A Meta-model for the Guideline Definition Language

Reyes Grangel[1][a], Cristina Campos Sancho[1][b], Begoña Martínez-Salvador[2][c] and Mar Marcos[2][d]
[1]*Departament de Llenguatges i Sistemes Informàtics, Universitat Jaume I, Castelló, Spain*
[2]*Departament d'Enginyeria i Ciència dels Computadors, Universitat Jaume I, Castelló, Spain*

Keywords: Computer-Interpretable Guidelines, openEHR, GDL, Meta-model, Case Study.

Abstract: Computer-Interpretable Guidelines (CIGs) are a key issue to implement decision support systems that could help clinical practice. To represent these guidelines numerous modeling languages, such as PROforma, Asbru, or GLIF, have been defined and they are currently used in different contexts with more or less acceptance. The Guideline Definition Language (GDL) is a rule-based modeling language for CIGs proposed recently by the openEHR Foundation. This language might have a wider acceptance because of the fact that it is defined as an open standard and, as such, open detailed specifications are provided. In this paper, we focus on the most recent specification of GDL, GDL2. Our objective is to gain insight and knowledge about this language and its specifications for software engineering application purposes, such as Model-Driven solutions. In this context, we present a proposal for a GDL meta-model as an attempt to formalise the GDL2 specification in a UML meta-model. Additionally, in order to validate this proposal, we present a sample GDL model, based on a clinical guideline for the diagnosis of heart failure, developed using the GDL meta-model implemented in Eclipse.

1 INTRODUCTION

According to the most recent definition (Graham et al., 2011), clinical practice guidelines are defined as 'statements that include recommendations intended to optimize patient care that are informed by a systematic review of evidence and an assessment of the benefits and harms of alternative care options'. The definition of clinical practice guidelines (henceforth, clinical guidelines) is a complex task that requires the participation of expert groups and many evaluation and verification activities. However, clinical guidelines have become a powerful resource for practitioners during decision-making processes in daily care practice.

Aiming to obtain guidelines that were computer-interpretable, and therefore provide the basis for clinical decision support systems, several representation languages and notations have been proposed in the field of Artificial Intelligence and Medical Informatics. The representation of the clinical guidelines using these modeling languages and notations are often called Computer-Interpretable Guidelines (CIGs).

However, one critical issue that arises in practice is that none of these numerous existing languages (de Clercq et al., 2004), (Peleg et al., 2003), Arden Syntax, PROforma, Asbru, EON, Prodigy, GLIF, and GUIDE, has become a de facto standard to be widely used by the scientific and medical community.

In this context, the openEHR Foundation (openEHR Foundation, 2021) proposes a set of open standards to represent this kind of guidelines aiming mainly to provide an open framework to manage Electronic Health Records (EHR) in an interoperable way. Two of the included specifications are the Guideline Definition Language (GDL), that is a rule-based language to represent clinical guidelines, and the Task Planning (TP) language jointly with its graphical version the Task Planning Visual Modelling Language (TP-VML) to support clinical task planning processes.

Even though openEHR Foundation provides a detailed specification of GDL2, this specification cannot be considered a formal meta-model to be used for software engineering purposes. The research work presented in this paper includes a first proposal of a meta-model for GDL, that is completed and validated by means of a case study developed for the clinical

[a] https://orcid.org/0000-0002-4049-3888
[b] https://orcid.org/0000-0003-1140-1603
[c] https://orcid.org/0000-0001-5959-9415
[d] https://orcid.org/0000-0001-9672-4190

Grangel, R., Campos Sancho, C., Martínez-Salvador, B. and Marcos, M.
A Meta-model for the Guideline Definition Language.
DOI: 10.5220/0010543400590068
In *Proceedings of the 16th International Conference on Software Technologies (ICSOFT 2021)*, pages 59-68
ISBN: 978-989-758-523-4

guideline dealing with the diagnosis and management of heart failure (The Task Force for the diagnosis and treatment of acute and chronic heart failure of the European Society of Cardiology (ESC), 2016).

The paper is organised as follows. Section 2 outlines the standards related to healthcare used in this paper, and the background in the context of meta-modeling. Section 3 shows the methodology followed in order to obtain the GDL meta-model and the meta-model proposal itself. Finally, the case study is described in Section 4, and Section 5 presents the conclusions together with some prospects for future work.

2 HEALTHCARE AND SOFTWARE ENGINEERING BACKGROUND

This section analyses the main concepts used in the research work linking healthcare and software engineering context.

2.1 openEHR Standards

The openEHR Foundation is one of the most noteworthy international standards development organisations inside the healthcare community. The mission of this international non-profit organisation is to enable semantic interoperability of health information, within and between EHR systems (Health Information and Quality Authority (HIQA), 2013). One of openEHR Foundation main achievements is to have provided widely accepted open specifications in this field.

openEHR Specifications is one of the openEHR Foundation initiatives that provides technology for e-health, which includes open specifications, clinical models, and software for researchers and developers, so that they can create interoperable solutions for healthcare (openEHR Foundation, 2021). These specifications and solutions are based on the requirements captured over many years in projects such as the EU FP3 Good European Health Record (GEHR) project (1992-1995), with the aim of developing systems that would be interoperable.

Figure 1 shows the openEHR architecture including the abstract specifications, also known as the 'Platform Independent Model' (PIM). The specifications relevant to CIG modeling are the Archetypes shown as a Formalisms component, the openEHR Terminology included as a Content component, and 'Process & CDS' (CDS for Clinical Decision Support). This last component includes the two main specifications regarding the guideline modeling, the GDL Language, for representing the decision rules of guidelines and the Task Planning Language, for representing the clinical processes of guidelines.

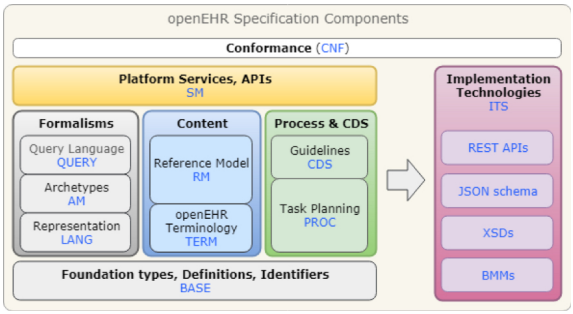

Figure 1: The openEHR Specification project (openEHR Foundation, 2021).

The knowledge contained in clinical guidelines is not only related to the decision-making processes that can be represented by rules, but also to patient data or to the tasks that need to be planned in the course of patient management. According to (Peleg et al., 2003), two main categories of knowledge could be identified in CPGs: structuring in plans of decisions and actions, and linking to patient data and medical concepts.

Modeling plans using GDL, i.e., modeling the workflow of clinical tasks to be performed, is a hard issue. According to (Marcos et al., 2019), it requires the use of ad hoc variables and a great number of rules in order to represent correctly the execution flow of the actions. The openEHR architecture includes within the Process and Clinical Decision Support module an additional specification for Task Planning (TP), as well as its graphical version, Task Planning Visual Modelling Language (TP-VML), which are useful to support flow control execution.

2.2 GDL

GDL is one of the specifications provided by openEHR community to represent part of the knowledge, specifically the decision logic, included in clinical guidelines (openEHR Foundation, 2019). The main advantage of this language over other formalisms developed in this context, such as PROforma (Sutton and Fox, 2003) or Asbru (Seyfang et al., 2002), is that it is based on an open standard. Moreover, it incorporates openEHR archetypes (openEHR Foundation, 2007) for the description of guideline data.

GDL is a formal language whose objective is to express decision support logic as a set of production rules. GDL rules contain 'when-then' statements,

which can be combined together as building blocks to support single decision making as well as more complex, chained, decision making processes. In this sense, GDL rules are suitable to be used in the task of modeling CIGs that can be part of decision support systems for healthcare (openEHR Foundation, 2021).

This language uses archetypes to model the clinical data collected and required in clinical guidelines. There exist some works that show how to use GDL jointly with archetypes in order to represent clinical guidelines in a rule-based manner (Anani et al., 2014; Lin et al., 2016).

GDL2 is the evolution of GDL version 1[1]. A distinctive characteristic of GDL2 with respect to GDL version 1 is that it provides a specification structured in packages. Although this structure cannot be used as a meta-model, it is a good starting point to develop it. The packages defined in the GDL2 specifications (openEHR Foundation, 2021) are the following:

- Guideline Package, which includes the classes needed to identify the archetypes used in the guideline and the defined rules.
- Expression Package, which incorporates the needed classes to make up the expressions (assertions and assignments) included in the rules, preconditions and default actions.
- Terminology Package, which incorporates the needed classes to define the terms and the term bindings.

GDL2 basically provides preconditions, default actions and rules to execute the logic of the guideline decisions, and openEHR archetypes used to interoperate with the EHR.

A GDL model includes a description, a definition and a terminology section. The description section contains the identification of the guideline: name, authors, version, keywords, etc.

The definition section includes the archetypes instantiation, the preconditions, the default actions and a list of rules to be executed. Besides, it can define different output formats that are identified in an output template. An archetype describes in detail the structure and content of clinical concepts. In a GDL model archetypes can be instantiated as input or output archetypes to deal with the input and output data respectively. Preconditions are defined assertions that have to be met before the rules of the guideline can be executed. Default actions define assignments to be always executed. A rule is defined by means of an assertion (when) and one or more assignments (then), i.e., (if-then) statements. Rules are executed following the priority provided by their order. Assertions and assignments are logical expressions composed by items and operators. Items can be in turn logical expressions, data elements instantiated from the archetypes or constant values. Finally, the terminology section defines how the terms used in the guideline are linked to user interface labels and to the term description in natural language. Moreover, it can include the local terms bound to external concepts by means of external references named *term_bindings*.

Figure 4 shows an example of the rules defined for a GDL model implemented using the GDL2 Editor Primer[2]. Figure 5 shows the rule `Step 1 HF or other diagnoses decision - suspected`. First the rule sets up the the conditions composed by logical expressions that assess values of the instantiated elements from archetypes, and then it includes the actions to be executed if the previous nested conditions are true.

2.3 Use of Meta-models in Software Engineering

The model and meta-model concepts are usually confused in the context of Software Engineering, often due to the numerous definitions that can be found in the literature. Therefore, and according to (Do et al., 2012), it is needed to specify the model and meta-model concepts for a better comprehension.

A model is an abstract representation of the real world or more specifically of a system. Therefore, it must necessarily be a simplification of that reality developed with a specific goal and purpose (Favre, 2004).

On the other hand, a meta-model is a model itself, but moreover, it is the explicit specification of an abstraction. It needs to set up a list of constructs, concepts and relationships among then, in order to develop the abstraction, that is to say the corresponding models (Bézivin and Gerbé, 2001).

Therefore, the meta-model represents the information of the model, which represents the real world that we want to abstract. In fact, the representation by means of the concepts and relationships included in the meta-model becomes then our understanding of the real world (Do et al., 2012).

According to (García-Magariño et al., 2010) meta-models need to formally specify modeling languages, providing a precise definition of the primitives, constraints and rules required for creating mod-

[1] In this paper, the terms 'GDL' and 'GDL2' are equally used to refer to the version 2 of GDL, while 'GDL version 1' is used to identify the version 1 of GDL.

[2] https://gdl-lang.org/the-project/guides-tutorials/gdl2/gdl2-editor-primer/

els given a specific problem or with one particular goal.

In the software engineering domain, the meta-modeling approach can be used to define domain-specific modeling languages for one specific context (Cuadrado and Molina, 2009; Viana et al., 2013), but also to provide an improved specification of some modeling languages whose syntax and semantics are expressed in other kind of formalisms (Karagiannis and Kühn, 2002). The final aim of this enhanced specification would be to provide more insight and knowledge about the modelling language, and also to apply other approaches such as Model-Driven solutions (Bézivin and Gerbé, 2001).

There exist different meta-modeling approaches according to (Karagiannis and Kühn, 2002). In this research we follow the OMG proposal Meta Object Facility (OMG, 2021b), which provides four layers of abstraction for meta-modeling: the Meta-Metamodel layer (M3), with meta-modeling languages like MOF and Ecore; the Metamodel layer (M2), where the meta-models of languages are defined using meta-modeling languages from M3; the Model layer (M1), with the specific models defined from those meta-models in M2; and the User Objects layer (M0), with the concrete objects of a system as instances of models in M1. There are several available alternatives as meta-modelling languages, such as Ecore, MOF and Graph, Object, Property, Relationship, and Role (GOPRR) (García-Magariño et al., 2010).

2.4 Related Work

There are no examples of the use of meta-modelling in the context of openEHR specifications. However, there are numerous research works trying to apply the meta-modeling approach to the healthcare domain with different purposes and results.

An example of using meta-modeling to avoid the problem that supposes to use a HL7 proprietary model language that is not supported by all Model-Driven Engineering (MDE) tools is shown in (Martínez-García et al., 2015). The proposal uses HL7 in a MDE approach considering the MIF (Model Interchange Format) meta-model proposed by HL7 by making use of a plug-in developed in the EA (Enterprise Architect) tool.

We can find a lot of experiences in the MDE context applied to the healthcare domain in which the use of a meta-modelling approach is followed. In this sense, a formalization is presented in (Rabbi et al., 2014) to coordinate the integration of different meta-models proposed in order to capture the complexity of healthcare systems with purposes of software development. While this is a more theoretical work, it shows how meta-models are useful when trying to represent the complexity inherent to healthcare systems, and how then, they can be used as part of an MDE approach to generate software.

Other kinds of research works are focused on integrating different modeling proposals of specific areas of healthcare software systems. For instance, a Disaster Management Meta-model is proposed in (Othman et al., 2014) using different models to unify approaches, which allows knowledge sharing and efficient disaster activities management. Besides, this work provides a wide description of the meta-model development and validation process followed.

Unlike the previously described results, the meta-model developed in this work is based on the defined openEHR specifications and the proposal is validated using a case study.

3 GDL META-MODEL PROPOSAL

This section describes the steps followed in order to obtain the GDL meta-model proposal, and its description.

3.1 Methodology

The methodology used to obtain the meta-model is based on Action-Research approach (Baskerville and Myers, 2004). The steps followed are: domain analysis, interpretation, and meta-model validation.

Domain analysis has consisted in identifying the concepts and relationships that are needed to define a GDL meta-model. To identify these contents we have reviewed and analyzed both the GDL2 specification provided by openEHR (openEHR Foundation, 2021), and the last version of the open source modelling tool, the GDL2 Editor Primer developed by Cambio Healthcare Systems (Cambio Group, 2021). This meta-model is focused on the representation of rules and terms, and for the time being, the modeling of archetypes has not been considered.

The interpretation of previous results has led us to define the concepts that make up the meta-model. We have applied different iterations in this phase until we have obtained a meta-model that allows designing a GDL model equivalent to the model created with the GDL2 Editor Primer. The tasks performed in each iteration are:

- Creation of the GDL meta-model using the

Eclipse Papyrus tool[3].

- Generation of the editing modeling tool based on this meta-model using the Enterprise Modeling Framework (EMF) utilities[4].
- Development of a model for an excerpt of a case study using the generated tool, and assessment (expressiveness, completeness, traceability, etc.) of meta-model constructs.
- Comparison and analysis of the developed model with respect to the one developed using the GDL2 Editor Primer. Conclusions drawn from the result of each iteration have served to improve the meta-model versions.

The 2016 ESC Guidelines for the diagnosis and treatment of acute and chronic heart failure (The Task Force for the diagnosis and treatment of acute and chronic heart failure of the European Society of Cardiology (ESC), 2016) has been used both as a case study to review the different iterations of the meta-model and to validate the final GDL meta-model. In order to achieve these two goals we have modeled an excerpt of this guideline using the modeling tool generated from the designed Papyrus meta-model, and also using the GDL2 Editor Primer.

3.2 Meta-model Description

Technically, the definition of a meta-model with the aim of generating a modeling editor tool by means EMF has specific requirements apart from the normal classes and relationships that make up the meta-model. The meta-model must adopt certain rules so that the generated modelling tool works correctly to generate the corresponding model. First, it is needed to define as a directed composition the relationship between two classes to generate an element (target of composite) from the other (source of composite) into the model. Besides, the relationship between two classes of the meta-model must be a normal directed association in order to establish a relationship between two elements in the model.

In the GDL meta-model proposal (see Figure 2) the classes representing the three sections that compose a GDL model (description, definition and terminology) are clearly shown in the upper part of the Figure.

Regarding the description section the classes defined are:

- Description: this class represents the identifying information of the Guideline. Each Guideline has a single description section including relevant information about the owner, author and contributors, descriptive data, version and keywords. Keyword, Contributor and Reference are classes as well, since they can be multi-valued concepts.
- Keyword: it represents each concept that can be defined as a tag within a Guideline.
- Contributor: it represents each person who has participated in the Guideline development.
- Reference: it represents multiple links to information sources regarding the Guideline.

Regarding the definition section the main class is Definition, that represents the contents needed to design a GDL model. These contents are DataBinding that sets up the archetype list, a list of rules, a list of preconditions, and a list of default actions. Moreover, a definition section can define different output formats that are identified in an output template.

According to the DataBinding concepts the classes defined are:

- DataBinding: this class defines the set of specific archetypes (ArchetypeInstantiation) used in the model.
- ArchetypeInstantiation: it represents each single archetype in a GDL guideline.
- ElementInstantiation: it defines each single instantiated element from an instantiated archetype.

Regarding the rules, the classes included are:

- RuleList: this class represents the set of rules defined for the execution of the Guideline. A Guideline has a single RuleList.
- Rule: it represents each pair of conditions and actions defined for the execution of the guideline. Each rule has an identification and a priority that sets the execution order. A rule is composed by one or more expressions, as the condition or the when part, that must be met to fire the rule, and a set of one or more actions, as the then part of the rule that are executed if the when part is met.
- RuleCondition: it represents the Boolean expressions that must be assessed in a rule.
- RuleAction: it represents the Boolean expressions that assign a value or an expression of any type to a variable (ElementInstatiation).

The other classes related to the GDL definition section are:

- PreconditionList: it represents the set of logical conditions that must be met before the execution of the Guideline.

[3]https://www.eclipse.org/papyrus/

[4]https://www.eclipse.org/modeling/emf/

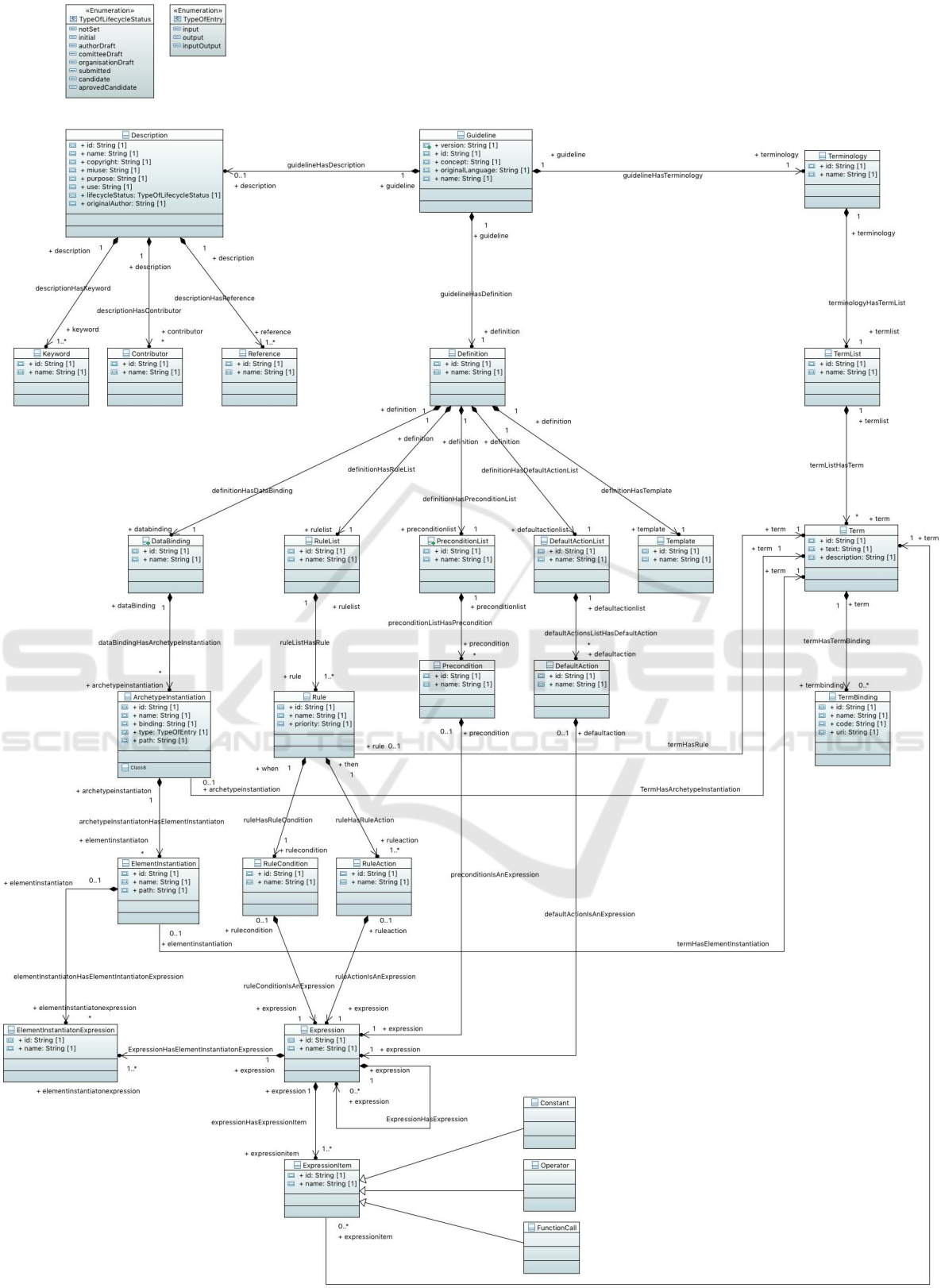

Figure 2: UML representation of the GDL meta-model proposal.

- Precondition: it represents the preconditions that apply to the guideline as a whole.
- DefaultActionsList: it represents actions defined to be executed without any previous condition.
- Template: this class represents each defined structured format for the output data reporting the result of the guideline execution.

Conditions, pre-condition and actions are defined using expressions represented by the following classes:

- Expression: it represents the logical statements to be assessed or assigned. Expressions are defined for the rules (conditions and actions), preconditions and default actions.
- ExpressionItem: it represents the elements used in the expressions. An expression has three kind of elements: constants, operations and variables.
- ElementInstantiationExpression: this class represents the use of an element (ElementInstantiation) in an expression.

Regarding the terminology section the classes are:

- Terminology: this class represents the section which includes the definition of the semantic terms (rules, data, etc.) used in the guideline.
- TermList: it represents the set of terms that are used in the guideline.
- Term: this class represents a concrete concept and its tag that can be used inside the guideline.
- TermBinding: it represents links to external terminology.

4 CASE STUDY FOR META-MODEL VALIDATION

The case study selected to validate the GDL metamodel is the 2016 ESC Guidelines for the diagnosis and treatment of acute and chronic heart failure (henceforth, ESC-HF guideline) (The Task Force for the diagnosis and treatment of acute and chronic heart failure of the European Society of Cardiology (ESC), 2016).

As mentioned before, we have considered a small part of this ESC-HF guideline focused on the algorithm for the diagnosis of the heart failure in the non-acute setting (see Figure 3). This excerpt of the ESC-HF guideline had been already modelled in GDL version 1 in a previous research work (Marcos et al., 2019). In this work we have used the new version of the GDL2 modeling tool to review and update this algorithm, which is presented in Figure 4 and 5.

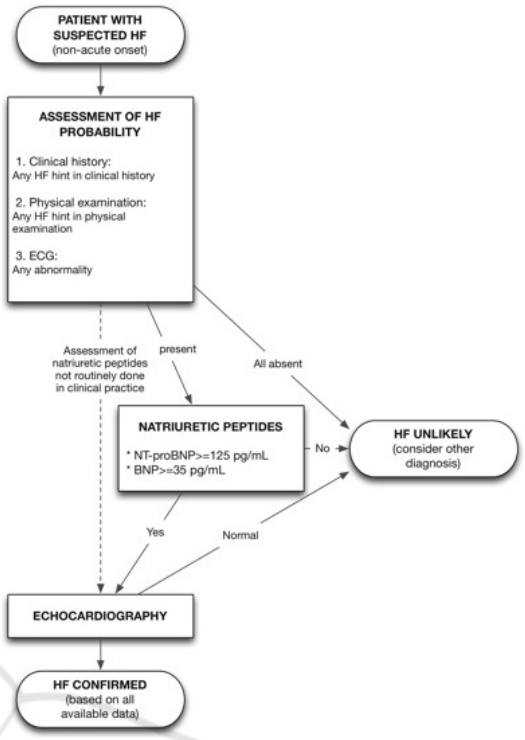

Figure 3: Diagnostic algorithm included in ESC-HF guideline (The Task Force for the diagnosis and treatment of acute and chronic heart failure of the European Society of Cardiology (ESC), 2016).

Figure 4: Rules of the GDL model for the case study.

The set of rules defined for this excerpt is shown in Figure 4. The rule order sets up the priority in the rule execution. From the list of rules, the rule selected in red, `Step 1 HF or other diagnoses decision - suspected`, is shown in detail with its

assertions and actions in Figure 5. This rule models the assessment of HF probability shown in Figure 3.

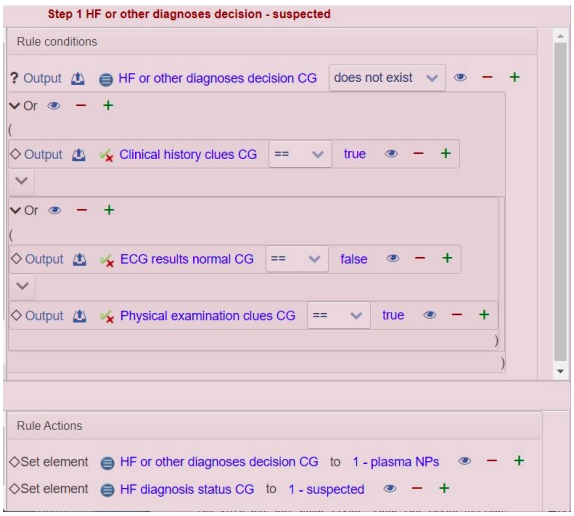

Figure 5: Detail of the rule `Step 1 HF or other diagnoses decision - suspected`.

As it can be seen in Figure 5, the rule has a compound assertion in the Rule Condition panel, and two rule actions in the Rule Actions panel. The assertion evaluates whether 'HF or other diagnoses decision CG' exists, i.e., has a defined value, and if one of the conditions in the 'or' expression ('Clinical history clues = true', or 'ECG results normal CG = false' or 'Physical examination clues = true') is true. Then, if the condition is accomplished the two actions that assign specific values to the elements 'HF or other diagnoses decision' and 'HF diagnosis status' are executed.

Once the GDL meta-model was defined and using the Eclipse EMF plug-in, an editing tool based on the meta-model was obtained. Figure 6 shows the same excerpt of the guideline shown in Figure 3 but this time modeled with the editing tool based on the GDL meta-model. The left-hand side of Figure 6 shows the overview tree of the implemented guideline. The definition part includes the data binding using archetypes instantiation and the set of rules defined for modeling the diagnostic algorithm. It can be observed at a glance that this set of rules is the same that the one shown if Figure 4 implemented with the GDL2 Editor Primer. The right-hand side of the Figure 6 shows with more detail the same rule displayed in Figure 5. On the top, in first place, the tree of the rule condition (when part) is shown. This condition comprises four expressions joined with logical or operators. Hereafter, the details of the two rule actions are shown (then part). Each rule action is an expression which is composed in turn of two expression items. Finally, at the bottom of the Figure 6, the properties window of the rule is shown. One of the elements of the properties is the term for labelling the rule.

We can conclude comparing both models, presented in Figures 4 and 5, and in Figure 6, that the modeling tool generated from the proposed GDL meta-model allows to properly define the ESC-HF guideline. Thus, the meta-model comprises and enables the definition of all the elements needed in a guideline according to the GDL specifications.

5 CONCLUSION

GDL is a rule-based language suitable to represent part of the knowledge included in clinical guidelines based on open specifications provided by openEHR Foundation.

In this paper, we present a proposal for a GDL meta-model that could be used jointly with these open specifications by the scientific and medical community. Furthermore, the meta-model is validated by means of a case study based on the diagnosis of the heart failure in the non-acute setting.

Taking into account that the new version of GDL, GDL2, has a specification but not a formal meta-model, the main contribution of this research work is to provide a meta-model useful in the software engineering context. For instance, this meta-model can be used to solve problems related to the healthcare domain, as for example when it comes to applying a MDE approach.

Moreover, the meta-model contributes to a deeper comprehension of GDL specifications and this fact could increase the applied use of this language by health care practitioners.

The limitations of the work are related to the validation process. More experiences with different and wider case studies would be carried out in order to improve and refine the current version of the GDL meta-model. Besides, the meta-model includes the needed classes to instantiate archetypes although their detailed modeling has not been considered because it is out of scope of this project. However, it is a question that will be addressed in future research work.

Finally, the future research work planned is to use this GDL meta-model in a MDE project to transform BPMN Models (OMG, 2021a) into GDL models, in a similar way that the work performed in (Martínez-Salvador et al., 2015), but using GDL instead of PROforma and following a Model-Driven approach.

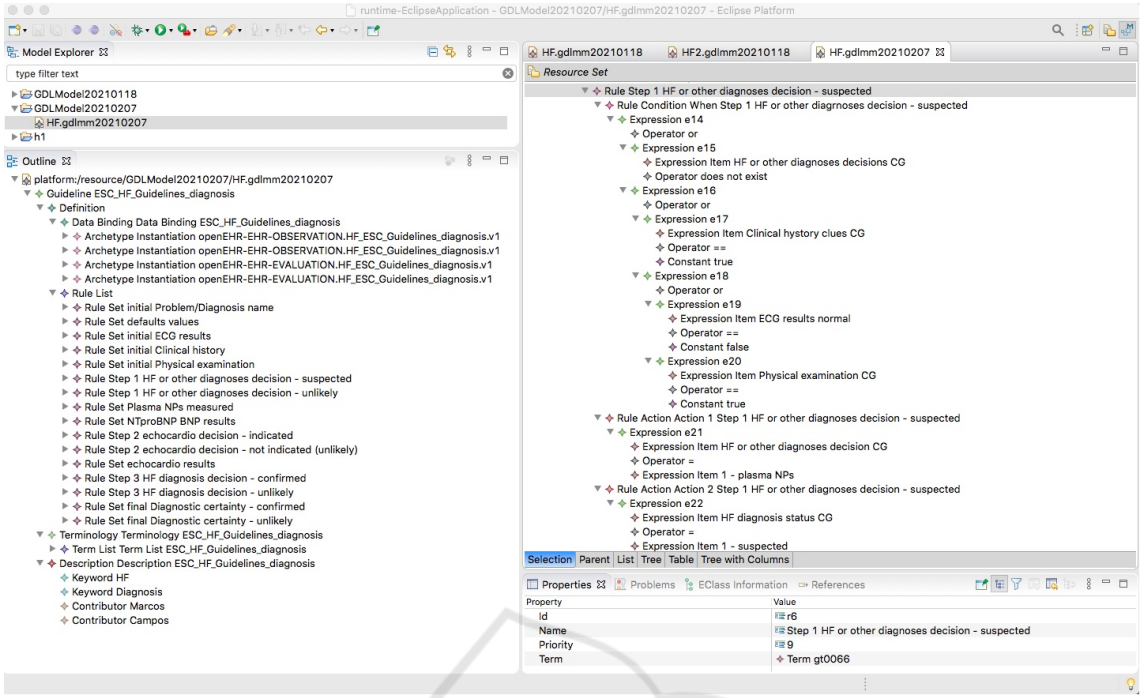

Figure 6: GDL model of the case study defined using the editing tool based on the GDL meta-model.

ACKNOWLEDGEMENTS

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness through the grant TIN2014-53749-C2-1-R.

REFERENCES

Anani, N., Chen, R., Moreira, T. P., and Koch, S. (2014). Retrospective checking of compliance with practice guidelines for acute stroke care: a novel experiment using openEHR's Guideline Definition Language. *BMC medical informatics and decision making*, 14(1):1 – 18.

Baskerville, R. and Myers, M. D. (2004). Special Issue on Action Research in Information Systems: Making IS Research Relevant to Practice: Foreword. *MIS Quarterly*, 28(3):329 – 335.

Bézivin, J. and Gerbé, O. (2001). Towards a precise definition of the OMG/MDA framework. In *Proceedings of 16th Annual International Conference on Automated Software Engineering, 2001 (ASE 2001)*, pages 273 – 280. IEEE.

Cambio Group (2021). GDL2 Editor Primer. In https://www.cambiogroup.com/. Accessed: 2021-02-26.

Cuadrado, J. S. and Molina, J. (2009). A Model-Based Approach to Families of Embedded Domain-Specific Languages. *Software Engineering, IEEE Transactions on*, 35:825 – 840.

de Clercq, P. A., Blom, J. A., Korsten, H. H. M., and Hasman, A. (2004). Approaches for creating computer-interpretable guidelines that facilitate decision support. *Artificial Intelligence in Medicine*, 31(1):1 – 27.

Do, Q., Cook, S. C., Campbell, P., Scott, W., Robinson, K., Power, W., and Tramoundanis, D. (2012). Requirements for a Metamodel to Facilitate Knowledge Sharing between Project Stakeholders. In Dagli, C. H., editor, *Proceedings of the Conference on Systems Engineering Research, CSER 2012, St. Louis, MO, USA, March 19-22, 2012*, volume 8 of *Procedia Computer Science*, pages 285 – 292. Elsevier.

Favre, J.-M. (2004). Towards a basic theory to model model driven engineering. *3rd Workshop in Software Model Engineering, WiSME*.

García-Magariño, I., Fuentes-Fernández, R., and Gómez-Sanz, J. J. (2010). A framework for the definition of metamodels for Computer-Aided Software Engineering tools. *Information and Software Technology*, 52(4):422 – 435.

Graham, R., Mancher, M., Wolman, D. M., Greenfiled, S., and Steinberg, E. (2011). Institute of Medicine: Clinical Practice Guidelines We Can Trust. *The National Academies Press*.

Health Information and Quality Authority (HIQA) (2013). Overview of Healthcare Interoperability Standards.

Karagiannis, D. and Kühn, H. (2002). Metamodelling Platforms. In Bauknecht, K., Tjoa, A. M., and Quirchmayr, G., editors, *E-Commerce and Web Technologies*, pages 182 – 196. Springer Berlin Heidelberg.

Lin, C.-H., Lo, Y.-C., Hung, P.-Y., and Liou, D.-M. (2016).

Building Chronic Kidney Disease Clinical Practice Guidelines Using the openEHR Guideline Definition Language. *Methods of information in medicine*.

Marcos, M., Campos, C., and Martínez-Salvador, B. (2019). A Practical Exercise on Re-engineering Clinical Guideline Models Using Different Representation Languages. In Marcos, M., Juarez, J. M., Lenz, R., Nalepa, G. J., Nowaczyk, S., Peleg, M., Stefanowski, J., and Stiglic, G., editors, *Artificial Intelligence in Medicine: Knowledge Representation and Transparent and Explainable Systems - AIME 2019 International Workshops, KR4HC/ProHealth and TEAAM, Poznan, Poland, June 26-29, 2019, Revised Selected Papers*, volume 11979 of *Lecture Notes in Computer Science*, pages 3 – 16. Springer.

Martínez-García, A., García-García, J. A., Escalona, M. J., and Parra-Calderón, C. L. (2015). Working with the HL7 metamodel in a Model Driven Engineering context. *Journal of Biomedical Informatics*, 57:415 – 424.

Martínez-Salvador, B., Marcos, M., and Riano, D. (2015). An Algorithm for Guideline Transformation: from BPMN to SDA. *The 5th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH 2015)*, 63:244 – 251.

OMG (2021a). Business Process Model and Notation. In http://www.bpmn.org/. Accessed: 2021-02-26.

OMG (2021b). The MetaObject Facility Specification. In https://www.omg.org/mof/. Accessed: 2021-02-26.

openEHR Foundation (2007). Archetype Definitions and Principles. In https://specifications.openehr.org/releases/1.0.2/architecture/am/archetype_principles.pdf. Accessed: 2021-02-26.

openEHR Foundation (2019). Guideline Definition Language v2 (GDL2). In https://specifications-test.openehr.org/releases/CDS/latest/GDL2.html. Accessed: 2021-02-26.

openEHR Foundation (2021). Open industry specifications, models and software for e-health (openEHR). In https://www.openehr.org/. Accessed: 2021-02-26.

Othman, S. H., Beydoun, G., and Sugumaran, V. (2014). Development and validation of a Disaster Management Metamodel (DMM). *Inf. Process. Manag.*, 50(2):235 – 271.

Peleg, M., Tu, S., Bury, J., Ciccarese, P., Fox, J., Greenes, R. A., Miksch, S., Quaglini, S., Seyfang, A., Shortliffe, E. H., Stefanelli, M., and et al. (2003). Comparing Computer-Interpretable Guideline Models: A Case-Study Approach. *Journal of the American Medical Informatics Association (JAMIA)*, 10:52 – 68.

Rabbi, F., Lamo, Y., and MacCaull, W. (2014). Coordination of Multiple Metamodels, with Application to Healthcare Systems. In *The 5th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN-2014)/ The 4th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH 2014)/ Affiliated Workshops*, volume 37 of *Procedia Computer Science*, pages 473 – 480. Elsevier.

Seyfang, A., Kosara, R., and Miksch, S. (2002). Asbru's Reference Manual, Asbru Version 7.3.

Sutton, D. R. and Fox, J. (2003). The Syntax and Semantics of the PROforma Guideline Modeling Language. *Journal of the American Medical Informatics Association (JAMIA)*, 10(5):433 – 443.

The Task Force for the diagnosis and treatment of acute and chronic heart failure of the European Society of Cardiology (ESC) (2016). 2016 ESC Guidelines for the diagnosis and treatment of acute and chronic heart failure.

Viana, M. C., Penteado, R. A. D., and do Prado, A. F. (2013). Domain-Specific Modeling Languages to improve framework instantiation. *Journal of Systems and Software*, 86(12):3123 – 3139.