UNIVERSITEIT VAN AMSTERDAM

MASTERS PROJECT

Representation Mismatch Reduction for Development in Rules-Based Business Engines

Author:
Paul SPENCER

Supervisor:
Dr. Clemens GRELCK

A thesis submitted in fulfillment of the requirements for the degree of Master of Software Engineering

in the

Graduate School of Informatics
Faculty of Science

July 19, 2021

UNIVERSITY OF AMSTERDAM

Declaration of Authorship

I, Paul SPENCER, declare that this thesis titled, "Representation Mismatch Reduction for Development in Rules-Based Business Engines" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. Except for such quotations, this thesis is entirely my work.
- I have acknowledged all of the main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

UNIVERSITEIT VAN AMSTERDAM

Abstract

Graduate School of Informatics
Faculty of Science

Master of Software Engineering

Representation Mismatch Reduction for Development in Rules-Based Business Engines

by Paul SPENCER

Context: Declarative rules engine languages, such as Drools, can become difficult to reason about when there are many rules.
Objective: This project investigates how different projections of the code can ease the comprehensibility of the code
Method: We created an implementation of the Drools language using the MPS language workbench and made innovative projections of large ASTs
Results:
Keywords:
Paper type: Research paper

Acknowledgements

We would like to acknowledge the School of The Gradate school of Informatics in the Faculty of Science at The University of Amsterdam for their guidance, specifically Dr. Clemens Grelck, who has been a supportive, understanding, and available academic adviser.

My greatest thanks go out to Toine Khonraad, an alum of this course, who provided me with moral and monetary support, as well as wisdom and friendship that aided in the completion of this, my fourth attempt at getting this project behind me.
...

Contents

List of Figures

List of Tables

Chapter 1

Introduction

> The limits of my language mean the limits of my world.
>
> *Logico-Tractatus Philosophicus*
> *Ludwig Wittgenstein*

1.1 Motivating example

1.2 Research questions

1.3 Contributions

1.4 Project context

1.5 Thesis outline

In Chapter 3 we present the research questions. Further, the chapter describes the protocol that we use for search strategy, selecting our studies, extracting data from them, and synthesizing the results. Chapter 4 presents the results of our synthesis of data from the primary studies. This is followed, in chapter 5, by a discussion of both the validity of the work and the implications of the findings. We discuss the implications of this study in chapter 6. Finally, the conclusions are presented in chapter 7.

Chapter 2

Background

2.1 RulesEngines

2.1.1 What is a rules engine?

In this section we will describe what a rules engine is and a little of its history.

The Aristotelian doctrine of essentialism declares that a thing has properties that are essential and properties that are accidental. If one takes away accidental properties, then the thing remains the thing. If one takes away essential properties, the thing is no longer the thing. If the thing is a business application, then its essential properties are its business rules.

Simply put, business rules are the rules by which an organization carries out the tasks needed to achieve their goals. When properly defined these rules can be encoded. In the typical application architecture, these rules are mostly distributed in the source code or database. Documentation describing these rules may be found in the design documentation or user manuals. However, as applications evolve documentation gets out of sync with codebase. Once this desynchronization occurs, to know what the rules that govern the application, one has to navigate the codebase and decode the rules from their, often scattered, locations.

Rule engines arose from the expert systems of the late 70s and early 80s. Expert systems initially had three main techniques for knowledge representation: Rules, frames and logic[1]. "The granddaddy" of the expert systems, MYCIN, relied heavily on rules based knowledge representation[2], rather than long inference chains. MYCIN was used to identify bacteria and recommend antibiotic prescriptions. MYCIN and its progenitor, DENDRAL, spawned a whole family of Clinical Decision Support Systems that pushed the rules engine technology until the early 1980's. Research into rules engines died out in the 1980s as it fell out of fashion.

[TODO: ADD MORE HISTORY HERE]

The goal of a rules engine is the abstraction of business rules into encoded and packaged logic that defines the tasks of an organization. Date[3] defines the concept of the rules engine as to "specify business process declaratively, via business rules and get the system to compile those rules in to the necessary procedural (and executable) code." Fowler[4] describes rules engine as follows: " ... providing an alternative computational model. Instead of the usual imperative model, which consists of commands in sequence with conditionals and loops, a rules engine is based on a Production Rule System. This is a set of production rules, each of which has a condition and an action ...".

[TODO: ADD MORE ADVANTAGES HERE]

In summary a rules engine, is the executor of a rules based program, consisting of discreet declarative rules which model a part of the business domain.

2.1.2 What is Drools?

2.2 Projectional Editing

2.2.1 What is projectional editing?

2.2.2 what are Language Workbenches?

2.2.3 What is MPS?

Chapter 3

Method

Chapter 4

Results

the purpose of abstraction is not to be vague but to create a new semantic level in which one can be absolutly precise.

Logico-Tractatus Philosophicus
Edsger W. Dijkstra

Chapter 5

Discussion

5.1 Threats to Validity

5.1.1 Construct Validity

5.1.2 Internal Validity

5.1.3 External Validity

5.1.4 Reliability

5.1.5 Repeatability vs Reproducibility

5.1.6 Method improvement

Chapter 6

Implications to research and practice

6.1 Implications to research

6.2 Future research directions

6.3 Implications to practice

Chapter 7

Conclusion

Appendix A

InterviewTranscripts

Write your appendix content here.

Bibliography

[1] P. Jackson, "Introduction to expert systems," 1986.

[2] E. H. Shortliffe, "Mycin: A rule-based computer program for advising physicians regarding antimicrobial therapy selection.," STANFORD UNIV CALIF DEPT OF COMPUTER SCIENCE, Tech. Rep., 1974.

[3] C. J. Date, *What not how: the business rules approach to application development*. Addison-Wesley Professional, 2000.

[4] M. Fowler, *Should i use a rules engine?* https://martinfowler.com/bliki/RulesEngine.html, Accessed: 2021-07-18, 2009.