dHealth 2022
G. Schreier et al. (Eds.)

doi:10.3233/SHTI220348

Modeling Medical Guidelines by Prova and SHACL Accessing FHIR/RDF. Use Case: The Medical ABCDE Approach

Gerhard KOBER[a,1], Livio ROBALDO[b] and Adrian PASCHKE[c]
[a] *Tiani "Spirit" GmbH, Vienna, Austria*
[b] *Legal Innovation Lab Wales, Swansea University, UK*
[c] *Fraunhofer FOKUS, Berlin, Germany*

Abstract. Decision-making based on so-called medical guidelines supported by semantic AI solutions is an essential and significant task for medical personnel in both a pre-clinical setting and an inner-clinical environment. Semantic representations of medical guidelines and Fast Healthcare Interoperability Resources (FHIR) using Semantic Web technologies, i.e., Resource Description Framework (RDF), rules (RuleML and Prova), and Shape Constraint Language (SHACL), provide a semantic knowledge base for the decision-making process and ease technical implementation and automation tasks. Current medical decision support systems lack Semantic Web integration using FHIR-RDF representations as a data source. In this paper, we implement a particular medical guideline using two different approaches: Prova [8] and SHACL [13]. We generate a series of raw FHIR-data for a selected guideline, the ABCDE approach, and compare the implemented two programs' (Prova and SHACL) results. Both approaches deliver the same results in terms of content. Both may be used within a distributed medical environment depending on the need of organizations.

Keywords. FHIR-RDF, Prova, SHACL, Rules, Medical guidelines

1. Introduction

The daily routine of medical personnel includes making decisions for optimal patient treatment. The basis for decisions is, firstly, clinical data and, secondly, the physician's knowledge about diseases and their treatment. Sensors like an ECG (electrocardiogram) generate clinical data. Furthermore, data about clinical observations like patients' movements and moisture of the skin and measurements like blood pressure is collected. These primary sources can be technically stored as Fast Healthcare Interoperability (FHIR) observations [4] in an FHIR-store.

Physicians gain much knowledge during their work, but they need to take track of every new achievement done in the specific field[17]. To ensure the best treatment for patients on every disease, expert groups provide state-of-the-art care, assemble best practices and therapies and formulate the recommendations as medical guidelines [5].

[1] Corresponding Author: Gerhard Kober, Tiani "Spirit" GmbH, Vienna, Austria, E-Mail: gerhard.kober@tiani-spirit.com

For example, if a patient shows signs of a particular disease, there is a general, evidence-based consensus on treating this patient to achieve the most acceptable result.

The research questions of this paper are: how can medical information stored in electronic health records, that is available as linked data, be used for clinical decision support; how can we use state of the art technology like SHACL and RuleML for expression and evaluation of medical guidelines; are there options to use semantic web standards (technologies), to address the need of medical decision support?

This paper addresses a well-known medical guideline called the "ABCDE-approach". This guideline is used for initial and continuing patient assessment in the inner-clinical and pre-clinical settings to find critically ill patients as quickly as possible.

When using this guideline, a physician, a nurse, or a paramedic can quickly determine if a patient is in a critical (life-threatening) situation and needs immediate support or if a patient is suffering from something less critical. This paper assumes that all observations made either by sensors from a medical device or by the medical personnel are represented as FHIR objects.

During the execution of the medical guideline, there is a need to find dependencies within the data of FHIR objects to take care of the lower and upper limits of the measured values. These dependencies are the basis for the decision of critical or not critical.

The FHIR standard provides a Resource Description Framework (RDF) representation [10] of FHIR-resources (e.g., Patient-resource, Observation-resource), as well as an ontology for the available resources[2].

By using RDF, we can model dependencies in patients' FHIR-observations and the included FHIR ontology. Afterward, the resulting RDF knowledge base might be queried via SPARQL [14], the standard query language for RDF.

However, SPARQL is not a full programming language but only a query language. In other words, SPARQL is unable to model sequences or flows of operations. Therefore, to properly implement medical guidelines, we need to embed SPARQL queries within another language that provides basic programming constructs. In this paper, we implement and compare rules embedding SPARQL queries in two logical programming languages: Prova [8] and SHACL [13]. The rules shown in this paper implement the ABCDE approach, which we chose as our use case.

The ABCDE approach is a technique for patient assessment to find critical illnesses or injuries and prioritize treatment [16]. ABCDE is an abbreviation for Airway, Breathing, Circulation, Disability, and Exposure[6]. Medical staff or paramedics use this acronym to remember the order of the checks. These checks are ordered by importance, and the purpose is to treat first what kills first [18].

Every step has a specific outcome - either a "check ok"-result or a "check not ok"-result. These checks contain more detailed instructions like particular measurements, treatments to apply, and limits for vital signs during processing.

Finally, the result impacts patient treatment and priorities within a treating facility (e.g., hospital, the emergency scene). For example, such a high-priority issue is a great wound, a blocked airway, or no pulse.

[2] See https://hl7.org/fhir/downloads.html

2. Methods

This work proposes and compares two different rule-based approaches to check a medical guideline for patient medical data. By "Prova-based approach", we intend the definition of the medical guideline workflow as a set of Prova rules. The other comparative approach defines the workflow rules in SHACL. Both approaches embed SPARQL queries to access the FHIR-RDF data. The workflow is implemented by defining, either in Prova or SHACL, an order among the rules and conditions to either allow or block some of them depending on previous queries' results (cf. [1], [11]).

In both implementation methods, the ABCDE approach has the option to discontinue after each evaluation step. The discontinuation is caused if a patient assessment is critical. The ABCDE approach also discontinues when approaching the end of the workflow, and a non-critical emerges. If one check out of ABCDE is successful, the workflow continues. In FHIR-resources, the measurements and observations are persisted. In the FHIR observation, the essential facts are the measurements stored as "FHIR values" (represented as FHIR-valueQuantity, FHIR-valueString, FHIR-valueBoolean) and an associated code. Codes are unique identifiers precisely describing examination or laboratory results. We are using LOINC-codes for the description of subjects. LOINC is an international standard to identify health measurements and observations from laboratories and clinics[3].

So, suppose a patient has various observations, and all are considered during clinical evaluation. In that case, the code helps to identify the correct observation within the entire FHIR-bundle [2], which can contain more information about a patient. For example, if it is needed to check the heart rate, the identifying LOINC-code is 8867-4. Observations with this code are delivering the appropriate entry. There are also textual descriptions of the LOINC code stored within the FHIR observation, but using text fields might lead to misinterpretation and wrong results. Therefore, we are using in our experiments the LOINC codes to avoid any ambiguity.

The following subsections describe the representation of FHIR in RDF, the Prova-based approach, the SHACL-based approach, and the generation of the FHIR-sample-data. The corresponding code for the implementation and the FHIR-sample-data generation is available on Github [4].

2.1. The representation of FHIR-data as RDF-Structure including the FHIR-ontology

Fast Healthcare Interoperability Resources (FHIR) are mainly represented in JSON or XML format as defined in the standard [3]. These JSON representations are mainly used for operational data exchange. Moreover, the FHIR-standard also defines FHIR resources represented as an RDF-graph. Such a graph is serialized in Turtle format [5]. FHIR as RDF is needed for semantic data validation, SPARQL queries, and further data integration.

Also, the available FHIR ontology supports reasoning tasks. However, the current open-source implementation of HAPI-FHIR[6] lacks the integration of the ontology, so this needs to be considered during data generation for the test data to have the semantic

[3] See https://loinc.org
[4] See https://github.com/gkober/Medical_ABCDE_Rules_SHACL_PROVA
[5] See https://www.w3.org/TR/turtle/
[6] See https://hapifhir.io/

relationship between the FHIR-objects (e.g., Patient, Observation) and the specific content of the resources.

2.2. The Prova-based approach

In the Prova-based approach, we formulate the ABCDE guideline as Horn-clauses. For the execution, we use Prova, an open-source rule language for reactive agents and event processing [8]. It combines logic programming aspects of Prolog style in conjunction with Java at runtime. By using Prova, we can represent a guideline as a logic program, as described in [7].

Additionally, Prova allows SPARQL queries to be injected into the rule conditions as built-ins to query the FHIR data represented in RDF format. Within these queries, SPARQL property paths are applied to find the relations from LOINC-codes to the corresponding values of the observations in an FHIR-Bundle.

SPARQL-ASK-queries return a true or false result. This helps collect true/false results within the entire workflow. Also, different types of values are considered: Boolean, strings, and integer. The SPARQL-queries can be very granular in terms of content. The filter of SPARQL allows searching for a particular LOINC-code and the target values (e.g., true/false, normal/not normal, or a numeric value within a specific range). An example of a particular check (in here the "A" for checking the Airway) is shown in Listing 1.

```
checkAirwayFree(Connection) :-
  QueryString = '<SPARQL-ASK-Query - TRUE Result>',
  sparql_ask(Connection, QueryString, QueryId),
  sparql_results(QueryId),
  checkBreathes(Connection),
  fail().
checkAirwayFree(Connection) :-
  QueryString = '<SPARQL-ASK-Query - FALSE Result>',
  sparql_ask(Connection, QueryString, QueryId),
  sparql_results(QueryId),
  sendMsg(XID,osgi,"FHIR",inform,{"ABCDECheckSPARQL"-
     >"Issue in CheckAirway"}),
  fail().
```

Listening 1. The airwayFreeCheck in Prova

In listening 1, Prova decides based on the result of the SPARQL-Query which check-Airway path to choose. If the result is positive, a subsequent check is called (in here, the checkBreathes). Otherwise, the message containing the problem is sent back to the execution engine and forwarded to the request's initiator.

Listening 1 also defines a "QueryString". It contains a SPARQL-ASK-query using SPARQL-Property-Path [15] on the FHIR-RDF-Graph. The result of this query is then either "true" or "false"; this result impacts the overall result of the rule.

The FHIR-RDF-graph represents an FHIR-Bundle containing the full information of a patient. An example, the SPARQL-ASK-query executed towards an FHIR-Bundle is in Listing 2.

```
PREFIX fhir:<http://hl7.org/fhir/> ASK WHERE {
 ?root fhir:Bundle.entry ?resource.
```

```
?resource
   fhir:Bundle.entry.resource/fhir:Observation.code/fhir:Code
   ableConcept.coding/fhir:Coding.code/fhir:value ?code.
?value
   ^fhir:value/^fhir:Observation.valueBoolean/^fhir:Bundle.en
   try.resource ?resource.
FILTER(?code="69046-1" && ?value= "true"^^xsd:boolean).
}
```

Listening 2. SPARQL on FHIR-Bundle

This query shows that the root node refers to a single resource, being the starting point, for the following route through the graph. We extract the path from *resource* to *code* and from the *value* to the *resource*, later applying a filter for a certain LOINC code and the expected value.

Additionally, there is an option to enrich the SPARQL query with a specific patient identifier, allowing searching for only on-site patient-related observations within an entire FHIR-store.

The Prova-based method expresses the guideline as Horn-clauses and concatenates by rule chaining, all values, and functions. This means that during the runtime of the Prova implementation, if one single condition does not hold, the rule resolves to false and further processing is stopped.

Furthermore, the SPARQL built-ins in the Prova-rules allow connecting and querying different SPARQL-endpoints during runtime. If the FHIR-observations are stored distributed in multiple locations, this feature can be used for finding additional results. For example, suppose a patients' smartwatch generates data in a private FHIR store, and data is generated and persisted within a hospital environment. There is the need for accessing two different endpoints during the execution of the Prova-rule.

2.3. The SHACL-based approach

SHACL is a recent W3C recommendation to validate and reason with RDF triplestores. In this work, we specifically use a special type of SHACL rules, called “SPARQLRules”, that allows embedding SPARQL queries in SHACL. SHACL rules can be assigned priorities to specify the order of their execution. On the other hand, special SPARQL operators can test whether some previous rules had triggered and then block the execution of subsequent rules based on that.

An example is shown in listening 3. This rule is executed first because it specifies “sh:order 0”, where “sh:order” is the SHACL property that defines the rules’ priority. All other rules in the knowledge base specify a higher numerical value for this property. The rule checks if the FHIR observation with LOINC code “69046-1”, i.e., the airway status[7], is not ok. In such a case, it adds to the knowledge base two triples specifying that for the patient identified by ?referenceValue, the “Airway is not free”.

```
sh:rule[ rdf:type sh:SPARQLRule; sh:order 0;
sh:prefixes[sh:declare
[sh:prefix"rdf";sh:namespace"http://..."^^xsd:anyURI],
...
[sh:prefix"fhir";sh:namespace"http://..."^^xsd:anyURI];
sh:construct """
```

[7] See https://loinc.org/69046-1

```
CONSTRUCT {$this rdfs:label ?referenceValue.
   $this rdfs:comment "Airway is not free". }
WHERE {$this fhir:Bundle.entry ?entry.
   ?entry fhir:Bundle.entry.resource ?obs.
   ?obs fhir:Observation.subject ?subject.
   ?subject fhir:Reference.reference ?reference.
   ?reference fhir:value ?referenceValue.
   ?obs fhir:Observation.code ?obsCode.
   ?obsCode fhir:CodeableConcept.coding ?coding.
   ?coding fhir:Coding.code ?code.
   ?code fhir:value ?codeValue.
   FILTER (?codeValue="69046-1").
   ?coding fhir:Coding.system ?system.
   ?system fhir:value ?systemValue.
   FILTER (?systemValue="http://loinc.org"^^xsd:anyURI).
   ?obs fhir:Observation.valueBoolean ?obsValueBoolean.
   ?obsValueBoolean fhir:value ?obsValue.
   FILTER (?obsValue!="true"^^xsd:boolean). }""";];
```

Listening 3. The airwayFreeCheck in SHACL

According to the ABCDE approach, all subsequent SHACL rules must not be executed. If the Airway is not free, there is no need for further checks. For this reason, the subsequent rules include the SPARQL clause:

```
NOT EXISTS$this rdfs:label ?referenceValue.
```

that blocks the execution of the rule in case the patient has been already labeled with his/her ?referenceValue by some previous rule, e.g., the one in listening 3.

2.4. Generation of the Sample-FHIR-data

For the upcoming execution of the two implementations of the ABCDE approach, we generated 1000 random FHIR-bundles containing all relevant information. During the data generation, we also included the interpretation of the expected result. The expected result is needed to compare the Prova-based and the SHACL-based approaches if they deliver correct results.

Since our rules are based on the LOINC standard and the ABCDE approach is formulated as text, we need to map the needed observations to LOINC codes. Table 1 shows the appropriate mapping and possible values, including the type in the resulting FHIR-resource.

The FHIR-data-generator provides all bundles by storing them locally for further processing. Therefore, the Prova-based and the SHACL-based implementations can directly access the observations.

Table 1. Code mapping

Check	LOINC-Code	FHIR-Type	Possible Values
airwayIsFree	69046-1	valueBoolean	Yes/no
Breathes	9278-3	valueBoolean	Yes/no
Cyanosis	39107-8	valueBoolean	Yes/no
respiratoryRate	9279-1	valueQuantity	e.g. 14
chestExpansion	67528-0	valueString	Normal/not Normal

respiratoryRhythm	9304-7	valueString	Normal/not Normal
symmetricChestExpansion	248562002	valueBoolean	Yes/no
oxygenSaturation	20564-1	valueQuantity	e.g. 96
pulseRate	8867-4	valueQuantity	e.g. 100
SystolicBloodPressure	8480-6	valueQuantity	e.g. 120
pulseCentral	8900-3	valueBoolean	Yes/no
pulsePeripheral	8911-0	valueBoolean	Yes/no
glucoseLevel	2339-0	valueQuantity	e.g. 105
Gcsnumber	35088-4	valueQuantity	e.g. 15
Pupilsisocore	80313-0	valueBoolean	Yes/no
strokeSigns	72089-6	valueBoolean	Yes/no
hasBleeding	81661-1	valueBoolean	Yes/no
hasBrokenBone	66571-1	valueBoolean	Yes/no
has Allergies	52571-7	valueBoolean	Yes/no
hasPain	52604-6	valueBoolean	Yes/no

3. Results and Discussion

The two presented approaches embed SPARQL-queries for the integration to FHIR-RDF-data. Using Prova, there is communication to an external RDF-store included. In the SHACL rules, the direct connection to the RDF data, containing relevant information for the ABCDE approach, was done.

The execution of the Prova-based approach takes more steps into account since it is capable of multiple sources that are considered for the result. Also, performing the setup in terms of publishing the FHIR-Resources to the RDF-store takes additional time.

When validating the implemented procedures, we had to crosscheck the correct LOINC-codes for the target values and the limits. Finally, when all settings are correct, the same results for the Sample-data set were generated.

In the Prova-based approach, we can integrate SPARQL-queries in the Prolog-style execution environment, which allows solving a medical workflow like the ABCDE approach with a specific result.

The errors (critical-results) that happen during execution can be enriched with particular error messages that describe the detailed issue - not only a “critical” result but a “critical” and, e.g., heart rate too low.

Prova also allows stopping processing on failure. This means if an error occurs in the beginning, the rest of the data and the checks are not needed, and therefore this feature can help reduce unnecessary queries.

Another benefit of the Prova-based implementation is that it can be translated to RuleML [12] and use RuleML to interchange and distribute the rules (so the guidelines). So, the same rules can be used in different environments without re-implementation.

The potential downside of submitting FHIR observations to an RDF-store is, for a prospective integrated solution, insignificant since the measurements are stored for different (e.g., medical, legal) reasons in a real-world environment.

SHACL is the state-of-the-art recommendation of W3C by now for validation and reasoning. We found that the reasoning is more performant (in terms of transaction duration). For the implementation, the SHACL-based approach does not need that deep integration in a workflow compared to the Prova-based approach. Therefore, depending on the particular use case, the SHACL based approach is the appropriate one to choose.

From a more general perspective, both methods have their justification, depending on the application. Prova if the expressive power of first-order logic (horn-logic subset) and rule inference/chaining is needed. SHACL helps with the integration of RDF data.

4. Conclusion

In this work, based on the simple "ABCDE-approach" medical workflow, we introduced the integration of FHIR/RDF-Data into two different approaches for the execution of a medical guideline. We formally represented and implemented the guideline in two approaches for this specific guideline. The implementation is available on Github.

We used Prova, where we created a "set of rules" enriching with SPARQL queries. Additionally, we deliver an implementation of SHACL rules to execute the guideline using FHIR-Observations and provide a "critical" or "not critical" decision to a medical person.

The evaluation shows that both implementations deliver the same results for a set of sample data. SPARQL queries allow a standard interface to triplestores that integrate to each solution and enable a highly flexible decision path. In the use-case, the behavior of the two implementations is the same since both stop execution once a "critical" result is generated.

In the future, it is foreseen to evaluate if both approaches are appropriate in a broader range of medical guidelines, limited to diagnosis tasks, where various outcomes are possible and might deliver notifications.

References

[1] Bartolini, C. and Giurgiu, A. and Lenzini, G. and Robaldo, L.: Towards legal compliance by correlating Standards and Laws with a semi-automated methodology, in proc. of the 28th Annual Benelux Conference on Artificial Intelligence (BNAIC2016). 2016.
[2] Bundle - FHIR v4.0.1., http://www.hl7.org/fhir/bundle.html, Accessed: 2022-01-30
[3] FHIR Overview - Developers., https://www.hl7.org/fhir/overview-dev.html, Accessed: 2022-01-30.
[4] Observation - FHIR v4.0.1., https://www.hl7.org/fhir/observation.html, Accessed: 2022-01-30.
[5] Natalia Iglesias, Jose M. Juarez, and Manuel Campos. Comprehensive analysis of rule formalisms to represent clinical guidelines: Selection criteria and case study on antibiotic clinical guidelines. Artificial Intelligence in Medicine, 103(October 2019):101741, 2020.
[6] Phil Jevon. Assessment of critically ill patients: the abcde approach. British Journal of Healthcare Assistants, 4(8):404–407, 2010.
[7] Gerhard Kober and Adrian Paschke. Representing and executing a medical guideline using prova. In Proceedings of the 15th International Rule Challenge, 7th Industry Track, and 5th Doctoral Consortium @ RuleML+RR 2021 (RuleML+RR-Companion 2021), 2021.
[8] A. Kozlenkov and A. Paschke. Prova rule language, https://github.com/prova/prova, 2014.
[9] Adrian Paschke. Provalets: Component-based mobile agents as microservices for rule-based data access, processing and analytics. Business & Information Systems Engineering, 58(5):329–340, 2016..
[10] Resource Description Framework (RDF), https://www.w3.org/RDF, 2014.
[11] Robaldo, L. and Bartolini, C. and Palmirani, M. and Rossi, A. and Martoni, M. and Lenzini, G.: Formalizing GDPR provisions in reified I/O logic: the DAPRECO knowledge base, The Journal of Logic, Language, and Information, Vol 29.
[12] RuleML., http://wiki.ruleml.org/index.php/RuleML_Home, Accessed: 2022-01-30.
[13] Shapes constraint language (SHACL), https://www.w3.org/TR/shacl, 2017.
[14] SPARQL Query Language for RDF, https://www.w3.org/TR/sparql11-overview, 2008.
[15] SPARQL 1.1 Property Paths., https://www.w3.org/TR/sparql11-property-paths/, Accessed: 2022-01-30.
[16] Troels Thim, Niels Henrik Vinther Krarup, Erik Lerkevang Grove, Claus Valter Rohde, and Bo Lofgren. Initial assessment and treatment with the Airway, Breathing, Circulation, Disability, Exposure (ABCDE) approach. International Journal of General Medicine, 5:117–121, 2012.
[17] van de Wiel, M. W. J., van den Bossche, P., Janssen, S., & Jossberger, H. (2011). Exploring deliberate practice in medicine: How do physicians learn in the workplace? Advances in Health Sciences Education 16 (1), 81–95. https://doi.org/10.1007/s10459-010-9246-3
[18] World Health Organization. The ABCDE and SAMPLE History Approach. World Health Organization, page 70, 2018.