PHP Object-Relational Mapping Libraries in action

Apr 14, 2010

O'Reilly MySQL Conference and Expo

Santa Clara, CA

Fernando Ipar & Ryan Lowe
Percona Inc.

Agenda

- What are ORM libraries
- Object Oriented vs Relational models
- ORM patterns
- ORM libraries for PHP
- Conclusions
- Questions? Stories? Comments?

What are ORM libraries?

- Tools to
  - Isolate developers from SQL
  - Bridge the gap between two different models
- Model
  - Should help see things in a simple way, but still be powerful enough to reflect reality
- Mapper
  - *An object that sets up communication between two independent objects*
  - One or both of them can be a subsystem
  - (Fowler et al, Patterns of Enterprise Application Software)

What are ORM libraries?

- They're NOT the DB class
  - DB.execute("SELECT id,name FROM customers")
- They map OO domain models to R tables
- Help deal with changing OO domain models
- Help deal with changing relational schema
- Developers want to persist objects and not care how it's done
- Developers tend to have SQL aversion

Object Oriented vs Relational

- Object Oriented
  - Everything is an object (Not really in PHP)
  - Units of data and behavior collaborating to get the job done
  - Abstraction
  - Encapsulation
    - SQL is also a form of encapsulation
  - Polymorphism
  - Modularity
  - *Inheritance*.

Object Oriented vs Relational

- Relational
  - Everything is a relation (not really in any RDBMS that I know of)
  - Relations (table definitions), relational variables (tables), Tuples (rows), attributes (columns).
  - Based on first order predicate logic.
  - Relations
    - Heading → unordered set of tuples

Object Oriented vs Relational

- Object Oriented
  - Class: Customer
    - Defines a 'template' for all customers
    - Every customer object is an instance of this Class.
    - They all share the same behavior, maybe some common data, maybe some private data for each instance.
    - Can probably be stored in a non specified way (serialization? ORM? Hand written query?)
    - An object instance means 'I am Customer X, with age N, etc'

Object Oriented vs Relational

- Relational
  - Table : Customers
    - Defines a predicate for all customers
    - 'A customer has an id N, name X, etc'
      - This is the table definition
    - All customers share this predicate, have individual rows or tuples
    - A row (N,X,...) asserts a truth fact: 'The predicate for Customers has truth value TRUE for id N, name X, etc'.
    - In everyday work, we simplify this as 'There's a record with id N in the table Customers'
    - Data is also stored in a non specified way
      - In available products, we use SQL to store and retrieve data, but we don't know and mostly don't care how that's implemented
        - » Until we grow really big and/or things go really bad

Object Oriented vs Relational

- Object Relational Impedance Mismatch
  - Difficulty to map an OO domain model into a relational model
- Data type problems
- Design mapping problems
  - OO domain models and relational models are not isomorphic

Object Oriented vs Relational

- Design mapping problems
- Interfaces
- Inheritance
  - Single table mapping
    - Discriminator column
  - Table per class mapping
    - Foreign keys, outer joins
  - Table per *concrete* class mapping
    - Foreign keys, outer joins, less tables
  - Hybrid mapping

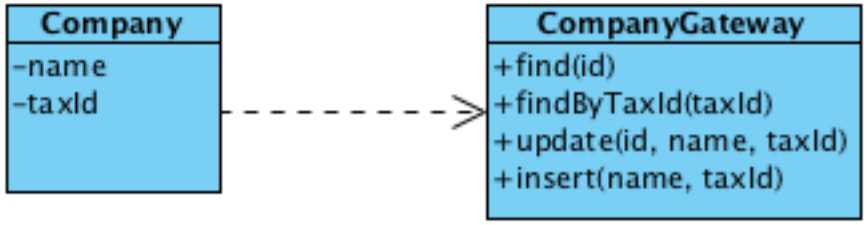
ORM patterns

- Table Data Gateway
- Row Data Gateway
- Active Record
- Data Mapper
- Metadata Mapping
- “Each pattern describes a **problem** which occurs over and over again in our environment, and then describes the **core** of the **solution** to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice” Christopher Alexander

Table Data Gateway

- Usually stateless
- Useful for direct or simple mapping between objects and RDBMS
- Good candidates for automatic generation
  - Can be used in combination with Data Mapper

Table Data Gateway



one CompanyGateway instance handles all rows in the *company* table

Holds all SQL for accesing the *company* table

Zend_Db_Table

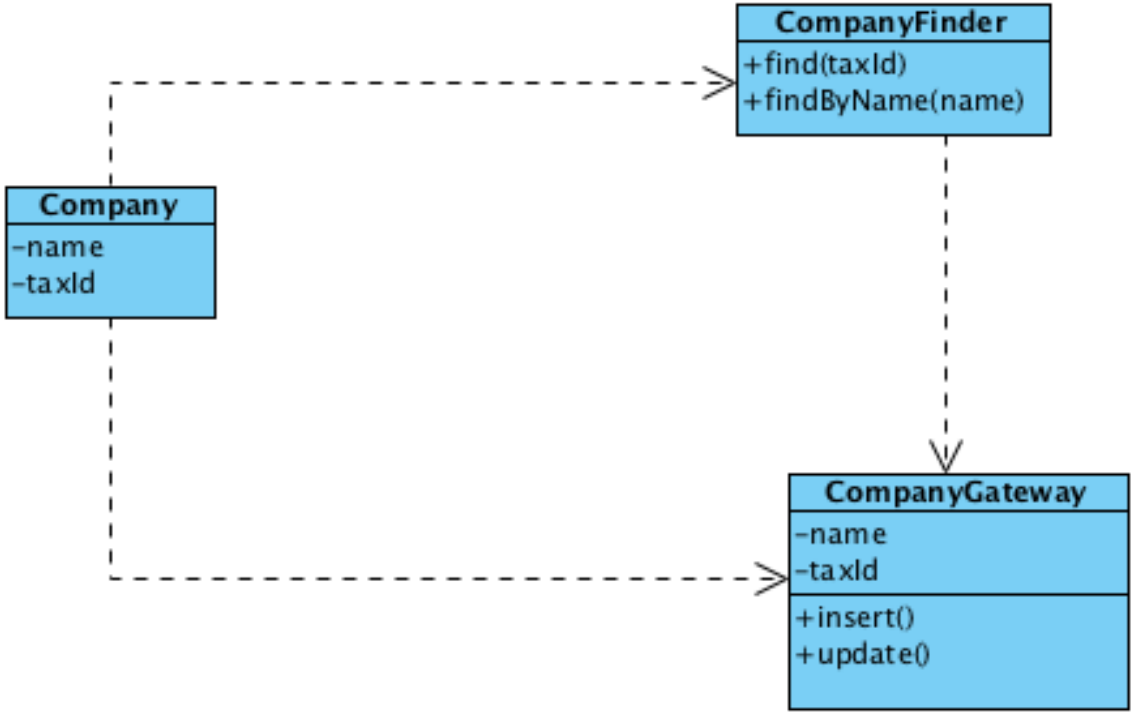
Propel (Table Data Gateway, Row Data Gateway)

Solar (Table Data Gateway, Data Mapper, Metadata Mapping)

Row Data Gateway

- Objects match records in the database
  - One object attribute per database field
- It's actually Active Record with no domain logic
- Good candidates for automatic generation
  - Can be used combined with Data Mappers

Row Data Gateway



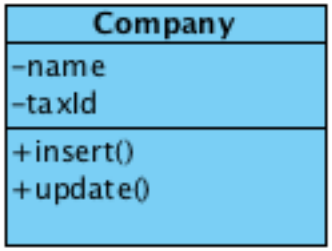
One instance per row.

SQL is decoupled from OO domain by two helper objects

Active Record

- Objects match records in the database
  - One object attribute per database field
- It's actually Row Data Gateway with domain logic
- It's also an OO domain model that's isomorphic to a relational model
- Good candidate for code generation
  - RoR, CakePHP

Active Record



Each *Company* instance wraps a row in the *company* table. An object has data and behavior. Data access logic is in the domain object.

Doctrine (Active Record, Data Mapper, Metadata Mapping)

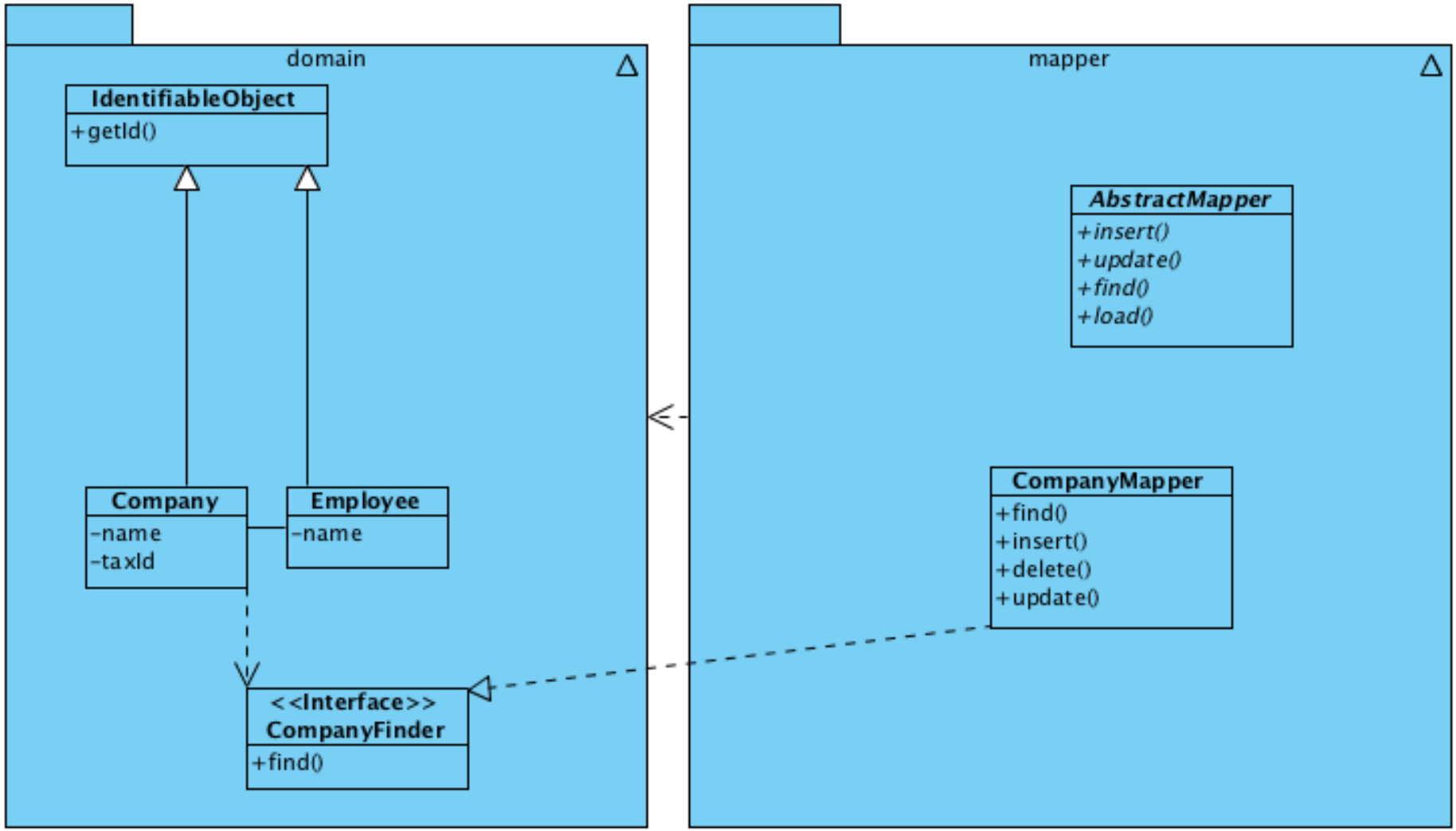
Solar (Table Data Gateway, Data Mapper, Metadata Mapping)

CodeIgniter (Active Record)

Data Mapper

- Use a layer of mappers to communicate objects and RDBMS
- Useful when
  - OO domain model has many differences with relational model
  - Both models may change independently
  - OO dependencies (an Order object that also stores all it's Item objects)
  - OO inheritance (Inheritance Mappers)
  - Foreign Key (Foreign Key Mapping)

Data Mapper



Metadata Mapping

- Defines mapping between object attributes and database columns
- Two alternatives
  - Code generation
    - Impact on build and deploy procedures
    - Need to regenerate and redeploy if model changes
  - Reflection
    - Impact on performance
    - Some model changes can be detected automatically

Metadata Mapping

Metadata mapping deals with how fields in the database match attributes in objects

Code generation, reflection, or mapping information in db schema

ORM Patterns

- Good match OO domain model ↔ relational model
  - Table Data Gateway
  - Table Row Gateway
  - Active Record
- Complex OO domain model
  - Data Mapper
- Metadata Mapping

ORM libraries for PHP

	MDM	RDG	TDG	AR	DM
Propel		x	x		
Doctrine	x			x	x
Zend			x		
Solar	x		x		x
CodeIgniter				x	
CakePHP				x	

Propel

- Table Data Gateway, Row Data Gateway
- Straightforward for single table mapping:

```
$company = new Company();
$company->setTaxId(1234);
$company->setName('Percona');
$company->save();
```

- No mapping support for MxN relationships, so:
  - You need to reference the cross-reference table
  - Or you can put your own methods in stub classes
- From user docs:
  - Fetch all readers for every book:
  - SELECT * FROM BOOK;
  - N x SELECT * FROM book_reader_ref LEFT JOIN reader ON reader.reader_id = book_reader_ref.reader_id WHERE book_reader_ref.book_id = $book->getBookId()
  - **select r.name, b.title from book b join book_reader_ref br on (b.id = br.book_id) join reader r using (reader_id);**

Propel

- Single table inheritance mapping
  - Table must have all columns needed by all classes
- Full query logging
- Supports read/write splitting
- Criteria class to simplify SQL

```
$c = new Criteria();
$c->add(CompanyPeer::NAME, “Percona”);
$companies = CompanyPeer::doSelect($c);

SELECT * FROM company WHERE company.NAME = 'Percona';
```

Examples - Doctrine

- Active Record, Data Mapping, Meta Data Mapping
- Create model
  - From DB
  - From YAML
- Doctrine Query Language

```
$q = Doctrine_Query::create()
  ->from('User u')
  ->leftJoin('u.Phonenumbers p');
SELECT u.id AS u__id, ....
               p.id AS p__id,
               p.phonenumber AS p__phonenumbe
FROM user u
LEFT JOIN phonenumber p ON u.id = p.user_id
```

PERCONA
Performance Consulting Experts

Examples - Zend

- Table Data Gateway

```
Zend_Db_Table::setDefaultAdapter($adapter);
$companies = new Zend_Db_Table('companies');

$data = array( 'taxId' => 1234, 'name' => 'Percona');
$companies->insert($data);

INSERT INTO companies (taxId,name) VALUES (1234,'Percona');
```

Conclusions

- It's back to impedance mismatch
- Choose the framework with the right model / pattern for your OO domain model
- Beware of default data types
- Beware of subjective advice
  - “A JOIN wouldn't really be the best choice here”
- Watch the SnR in queries
  - i.e. SHOW CREATE TABLE
- Decomposed joins, inheritance mapping, ...

Questions

- fernando.ipar@percona.com
- Booth 308
- LAMP performance problems? Yes, we can help
  - http://www.percona.com

PERCONA
Performance Consulting Experts