The Health eDecisions Authoring Environment for Shareable Clinical Decision Support Artifacts

Davide Sottara[1], Peter J. Haug[2], Matthew Ebert[2], Edinardo Potrich[2], and Robert A. Greenes[1]

[1] Biomedical Informatics Department, Arizona State University, Scottsdale (AZ)
[2] Intermountain Healthcare, Salt Lake City (UT)

Abstract. In 2012, the U.S. Office of the National Coordinator for Health Information Technology established the Health eDecisions (HeD) Initiative. One of the goals of this initiative was the development of a standard model-based representation and XML exchange format for best-practice clinical knowledge in the form of decision support rules, clinical order sets and documentation templates. The standard would later become a candidate requirement for electronic health record systems as part of Meaningful Use Stage III. To facilitate the rapid diffusion of the standard, a project was started to build an authoring and editing tool for the HeD knowledge artifacts in the context of the SHARPc-2B grant. The tool, whose initial architecture and prototype is presented in this work, is model driven, based on semantic web technologies, compatible with a number of preexisting standards and ultimately designed to enable authoring not only by knowledge engineers but also by subject matter experts working at a non-technical level.

1 Introduction

A major challenge for health care organizations is the development of computable clinical decision support (CDS) rules from narrative recommendations and guidelines. The process involves the customization of the medical knowledge to smoothly integrate into the local clinical practices. Failure to do this appropriately will often impede a successful implementation. Due to the complexity of the problem, a four-stage knowledge refinement paradigm has been developed to describe the process [3]:

1. Initial markup and categorization of a recommendation, based on purpose, user, domain, and other components, using narrative text entries.
2. Formalization of the above using information models, coding systems, and value sets.
3. Contextualization of the rule based on "setting specific factors", such as how a rule would be triggered in a particular setting, in what clinical context, refinement of applicability criteria, incorporation of timing considerations, etc.

4. Conversion to an executable form for use in a particular environment, typically involving translation to a proprietary electronic health record (EHR) internal language, and mapping of the information model (the patient data and the rules clinical knowledge) to the EHR's internal representation.

The Health eDecisions[3] initiative has been established to facilitate the interchange of CDS knowledge between different institutions. In its first version, knowledge to be exchanged was intended to include decision rules, order sets, and documentation templates. The HeD community has then reached a consensus on a common, modular XML schema to encode such knowledge artifacts (KAs) in a shareable format that is also machine-understandable, effectively targeting the stage II of our original model. In HeD, an artifact typically consists of metadata about authorship, focus, provenance, version, etc.; a possible set of triggering events; a standards-based description of the data necessary for the evaluation; a logical condition expression to define the general applicability of the artifact; and set of actions to be suggested or performed when the preconditions are satisfied. Actions themselves can be further constrained using localized conditions and composed using a simple action algebra. Rules, order sets and documentation templates can be considered special cases of artifacts, where the sections are constrained in different ways. CDS rules generally require the specification of events, conditions and actions, whereas order sets and documentation templates may involve conditions as indications, but most of the knowledge is specified in the action part. In this context, it is evident that an HeD artifact only loosely corresponds to a rule (set) in the stricter sense of a language such as Reaction RuleML or RIF. However, a proper discussion of the semantics of HeD and its translation into a more formal rule language goes beyond the scope of this work.

Instead, this paper focuses on the authoring and editing tool for HeD knowledge artifacts (KAs) that was built as part of the SHARPc-2B grant at Arizona State University and Intermountain Healthcare. A key feature of this tool is that its intended users are not only knowledge engineers, but also subject matter experts (SMEs) who should work at a level that does not require deep technical knowledge of the HeD model and its XML schema.

The authoring process for health care-oriented KAs typically suffers from a disconnect between the ability of a human expert to comprehend the domain-specific logic and the level of detail required for mapping the knowledge to formal patient data/information model elements, coding schemes, value sets, etc. Thus, most knowledge authoring today is done by using custom or system-specific authoring/editing tools, usually provided by EHR vendors, and typically at a level that must be carried out by a knowledge or software engineer. Furthermore, there is little ability to organize the corpus of knowledge to review what it contains, query it by specific attributes (e.g. domain, setting, usage or mode of intended execution), to be able to manage the corpus of knowledge or to update it, or to identify gaps in knowledge requiring attention.

[3] http://wiki.siframework.org/Health+eDecisions+Homepage

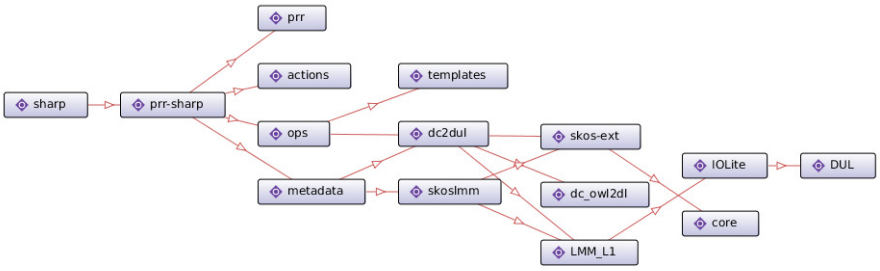
2 Approach

2.1 Editor models

The HeD schema has been balloted as an HL7 standard in 2013. Due to the iterations and revisions during the standardization process, the design and implementation of the editor had to be started way before a final version of the HeD specification was available. Moreover, the standard recommends the adoption of the HL7 virtual medical record (vMR) as a data model, in conjunction with common medical vocabularies such as SNOMED-CT, LOINC or RxNORM. However, other competing standards exist or are in process of being released. Despite the recommendation, HeD can potentially be used with any (clinical) data model and vocabulary, some of which may be specific to a practice or a domain such as genomics or pediatrics. Due to the initial uncertainty about the schema and the data models, and in order to facilitate the evolution of the tool, we have decided not to use HeD to develop the editor. Instead, we have designed an editor that is completely modular and model-driven, so that it could be adapted more easily as the language schema or the data models and vocabularies change. The core model is represented using Description Logic, a widely adopted formalism with descriptive and inferential capabilities. More specifically, we have chosen the Web Ontology Language v2 (OWL2-DL), a W3C standard designed for interoperability over the web. This choice facilitated the development of both the models and the software, and its grounding in the context of existing "upper ontologies". Some of these ontologies, or the metamodels from which they have been derived, inspired the creation of the HeD XML schema in the first place. While a number of alternatives exist (e.g. [5], [10]), we selected those which would require a minimal amount of conceptualization and transformation when interpreting or generating an HeD compliant serialization. A second advantage of reintroducing the full models is that the XML specification was mostly focused on the ability to deliver the content and could not capture (nor was aiming to represent) the complete semantics present in the original ontologies. The editor, instead, tries to leverage both the content and the context. In particular, the foundations of our work are as follows. SKOS[6], which was used to conceptualize clinical and medical terms and vocabularies; the Dublin Core[8] (DC), which was the basis for the HeD metadata; the Production Rule Representation[9] OMG standard (PRR), which provided the general structure of a KA; and a combination of Object Constraint Languages, inspiring the HeD expression language. We have also incorporated the LMM[7] ontology pattern, to capture concepts and the ability to reference and mention them, as well as the DULCE/IO-Lite ontologies, which allowed to contextualize our required concepts. We have extended and harmonized these ontologies to include the specific concepts needed to model HeD artifacts and their content. Notice that while some ontologies have been created manually, others have been generated dynamically. For example, the rule authoring process requires a description of the domain-specific information model used to deliver the data at runtime (HL7 vMR, in our case). This model is also described using an ontology, which

has been derived from the vMR schema. Likewise, the ontology module that covers the expression language is the result of a partially manual and partially automated generation process.

Fig. 1. The modular HeD ontology and its dependencies.



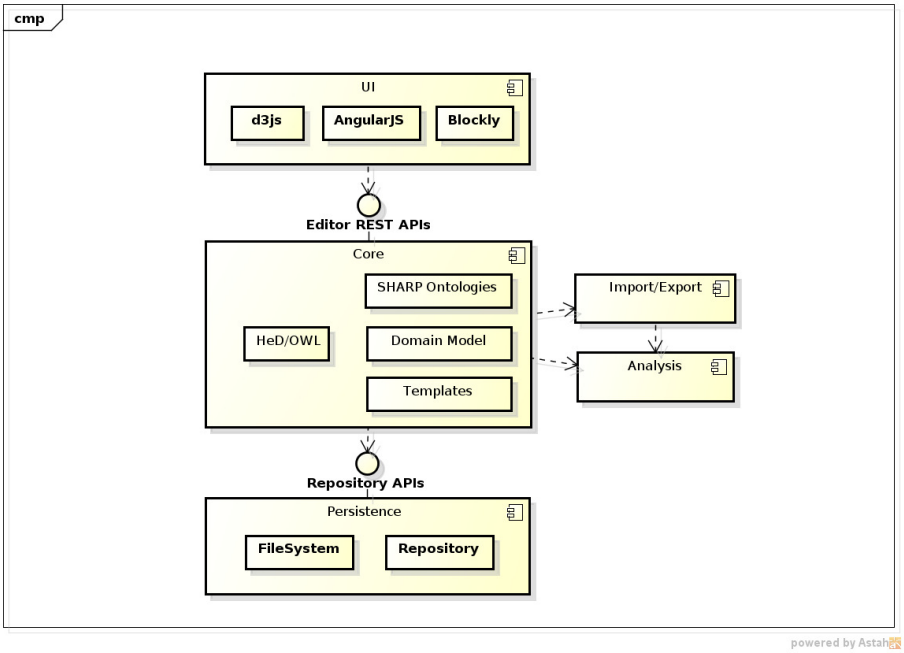
2.2 Editor architecture

The ontologies are the models driving the editor, which in turn is based on a simple 3-tier architecture. The persistence layer allows the storage and retrieval of a KA from a repository. The KAs are stored natively in RDF format rather than HeD/XML, to preserve the additional information in the semantic description. The editor core is responsible for loading the artifact being authored and the ontologies required to model it. The core will also analyze the artifact, generate the internal data structures required during the authoring process and apply the additions and transformations requested by the user through the user interface. Eventually, the core is responsible for exporting and serializing the internal model into redistributable and/or executable formats, of which HeD is the main example. The presentation layer is a pure web-based application, written in Javascript, which interacts with the core through a set of REST APIs. The interface is organized in several views that correspond to the different sections of an artifact. In each section, the artifact's current content is rendered and manipulated using Google Blockly [4], leveraging palettes of building blocks generated dynamically and based on the content of the editor's ontologies. The core is packaged as a Play application, which allows it to be deployed on the cloud, as well as in a web application container such as Tomcat. The user interface, instead, can be distributed as a web application (packaged in WAR format), deployed on a container and accessed using a normal web browser compatible with javascript. The editor has an optional dependency on a CTS-2 [2] compliant server, which is used to look up medical coded concepts and value sets.

Relying on the underlying formal model and its modular architecture, our basic approach to developing the editor was to create constructs (or "expression

[4] https://code.google.com/p/blockly/

Fig. 2. HeD Editor conceptual architecture.



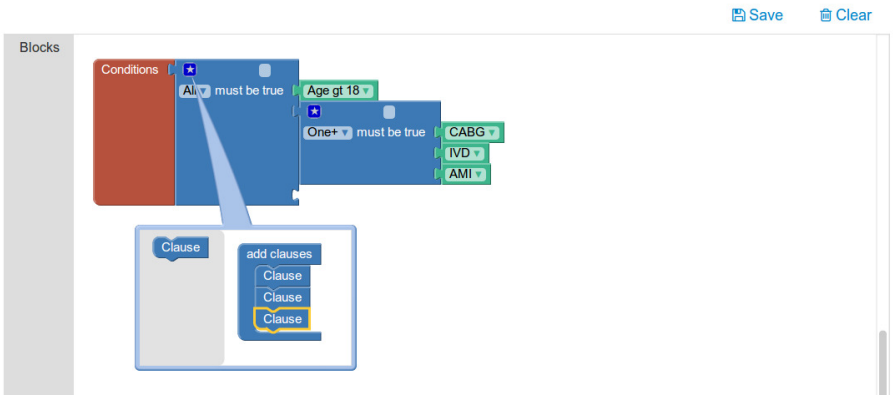
templates") that SMEs could use to define a KA at a somewhat high, conceptual level, and to associate with each construct the specific attributes needed to be specified in order to create a placeholder for it. A prior study [4] in reviewing rules at Partners Healthcare in 2004 showed that the many thousands of rules in use tended to be based on around 40, relatively simple templates. This suggests that creating templates for commonly used clause types would be both feasible and useful. The templates determine which data element properties are relevant, so selecting a template type means the author need only focus on specifying those properties needed for a specific clause type. Moreover, templates support default values and/or constraints on operations and values, which further simplifies the authoring and allows validation routines to be run. For example, if an artifact is to refer to the existence of a specific laboratory test result being available within a timeframe and above a threshold value, then only the name, timeframe and value need to be specified by a SME. These "templates" can be defined using a dedicated ontology and can be aggregated into libraries. In practice, they can also be pre-loaded from a spreadsheet compliant with a simple schema, derived from an official HeD template specification, which is used to instantiate the concepts in the template ontology. Once the parameter values have been specified by the user and validated, template instances become reusable, named expressions that can be used to define complex actions or conditions. There are cases, however, when artifacts require complex expressions that are not supported through

the available templates. For example, a number of clinical rules are based on "scorecard" models [1], e.g. to determine priorities or risk factors. Such expressions are very hard to capture using templates, but are important knowledge items to include in a shareable artifact. For such cases, the editor still allows to create free-form expressions. The HeD expression and OCL language has been conceptualized into an ontology, to add more semantics about the nature of the operations, and then translated into a set of Blockly components. The editor allows to compose the expression visually: as the author manipulates the blocks, an XML tree is built and delivered to the editor back-end to be parsed and integrated into the artifact. Likewise, any HeD expression can be parsed and transformed into a Blockly composition for display and further editing.

3 Example

As an example of the use of the HeD editor, we discuss the authoring of a CDS rule adapted from the NQF-0068[5] quality measure. This rule provides recommendations for antithrombotic therapy on discharge. In particular, patients 18 years and older with ischemic vascular disease (IVD) who were discharged alive for acute myocardial infarction (AMI), coronary artery bypass graft (CABG) or percutaneous coronary interventions (PCI) should be put on an aspirin or another antithrombotic drug regimen, unless contraindications are present. Moreover, an alert should be sent to notify the health care provider.

Fig. 3. Partially defined rule conditions.

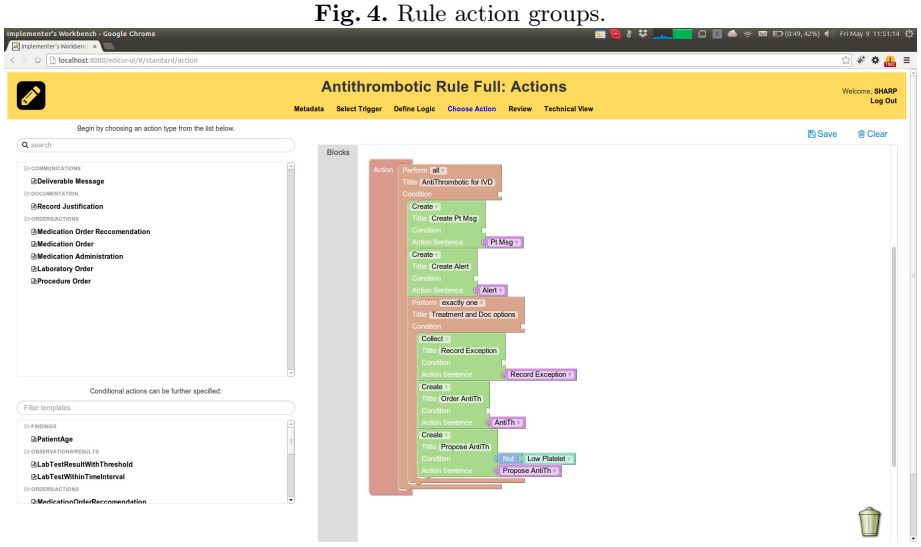


Clinical concepts of AMI, CABG, IVD, and antithrombotic medications are defined by specified value set groupings published by NCQA, and maintained by the NLM VSAC. Using the editor, we first modeled the trigger event as

[5] http://www.htsrec.com/janda/pdf/2012EP_MeasureSpecifications/NQF\%200068/NQF_HQMF_HumanReadable_0068.pdf

the patient being in pre-discharge status. Second, we defined the applicability criteria as shown in Fig. 3. In particular, we have defined a complex conditional expression combining a set of simple clauses using the traditional connectives AND ("all"), OR ("any") and NOT clauses. The terms in the logical expression are themselves sub-expressions, generated using three templates dealing with the patient's demographics, their problems and the procedures they have been subject to. Finally, we have modeled the recommendations, recognizing that there are two actions that should always be done, plus "exactly one" of three other actions.

Fig. 4. Rule action groups.



The details and the full outcome of the authoring process can also be seen in the video referenced in Section 5.

4 Conclusions and Future Works

This work has provided an opportunity to develop a tool that we believe, if properly positioned, can be foundational for future CDS knowledge representation, distribution, management, and incorporation into applications. Its current natural constituency is limited so far by lack of an appropriate stimulus or requirement for use, but assuming that limitation will be overcome, there is a broad agenda of potential application and use of this technology waiting to be carried out.

To improve its functionality and usability, several directions will be explored. First, more templates for commonly used constructs (i.e., trigger types, conditional expression clause types, and action types) should be added to the template

library, possibly enhancing them with domain-specific extensions, e.g., for pharmacogenomic CDS. The templates should also be grouped and indexed based on their underlying model to facilitate their retrieval. Second, the ability to include reusable definitions (e.g. as rules that make assertions rather than recommendations) might be considered as an extension of the HeD model. Third, it should be possible to process the internal, semantic model of an artifact to generate other serializations than HeD/XML, including standard and/or executable rule languages. Finally, the editor should be subject to a proper usability study to improve its user experience and test its impact in practice.

5 Resources

The editor is released under the Apache Software License v2 Open Source license. Its source code is hosted on github at the URL `https://github.com/SHARPC2B/HeD-editor`, together with its documentation. A recorded presentation and demo of the editor is available on YouTube at `http://www.youtube.com/watch?v=2WfWuqYX7NM`. A cloud-based, public instance of the editor will be released to the public in the near future.

6 Acknowledgments

The authors would like to thank other members of the SHARPC project 2B team who contributed at various earlier stages of this work and provided technical assistance during the latter stages. Earlier contributors included Mary Goldstein, MD, VA Medical Center, Palo Alto, CA; Samson Tu, MS, Stanford University; Emory Fry, MD, Cognitive Medical Systems; David Yauch, MS, Banner Health; Randy Kerber, MS, independent consultant.

References

1. Medical calculators, `http://www.mdcalc.com`
2. OMG CTS2 Statement Model and Services - FTF Beta 1. Object Management Group (Sep 2011), `http://www.omg.org/spec/CTS2/1.0/Beta1/20110907/11-09-07.pdf`
3. Greenes, R., Bloomrosen, M., Brown-Connolly, N.E., Curtis, C., Detmer, D.E., Enberg, R., Fridsma, D., Fry, E., Goldstein, M.K., Haug, P., Hulse, N., Hongsermeier, T., Maviglia, S., Robbins, C.W., Shah, H.: The morningside initiative: collaborative development of a knowledge repository to accelerate adoption of clinical decision support. Open Med Inform J 4, 278–290 (2010)
4. Greenes, R.A., Sordo, M., Zaccagnini, D., Meyer, M., Kuperman, G.J.: Design of a standards-based external rules engine for decision support in a variety of application contexts: report of a feasibility study at Partners HealthCare System. Stud Health Technol Inform 107(Pt 1), 611–615 (2004)

5. Kontopoulos, E., Zetta, T., Bassiliades, N.: Semantically-enhanced authoring of defeasible logic rule bases in the semantic web. In: Proceedings of the 2Nd International Conference on Web Intelligence, Mining and Semantics. pp. 56:1–56:4. WIMS '12, ACM, New York, NY, USA (2012), http://doi.acm.org/10.1145/2254129.2254199
6. Miles, A., Matthews, B., Wilson, M., Brickley, D.: Skos core: Simple knowledge organisation for the web. In: Proceedings of the 2005 International Conference on Dublin Core and Metadata Applications: Vocabularies in Practice. pp. 1:1–1:9. DCMI '05, Dublin Core Metadata Initiative (2005), http://dl.acm.org/citation.cfm?id=1383465.1383467
7. Picca, D., Gliozzo, A.M., Gangemi, A.: Lmm: an owl-dl metamodel to represent heterogeneous lexical knowledge. In: LREC. European Language Resources Association (2008)
8. Sutton, S.A., School, T.I., Mason, J., Limited, E.A.: The dublin core and metadata for educational resources. In: Paper presented at the International Conference on Dublin Core and Metadata Applications (2001)
9. Tabet, S., Wagner, G., Spreeuwenberg, S., Vincent, P.D., Jacques, G., de Sainte Marie, C., Pellant, J., Frank, J., Durand, J.: Omg production rule representation - context and current status. In: Rule Languages for Interoperability. W3C (2005)
10. Wagner, G., Damasio, C.V., Antoniou, G.: Towards a general web rule language. Int. J. Web Eng. Technol. 2(2/3), 181–206 (Dec 2005), http://dx.doi.org/10.1504/IJWET.2005.008483