Object-oriented programming
Persistence, post-object paradigms

Marcin Młotkowski

22nd May, 2012
Outline

1. Programming by contract
2. Aspect-oriented programming
3. Object persistence
   - Combine objects with relational databases
   - Persistence in Java
   - Persistence C#
   - Object-oriented databases
4. Distributed objects
Brief story of DbC

- a concept of designing and implementing software proposed and designed in 80’s by Bertrand Meyer;
- supported by Eiffel programming language;
The main idea

Prepare a contract between a provider (a method/function/class) and a client who uses this method.
How the contract looks like

**Obligations**
A client must provide data or arguments which satisfies certain conditions.

**Benefits**
A client gets a result satisfying its requirements.
How to control the contract

Assertions

logical conditions
How to control the contract

Assertions

logical conditions

Types of assertions

- preconditions
- postconditions
- invariants
Example

Add a node to a tree

put_child(new: NODE) is
    -- add a subnode
require
    new /= Void
do
    ...
ensure
    new.Parent = Current
    child_count = old child_count + 1
end
Invariants

Assertions, which must be always satisfied by object’s state.
Invariants

Assertions, which must be always satisfied by object’s state.

Implementation

Invariant is assigned to preconditions and postconditions.
Invariant: an example

class BINARY_TREE[T] feature

    ...

    invariant

    left /= Void implies (left.Parent = Current)
    right /= Void implies (right.Parent = Current)

end
More on contracts

- Invariants are inherited
- Assertions are checked dynamically
- If a condition is not satisfied, program raises a runtime error or print a warning
static void Main(string[] args)
    
    requires args.Length > 0;
{
    
    foreach (string arg in args)
    {
        
        Console.WriteLine(arg);
    }
}
Other implementation of DbC concept:

- SPARC: based ADA for critical software;
- CoFoJa: Contracts for Java (google project)
Outline

1. Programming by contract
2. Aspect-oriented programming
3. Object persistence
   - Combine objects with relational databases
   - Persistence in Java
   - Persistence C#
   - Object-oriented databases
4. Distributed objects
Methods and procedures besides their 'core' functionality do some auxiliary tasks (concerns), such as authorization, logging etc. Those tasks interleave, e.g.

- client authorization;
- receiving a transaction;
- acceptance;
- logging.
Diagnosed problems

- method has more than one responsibility
- task interleaving decreases reusing
Aspect
A programming unit which implements secondary unit.

join points
A place where aspect is assigned
A simple interactive system, with manner aspects.
public class HelloWorld {
    public static void say(String message) {
        System.out.println(message);
    }

    public static void sayToPerson(String message, String name) {
        System.out.println(name + " , " + message);
    }
}

Implementation (AspectJ)
public aspect MannersAspect {
    pointcut callSayMessage() :
        call(public static void HelloWorld.say*(..));
    before() : callSayMessage() {
        System.out.println("Good day!");
    }
    after() : callSayMessage() {
        System.out.println("Thank you!");
    }
}
Implementations

- AspectJ: Java extension (Georg Kiczales, Parc XEROX)
- Demeter (C++/Java)
- LOOM.NET and WEAVER.NET
Outline

1. Programming by contract
2. Apect-oriented programming
3. Object persistence
   - Combine objects with relational databases
   - Persistence in Java
   - Persistence C#
   - Object-oriented databases
4. Distributed objects
Definition

It is a feature which allows to store the state of an object.
Saving state in database

Application

Class definitions
method implementations

save a state

save a state

database

states of objects

retrieve a state
Typical scenario of usage od persistence object

- retrieve from a storage
- modify a state/call a method
- save
Problems

- state is stored, not methods
- Lazy or eager updates?
- problem with changes: new fields, changes of types, changes of fields in a superclass
- which part of state should be stored?
Different approaches

- combine object-oriented application with existing database
- Extend a object environment by processing large number of objects
- Build a new environment from scratch
- extend a database system by object-oriented features
Combine objects with databases

Current state

- Object-oriented environment: JAVA, C#, C++, Python ...
- Database engines: MySQL, Oracle, PostgreSQL, Sybase, Microsoft SQL Server ...
Implementation strategy

- design projection of object features (fields, inheritance, references) into relations (object-relational mapping, ORM)
- implement intermediate layer
## Object and relational approaches comparison

<table>
<thead>
<tr>
<th>Object model</th>
<th>Relational model</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Hiding data, access only by interface</td>
<td>- record</td>
</tr>
<tr>
<td>- Relations among objects: associations, inheritance, composition</td>
<td>- table: collection of</td>
</tr>
<tr>
<td>- Object identity</td>
<td>records</td>
</tr>
<tr>
<td></td>
<td>- query language</td>
</tr>
</tbody>
</table>
Java Data Object

- defines **a specification**;
- object may be stored in files, relational and object databases;
- details (eagerness, laziness, etc) are defined in external xml files;
- example implementations: Apache JDO, DataNucleus
Java Persistence API

- it is a specification;
- dedicated to store objects in relational databases;
- configuration is defined via annotations or in external files;
- Java Persistence Query Language: query language similar to SQL;
- implementations: Hibernate, TopLink.
C#: ADO.NET

- ADO - ActiveX Data Objects
- ADO.NET – environment to DB
Entity Framework

- part of ADO.NET environment
- data is defined in abstract form: Entity Data Model
- system defines the structure of relational database and declarations of classes automatically
LINQ: Language Integrated Query

Example

Northwind db = new Northwind(connectionString);
var q = from o in db.Orders, c in db.Customers
    where o.Quality == "200"
    select new { o.DueDate, c.ItemID};
foreach (var t in q) { ... }
Extensions of relational database: Oracle

PL/SQL – extension of SQL:
- class declaration (User Data Type);
- SQL extended by method invocation.
STONE environment

- All objects are stored on a server;
- Objects are not copied on client’s server;
- Client has only proxies, methods invocations are forwarded to server.
Smalltalk

The state of application is stored in one file (image of session).

Disadvantages

- Problems with large data
- Problems with data sharing
Outline

1. Programming by contract
2. Aspect-oriented programming
3. Object persistence
   - Combine objects with relational databases
   - Persistence in Java
   - Persistence C#
   - Object-oriented databases
4. Distributed objects
The concept of distributed objects

- An object (its state) and implementation is stored on a server;
- All methods are executed by a server;
- The server takes care on data consistency, multiple access etc.
A uniform access to local and remote objects;
Easy access to remote objects;
Independent of platform
Implementations

- Java RMI
- CORBA
- SOAP
- DCOM
Java RMI

RMI – Remote Method Invocation

- only for Java applications
- java.rmi.*
- it is a part of standard SDK
CORBA

- it is a set of standards developed by OMG (Object Management Group)
- independent of platform/hardware/programming language
- Implementations: VisiBroker, ORBit
Simple Object Access Protocol
- based on XML
- W3C standard
- Implementations: .NET Remoting, Apache SOAP
- .NET Remoting
Distributed Component Object Model

- Microsoft Win* only
- deprecated
finally ...