HaDEsclipse – Integrated Environment for Rules (Tool Presentation)*

Krzysztof Kaczor[1], Grzegorz J. Nalepa[1], Krzysztof Kutt[1]

AGH University of Science and Technology,
Al. Mickiewicza 30, 30-059 Kraków, Poland
kk@agh.edu.pl, gjn@agh.edu.pl, kutt@agh.edu.pl

Abstract. In the paper a presentation of HaDEsclipse is given. It is an environment for design and implementation of rule-based systems within the Semantic Knowledge Engineering (SKE) approach. It is build with the use of the Eclipse framework, integrating the previously developed components of the HaDEs environment. HaDEsclipse integrates modules for conceptual prototyping of rule bases, and a visual editor for logical design of extended decision tables that group rules working in similar context. It also allows for generating an executable form of the system, that can be later executed by an inference engine. While the SKE is targeted mainly at knowledge engineers, the use of the Eclipse framework makes the development easier for software engineers.

1 Introduction and Motivation

Rule-based systems (RBS) play an important role in knowledge engineering and software engineering, e.g. with the business rules approach [1]. However, practical design of rules is a challenging task. It requires both efficient knowledge representation methods for rule bases, as well as practical design tools that support them. The Semantic Knowledge Engineering (SKE) [2] addresses these problems, by providing the XTT2 [3] and ARD+ [4] representation methods, and a dedicated design framework HaDEs, previously presented at KESE [5].

However, HaDEs turned out to be hard to use for knowledge engineers not familiar with SKE as well as for software engineers. This gave the motivation to develop a new front end to HaDEs, based on the popular Eclipse IDE. In this paper we present this new tool called HaDEsclipse [6]. First, we shortly discuss the SKE design process and how it is supported by HaDEs. Then we present the architecture and selected aspects of implementation of HaDEsclipse.

2 SKE Design Process with HaDEs

Our research concerns the representation and formal verification of RBS. An important result of our research is the SKE (*Semantic Knowledge Engineering*) [2]

* The paper is supported by the AGH UST Grant 15.11.120.361.

methodology, which derives from the HeKatE (*Hybrid Knowledge Engineering*) research project [7]. It aims at providing an integrated process for design, implementation, and analysis of the RBS supported by HADEs (HeKatE *Design Environment*) framework.

The main features of this methodology are:

1. *Visual rule representation.* The provided XTT2 [3] rule representation method that visualizes the rule base in the form of interconnected decision tables, which makes the design more transparent.
2. *Supported rule modeling.* The HADEs framework provides a set of dedicated tools, which facilitate the design process.
3. *Easy rule maintenance.* The HADEs-based design process consists of three stages. The transitions between stages are formally defined and automatically performed. The modification made in one stage can be automatically propagated into the following stages.
4. *One rule type.* As opposed to Business Rules, SKE provides only one type of rule – production rule. However, the methodology provides different inference strategies that correspond to different types of Business Rules, e.g. derivation rule type corresponds to backward chaining inference mode.
5. *Formal rule description and verification.* The provided formal rule language based on the ALSV(FD) (*Attributive Logic with Set of Values over Finite Domains*) logic [7] allows for formalized representation and verification of rules. Moreover, the semantics of rules is precisely defined.

The SKE approach can be applied to a wide range of intelligent systems. In this context, two main areas have been identified in the project: control systems, in the field of intelligent control, and Business Rules [1] and Business Intelligence systems, in the field of software engineering.

The HADEs framework aims at supporting the SKE approach. In this approach, the application logic is expressed using forward-chaining decision rules. They form an intelligent rule-based controller or simply a business logic core. The logic controller is decomposed into multiple modules represented by decision tables. HADEs supports a complete hierarchical design process for the creation of knowledge bases. The whole process consists of three stages: conceptual, logical and physical design and is supported by a number of tools providing the visual design and automated implementation[1].

The conceptual design is the first stage of the process. During this step, the ARD+ (*Attribute Relationships Diagrams*) method is used. The principal idea for this stage is to build a graph defining functional dependencies between attributes on which the rules are built. This stage is supported by two visual tools: VARDA (*Visual* ARD+ *Rapid Development Alloy*), and HQEd.

The logical design is the second stage of the process. During this stage, rules are designed using the visual XTT2 (*Extended Tabular Trees version 2*) [3] method. This phase can be performed as the first one in the design or as the

[1] See: `https://ai.ia.agh.edu.pl/wiki/hekate:hades`

second one, when the input is provided from the conceptual design. It is supported by the dedicated editor HQEd (HeKatE *Qt Editor*). HQEd supports the HML format, which allows for importing models generated by VARDA, as well as for saving and loading the state of the design.

Having a complete XTT2-based model the physical implementation can be generated automatically. In this stage, a logical model is transformed into an algebraic presentation syntax called HMR[2] (HeKatE *Meta Representation*). HMR is a textual representation of the XTT2 logic. It is a human readable form, as opposed to the machine readable HML format. The HMR representation can be directly executed by the dedicated inference engine tool, called HeaRT[3] (HeKatE *Run Time*) [8]. The HeaRT engine has communication and integration facilities. It supports Java integration based on callback mechanism and Prolog JPL library, called JHeroic.

HaDEs proved to be an efficient framework for designing rule bases within the SKE approach. However, its main limitation is that it is a set of loosely connected tools. Moreover, these tools have custom GUIs, which is problematic for engineers not familiar with SKE. This gave motivation for the development of a new platform, providing a more user friendly front end to HaDEs.

3 Architecture of HaDEsclipse

A decision was made to use the popular Eclipse IDE, which is a widely used tool in the software engineering community. Using it a new integrating front end to HaDEs was developed [6]. *HaDEsclipse* was implemented as a plugin for Eclipse. It integrates modules for conceptual prototyping of rule bases, and a visual editor for logical design of extended decision tables grouping rules. It also allows for generating an executable form of the system, that can be later executed by an inference engine. Within this plugin, one can manage the whole SKE design process described in the previous section.

The main functional requirements of HaDEsclipse are aimed at integrating the existing components of HaDEs using Eclipse:

1. ARD+ support:
   (a) Code editor with syntax highlighting, formatter, content assistant and error checking,
   (b) Integration with VARDA,
   (c) Wizard to create new ARD+ files.
2. HML support:
   (a) Code editor with syntax highlighting and checking, content assistant,
   (b) Integration with HQEd,
   (c) Wizard to create new HML files.
3. HMR support:

[2] See `https://ai.ia.agh.edu.pl/wiki/hekate:hmr`.
[3] See `https://ai.ia.agh.edu.pl/wiki/hekate:heart`

(a) Code editor with syntax highlighting, code formatter, content assistant and error checking,
(b) Integration with HeaRT.

4. Preferences card:
   (a) Code editors settings,
   (b) HaDEs environment parameters.
5. Intuitive Eclipse wizards, views and perspective.

The architecture of HaDEsclipse is presented on Fig. 1. It consists of 5 parts: three of them support HaDEs languages and the other two are responsible for view and wizards. Communication with HaDEs environment (VARDA, HQEd, HeaRT) is handled using JHeroic library.

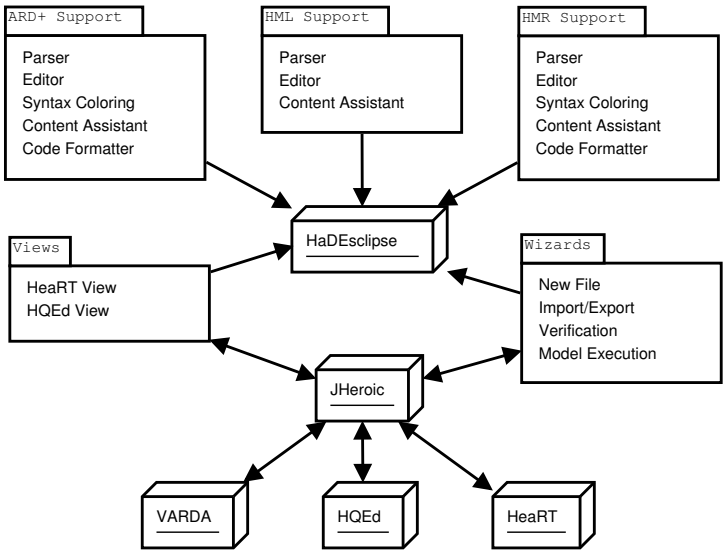

Fig. 1. Architecture of HaDEsclipse

The tool was implemented in Java as a plugin for Eclipse. All of the functional requirements where met. Five modules of HaDEsclipse successfully support the design with SKE. Thanks to HaDEsclipse the models created in the subsequent design phases are easily interchanged between the HaDEs tools. Moreover, the tool allows to run and verify rule models in HeaRT. For the visual design of the XTT2 tables HQEd is used, but files produced with it are exchanged with other tools transparently for the user.

An example session with the tool is presented in Figures 2 and 3. In the first figure the conceptual design with ARD+ is presented. The conceptual model of the rule base is described using a set of attributes and dependencies between them. The HML file contains prototypes of decision tables holding the conditional and decision attributes. In the second figure the HMR editing process is presented. The plugin supports both syntax highlighting and hinting, as well as

a structured XML editor in the case of ARD+. Schemas (headers) of the tables are defined base on the HML description. In given tables rules are defined using the `xrule` construct.

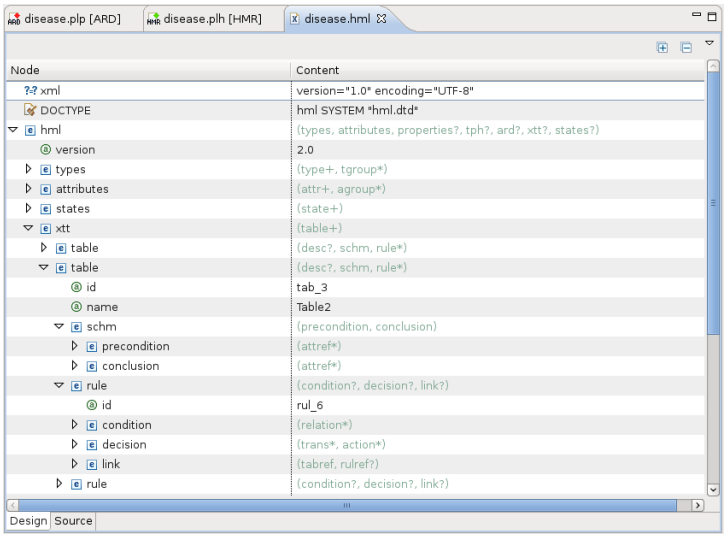

Fig. 2. HML Editor

4 Summary and Future Work

In the paper the HaDEsclipse tool was presented. It is an integrating front-end for the HADEs framework, which supports the knowledge engineering process in the SKE approach [2]. The new tool makes HADEs more accessible and useful for software engineers.

Our future works include further integration of HaDEsclipse with other tools we developed. This includes design tools for business processes, and integration with business process engines. Finally, recent results include an Eclipse-based tool for generating test cases for unit testing based on a rule-based specification. This framework will be integrated with HaDEsclipse, bringing more practical benefits from the area of knowledge engineering to software engineers [9].

References

1. von Halle, B.: Business Rules Applied: Building Better Systems Using the Business Rules Approach. Wiley (2001)
2. Nalepa, G.J.: Semantic Knowledge Engineering. A Rule-Based Approach. Wydawnictwa AGH, Kraków (2011)

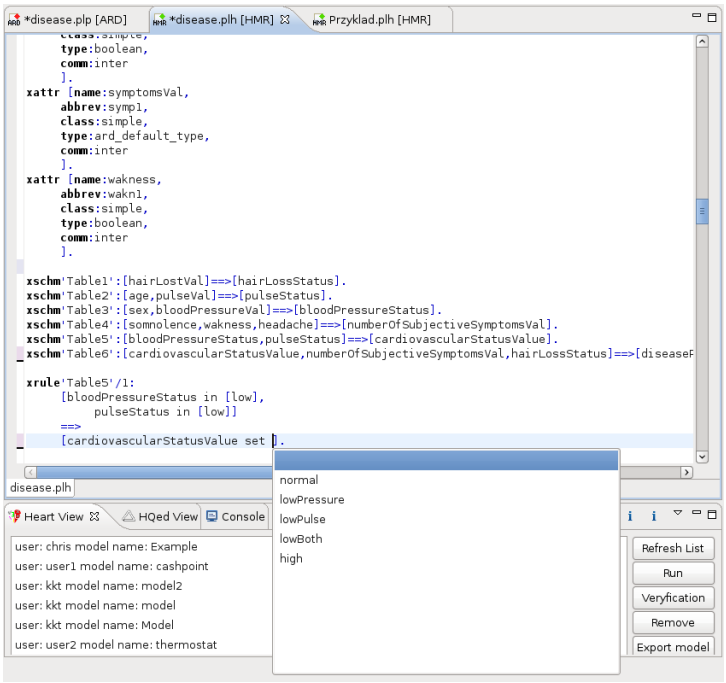

Fig. 3. HMR Editor

3. Nalepa, G.J., Ligęza, A., Kaczor, K.: Formalization and modeling of rules using the XTT2 method. International Journal on Artificial Intelligence Tools **20**(6) (2011) 1107–1125
4. Ligęza, A.: Logical Foundations for Rule-Based Systems. Springer-Verlag, Berlin, Heidelberg (2006)
5. Kaczor, K., Nalepa, G.J.: HaDEs – presentation of the HeKatE design environment. In Baumeister, J., Nalepa, G.J., eds.: 5th Workshop on Knowledge Engineering and Software Engineering (KESE2009) at the 32nd German conference on Artificial Intelligence: September 15, 2009, Paderborn, Germany, Paderborn, Germany (2009) 57–62
6. Bator, P.: Projekt i implementacja narzędzi do edycji wiedzy regułowej HeKatE na platformie Eclipse. Master's thesis, AGH University of Science and Technology (2012) supervisor: Grzegorz J. Nalepa.
7. Nalepa, G.J., Ligęza, A.: HeKatE methodology, hybrid engineering of intelligent systems. International Journal of Applied Mathematics and Computer Science **20**(1) (March 2010) 35–53
8. Nalepa, G.J.: Architecture of the HeaRT hybrid rule engine. In Rutkowski, L., [et al.], eds.: Artificial Intelligence and Soft Computing: 10th International Conference, ICAISC 2010: Zakopane, Poland, June 13–17, 2010, Pt. II. Volume 6114 of Lecture Notes in Artificial Intelligence., Springer (2010) 598–605
9. Grzegorz J. Nalepa, K.K.: Proposal of a rule-based testing framework for the automation of the unit testing process. In: Proceedings of the 17th IEEE International Conference on Emerging Technologies and Factory Automation ETFA 2012, Kraków, Poland, 28 September 2012. (2012)