ITDevCon

Delphi European Conference

DORM, the Delphi ORM

Daniele Teti
R&D Director & Educational
bitTime Software

October 27/28 2011 VERONA

bit Time software

The Delphi ORM

- Delphi implementation of Data Mapper Enterprise Design Pattern
- It's not related to RAD, but LiveBindings allows a good «RAD compromise»

Why DORM?

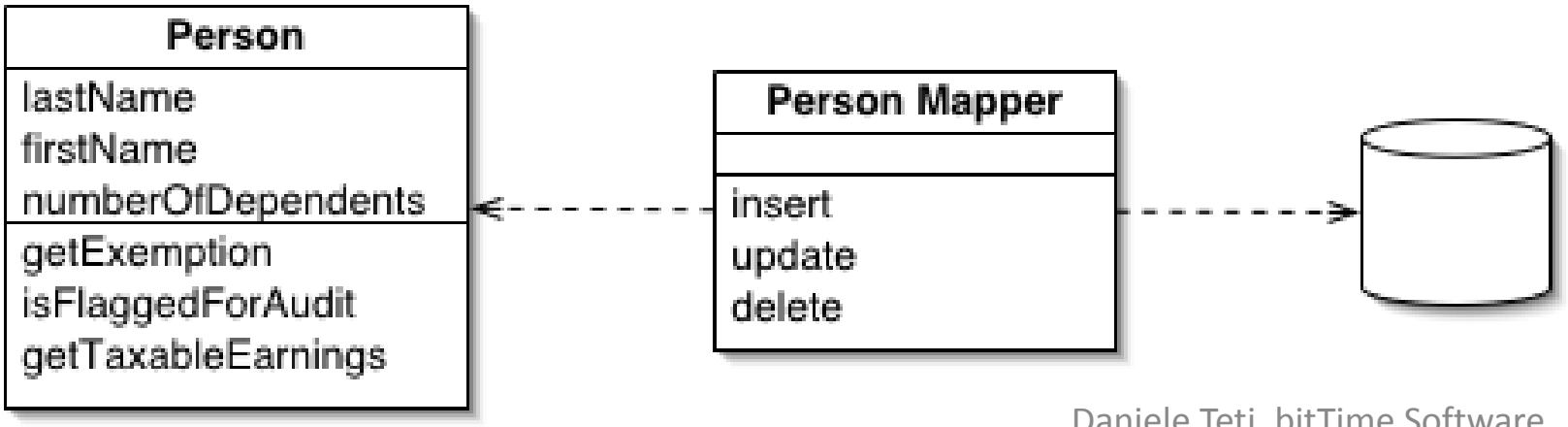
- I dont want (or I cant) change my database
- I want persistence ignorance
- I want to access to the «metal»
- I want to persist PODO
- I don't hate SQL, but I know that 90% of it is boilerplate and errors prone

Who is behind DORM?

- DORM’s author is Daniele Teti
- Contributors are welcome
- bitTime software is the main sponsor and offer support for DORM custom development, consultancy and so on.

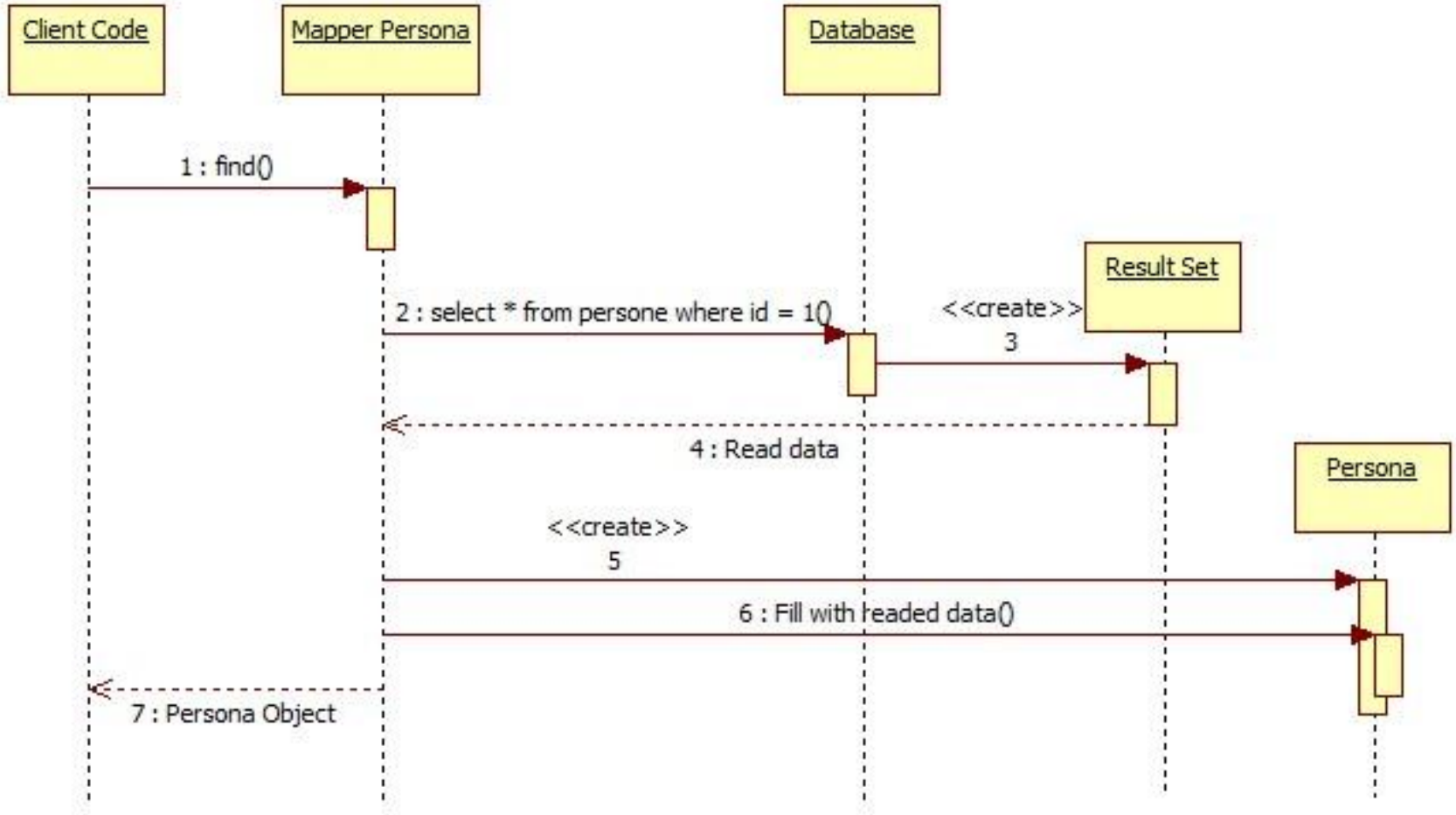
DORM foundation: Data Mapper Pattern

- A software layer that separe objects from the database
- Persists/retrieve objects state from/to database
- Complete separation between domain model and database
- Objects do not know the mapper



Is it a new idea?

DataMapper Sequence Diagram



Let's say that you have a PODO (or a set of PODOs)


```
TPerson = class(TObject)
  private
    FLastName: string;
    FAge: Int32;
    FFirstName: string;
    FID: UInt8;
  public
    constructor Create(FirstName, LastName: String; Age: UInt8);
    property ID: Integer read FID write SetID;
    property FirstName: string read FFirstName write FFirstName;
    property LastName: string read FLastName write FLastName;
    property Age: Int32 read FAge write FAge;
  end;

TManager = class(TPerson)
  private
    FBenefits: String;
  public
    property Benefits: String read FBenefits write FBenefits;
  end;
```

WOW effect slide. How to insert and retrieve an object form a sql database or other persistence systems


```
var
  person: TPerson; oid: Integer;
begin
  person := TPerson.Create('Daniele','Teti',32);
  Session.Save(person);
  oid := person.id;
  person.Free;
  person := Session.Load<TPerson>(oid);
  WriteLn('Person name is: ', person.FirstName);
  person.Free;
end;
```

DORM features

- Open Source «Apache License 2.0»
- Database agnostic
- Do not require database changes!
- «Has one», «has many» and «belongs to» relations support
- Save and retrieve objects graph, not only single objects
- Support persistence inheritance (TPerson, TManager)
- File based configuration
- Interfaces based!
- Event based validation (TdormObject)
- Persistence ignorance
  - Can persists everythig!
- It is already used in REAL BIG 3tier system
  - So it will be supported and mantained J

DORM features

- Very good performances
- Completely Unit tested
- SQL script generation from configuration file
- Multiple environments
  - Development, Test, Production
- Lazy Load for related objects
  - has_one, has_many, belongs_to
- Fluent interfaces for queries
- Unit Of Work for multiple operations (UOW)
- Use anonymous methods, generics
- Does not require a particular «editor» to design classes. Together (in Delphi) in enough.
- Tested on Delphi XE and XE2 (Win32/Win64)
  - Should works also on Delphi 2010

DORM configuration file

- It is a normal JSON file
- Can be linked as resource
- Defines
  - Mapping
  - Persistence
  - Log
  - Primary Key type
  - Primary key generator type

Supported databases

- Firebird SQL 2.1+
  - DBExpress
  - UIB
- Interbase 2009+
  - DBExpress
  - UIB
- SQLServer
  - DBExpress
- SQLite

DORM Samples

ITDevCon

Retrieve, update and delete an object with DORM

ITDevCon

```
var
  person: TPerson; oid: Integer;
begin
  person := Session.Load<TPerson>(1);
  try
    person.FirstName := ‘Daniele’;
    person.LastName := ‘Teti’;
    Session.Update(person);
    Session.Delete(person);
  finally
    person.Free;
  end;
end;
```

SELECT * FROM PEOPLE WHERE ID = 1
UPDATE PEOPLE SET FIRST_NAME = ‘Daniele’, LAST_NAME = ‘Teti’ WHERE ID = 1
DELETE FROM PEOPLE WHERE ID = 1

```
var
  p: TPerson;
begin
  p := TPerson.Create('Daniele','Teti');
  try
    p.Phones := NewList; //TdormCollection.Create
    p.Phones.Add(TPhone.Create('328-9823883'));
    p.Phones.Add(TPhone.Create('328-4242424'));
    Session.Save(p);
    Session.Commit;
  finally
    p.Free;
  end;
end;
```

```
var
  p: TPerson;
begin
  //Session.SetLazyLoadFor(
       TypeInfo(TPerson), 'Phones', false);
  p := Session.Load<TPerson>(12);
  try
    for phone in p.Phones do
      WriteLn(phone.number);
  finally
    p.Free;
  end;
end;
```

Simple search based on attributes


```
var
  Criteria: TdormCriteria;
  People: TdormCollection;
begin
 Criteria := TdormCriteria.
    NewCriteria('FirstName',
       TdormCompareOperator.Equal, 'Daniele').
    Add('LastName',
       TdormCompareOperator.Different, 'Smith');
  People := Session.List<TPerson>(Criteria);
  //do something with people
end;
```

SELECT * FROM PEOPLE WHERE FIRST_NAME = 'Daniele' AND LAST_NAME != 'Smith'

Complex queries (fast and dirty way)

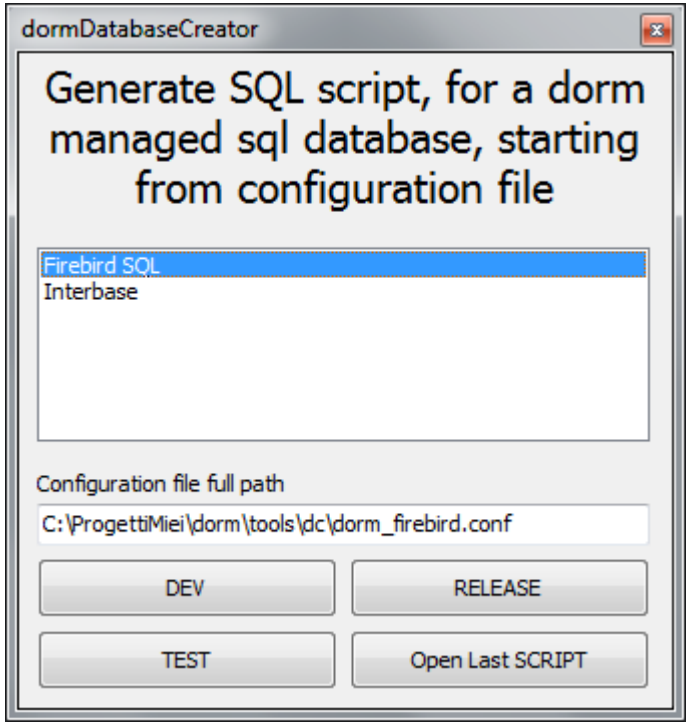

```
var
  intf: IdormSearchCriteria;
  SQL: String;
begin
  SQL := 'SELECT * FROM PEOPLE ... a lot of SQL';
  intf := TdormSimpleSearchCriteria.Create(
      TypeInfo(TPerson), SQL);
  People := Session.List(intf);
end;
```

Complex queries – the correct way. Finders


```
TUsersByRole = class(TdormInterfacedObject,
                                    IdormSearchCriteria)
  private
    FRoles: TArray<string>;
  public
    function GetItemClassInfo: PTypeInfo;
    function GetSQL: string;
    constructor Create(Role: string); overload;
    constructor Create(Roles: TArray<string>); overload;
  end;

//Use a finder
Users := Session.List(TUsersByRole.Create('admin'));
```

DORM tools: dormDatabaseCreator



- Useful for new applications
- Generate SQL script from dorm conf file
- It’s aware to the «environment» settings
- Simple to extend

DORM road map

- Many to many relations
- Support for strongly typed collections
  - TObjectList<TPerson>
- Per-Object custom mappers
- More databases support
  - MySQL, Oracle, SQLite, MSSQLServer, ??
- File based (not SQL) supports
  - JSON, CSV, XML, ??
- More tools
  - Create config file from existing databases
  - Create objects from config file
- Demos, demos, demos
- Build a strong community

What about flexibility?

System Component	Can be customized implementing	Notes
Persistence	`IdormPersistenceStrategy`	Currently there are • Firebird SQL • Interbase • SQLServer • SQlite
Logging	`IdormLogger`	Currently there are a CodeSite logger and a flat file logger
Primary key generators	`IdormKeysGenerator`	Currently there are a sequence based generator and a GUID generator
Finders	`IdormSearchCriteria`	You can create your own queries arbitrary complexes

DEMOS

- ObjectsLists with LiveBindings
- Persistence Ignorance

Where is DORM?

The Delphi ORM

http://code.google.com/p/delphi-orm/

Are you interested in DORM?

- Try the DORM demos
- Add issues on google code.
  - I'll try to respond to all the requests support
- Follow my blog (*www.danieleteti.it*)
  - There will be demos, articles and others stuff

The Delphi ORM

Thank You