPanel.add(.Box);
Box.createHorizontalBox();
.Box.add(level.getMoveUpButton());
.Box.add(Box.createHorizontalStrut(1));
.JPanel.add(.Box);
```

Based on http://mmax2.sourceforge.net/ sources

Referential Ambiguity and Activation

```
getDiscourseElementNode(id);
new MMAX2DiscourseElement(
  .Node.getFirstChild().getNodeValue(),
  .Node.getAttributes().getNamedItem("id")
```

Based on http://mmax2.sourceforge.net/ sources

Conclusions

- Knowledge-based language processing can be applied to source code
- New, easily learnable PL features can be derived from linguistics
- Need to progress from test-first development to corpus analysis (and on to experimental evaluation)
- Need to increase cognitiveness through activation and focus
- Morphological processing should increase applicability
- We need you!
- You can ease your own life as a computer linguist (long-term)

Q&A

Documentation, sources, updates:
http://www.monochromata.de/lop

Diaspora handle:
monochromata@pod.geraspora.de

References I

Peter Clark, William Murray, Phil Harrison, and John Thompson. Naturalness vs. predictability: A key debate in controlled languages. In Norbert Fuchs, editor, *Controlled Natural Language; Workshop on Controlled Natural Language, CNL 2009*, volume 5972 of *LNCS*, pages 65–81, Berlin and Heidelberg, 2010. Springer. URL `http://dx.doi.org/10.1007/978-3-642-14418-9_5`.

Torbjörn Ekman and Görel Hedin. The JastAdd extensible Java compiler. In *Companion to the 22nd ACM SIGPLAN conference on Object-oriented programming systems and applications companion*, OOPSLA '07, pages 884–885, New York, NY, USA, 2007. ACM.

James Gosling, Bill Joy, Guy Steele, and Gilad Bracha. *The Java(TM) Language Specification*. Addison Wesley, 3rd edition, 2005. ISBN 0321246780.

References II

Roman Knöll and Mira Mezini. Pegasus: first steps toward a naturalistic programming language. In *OOPSLA '06: Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications*, pages 542–559, New York, NY, USA, 2006. ACM. URL `http://doi.acm.org/10.1145/1176617.1176628`.

Roman Knöll, Vaidas Gasiunas, and Mira Mezini. Naturalistic types. In *ONWARD '11: Proceedings of the 10th SIGPLAN symposium on New ideas, new paradigms, and reflections on programming and software*, pages 33–47, October 2011.

Sebastian Lohmeier. *Continuing to Shape Statically-Resolved Indirect Anaphora for Naturalistic Programming: A transfer from cognitive linguistics to the Java programming language*. 2011. URL `http://monochromata.de/shapingIA/`.

References III

Cristina Videira Lopes, Paul Dourish, David H. Lorenz, and Karl Lieberherr. Beyond AOP: toward naturalistic programming. *SIGPLAN Not.*, 38(12):34–43, 2003. URL `http://dx.doi.org/10.1145/966051.966058`.

Monika Schwarz. *Indirekte Anaphern in Texten*. Niemeyer, Tübingen, 2000.

Monika Schwarz-Fiesel. Indirect anaphora in text: A cognitive account. In Monika Schwarz-Fiesel, Manfred Consten, and Mareile Knees, editors, *Anaphors in Text : cognitive, formal and applied approaches to anaphoric reference*, volume 86 of *Studies in Language Companion Series*, pages 3–20. John Benjamins, Amsterdam, 2007.