Master Project Proposal Template

Paul Spencer: 11721677
University of Amsterdam
paul.spencer@student.uva.nl

January 2021

Project details

DONE

- **Project title:** Projecting Drools
- **Host organization:** Khonraad Software Engineering B.V.
- **Contact Person:** Toine Khonraad, MD, a.khonraad@khonraad.nl +31 (0)6 51 61 47 55

1 Project summary

DONE

Drools is an open source production rule system for complex event processing, using an implementation of the Rete algorithm. It has it's own Domain Specific Language, which are written in Drools files. When there are many rule in an environment it can be difficult to reason about the rules and how they interact.

Although Drools is nearly 20 years old and has wide use, it does not have strong cross platform IDE support. This masters project will approach fixing this by creating a projectional editing IDE. There are two popular editing methods for language workbenches, these are free-form editing, where the user typically edits the source code, and projectional editing, where the user edits one or more projections of the persisted model[3]. Projectional editing could be seen as a method of bypassing the parser and programming directly into the Abstract Syntax Tree. In this project we will use the opensource language workbench MPS from JetBrains[6]. There is no existing implementation of the Drools language in MPS.

Khonraad Software Engineering's product, Khonraad, is a SaaS used by the Dutch local governments, implementing the Wvggz, Wzd and Wth laws. These laws deal with very sensitive details about mental health and domestic violence.

It goes without saying that this data must not be accessed by the wrong people. Drools is the Technology choice Khonraad made to govern this critical function. Thus, they would like to improve the reasonability of this code.

This project will attempt to answer the following research questions:

- **RQ1:** How can we apply projectional editing to Drools to increase reasoning?
- **RQ2:** Which refactorings will most benefit a Drools code clarity?

2 Problem analysis

Here you present your analysis of the problem that your research will address. Also summarize existing scientific insight into the problem (result of your literature survey, see below). You may also touch on how this problem manifests itself at your host organization.

3 Research method

Gregor [4], gives "A Taxonomy of Theory Types in Information Systems Research". To answer our research questions we will conduct what she calls "Type V: Theory for Design and Action". The criteria for success of this type of research "include utility to a community of users, the novelty of the artefact, and the persuasiveness of claims that it is effective".

We have observed the difficulty with which developers are having to reason about and edit large collections of Drools files. We hypothesize that developers can be presented with different views on their code that will allow them to better understand the code. The novelty of our approach will be to add different views specific to the needs of a drools programmer.

The major tasks in this prototype development will be:

- modelling the Drools language.
- developing the alternative projections.

We will be relying on MPS as well as other open source components to work together with acceptable performance such that the user experience is acceptable.

The projection design, which will run in parallel to the Drools language modelling, will depend on the

Present how you are going to find the answers to your research question. This section should cover:

- What will make the research difficult?
- What is the input you expect from the literature survey
- What sources will you use and how will you use / document them?

- What experiments / research will you do? What proof of concept will you make?
- What method will you use?
- Which hypothesis do you have?
- Present a time line
- How will you validate your research?

4 Expected results of the project

DONE

We expect the following from this project:

- We will be able to model Drools in MPS.
- A suite of novel and useful projectional editors for the Drools language.
- We will reduce the thought to execution cycle for Drools Developers. resulting in a reduction of the "cognitive distance and representation impedance mismatch" [18] that they currently experience.

A happy side effects of this project is that the following open source products will become generally available:

- An improved Drools Editor plugin for those using the JetBrains IntelliJ community edition.
- An MPS implementation of drools that can be used for cross generation by other DSLs.

5 Required expertise for this project

DONE

Skill	Required	Acquired	Notes
MPS	★★★★★	★★½☆☆	Currently taking various courses
Drools	★★★½☆	★★☆☆☆	The language is simple
Java	★★★★☆	★★★☆☆	15 years of C#, these are similar
Swing	★★★★☆	☆☆☆☆☆	never played with this
Language Design	★★★☆☆	★★☆☆☆	more for deconstructing Drools than creating a new language

Table 1: Expertise required.

6 Timeline

DONE

This project consists of two main parts. First is modelling the Drools structure, behaviour, constraints, editors and generators. The second will be creating non-standard projections of the structure.

Time will be allocated as 20 hours of my work time per week will be dedicated to design and development of the software. Currently estimating 4-8 hours at the weekends to research and project writing. There is an added period of 4 weeks at the end to rewriting the thesis.

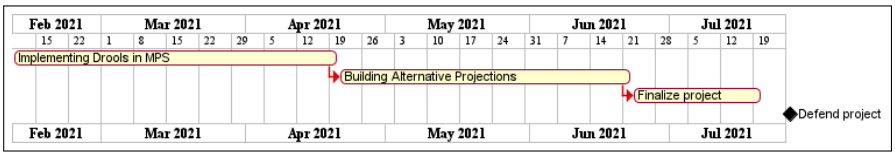

Figure 1: Predicted timeline.

7 Risks

DONE

Description	Risk Level	contingency
Project goals are too ambitious	★★★☆☆	Reduce Drools implementation to a useful subset and reduce number of projections.
MPS is not as flexible as needed	★★☆☆☆	we're screwed, but the papers from mbeddr indicate low risk.
I'm a terrible MPS developer	★★★★★	currently taking training and reading lots of books.
Academic overlap	★★☆☆☆	Panic or ignore.

Table 2: Project Risk.

8 Literature survey

To get an overview of the field we mainly looked at MPS and Drools based papers. For MPS we started with an expert recommendation and did some forward and backward snowballing. For Drools we were limited by a lack of access to papers behind paywalls, so we used a google scholar search and chose those that were open access. The Drools papers did not present any work on code visualization. The MPS papers and associated DSL papers covered some aspects of visual projectional editing, especially the papers relating to the produce mbeddr.

During our research we will keep a document database, using the Zotero Personal research assistant software [16]. Also we will create an annotated bibliography of the most relevant papers.

	Papers				
citations	Creating DSLs	How Drools works	Comparing Workbenches	How MPS works	Projectional Editing
[1]		⊕			
[2]			⊕	⊕	
[3]			⊕		
[5]					⊕
[7]		⊕			
[8]			⊕	⊕	
[9]		⊕			
[10]				⊕	
[11]				⊕	
[12]				⊕	
[13]		⊕			
[14]				⊕	
[15]	⊕			⊕	
[17]		⊕			
[19]				⊕	
[20]	⊕		⊕	⊕	
[21]	⊕				
[22]				⊕	⊕
[23]				⊕	
[24]					⊕
[25]				⊕	⊕
[26]	⊕		⊕	⊕	⊕
[27]				⊕	
[28]	⊕			⊕	

Table 3: Papers about the Drools, MPS and Language workbenches

References

[1] Erwin De Ley and Dirk Jacobs. "Rules-based analysis with JBoss Drools: adding intelligence to automation". In: *Proceedings of ICALEPCS 2011* (2011), pp. 790–793.

[2] Sebastian Erdweg et al. "Evaluating and comparing language workbenches: Existing results and benchmarks for the future". In: *Computer Languages, Systems & Structures* 44 (2015), pp. 24–47.

[3] Sebastian Erdweg et al. "The state of the art in language workbenches". In: *International Conference on Software Language Engineering*. Springer. 2013, pp. 197–217.

[4] Shirley Gregor. "The nature of theory in information systems". In: *MIS quarterly* (2006), pp. 611–642.

[5] Stian M Guttormsen, Andreas Prinz, and Terje Gjøsæter. "Consistent Projectional Text Editors." In: *MODELSWARD*. 2017, pp. 515–522.

[6] *Jetbrains MPS Product Page*. `https://www.jetbrains.com/mps/`. Accessed: 2021-01-19.

[7] Narendra Kumar, Dipti D Patil, and Vijay M Wadhai. "Rule based programming with Drools". In: *International Journal of Computer Science and Information Technologies* 2.3 (2011), pp. 1121–1126.

[8] Sofia Meacham, Vaclav Pech, and Detlef Nauck. "Classification Algorithms Framework (CAF) to Enable Intelligent Systems Using JetBrains MPS Domain-Specific Languages Environment". In: *IEEE Access* 8 (2020), pp. 14832–14840.

[9] Katanosh Morovat. "–Business Rules and DROOLS". In: ().

[10] Domenik Pavletic et al. "Extensible debuggers for extensible languages". In: *GI/ACM WS on Software Reengineering* (2013).

[11] Vaclav Pech, Alex Shatalin, and Markus Voelter. "JetBrains MPS as a tool for extending Java". In: *Proceedings of the 2013 International Conference on Principles and Practices of Programming on the Java Platform: Virtual Machines, Languages, and Tools*. 2013, pp. 165–168.

[12] Andreas Prinz. "Multi-level Language Descriptions." In: *MULTI MoDELS*. 2016, pp. 56–65.

[13] Mark Proctor. "Drools: a rule engine for complex event processing". In: *International Symposium on Applications of Graph Transformations with Industrial Relevance*. Springer. 2011, pp. 2–2.

[14] Daniel Ratiu, Vaclav Pech, and Kolja Dummann. "Experiences with teaching MPS in industry: towards bringing domain specific languages closer to practitioners". In: *2017 ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems (MODELS)*. IEEE. 2017, pp. 83–92.

[15] Daniel Ratiu, Markus Voelter, and Domenik Pavletic. “Automated testing of DSL implementations—experiences from building mbeddr”. In: *Software Quality Journal* 26.4 (2018), pp. 1483–1518.

[16] *Roy Rosenzweig Center For History And New Media. Zotero product page.* `https://www.zotero.org/`. Accessed: 2021-01-24.

[17] Kay-Uwe Schmidt, Roland Stühmer, and Ljiljana Stojanovic. “Blending complex event processing with the rete algorithm”. In: *Proceedings of iCEP2008: 1st International Workshop on Complex Event Processing for the Future Internet*. Vol. 412. Citeseer. 2008.

[18] Tijs van der Storm and Felienne Hermans. “Live literals”. In: *Workshop on Live Programming (LIVE)*. Vol. 2016. 2016.

[19] Markus Voelter. “Fusing modeling and programming into language-oriented programming”. In: *International Symposium on Leveraging Applications of Formal Methods*. Springer. 2018, pp. 309–339.

[20] Markus Voelter. *Generic tools, specific languages*. Citeseer, 2014.

[21] Markus Voelter et al. *DSL engineering: Designing, implementing and using domain-specific languages*. dslbook. org, 2013.

[22] Markus Voelter et al. “Lessons learned from developing mbeddr: a case study in language engineering with MPS”. In: *Software & Systems Modeling* 18.1 (2019), pp. 585–630.

[23] Markus Voelter et al. “Shadow models: incremental transformations for MPS”. In: *Proceedings of the 12th ACM SIGPLAN International Conference on Software Language Engineering*. 2019, pp. 61–65.

[24] Markus Voelter et al. “Towards user-friendly projectional editors”. In: *International Conference on Software Language Engineering*. Springer. 2014, pp. 41–61.

[25] Markus Voelter et al. “Using C language extensions for developing embedded software: a case study”. In: *Proceedings of the 2015 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications*. 2015, pp. 655–674.

[26] Markus Voelter et al. “Using language workbenches and domain-specific languages for safety-critical software development”. In: *Software & Systems Modeling* 18.4 (2019), pp. 2507–2530.

[27] Premysl Vysoky. “Grammar to JetBrains MPS Convertor”. In: (2016).

[28] Andreas Wortmann and Martin Beet. “Domain specific languages for efficient satellite control software development”. In: *ESASP* 736 (2016), p. 2.