

# *Architecture and Design of Adaptive Object-Models*

**Joseph W. Yoder  
Federico Balaguer  
Ralph Johnson**

**The Refactory, Inc.  
University of Illinois**

**yoder@refactory.com  
[balaguer|johnson]@cs.uiuc.edu  
<http://www.adaptiveobjectmodel.com>**

## Table of Contents

- ◆ Overview
- ◆ Adaptive Object-Model Description
- ◆ Architectural Elements of AOM
- ◆ An Example in the Medical Domain
- ◆ Implementation Issues
- ◆ Summary

## Meta Collaborators

- ◆ Ali Arsanjani
- ◆ Krzysztof Czarnecki
- ◆ Martine Devos
- ◆ Brian Foote
- ◆ Martin Fowler
- ◆ Dragos Manolescu
- ◆ Jeff Oaks
- ◆ Nicolas Revault
- ◆ Dirk Riehle
- ◆ Reza Razavi
- ◆ Michel Tilman
- ◆ Others...

## General Problem

- ◆ Requirements change within applications' domain.
- ◆ Business Rules are changing rapidly.
- ◆ Applications have to quickly adapt to new business requirements.
- ◆ Changing the application is costly, it generally includes code and data-storage.
- ◆ There are cycles of: build-compile-release.

## Forces – Shearing Layers

- ◆ Who (Business Person, Analyst, Developer)
- ◆ What (Business Rule, Persistence Layer,...)
- ◆ When (How often, How fast)

There is a different rate of change  
on the system.

Foote & Yoder - Ball of Mud PLoPD4

## General Solution

- ◆ Create an object design (meta-model) that describes the domain objects which includes attributes, relationships, and business rules as instances rather than classes.
- ◆ The domain objects are instantiated through a description given by the user or domain expert.
- ◆ Each new requirement is satisfied by creating a new description and a new instantiation.
- ◆ Separate what changes from what doesn't.

## Adaptive Object-Models

- ◆ Architectures that can dynamically adapt to new user requirements by storing descriptive (metadata) information about the business rules that are interpreted at runtime.
- ◆ Sometimes called a "reflective architecture" or a "meta-architecture".
- ◆ Highly Flexible – Business people (non-programmers) can change it too.

## Adaptive Object-Model (Active|Dynamic Object-Model)

- An ADAPTIVE OBJECT-MODEL is an object model that provides "meta" information about the domain so that it can be changed at runtime
  - ◆ explicit object model that it interprets at run-time
  - ◆ change the object model, system changes its behavior
- ADAPTIVE OBJECT-MODELS usually arise from domain-specific frameworks
- Business rules are stored as descriptive (meta) information in ADAPTIVE OBJECT-MODELS

## Adaptive Object-Models

- ◆ Represents classes, attributes, relationships, and behavior as *metadata*.
- ◆ Based on instances rather than classes.
- ◆ Users change the *metadata* (object model) to reflect changes in the domain.
- ◆ Stores its *Object-Model* in a database or in files and interprets it (can be XML/XMI).

Consequently, the object model is adaptable, when you change it, the system changes immediately.

## Architectural Elements of Adaptive Object Models

- Metadata
- TypeObject
- Properties
- Type Square
- Entity-Relationship
- Strategy/RuleObjects
- Interpreters/Builders
- Editors/GUIs

If you want something to change quickly,  
you must push it into the data.

# Metadata and Adaptive Object-Models

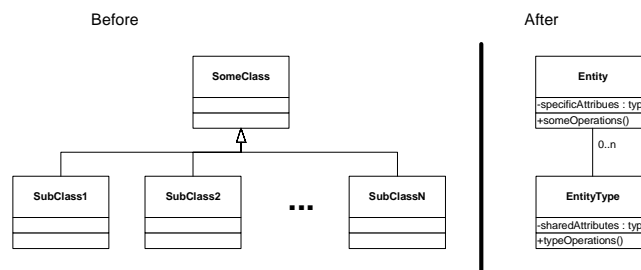
*"Anything you can do, I can do Meta"*

Metadata: If something is going to vary in a predictable way, store the *description* of the variation in a database so that it is easy to change....Ralph Johnson

*"Meta is Beta"*

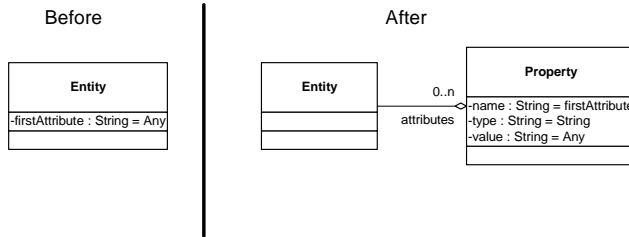
Code is Data, Data is Code – Everything is Data

# Type-Object



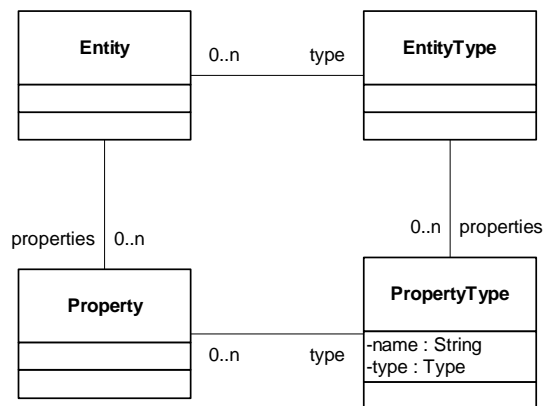
PLoPD3 - Johnson and Woolf

# Properties



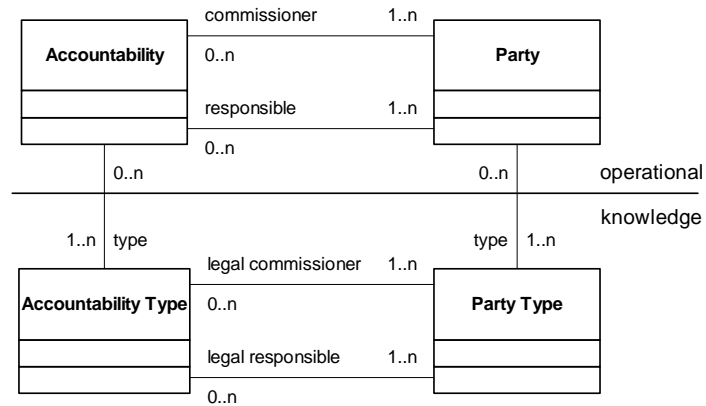
PLoP98 - Foote and Yoder

# Type Square



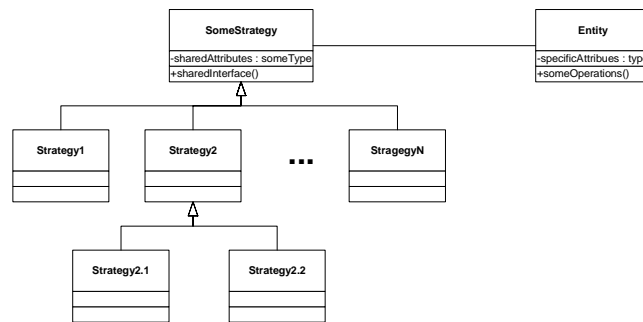
ECOOP & OOPSLA 2000, 2001 – Yoder, Balaguer, Johnson

# Entity-Relationship



Analysis Patterns - Fowler

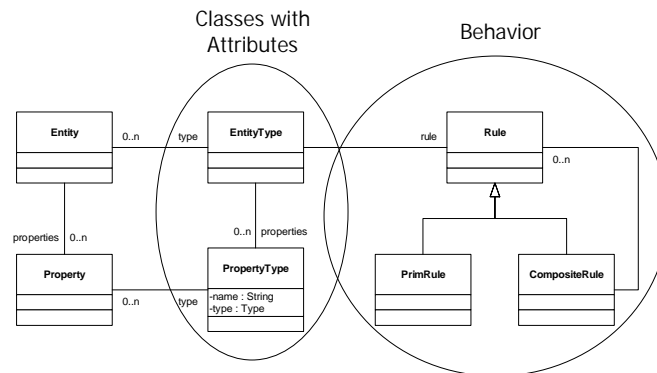
# Strategies/RuleObjects



Design Patterns - GOF95

# Putting It All Together

(Very Common Structure)



ECOOP & OOPSLA 2001 Yoder, Balaguer, Johnson

17

Architecture and Design of Adaptive Object-Models – Intriguing Technology Session, OOPSLA'2001 – October 2001, Tampa FL. Copyright 2001, The Refactory, Inc.

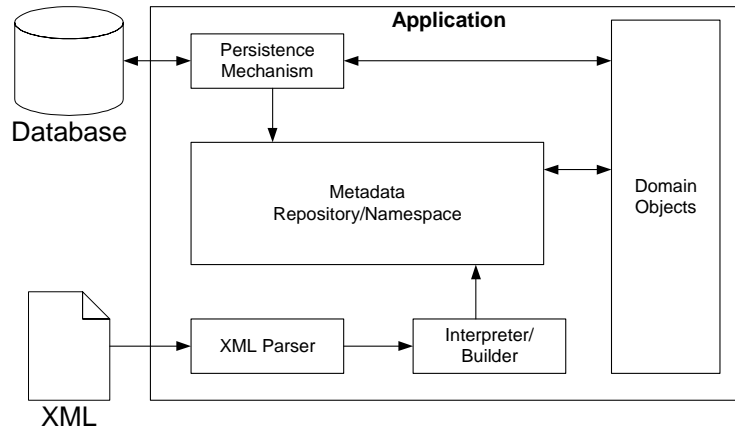
## Other Patterns

- ◆ Composite – GOF
- ◆ Interpreter – GOF
- ◆ Builder – GOF
- ◆ Mediator/Adaptor – GOF
- ◆ History – Francis Anderson PLoPD4
- ◆ Roles – Baumer, Riehle, Siberski, & Wulf  
Fowler PLoP '97
- ◆ RuleObject – Arsanjani PLoP2000

18

Architecture and Design of Adaptive Object-Models – Intriguing Technology Session, OOPSLA'2001 – October 2001, Tampa FL. Copyright 2001, The Refactory, Inc.

# Interpreters / Builders



19

# Adaptive Object-Model Example

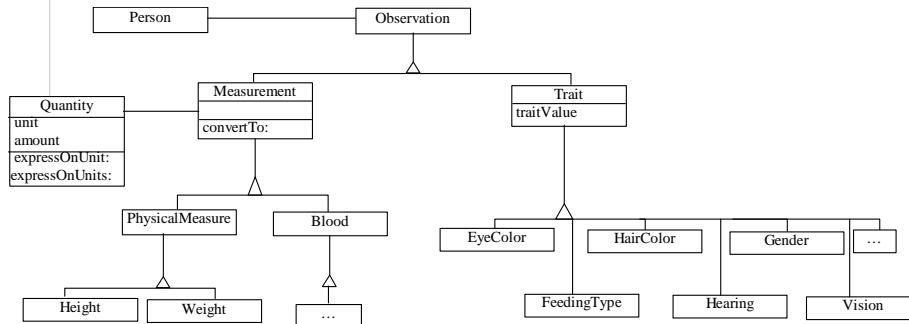
20

# Medical Observations



21

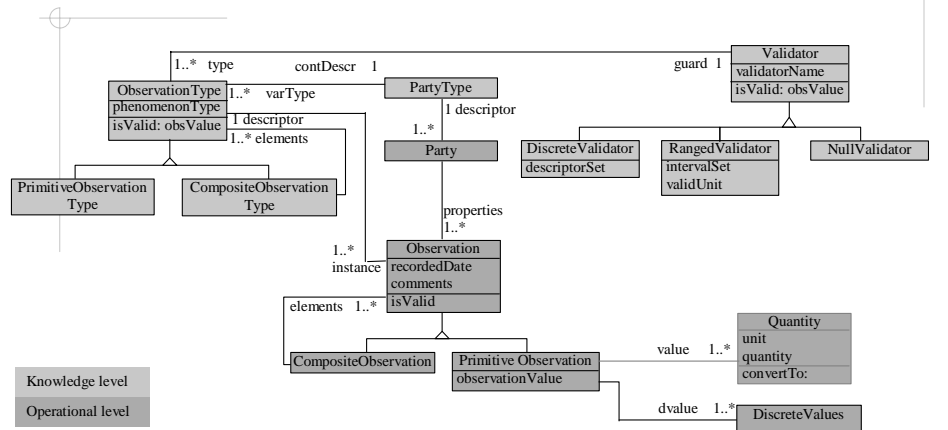
# Observation - First Model



What happens when a new observation is required?

22

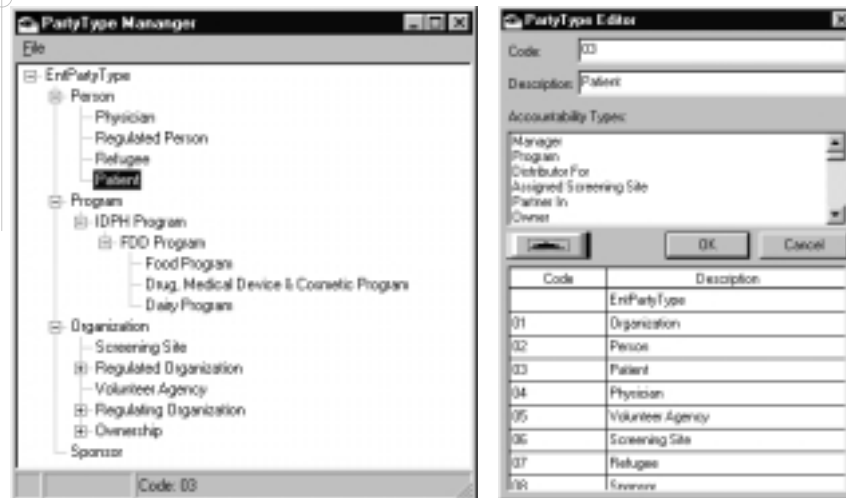
# Observation Example



# Metamodel and GUI

- ◆ Generating Dynamic GUIs is Hard!
- ◆ Can generate GUIs using metadata.
- ◆ Special GUI components can be developed for using the metadata.

## PartyType: Metadata-Editors



Architecture and Design of Adaptive Object-Models – Intriguing Technology Session, OOPSLA'2001 – October 2001, Tampa FL. Copyright 2001, The Refactory, Inc. 25

## Advantages of Adaptive Object-Models

- ◆ Can more easily adapt to new business requirements.
- ◆ Smaller in terms of classes so possibly easier to maintain by experts.
- ◆ Changes do not require recompiling the system.
- ◆ Business People can make changes.
- ◆ Time to market can be reduced.

Architecture and Design of Adaptive Object-Models – Intriguing Technology Session, OOPSLA'2001 – October 2001, Tampa FL. Copyright 2001, The Refactory, Inc. 26

## Disadvantages of Adaptive Object-Models

- ◆ It demands having infrastructure for storing, building, interpreting metadata.
- ◆ Developing AOM can be expensive.
- ◆ Can be hard to understand and maintain.
- ◆ It requires skilled human resources.
- ◆ Can have poor performance.

## Related Approaches and Technologies

- ◆ Generative Techniques
- ◆ Black-box Frameworks
- ◆ Metamodeling Techniques
- ◆ Code Generators
- ◆ Table-driven Systems
- ◆ UML Virtual Machine

## Successfully Used For:

- ◆ Represent Insurance Policies
- ◆ Telephone Billing Systems
- ◆ Model Workflow
- ◆ Medical Domain
- ◆ Financial Domain
- ◆ Validate Equipment Configuration
- ◆ Model Documents
- ◆ Model Databases

29

Architecture and Design of Adaptive Object-Models – Intriguing Technology Session, OOPSLA'2001 – October 2001, Tampa FL. Copyright 2001, The Refactory, Inc.

## Summary

- ◆ Adaptive Object-Models can take time to develop -- but the payoff can be enormous!
- ◆ Adaptive Object-Models work based upon domain expert knowledge.
- ◆ Applying well-known design principles such as TypeObject, Properties, and Strategies & RuleObjects works well for developing Adaptive Object-Models.

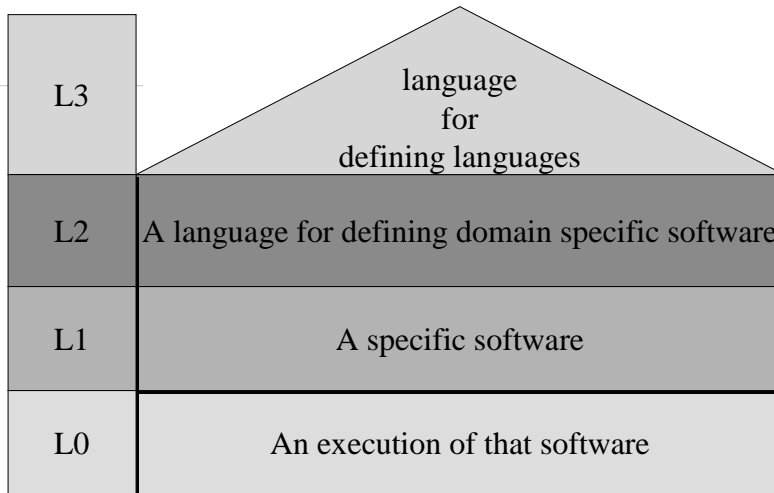
30

Architecture and Design of Adaptive Object-Models – Intriguing Technology Session, OOPSLA'2001 – October 2001, Tampa FL. Copyright 2001, The Refactory, Inc.

# That's All



## Dimensions of Abstraction



Dimensions of abstraction in Adaptive Object-Models, Reflection and OMG's metamodeling Architecture

## Points to Remember

- ◆ AOMs Separates what changes quickly from what changes slowly
- ◆ Takes into account who changes what and where
- ◆ Objects constitute a domain specific language
- ◆ Building languages out of objects can be good...reflection guys say this.