Design and Implementation of a MongoDB Driver for Prolog

Sebastian Lundström

April 28, 2011

Abstract

Prolog is good stuff, and MongoDB ...

Contents

1 Introduction

[Write this, or at least finalize it, near the end.]

From spec:

[MongoDB is a young document-oriented database system that has started to gain much attention recently. Document-orientation involves removing rigid database schemas and advanced transactions, in favor of flexibility. Document-orientation also promotes a certain degree of denormalization which allows embedding documents into each other, leading to potentially much better performance by avoiding the need for expensive join operations.

Prolog, being an untyped language, agrees with the document-oriented approach of relaxing manifests in order to create more dynamic and flexible systems. Embedding terms in other terms is natural in Prolog, and embedding documents in other documents is natural in MongoDB.

Many drivers exist, both official and unofficial, that enable the use of MongoDB from various programming languages. At the time of writing, no such driver for Prolog seems to exist.]

2 Background

This chapter briefly discusses what MongoDB and Prolog are.

2.1 MongoDB

MongoDB is a document-oriented database management system that emphasizes speed and flexibility. This is accomplished by sacrificing statically defined database schemas and allowing a certain degree of denormalization by embedded documents into each other.

2.1.1 Document-Orientated vs. Relational Databases

2.2 Prolog

Prolog is an interpreted dynamic language with its roots in logic.

2.2.1 Type System

XXX

3 Scope

From spec:

[Creating a driver that is usable with, or at least easily portable to, other Prolog environments is desirable, but development and testing will focus on SWI-Prolog.

More complex features of MongoDB, such as advanced connection management and GridFS, will not be implemented.]

4 Method (necessary?)

Research other drivers, docs.

Test-driven development.

5 Requirements

Various requirements on the implementation. See for example

http://www.mongodb.org/display/DOCS/Writing+Drivers+and+Tools

6 Design

- BSON encoder/decoder
  - Some parts written in C. Why? (Basically didn't know how to easily handle bytes-to-float in Prolog. And perhaps some efficiency.)
  - Discuss data structures, term [key:value] maps to bytelist [4,1,7,9,3,...] etc.
  - Design choices: text as atoms (why not list of codes, [97,98,99]?) Inspired slightly by JSON parser: http://www.swi-prolog.org/pldoc/doc_for?object=section%283,
- Network communication
  - How does the communication work? How simplistic is the communication administration? Sockets, connections, etc.
- MongoDB API
  - Thoroughly discuss design of wrapper API, how lists and structures are represented etc. What algorithms are used etc.

7 Implementation

How do I solve it?

8 Evaluation

Did it work? Is it usable? What should have been done differently?

9 Related Work

Compare to existing drivers? Erlang? And something completely different?

10 Conclusion/Future Work

Not sure how much this section relates to Evaluation above.

Portability (Tested on Mac, SWI, GCC/Clang, etc.)? Efficiency? How to improve in the future? Critical parts (BSON) in pure C? Even write a C extension with more/most functionality? Don't know how portable that would be though, but SWI (and probably SICStus also?) has a mature interface to C. What is missing for it to be a "real" driver?

11 References

References for Prolog DCG? References for various stuff used in SWI? At all relevant? How much web references? Most MongoDB driver stuff is web.

Chodorow, K. & Dirolf, M. (2010) *MongoDB: The Definitive Guide.* Sebastopol, United States of America: O'Reilly Media, Inc.