Charles University in Prague

Faculty of Mathematics and Physics

MASTER THESIS

Bc. Radovan Duga

Querying NoSQL databases in MPS

Department of Distributed and Dependable Systems

Supervisor of the master thesis:	RNDr. Pavel Parízek, Ph.D.
Study programme:	Computer Science
Specialization:	Software Systems

Prague 2014

Dedication.

Název práce: Dotazování NoSQL databáz v prostředí MPS

Autor: Bc. Radovan Duga

Katedra: Katedra distribuovaných a spolehlivých systémů

Vedoucí diplomové práce: RNDr. Pavel Parízek, Ph.D., Katedra distribuovaných a spolehlivých systémů

Abstrakt: S příchodem NoSQL databází se objevila i potřeba pro vznik doménově specifických dotazovacích jazyků. Jednou ze zajímavých domén jsou grafové databáze jako například Neo4j s dotazovací jazykem Cypher. Doménově specifické jazyky (DSLs) může být navržena a snadno použita pomocí speciálních vývojových prostředích zvaných Language Workbenche. Velmi populární Language Workbench je MPS, který implementuje koncept projekčních DSLs.

Tato práce zodpovídá otázku, zda Language Workbenche a projekční DSLs mohou být přínosem v doméně NoSQL databází, vystihnout výhody projekčních DSLs použitím různých typů přístupu. Dalším specifickým cílem je navrhnout a implementovat dotazovací DSL jazyk pro vybranou NoSQL databázi (např. Neo4J nebo Redis) jako případová studie.

Klíčová slova: NoSQL, MPS, dotaz, Cypher

Title: Querying NoSQL databases in MPS

Author: Bc. Radovan Duga

Department: Department of Distributed and Dependable Systems

Supervisor: RNDr. Pavel Parízek, Ph.D., Department of Distributed and Dependable Systems sl Abstract: With the advent of NoSQL databases, a need for targeted domain-specific query languages has become evident. One of the interesting domains are graph databases, such as Neo4j with the query language Cypher. Domain specific languages (DSLs) can be designed and easily used with the help of special development environments called Language Workbenches. A very popular Language Workbench is MPS, which implements the concept of projectional DSLs.

This work will answer the question whether Language Workbenches and projectional DSLs can make a contribution in the domain of NoSQL databases, and identify the benefits of projectional DSLs over different approaches. An additional specific goal is to design and implement a practical MPS-based query DSL for a chosen NoSQL database (e.g., Neo4J or Redis) as a case study.

Keywords: NoSQL, MPS, query, Cypher

Contents

1. Introduction

- there exist lot of general dsl languages - universal dsl language xml - domain specific

1.1 Motivation

1.2 Goals

The goals of this thesis are these:

- Discuss MPS contribution in DSL languages
- Discuss MPS contribution in NoSQL domain
- Design and implement Neo4jCypher language and projectional editor and integrate it into MPS BaseLanguage (Java)

2. Background

2.1 NoSQL databases

2.1.1 Neo4j graph database

Neo4j is the most famous graph database system which is subset of NoSQL databases - databases which offer more expresiveness than the classic SQL databases. The graph database is consisted of nodes and relationships which gives us the power to query and update more complex expressions than in the classic table database approach and gives us helpful graph functions like function for searching shortest path and so. We chose this system because this system offers in addition to classical java api approach also the DSL approach - Cypher DSL language which gives us the more powerful tool to express what we want to process.

2.1.2 Neo4j Cypher query language

Cypher is a declarative domain specific query language for Neo4j graph database. It offers high expressivity and efficient querying and updating of the graph in this database. It is designed both for developers and for domain exprerts, so the syntax of the language is easy to understand.

Syntax of Cypher

To understand the principles and problems of design and implementation of Neo4jCypher (Cypher implementation in MPS language workbench) we provide you the short overview of Cypher syntax.

Cypher query language is composed of these basic clauses:

- `START` - specifies starting inputs
- `MATCH` - specifies pattern to match from starting nodes
- `WHERE` - next filtering of matched nodes
- `CREATE` - creates nodes, relationships
- `DELETE` - deletes nodes, relationships
- `SET` - updates nodes, relationships
- `RETURN` - returns nodes, relationships, etc.

Now we give some examples of queries with explanation what they do:

Read-Only Query:

START
[**MATCH**]
[**WHERE**]
RETURN [**ORDER BY**] [SKIP] [**LIMIT**]

Write-Only Query:

CREATE [**UNIQUE**] ∗
[**SET** | **DELETE** | FOREACH] ∗
[RETURN [**ORDER BY**] [SKIP] [**LIMIT**]]

Read-Write Query:

START
[**MATCH**]
[**WHERE**]
[**CREATE** [**UNIQUE**]] ∗
[**SET** | **DELETE** | FOREACH] ∗
[RETURN [**ORDER BY**] [SKIP] [**LIMIT**]]

2.2 Domain Specific Languages

2.2.1 Neo4jCypher DSL

2.3 Language Workbenches

2.3.1 How to define DSL in MPS

2.4 MPS Language Workbench

2.4.1 How to Define DSL in MPS

Structure DSL

Editor DSL

Constraints DSL

Typesystem DSL

Intensions and Other Parts

2.4.2 MPS Pros and Cons

3. Design of Neo4jCypher

3.1 Problem analysis

3.2 Design decisions

3.2.1 Projection editor

MPS Projection editor is a base approach of MPS to editing language. It differs from pure textual editors in these points:

- Unparsable editor notation - editor is composed of cells, every cell represents some node of the abstract syntax tree (AST)
- Possible switching among multiple notations using alternative editors, where every editor can show the AST in different way
- Combine notations from different authors and be extensible. It is possible to extend MPS languages, compose them and so on

3.2.2 Text-like editor

3.2.3 Graphical extensions

4. Implementation details of Neo4jCypher language

4.1 Patterns

4.2 References

4.3 Integration into BaseLanguage

MPS has a big advantage that we not only can design the new language and create the full ide support for this language but also we can integrate this language into some existing language, extend it and provide the way to execute our code with possibility to debug our code.

We chose these 4 steps, where each next step integrates the Neo4jCypher language more closely into its BaseLanguage:

- Design Neo4jCypher queries in Query Sheet editor - editor with full ide support:
  - code completition with correct reference visibility and full cypher function list
  - intensions for query transformation, parts of query grouping, relationship type conversion, properties definition
  - identifier uniquiness checking
  - items type checking where necessary
- Generation of MPCypher query into BaseLanguage string type - this approach provides us easy creation of query with full ide editor support. The next query manipulation depends on user's will. User can store this query or execute it using Neo4j Execution Engine from Neo4j java library.
- Using CypherExecutor object which adds us easy access to execution of Neo4jCypher query without need of implementation any further code from Neo4j java library. This approach tries to provide very easy way to query the database without need to know any king of Cypher java interface.

5. Evaluation

MPS platform is a robust tool for creating DSL languages

5.1 Experience with MPS

5.2 MPS Contribution in DSL languages

5.3 MPS Contribution in NoSQL Domain

5.4 Related work

5.5 Case Study

Conclusion

Bibliography

[1] Fowler, Martin. *Domain-Specific Languages.* Massachusetts: Addison Wesley, 2013. ISBN 0-321-71294-3.

[2] JetBrains. *MPS Documents and Live Demos.* `http://www.jetbrains.com/mps/documentation/index.html`.

[3] JetBrains. *MPS User's Guide.* `http://confluence.jetbrains.com/display/\MPSD25/MPS+User's+Guide`.

[4] Voelter Marcus. *Language and IDE Modularization, Extension and Composition with MPS.* `http://voelter.de/data/pub/Voelter-GTTSE-MPS.pdf`.

[5] Neo Technology. *The Neo4j Manual v1.9.5.* `http://docs.neo4j.org/chunked/1.9.5/cypher-introduction.html`.

List of Tables

List of Abbreviations

Attachments