JavaOne℠

Sun's 2004 Worldwide Java Developer Conference℠

Java™ Metadata (JSR 175) and the Semantic Web

java.sun.com/javaone/sf

Joshua Fox
Senior Software Architect
Unicorn Solutions, Inc.
joshua.fox@unicorn.com
http://www.unicorn.com

Adding Semantics to Java

Java and the next generation World Wide Web

Exposing the semantics of Java classes, interfaces, and methods, for automated integration

Agenda With Section Highlights

- Java Metadata
- The Semantic Web
- Ontology
- Model-Driven Architecture
- The Power of Semantics

Connect code to the world

Give business meaning to Java classes

Example from Industry

- Integrate a modern application with a partner's mainframe application
  - One side is an insurance carriers which talks insurance concepts like ACORD
  - The other is a benefits company which talks HR language such as HR-XML

Java Metadata

Adding attributes to Java code

- JSR 175
- Tiger/JDK 1.5
- Can be retained in bytecode
- On the metadata level

Java Metadata: Sample Uses

unicorn®

Additional information on code

Example uses

- Expose a method as Web Service
- Indicate copyright information on code
- Mark methods as JavaBean-style properties
- Declare serializable fields
- Set permission categories on classes/methods

and

- Declare the *meaning* of the code

Example

```
public class IgnoreTestCase extends TestCase{
   @Ignore(reason="why not")
   public void testCase() {
      fail("");
   }

   public void testNotIgnored() {
      //…
   }
}
```

Thanks to Jeff Langr of Langrsoft

The Semantic Web

unicorn

Web Interfaces Have Real-Life Meaning

- Connect Web Services so that they can be found, aggregated and used without prior knowledge of interfaces/schemas
- Software can interact based on real-life business concepts.

Ontology

unicorn

Formalization of Real-World Meaning

- Express the real world in classes, properties, and business rules
- Deceptive similarity to OO: ontological models represent the world, OO models represent software (more later)

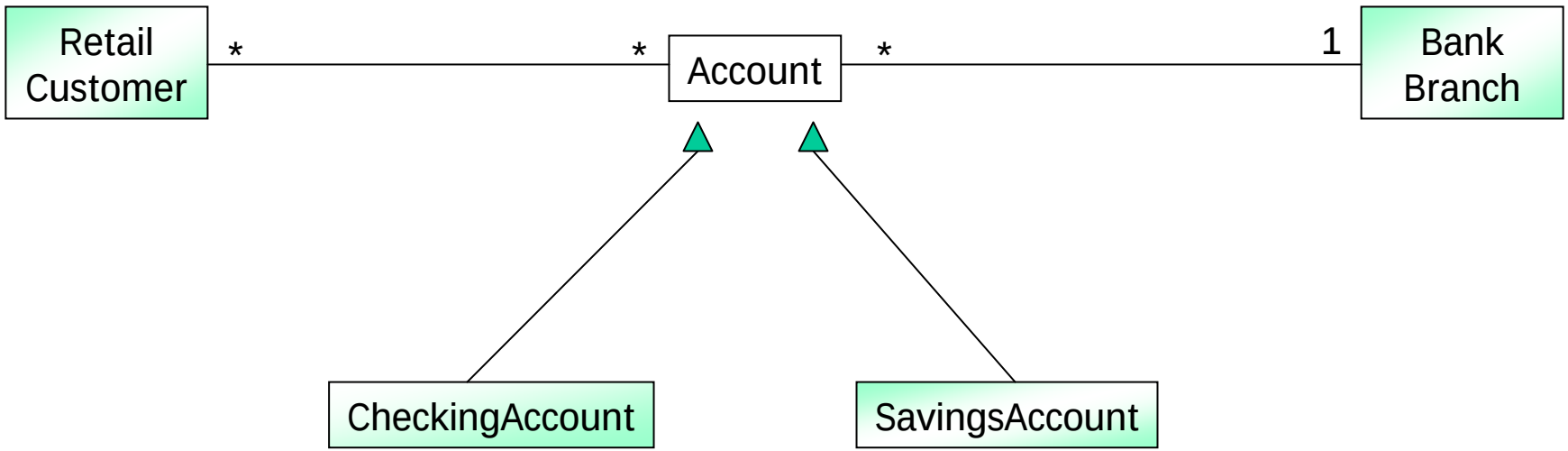
Ontology: Standards

From the World Wide Web Consortium

- W3C: The organization of Tim Berners-Lee, WWW inventor and Semantic Web visionary
- OWL
- RDF and RDF/S

Ontology: Example

The real-world concepts of banking



Ontology: Classes

unicorn®

A classification of real-world instances

- (Multiple) Inheritance supported
- Comparable to Java classes, but
  - Describe *real-world* objects, not *software* objects
  - Are not “factories” for the objects they describe
  - No behavior: no “methods” or “executables”
- Capture views of the business at different levels of granularity
- Each usable without the other

Ontology: Packages

unicorn

Grouping sets of classes

- Similar to Java packages, but hierarchical
- Packages should reflect practical categories of real-life meaning

Ontology: Properties

unicorn

Characteristics of a class' instances

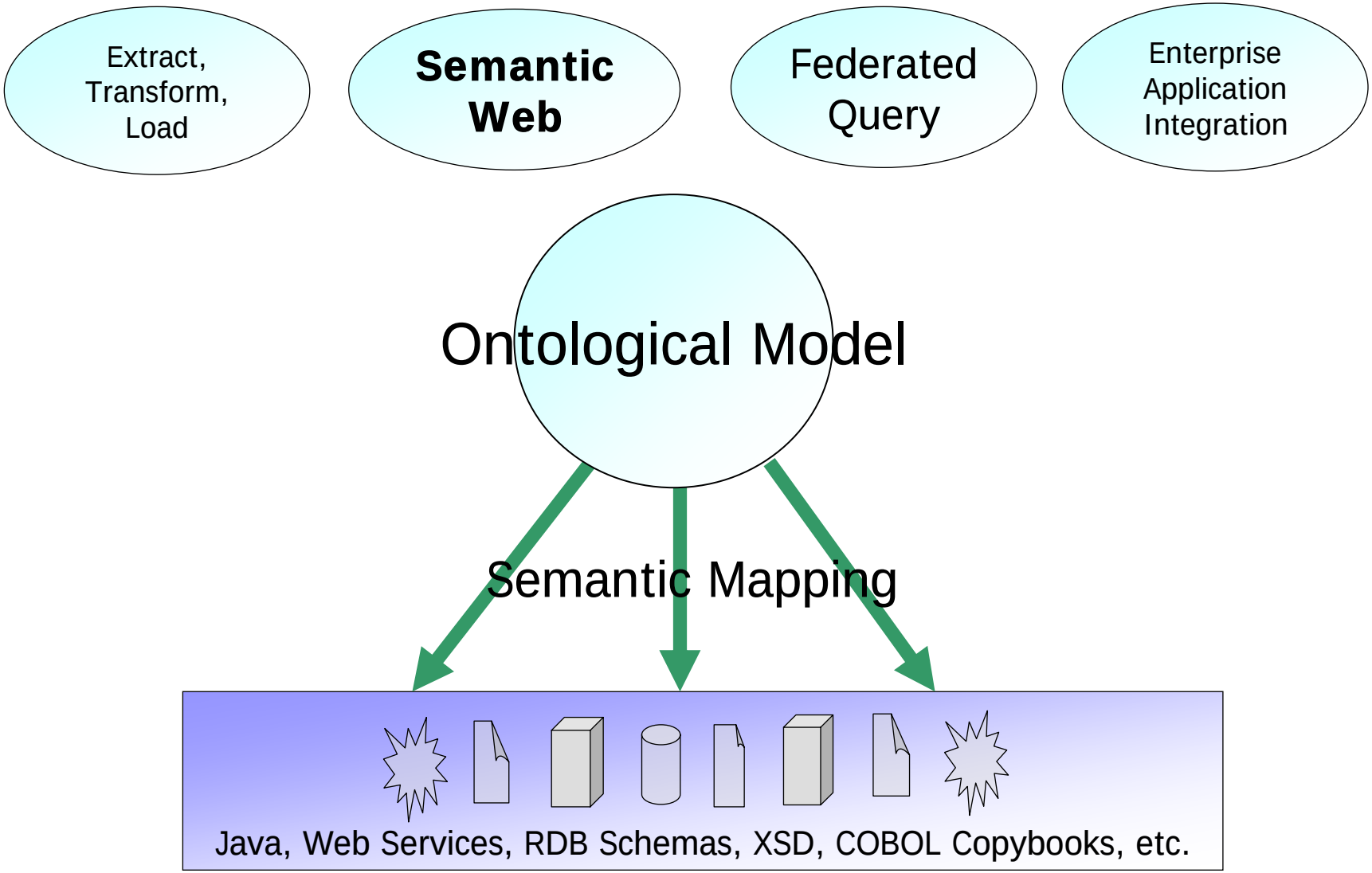
- Relationships between classes (including primitive classes)
- Resemble Java properties/fields, UML associations
- The "has a" relationship
- Inheritance supported

Ontology: Business Rules

Interrelate classes or properties

- Examples:
  - **grossPrice = netPrice+ netPrice* taxRate**
  - Value of **currency** is one of **{**“USD,” “EUR,” **…** ,”GBP”**}**
  - **accountNumber** is unique for each bank account
- Expressions, assertions about reality, rather than (as in Java) executable behavior
- An extension of the ontological standards

High Level Architecture



Model Driven Architecture

An ontological approach to MDA

- The model represents the real world
- Java code represents executable behavior, a simulation of the real world
- Java attributes (metadata) link between the two layers

The Power of Semantics

Give code real-world meaning

- Java class *Account* linked to a clear definition of what a "*Retail banking account*" really is
- Java field/property *balance* of class *Account* linked to a definition: e.g., "*current balance not including interest year-to-date*".
- These definitions written in plain English notation in the context of an ontological formalism

Using Semantics

Access code by its real-world meaning

Automated system can

- Find relevant classes/methods
  - in an API loaded in a jar
  - in Java code exposed as Web Services
- Transform when APIs do not provide the exact desired real-world meaning
  - E.g., property *price* is available in JavaBean *RetailItem.*
  - However, the mapping to the model indicates that this is “net price,” while “gross price” is needed.
  - The model has a rule *grossPrice = netPrice +net * taxRate*, and the system can generate the Java code to make the conversion automatically.

An Annotated Class

Linking Java code to its meaning

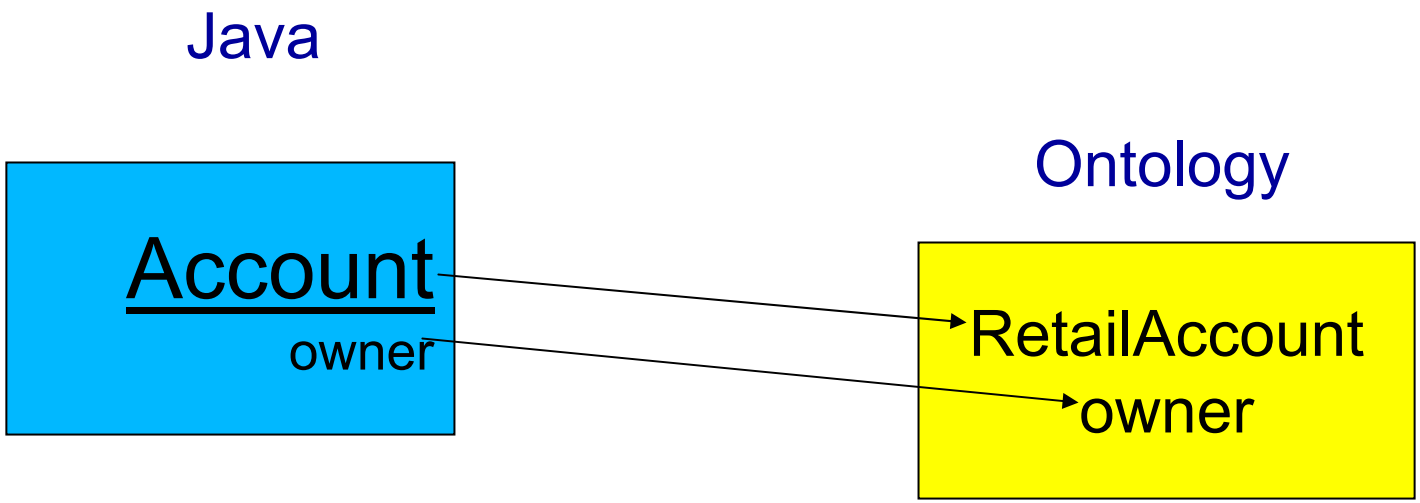
```
@OntologicalClass(url="http://ecommerce.org/
                       model/RetailBankingAccount")
@MappingRule("owner.type!=CORPORATE")

public class Account {

  @OntologicalProperty(url="http://ecommerce.org/
                        model/RetailBankingAccount/owner")
  public Customer getOwner(){
    return owner;
  }
}
```

Mapping Java to ontology

Implemented with metadata attributes



Declarations of Ontological Attributes

unicorn

Defining the metadata

```
@Retention (RetentionPolicy.RUNTIME)
@Target({ElementType.CLASS})
@interface OntologicalClass{
 String name();
 String url();
}

@Retention (RetentionPolicy.RUNTIME)
@Target({ElementType.FIELD})
@interface OntologicalProperty{
 String name();
 String url();
}
@Retention (RetentionPolicy.RUNTIME)
@Target({ElementType.CLASS})
@interface MappingRule{
 String value();
}
```

Ontological Attributes

unicorn

Identifiers for ontological concepts

- *URL* is standard adopted by W3C as globally unique identifier for various purposes, e.g., XML namespaces. Semi-human-readable
- *Name* is human-readable identifier; needs to be a fully qualified name with packages, similar to Java packages, for uniqueness
- Another approach uses a *Unique ID* for ontological concepts, a long pseudo-random number or String, not human readable, but compact and guaranteed unique

Isn’t a Naming Convention Enough?

No!

- The JavaBean *setXYZ/getXYZ* doesn’t tell you what *XYZ* means
- JSR 175 was developed to replace name-based conventions
- Map to ontological model to give meaning
- Ontological models built differently from java codebases
- Ontological model may be fixed standard
- There’s only so much you can pack into a string
- Get the power of classes, inheritance, properties, rules

Q&A

Joshua Fox

Unicorn Solutions, Inc.

joshua.fox@unicorn.com

www.unicorn.com

JavaOne℠

Sun's 2004 Worldwide Java Developer Conference℠

Java™ Metadata (JSR 175) and the Semantic Web

java.sun.com/javaone/sf

Joshua Fox
Senior Software Architect
Unicorn Solutions, Inc.
joshua.fox@unicorn.com
www.unicorn.com

Sun microsystems