



Data Modeling Using the UML

Dr. Paul Dorsey

Dulcian, Inc.

www.dulcian.com



December 8, 2005

The Problem

◆ Analysis

- Class Diagram – Data model
- Activity Model – Process flow
- Sequence Diagram – Module interaction

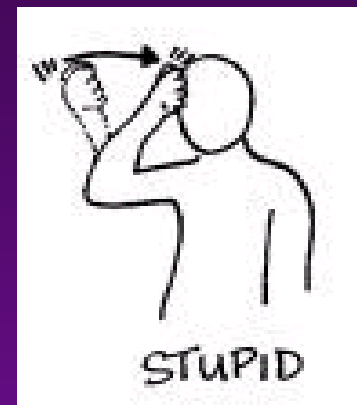
◆ Now what?



The goal from the relational perspective

- ◆ Standard relational design
- ◆ Put object-oriented applications on top.

Data-centric morons!



The goal from the OO perspective

- ◆ Build normal object-oriented applications.
- ◆ Tables are just places to store persistent copies of classes.

OO bigots!



The normal “shotgun” marriage

- ◆ Relational people design the database.
- ◆ OO people design the applications.
- ◆ Some techie is forced to build EJBs.
- ◆ Oracle TopLink is a marriage broker.
 - It solves a problem that shouldn't exist.



Divorce imminent!



The “Right” Goal

- ◆ Merge OO and relational thinking
- ◆ OO-influenced database design
- ◆ OO applications that cleanly work with the database



Analysis and Design

Model Driven Architecture (MDA)

Analysis

- ◆ Use cases
- ◆ Class models
- ◆ Activity models
 - (Object states)
- ◆ Sequence diagrams

Design

- ◆ RDBMS + Middle tier Objects
- ◆ Procedures
- ◆ Screens, major modules



Why is class to database translation so hard?

- ◆ Java programmers want:
 - Classes to map to database objects
 - Derived attributes
 - Abstract classes
 - Generalization
- ◆ Relational database developers want:
 - A nice BCNF database
- ◆ Oracle wants to make everyone happy.
(unwilling to push a solution on us)
- ◆ IBM/Rational follows the OO market.



Classes → Tables (1)

No Primary Keys (not directly)

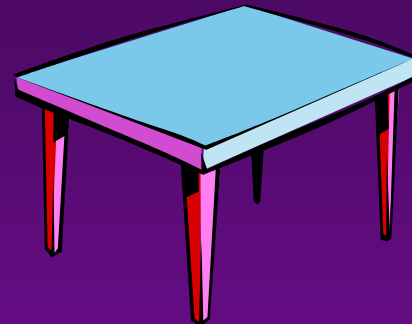
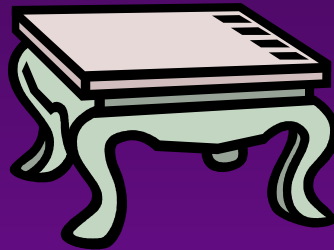
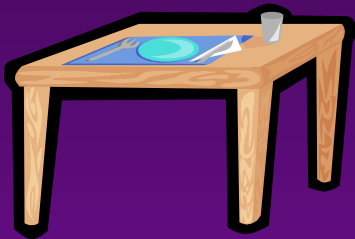
- ◆ Logical primary keys are annoying for physical PK.
- ◆ OIDs are nice for physical primary keys.
- ◆ Logical primary keys are essential for good design.



Classes → Tables (2)

No derived attributes

- ◆ Within table – “icky” triggers
- ◆ Between tables
 - Mutating tables
 - Frequently impossible
- ◆ Almost never updatable due to mutating tables



Classes → Tables (3)

No process flow

- ◆ Need:
 - Open object
 - Process object
 - Close object
- ◆ Triggers are inadequate.
 - Mutating tables on complex objects



Classes → Tables (4)

Limited generalization

- ◆ `Select * from EMP` is a 7-table join.
- ◆ What do you do with abstract classes?
- ◆ What about inherited attributes?



Classes → Tables (5)

Composition and Aggregation

- ◆ Should lock entire object
- ◆ Nasty triggers
- ◆ Frequent deadlocks



Classes → Tables (6)

Complex validation

- ◆ Mutating tables
- ◆ Performance impact (can't turn off)
- ◆ Difficult with complex multi-table validations



What is the big deal? (OO Perspective)

“Just make the tables copies of classes.”

- ◆ Lots of coding
- ◆ Thick middle tier
- ◆ Lots of redundant columns
 - Synchronization issues
- ◆ “Forced” to do most of the management in the middle tier by hand
- ◆ Bad database design



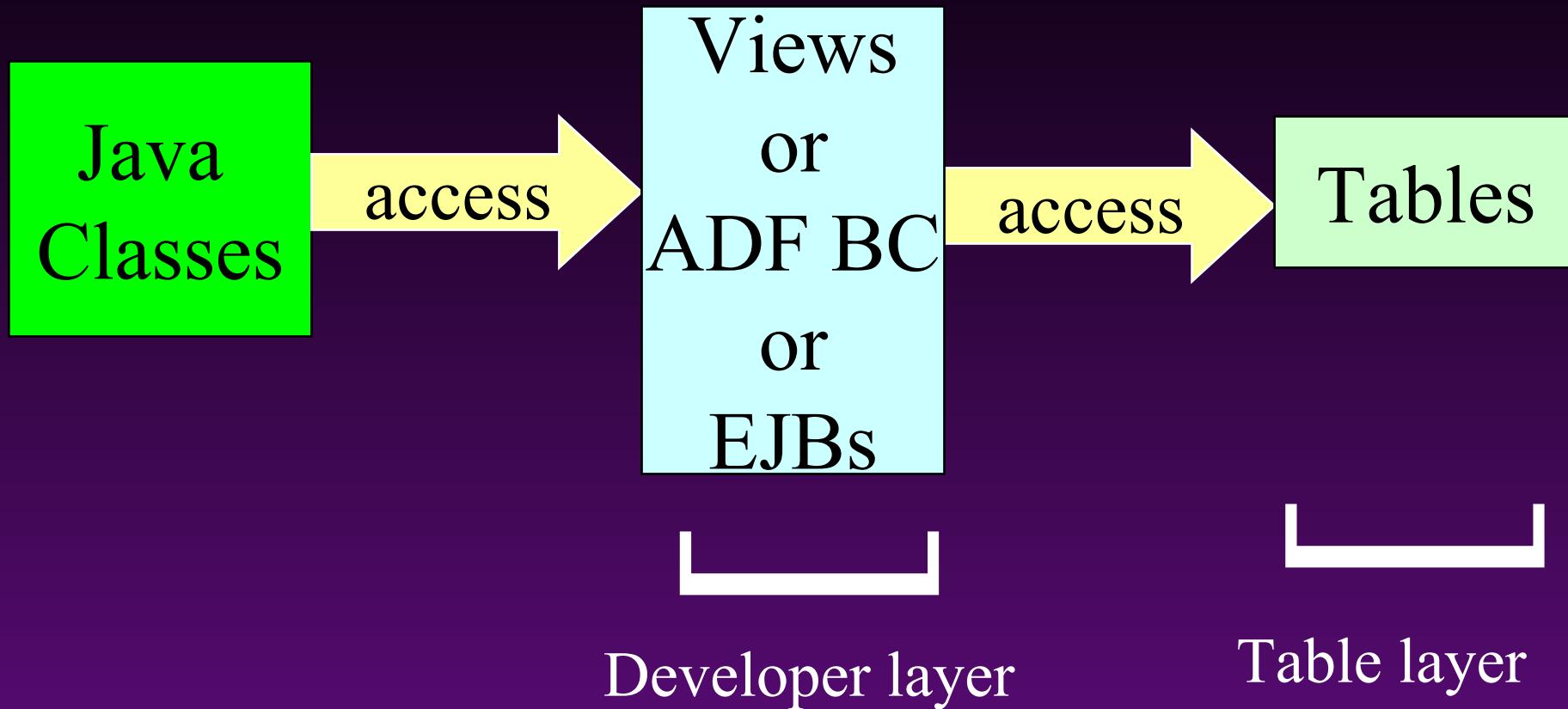
What is the big deal? (DB Perspective)

“Just design the way we always have. Let the developers build what they want when we are done.”

- ◆ OO hostile DB
- ◆ Not agile
- ◆ Huge mismatch
- ◆ Requires TopLink



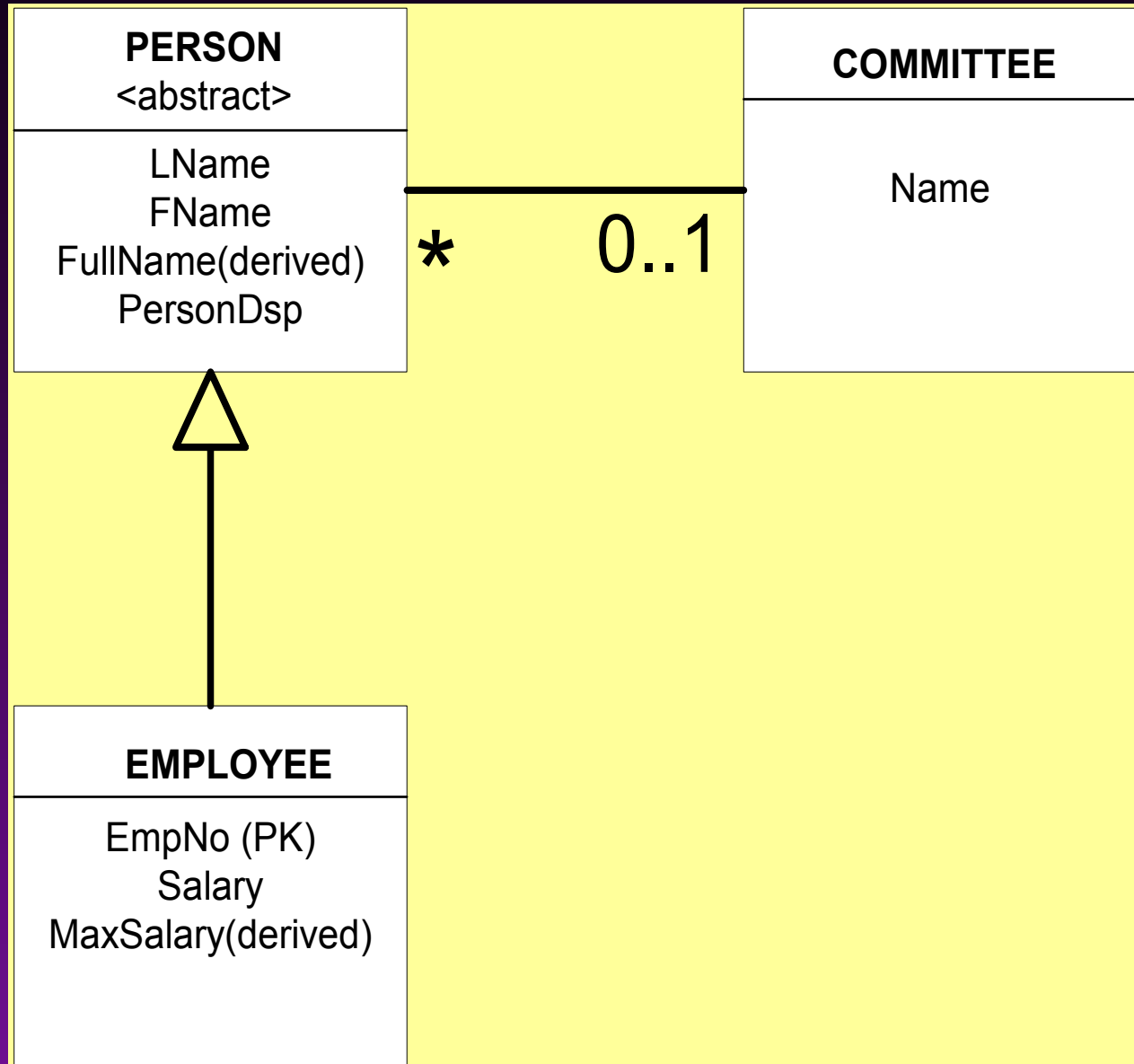
The Solution



Developer layer \neq Table layer

Class diagram generates EVERYTHING

Example



Logical structure

PERSON

Supported Operations

Person_OID	
Committee_OID	Delete
LName	Update
FName	LName
Type	FName
PersonDsp	Committee_OID
CommitteeDsp	

COMMITTEE

Name	Insert
Committee_OID	Delete
CommitteeDsp	Update
	Name

EMPLOYEE

Supported Operations

Employee_OID	Insert
LName	Delete
FName	Update
Full Name	LName
CommitteeDsp	FName
EmpNo	EmpNo
Salary	Salary
MaxSalary	Committee_OID
PersonDsp	
Committee_OID	

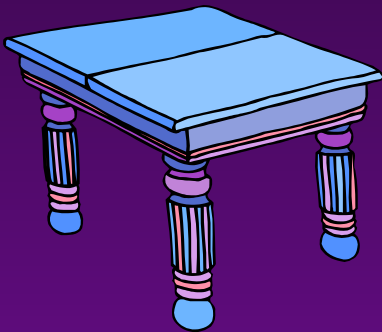
Note: Employee_OID \equiv Person_OID

Physical Database Table Structure

Committee

Committee_OID

Name



Employee

Employee_OID

Committee_OID

Salary

LName

FName

What are the logical structures?

◆ 1. Views – Thick DB

➤ Pros

- Fastest performance (usually)
- Most adaptable (.Net, J2EE agnostic)

➤ Cons

- DB Focused
 - Oracle, DB2, MS-specific
 - Developers need to know database coding

◆ 2. Enterprise JavaBeans (EJB) – Middle tier

➤ Pros

- Database agnostic
- Java only

➤ Cons - Less scalable

◆ 3. Proprietary framework (modified 2)

➤ ADF BC, TopLink

➤ Pros - May be easier to do

➤ Cons - Proprietary

What is best?

◆ Database views

- You have database skills
- Big application
- Database platform is stable
- User Interface platform is unstable

◆ EJB

- Java shop
- Product (Database-independent)
- Smaller application

◆ Proprietary

- Hand-coding



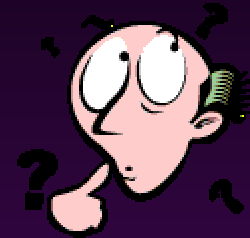
Class diagram “style”

- ◆ This is not a normal OO programming class diagram.
- ◆ It accommodates database design principles.
- ◆ It requires a “normalized” design.
- ◆ What is a “normal” class diagram?

What does all this mean?

◆ I am saying:

- The idea of logical + tables
- Logical \neq Tables
- Tables should be BCNF (mostly)
- IBM/Rational and Oracle have missed the boat.



◆ I am not saying:

- The columns in my logical views and tables are right.
- There is only one good algorithm.

Products using UML for data modeling

◆ JDeveloper 10g



◆ Rational Rose Data Modeler



◆ BRIM[®]



BRIM[®] (Business Rules Information Manager)

JDeveloper 10g Entity Object Modeler (1)

- ◆ The first is from the Entity Object Modeler in the Application Development Framework (ADF) Business Components (BC) portion of the product.
- ◆ Start with a fully-formed relational database and build middle tier components, generating most of the structure from the database.
- ◆ Developers can then modify this structure, adding significant business logic in the middle tier.
- ◆ To accommodate requests from users wanting to model within the same tool, Oracle added the capability of first building the middle tier components and then using these components directly to generate the database.

JDeveloper 10g Entity Object Modeler (2)

- ◆ Mapping from business component elements to the database is relatively simple-minded:
 - Business component entity objects are directly translated to relational tables.
 - Entities become tables.
 - Attributes become columns.
- ◆ Foreign key attributes must be manually specified prior to generation.
- ◆ Tables are dropped and recreated during generation.
 - Cannot add attribute if data already exists in tables

JDeveloper 10g class diagrams

- ◆ Second mechanism for data modeling in JDeveloper
- ◆ Classes can be stereotyped as tables.
- ◆ Users define tables, columns, foreign keys, check constraints.
- ◆ No notion of generalization or derived attributes.
- ◆ Both modeling mechanisms store metadata repository in XML files.

JDeveloper 10g Pros and Cons

PROS

- ◆ Physical database modeler that allows users to specify tables, columns, and foreign key relationships using UML class diagrams.
- ◆ JDeveloper is second to none as a Java IDE.
- ◆ OO-centric designers who are satisfied with the generation algorithm will be happy with the product.

CONS

- ◆ No inheritance, aggregation, and composition capabilities
- ◆ The Database Modeler cannot be used yet for complete logical and physical database design.
- ◆ Still need a full-featured database design tool such as Oracle Designer in conjunction with JDeveloper

Rational Rose Data Modeler (1)

- ◆ Marginally farther along than JDeveloper in the generation of Oracle databases.
- ◆ Possible to specify limited class diagrams which forward generate into data model class diagrams.
- ◆ Primary keys are automatically specified as foreign keys on relationship.
- ◆ The repository is in its own proprietary document format, which is accessible and editable but making changes is not very easy.
- ◆ Excels in the development of the software management process.



Rational Rose Data Modeler (2)

- ◆ Visual modeling tool that makes it possible for database designers, analysts, architects, developers and anyone else on your development team to work together.
- ◆ Captures and shares business requirements and tracks them as they change
- ◆ Realization of the ER methodology using UML notation to bring database designers together with the software development team.
- ◆ Can capture information such as constraints, triggers and indexes directly on the diagram rather than representing them with hidden properties behind the scenes.
- ◆ Can transfer between object and data models
- ◆ Takes advantage of basic transformation types such as many-to-many relationships.
- ◆ Intuitive way to visualize the architecture of the database and how it ties into the application.

Rational Rose Pros and Cons

PROS

- ◆ Industry standard for OO design and development.
- ◆ Mature, well-written tool
- ◆ Supports OO industry standard that “a database is nothing more than a place to store persistent copies of our classes.”
- ◆ Best tool for generating tables that look like classes

CONS

- ◆ Leads to poor database design
- ◆ Doesn't support the richness of class diagrams.
- ◆ Classes and attributes will get directly translated into tables and columns.

- ◆ Business rules approach to fully generate systems using an “executable UML” approach.
- ◆ Class diagram is specified including inheritance, derived attributes, etc.
- ◆ Views and relational database tables to support the class diagrams are simultaneously generated.
- ◆ The BRIM repository is stored in an Oracle database which can be queried or updated through APIs.
- ◆ Uses object IDs (OIDs) as the physical primary key for all tables.
- ◆ Views that interface with the tables keep the data from getting out of synch.

BRIM[®] Pros and Cons

PROS

- ◆ Generates both tables and views
- ◆ Allows for good database design as well as a sound OO structures
- ◆ Once business rules placed in the BRIM repository, system is generated.
 - Not necessary to wait for system to be complete before generating V1.
- ◆ Additional system pieces can be quickly generated,
 - RAD environment - virtually no cycle time.
- ◆ BRIM[®] repository = set of Oracle tables.
 - Complete set of APIs exists (some used by the Repository Manager), any or all of which can be used to manipulate the repository.

CONS

- ◆ Only works in the Oracle environment
- ◆ Takes a strong stand on the “right” way to generate a database (and indeed the whole system).
- ◆ Few options in the generation algorithms.
- ◆ No market share

Conclusions

- ◆ 1. UML class diagrams are a great way to do data modeling.
- ◆ 2. Tools to support this approach are still evolving.
 - OO-centric tools (like IBM's Rational Rose) cater to OO designers with little interest in good database design.
 - Mainstream relational vendors such as Oracle have yet to figure out how to use a class diagram to generate a well-designed database.
 - Fringe products like Dulcian's BRIM[®] may show promise but lack the credibility of products from larger companies.

The J2EE SIG

Co-Sponsored by:



Chairperson – Dr. Paul Dorsey



About the J2EE SIG

- Mission: To identify and promote best practices in J2EE systems design, development and deployment.
- Look for J2EE SIG presentations and events at national and regional conferences
- Website: www.odtug.com/2005_J2EE.htm
- Join by signing up for the Java-L mailing list:
 - <http://www.odtug.com/subscrib.htm>

J2EE SIG Member Benefits

- Learn about latest Java technology and hot topics via SIG whitepapers and conference sessions.
- Take advantage of opportunities to co-author Java papers and be published.
- Network with other Java developers.
- Get help with specific technical problems from other SIG members and from Oracle.
- Provide feedback to Oracle on current product enhancements and future product strategies.



Share your Knowledge: Call for Articles/Presentations

◆ Submit articles, questions, ... to

IOUG – The SELECT Journal

select@ioug.org

ODTUG – Technical Journal

pubs@odtug.com



New York Metro Area Users Group Meeting



September 29, 2005

To be added to email list – info@nyoug.org



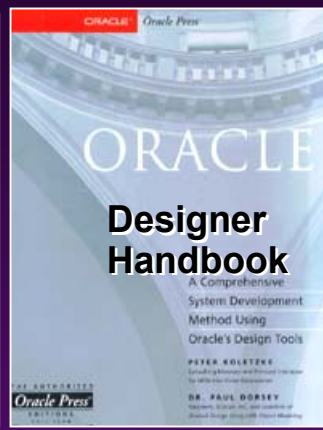
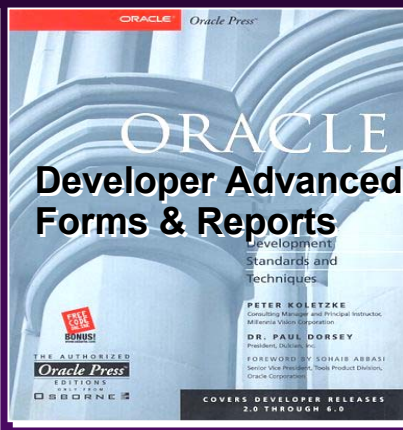
Dulcian's BRIM[®] Environment

- ◆ Full business rules-based development environment
- ◆ For Demo
 - Write “BRIM” on business card
- ◆ Includes:
 - Working Use Case system
 - “Application” and “Validation Rules” Engines



Contact Information

- ◆ Dr. Paul Dorsey – paul_dorsey@dulcian.com
- ◆ Michael Rosenblum – mrosenblum@dulcian.com
- ◆ Dulcian website - www.dulcian.com



Coming in 2006:
Oracle PL/SQL for Dummies

