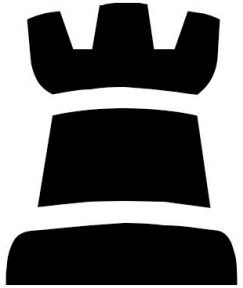


DATABASE MIGRATION

Open-Source Programming
Project: MigDB – Database Migration
Web (GitPage): http://migdb.github.com/migdb/
Git: https://github.com/migdb/migdb
Day: 21. 4. 2013
Řešitel: Petr Tarant

ISSUE: #131 EXTRACT CLASS

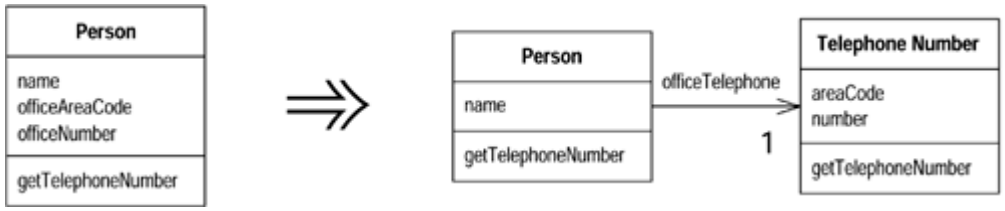
Mechanism of operation:

- Decide how to split the responsibilities of the class.
- Create a new class to express the split-off responsibilities.

  If the responsibilities of the old class no longer match its name, rename the old class.
- Make a link from the old to the new class.

  You may need a two-way link. But don't make the back link until you find you need it.

Example:



LAYER: APPLICATION META-MODEL

- First of all I had to define the operation in the application meta-model.

Code from meta-model source:

```
class ExtractClass extends AtomicOperation{
    attr String[1] sourceClassName;
    attr String[1] extractClassName;
    attr String[1] associationPropertyName;
}
```

Used technology and languages:

XMI, Ecore, Genmodel, JAVA, EMF

LAYER: APPLICATION EVOLUTION

- Then I had to create an evolution transformation on application layer.
- On appliacation layer must be defined set of restriction for operation.

Restrictions:

- Class with **@sourceClassName** must exist in model
- Class with **@extractClassName** cannot exist in model
- Class name **@extractClassName** must be valid
- Property name **@associationPropertyName** cannot exists in class **@sourceClassName**

Used technology and languages:

QVTO, OCL

LAYER: ORM MAPPING

- Every atomic (atomic mean irreducible operation) operation in our system has ORM mapping to database.

CODE:

```
addTable(schema, extractClass);
addColumn(schema, extractClass, idColumn, type);
addPrimaryKey(schema, extractClass, idColumn, pk);
addColumn(schema, sourceClass, assocProperty, type);
addForeignKey(schema, sourceClass, assocProperty, fk, extractClass);
```

Used technology and languages:

QVTO, OCL

LAYER: TESTING

- Testing is the last thing which have to be done.
- For testing must be create a lot of files (input struncture, transformation file, required structure)

Testing component:

```
component = TestComponent{
    transformationFile = "${TEST_TRANSFORMATION}„
    outputParentUri = "${OUTPUT_BASE_DIR}/14_extractClass„
    qvtInput = "${IN_STR}/extractClass_str.qvto„
    qvtInput = "${IN_OPS}/extractClass_ops.qvto„
    qvtComparison = "${COM_STR_DIR}/extractClass_required.qvto„
    qvtComparison = "${COMPARE_ERRORS_DIR}/empty_errors.qvto„
    testDescription = "extractClass„
}
```

Used technology and languages:

MWE2, JAVA, QVTO, OCL

RESULT: EVERYTHING DONE!

- My issue has been tested (by team tester) and accepted (by project leader).
- During development the team leader changed several input requirements.
- After accepting the task I created documentation for the newly formed operation (that documentation was accepted too).

COMMUNITY: COMMUNICATION

- Communication with community was excellent (email communication, video chat to Japan, GitHub messages).
- There is a strict hierarchy in the community (Tester, validator,…)

- Sometimes I had a problem with response time from some team member (for example tester).
- For Japanese is English very big problem.
- Documentation could be improved.

THANK YOU FOR YOUR ATTENTION

Petr Tarant