

Florian Deißeböck & Daniel Ratiu

A Unified Meta-Model for Concept-Based Reverse Engineering

October 1st 2006
ATEM '06 Genova



Outline

Reverse Engineering

Concept-based Reverse Engineering

Ontologies, WordNet, Meta-Model

Experiences

Synonymy, Logical Duplication, Polysemy

Conclusion & Future Work

Reverse Engineering

Reverse engineering is the process of analyzing a subject system to

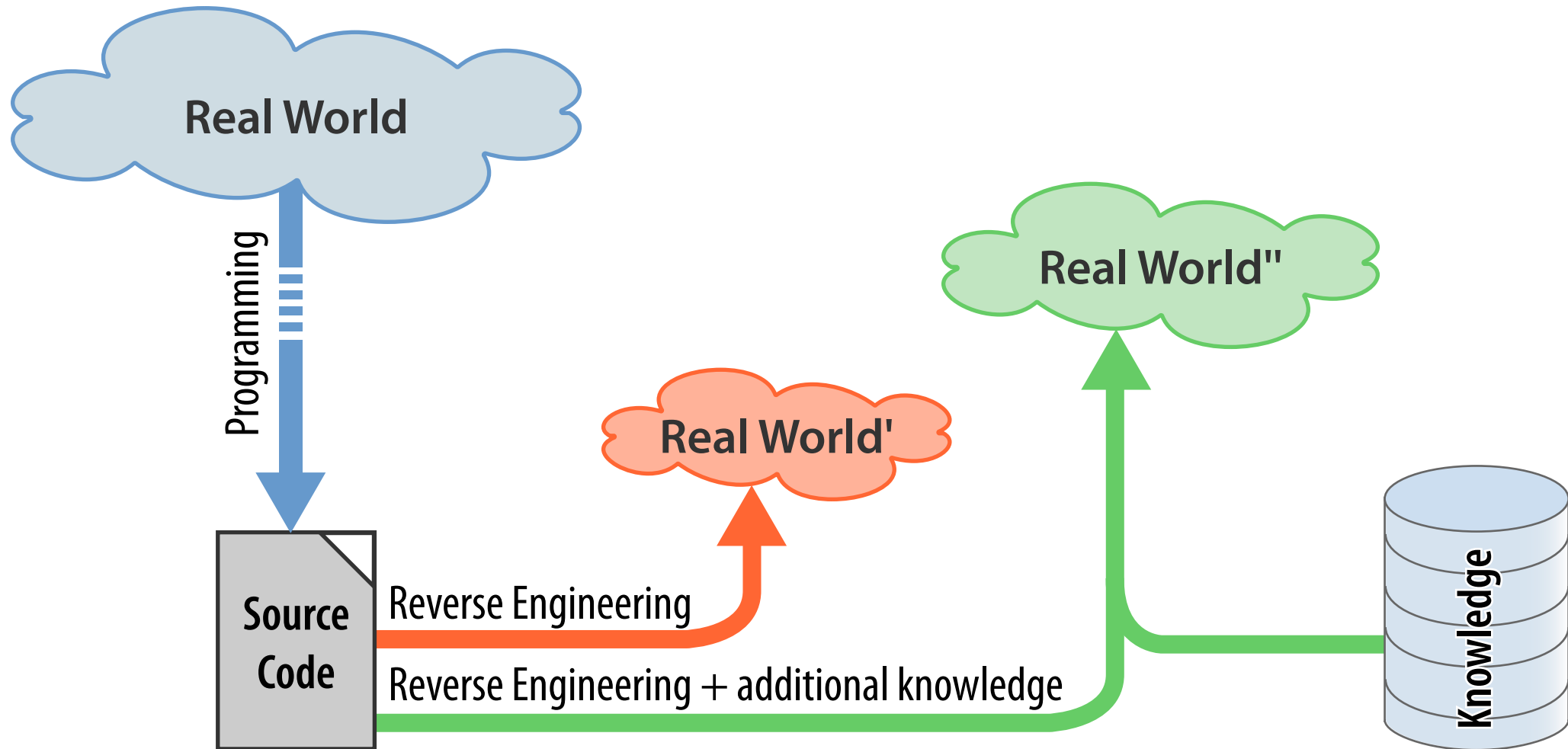
- identify the system's components and their interrelationships and
- create representations of the system in another form or at a **higher level of abstraction**.*

Pressing Issues

- Logical duplication
- Naming defects
- Documentation quality
- Impact analysis
- Dependency management
- Aspect mining

* E.J. Chikofsky, J.H. Cross, *Reverse engineering and design recovery: A taxonomy*, 1990

Information Loss

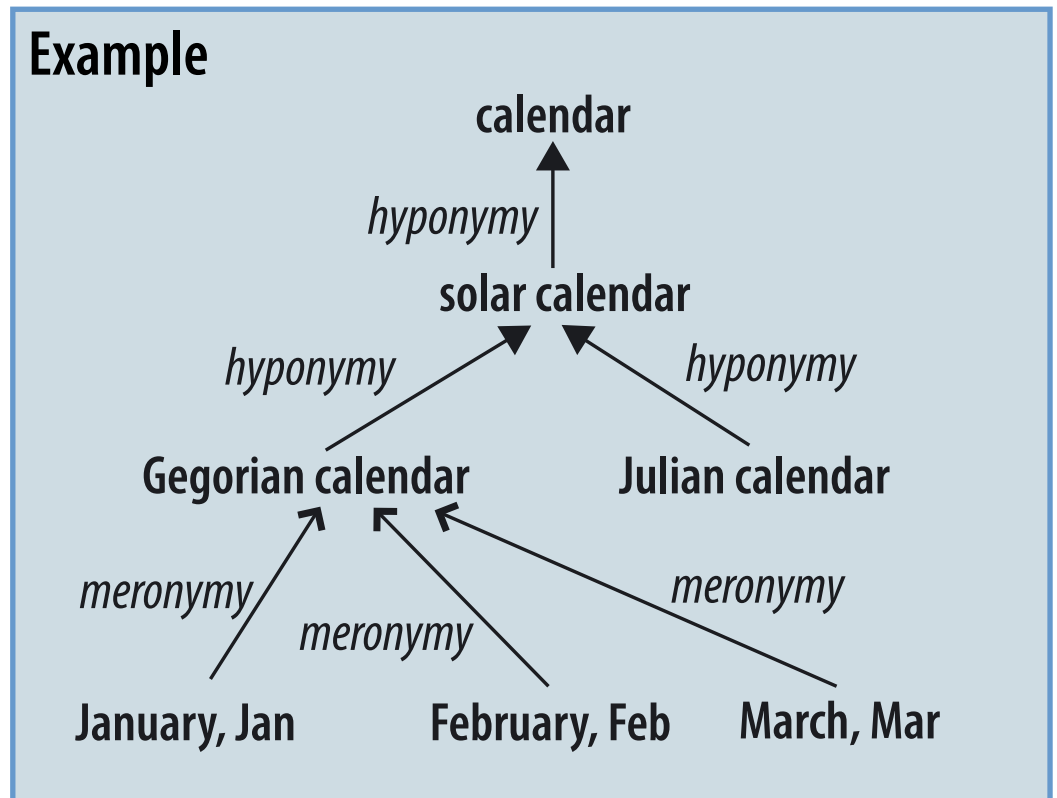


Unfortunately, source code does not contain much of the original design information, which must be reconstructed from only the barest of clues.*

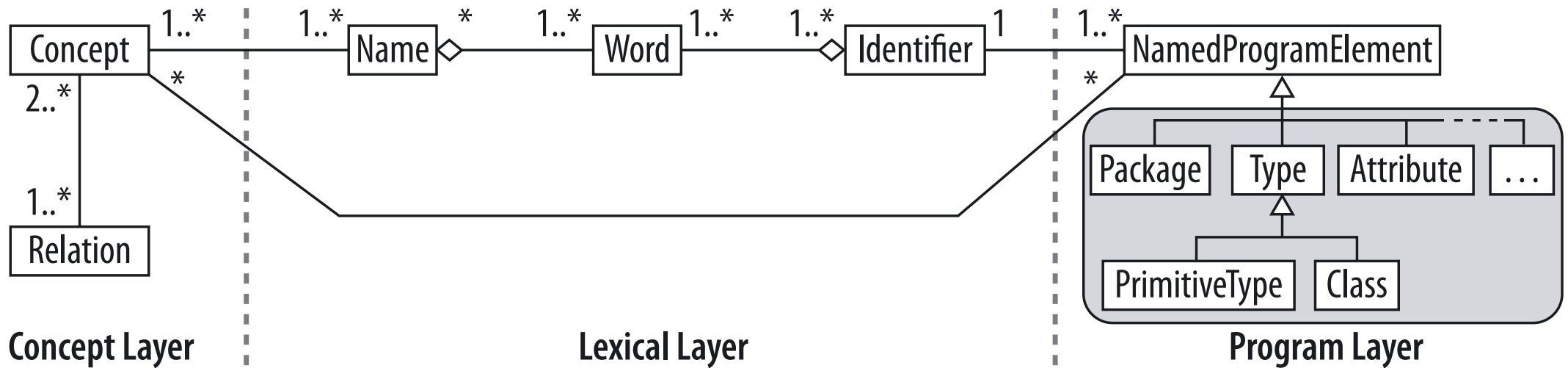
* T.J. Biggerstaff, *Design Recovery for Maintenance and Reuse*, 1989

WordNet Ontology

- Provides »Common Knowledge«
- Lexicalized concepts only
- ≈ 150.00 Words
- ≈ 115.00 Concepts
- Synonymy
- Polysemy
- Relations
 - Hypernymy/Hyponymy
 - Holonymy/Meronymy



Meta-Model



- Concept layer
- Lexical layer
- Program layer
- Extends structural models
- Names are the »glue«
- Explicit mapping between concepts and program elements

Instantiation

```
class Calendar {  
    ...  
}  
class GregorianCalendar  
    extends Calendar {  
    int JANUARY = 0;  
    int FEBRUARY = 1;  
    ...  
}
```

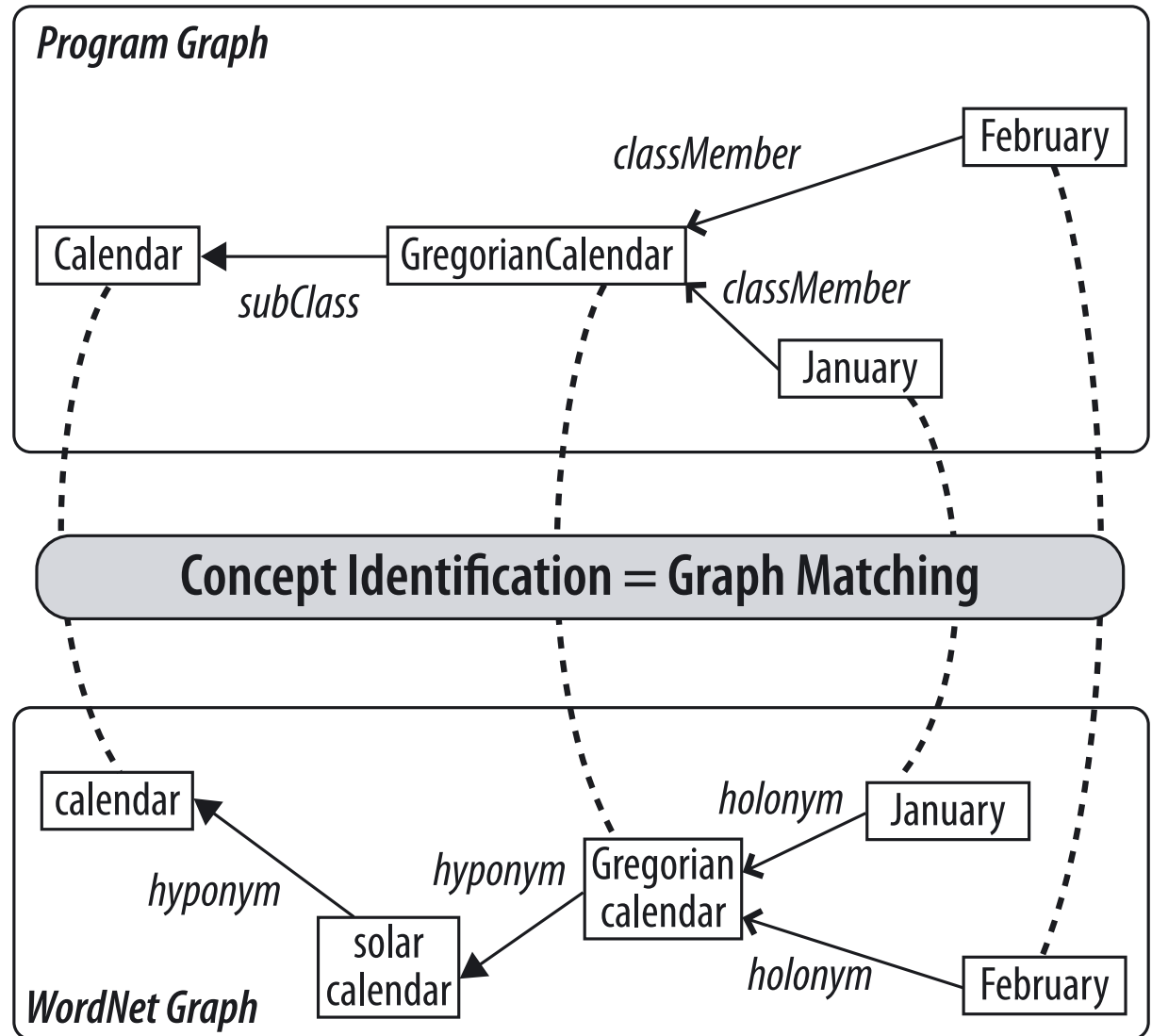
Source Code

□ □ □ □ →
Program hierarchy
between names
represented
as graphs

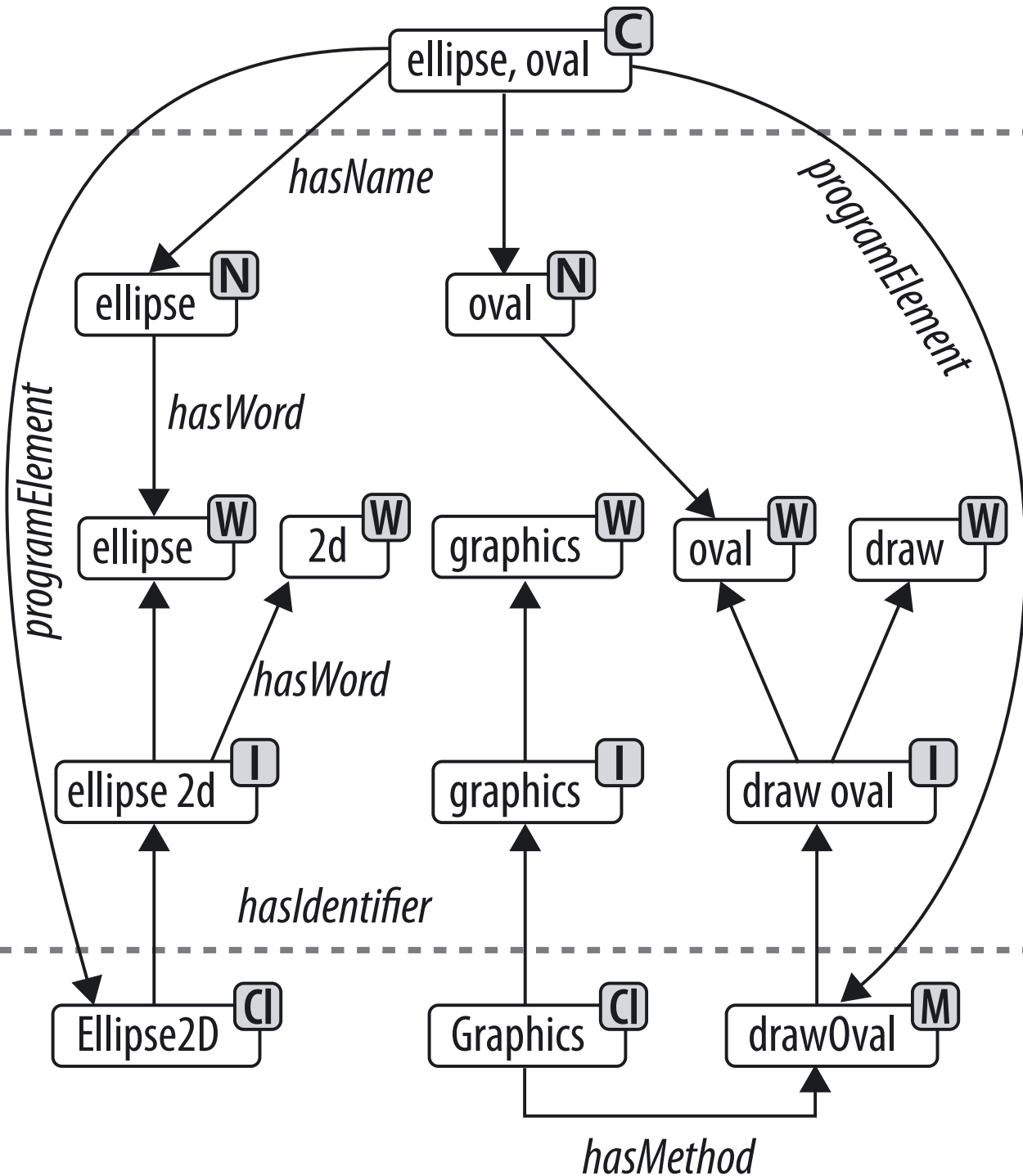


Ontology
WordNet 2.0

□ □ □ □ →
Ontology
represented
as Graph



Example 1: Synonymy



drawOval **JavaDoc**

```
public abstract void drawOval( int x,
                               int y,
                               int width,
                               int height)
```

Draws the outline of an oval. ...

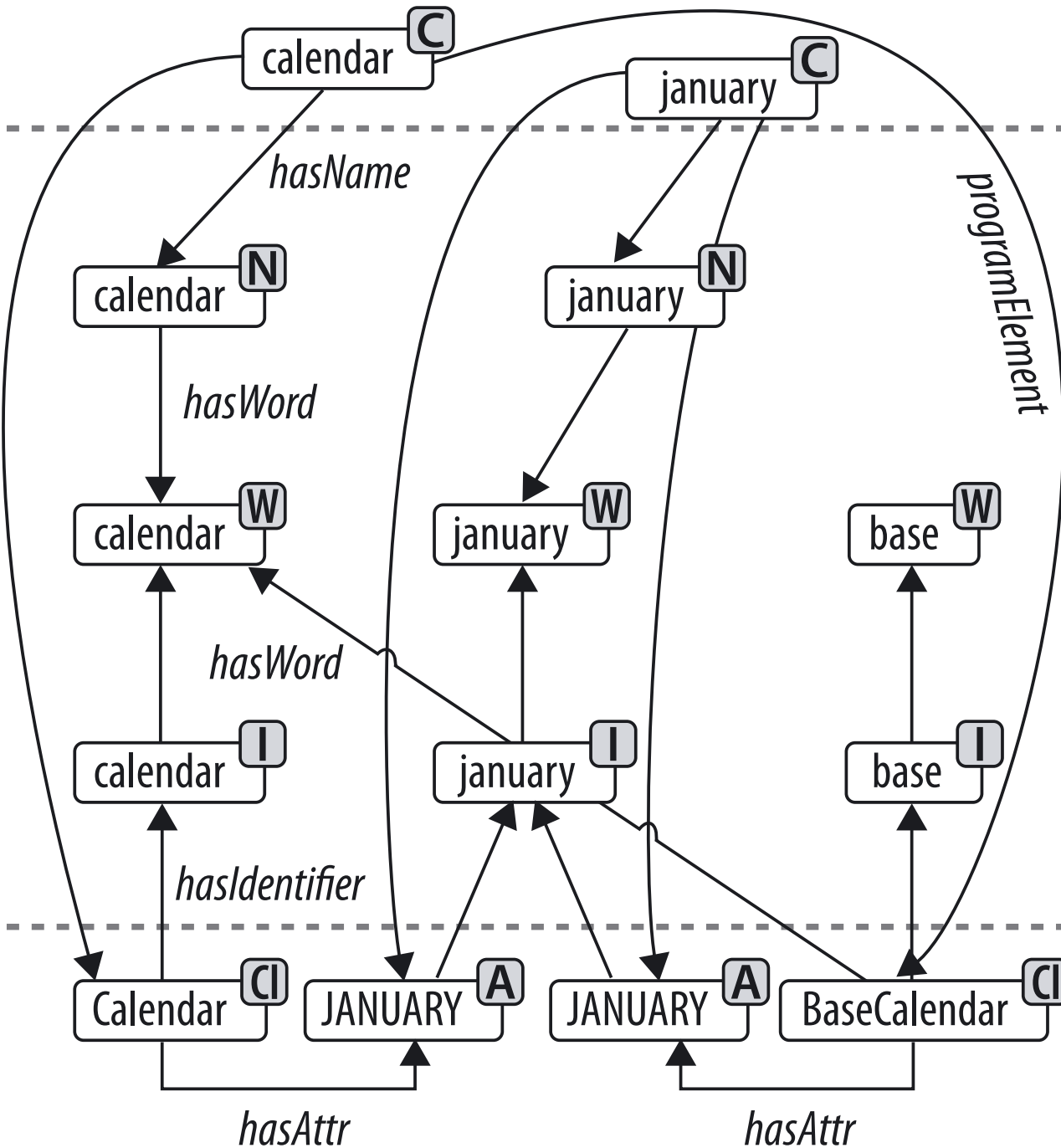
Ellipse2D.Double **JavaDoc**

```
public Ellipse2D.Double(double x,
                        double y,
                        double w,
                        double h)
```

Constructs and initializes an Ellipse2D ...

- | | |
|------------------|---------------------|
| C Concept | I Identifier |
| N Name | CI Class |
| W Word | A Attribute |
| M Method | |

Example 2: Logical Duplication



java.util.Calendar

```
public final static int JANUARY = 0;
public final static int FEBRUARY = 1;
public final static int MARCH = 2;
...
```

sun.util.calendar.BaseCalendar

```
public final static int JANUARY = 1;
public final static int FEBRUARY = 2;
public final static int MARCH = 3;
...
```

- | | |
|------------------|---------------------|
| C Concept | I Identifier |
| N Name | Cl Class |
| W Word | A Attribute |
| M Method | |

Example 3: Polysemy

java.awt.BorderLayout

```
/**
 * The north layout constraint (top of container).
 */
public static final String NORTH = "North";

...

/**
 * Constant to specify components location to be the
 * north portion of the border layout.
 * ...
 */
Component north;
```

Conclusions & Future Work

Conclusions

- Lost information needs to be recovered for effective reverse engineering.
- Additional information must be provided and...
- ...explicitly linked to the source code.
- Results are encouraging but need further research.
- Variation points need to be investigated.

Future work

- Chained ontologies
- Hand-crafted ontologies
- Different types of relations