Model-based Validation of HL7 CDA R2 Documents and Implementation Guides Using Gazelle ObjectsChecker and ART-DECOR®

Abderrazek Boufahja[1], **Kai U. Heitmann**[2], **Eric Poiseau**[1]

[1]IHE-Europe

[2]ART-DECOR® Expert Group, HL7 Germany

Abstract

Numerous tools define metamodels to describe the requirements included in CDA specifications, the most recent and standardized one is the HL7 Templates Standard. The Templates Design resulting of this standard allow distributing the CDA specification in a formal way. One application of this normalization is the validation of CDA documents. IHE-Europe/Gazelle team developed already a methodology named Gazelle ObjectsChecker in order to generate model-based validation of XML requirements, including HL7 CDA standard.

The aim of this paper is to describe the way the requirements from HL7 Templates Standard are imported on Gazelle ObjectsChecker, and the benefit of such methodology for CDA implementation guides and for CDA documents validation.

Keywords

ART-DECOR®; Gazelle ObjectsChecker; HL7 CDA R2; HL7 Templates Standard

Correspondence to:

Abderrazek Boufahja
Kereval
4, rue Hélène Boucher, Z.A. Bellevue, 35235 Thorigné Fouillard
E–mail: abderrazek.boufahja@ihe-europe.net

EJBI 2016; 12(1):en62-en69
received: April 19, 2016
accepted: May 1, 2016
published: May 20, 2016

1 Introduction

The specification of formal description for HL7 templates was a subject of discussion for several years in HL7 circles. HL7 Templates Standard DSTU R1 [1] was published in October 2014, and was the output of several years of work and discussions. ART-DECOR® consulting group supported this standard by participating in discussions and by development of a templates editor tool based on the HL7 Templates Standard. In recent years, IHE-Europe and Gazelle team developed a methodology to validate XML documents and especially HL7 CDA documents, named Gazelle ObjectsChecker. The aim of this methodology is to describe the requirements using formal language in order to generate validation tools.

The purpose of this paper is to explain how Gazelle ObjectsChecker succeeded to import requirements from Customer Templates Design into its models, and how this process was beneficial for all the intervenants, including the HL7 CDA implementation guides.

In the first section, we expose the state of the art: ART-DECOR®, HL7 Templates standard, and Gazelle ObjectsChecker. In the second part, we explain how Gazelle ObjectsChecker was coupled with ART-DECOR®, and how the coupling allows to validate HL7 CDA documents and to improve the quality of CDA implementation guides. And finally, we expose an application of this work: first, a comparison between the validation tools coming from the coupling of Gazelle ObjectsChecker and ART-DECOR®, with other validation tools, then we describe the result of validation of several HL7 Templates exchange documents coming from multiple domains.

2 State Of The Art

2.1 Introduction

CDA validation tools are generally based on a known process of validation [3] [4]:

- XML validation: check the validity of the CDA document regarding the XML standard [17]
- CDA schema validation [18]: check if the CDA document has a valid structure regarding the CDA schema
- Validation the CDA document regarding the Basic requirements of CDA Standard [3]
- Validation based on requirements coming from CDA implementation guides.

This paper is related to this last step of the CDA validation process. This last step could be performed by multiple validation tools like Gazelle ObjectsChecker, ART-DECOR® schematrons, MDHT, Trifolia schematrons, and MARC-HI Everest tool. Many technologies are used to modelise the requirements coming from CDA implementation guides, in order to use this modeling in the validation process.

2.2 ART-DECOR® and HL7 Templates Standard

ART-DECOR® is an open-source tool suite that supports the creation and maintenance of HL7 Templates, Value Sets as well as Data Sets and features cloud-based federated Building Block Repositories (BBR) for Templates and Value Sets. It supports comprehensive collaboration of team members within and between governance groups. The tool offers a Data Set and a Scenario editor, two Template editors, a Value Set editor and includes browsers for various international terminologies such as LOINC and Snomed CT. The tool covers all important phases of the creation artefacts for healthcare information exchange:

- capture of the clinical requirements in so called data sets and scenarios
- terminology mapping and associations
- template specifications with rules and value sets; it fully supports HL7's Templates Exchange Standard (see below)
- publication of all specifications, both online and offline available
- validation (schematron generation)
- support of the maintenance process (issue and ticketing system)

Since 2014 HL7 has an exchange standard (HL7 Templates Standard: Specification and Use of Reusable Information Constraint Templates, Release 1) that represents a big step forward in the process of use and re-use of templates. The standard's formal language enables governance groups busy with creation and maintenance of templates and associated artefacts to better express the constraints and vocabulary bindings. Template metadata captures the context in which this template has been created/updated and what relationships to other templates exist or are stated. This allows optimal support for template lifecycles and fosters the use of template registries and repositories (Building Block Repositories).

ART-DECOR® is used in over 30 projects throughout Europe and other parts of the world, e.g. the national infrastructure ELGA in Austria, the Dutch Nictiz (National Healthcare Standards Institute), the RIVM (National Institute of Public Health and the Environment in the Netherlands), HL7 and IHE Germany. The main ART-DECOR® servers host thousands of template definitions within the European Realm in HL7 Templates Standard format and an increasing number of re-used templates.

ART-DECOR® has a tight connection to IHE's Gazelle ObjectsChecker: the template definitions (in HL7 Templates Standard) along with the value sets captured in the tool and the generated schematron-based validation environment can be transferred and used in the IHE testing suite in a very straight-forward manner.

2.3 Other meta-models for CDA requirements description

Other meta-models exist to describe the CDA requirements, the best known are Trifolia XML description[9] and the description model of MDHT[7].

Trifolia Workbench model

Trifolia workbench is a web based application [9] that allows to edit CDA requirements and export the templates definitions in a proprietary XML format [5] [6], which can be interpreted as the metamodel description of the CDA requirements used by Trifolia.

MDHT

Model-Driven Health Tools (MDHT) [7] is a UML based tool allowing to formalize the representation of HL7 CDA requirements and implementation guides. The description of these requirements is based on the UML class diagram model, and a set of CDA profile stereotypes to provide information related to the CDA requirements and templates [8].

2.4 IHE Gazelle ObjectsChecker (GOC)

Introduction

Gazelle ObjectsChecker [2] [3] is a tool for the validation of XML clinical documents, and it is part of IHE-Europe Gazelle platform [10]. This methodology simplifies the treatment of XML requirements, like CDA specifications, and allows moving forward from using schema-

trons [19]. The tool provides a model-based validation using UML class diagram description [11] and OCL formal description of requirements (Object Constraint Language) [12]. The content of the UML models is filled based on the CDA implementation guides requirements.

Process of validation tools generation

Gazelle ObjectsChecker processes the UML models following this schema:

- Process the OCL constraints using an OCL processor
- Generate java code for validating CDA documents using the CDA model of requirements description
- Generate unit tests based on OCL constraints
- Generate documentation for each constraint

The generation of code is based on M2T technology, which allows the extraction of UML data and the creation of structured text files based on templates of generation [2].

All these steps are automatic steps, the only manual part is filling the UML models with OCL constraints, which is significant, because it takes time and is error-prone (we need a complete testing process to make sure there are no misunderstanding and bad interpretations of CDA requirements).

Advantages of Gazelle ObjectsChecker

The use of UML models allows benefiting from the strength of UML modeling tools, like searching for requirements, packaging of specifications constraints, friendly UML GUI editors, and constraints auto completion capability; such tools allow improving the maintainability of the validation tools based on Gazelle ObjectsChecker.

This tool has also large requirements coverage; it supports complex requirements like complex algorithms of validation, conditional and iterations checks, and allows datatypes [16] verification, which is a considerable advantage comparing to schematrons validation technology. Another advantage of this technology regarding the schematrons validation process is the time of processing [2]. This tool allows also runtime access to value sets repositories, and provides direct link to the original specifications, by coupling between constraints and requirements from specifications.

Usability

Gazelle ObjectsChecker is largely used by IHE-Europe for the validation of IHE CDA documents and many other XML based standards. Multiple CDA validators were developed based on this technology (over 40 validators between IHE, epSOS [13] and many national projects from Europe). These validation tools are heavily used during the connectathons [14] and European projectathons like epSOS [13] and EXPANDathon [15]. Some of the validation tools were also integrated in third party tools as a front-end validation. Developers' feedback regarding the process of managing models and generation of validators was positive [2], however the heaviest part in this process is the writing of OCL constraints into the UML models.

The aim of this paper is to describe the methodology used to extract automatically the CDA requirements from Templates Design. This methodology allows also improving the quality of the CDA implementation guides.

3 Coupling ART-DECOR® and Gazelle ObjectsChecker (GOC)

3.1 CDA documents validation based on ART-DECOR® Rules

Handwritten edition of OCL constraints in the UML models has always been the most complex task in Gazelle ObjectsChecker during the creation of CDA validators. This task is time consuming as it takes sometimes many days to manually formalize the requirements from CDA implementation guides into OCL constraints. Also, it can be a source of errors and false positive or false negative checks, due to human interpretation of the requirements. To deal with this problem, Gazelle ObjectsChecker generates an exhaustive list of unit tests for each handwritten constraint. Such testing is also time consuming.

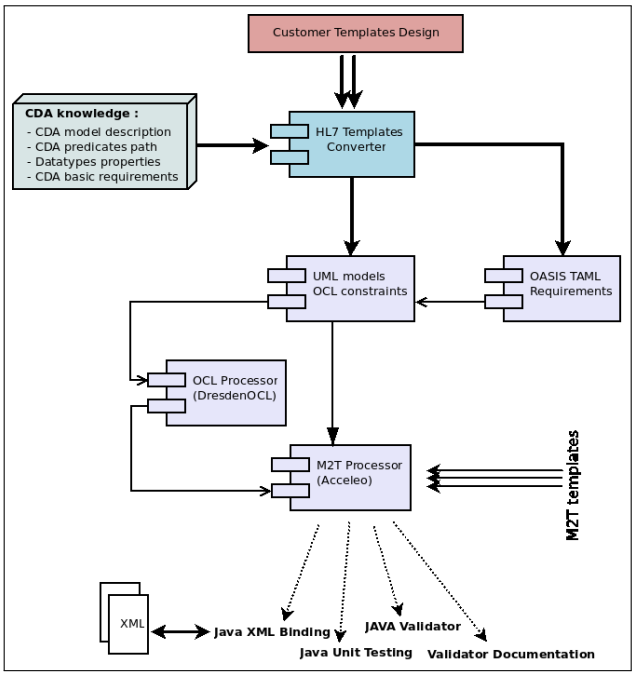

Figure 1: Principle of coupling ART-DECOR® and Gazelle ObjectsChecker.

The aim from coupling Gazelle ObjectsChecker with ART-DECOR® and HL7 Templates Standard is to deal with these problems by importing requirements from Cus-

tomer Templates Design based on HL7 Templates standard, and generating OCL constraints automatically.

The principle of coupling Gazelle ObjectsChecker and ART-DECOR® is described by figure 1; from the Customer Templates Design we generate TAML description of requirements [22], and we generate OCL constraints which are included into a UML class diagram using the stereotypes of Gazelle ObjectsChecker. The new module that allows such conversion is HL7 Templates Converter, which is developed by IHE-Europe/Gazelle team.

HL7 Templates Converter module allows to transform the requirements described in the Templates Design, including: checking of the cardinality of elements and attributes, fixed values and value sets attributed to coded elements, datatypes specialization, choices specification between CDA elements, templates inclusion, templates closing feature, and every kind of specification based on the structure of HL7 Templates description. The generation of OCL constraints based on the requirements in the HL7 Templates description is exposed in reference [26]: this document specifies when and which situations need a constraint to be generated by HL7 Templates Converter module. This document can also be used by other validation tools which take Templates Design as input for their validation tool, like the DECOR's schematrons generator.

This generation module takes advantages from knowing the model and the structure of CDA documents, the generation of OCL constraints is based on this knowledge:

- CDA model: knowing the CDA model allows to predict the elements used in the Templates Design, and inform the user if there is a misuse or an element which should not be used. Also, a good number of problems in the Templates Design are a bad specification of attributes values, especially when it comes from a CNE CDA vocabulary.

- CDA predicates paths: this knowledge allows better distinguishing between the CDA elements (like the templateId/@root, the observation/@code and the entryRelationship/@typeCode).

- Datatypes properties: such knowledge allows catching the extension between datatypes and allows preventing a misuse.

- CDA basic requirements: such requirements define how to use and to extend the CDA standard for implementation guides. These requirements are verified during the generation of OCL constraints in order to prevent a misuse of the CDA standard [21].

Using of Gazelle ObjectsChecker with ART-DECOR® simplifies the work of the HL7 Templates editors, as there is no need to specify an XPath distinguisher between the CDA elements described; this distinguisher is detected automatically based on the knowledge of CDA model.

Open and closed templates management

Open and closed templates can be defined as follows (see HL7 Templates Standard):

- **Open templates** permit anything to be done in the underlying standard that is not explicitly prohibited. This allows templates to be built up over time that extend and go beyond the original use cases for which they were originally designed.

- **Closed templates** only permit what has been defined in the template, and do not permit anything beyond that. There are good reasons to use closed templates, sometimes having to do with local policy. For example, in communicating information from a healthcare provider to an insurance company, some information may need to be omitted to ensure patient privacy laws are followed.

In most CDA-template libraries templates are defined as open.

Another typical situation is that templates in a repository for re-use are defined as open as when they are used within a document definition (document level template) a governance group may decide to use all templates as closed, i.e. no other content then specified is allowed. The same may temporarily apply during conformance testing, for example a connect-a-thon where it may be required to detect undefined content.

The following figure 2 shows a CDA template definition, CDA instance and the corresponding expected errors: Section B is defined as required (1..1) and therefore must be present in an instance. This gives errors in both open and closed template environments. If an "alien" section X is interspersed in the instance that is not in the definition this will be accepted with no errors in an open environment but will be rejected in with a closed template.

Sometimes one expects error message even with open templates that cannot be detected. A typical example is a typo in a template id in the instance. This will not be detected in open environments but only in closed ones.

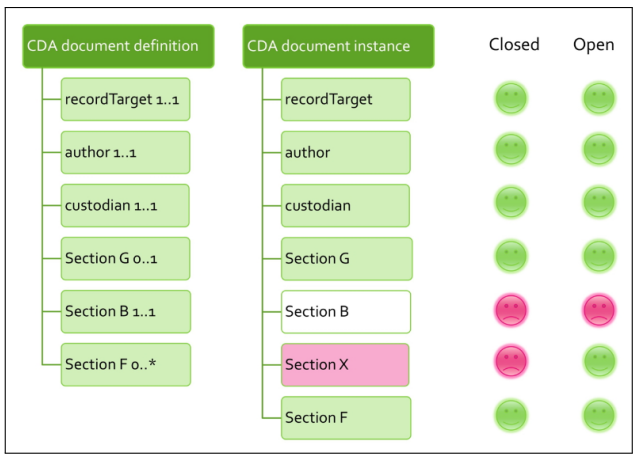

Figure 2: CDA template definition, CDA instance and the corresponding expected errors in open and closed environments.

ART-DECOR is the only known template tool that generates schematron for open and closed environments and Gazelle ObjectsChecker can validate open and closed environments.

3.2 Customer Templates Design validation

HL7 Templates Converter module allows verifying the information in the Customer Templates Design, before using them to create the Gazelle ObjectsChecker model with the OCL constraints. The requirements verified by this module are described in an independent document: HL7 CDA Requirements for HL7 Templates Standard [20], which contains the general CDA requirements which should be respected by CDA implementation guides. These requirements are divided into two kinds of requirements:

- CDA Model requirements: these requirements are directly related to the CDA schema, i.e. the XML structure.
- CDA Standard requirements: these requirements are related to the basic requirements which are described in the normative description, but not described in the CDA schema [21]

The HL7 Templates Standard allows the description of any kind of HL7 implementation guide, including non CDA guides. ART-DECOR® provides a schema for the exchange documents validation, which includes schematrons assertions in order to verify the conformity of the Templates Design, but this validation is not enough to verify if there are nonconformity between the written Templates Design and the CDA standard itself. The validation performed by HL7 Templates Converter module allows dealing with this lack. This module of validation can be seen as a validation of the specification itself, regarding the general rules of the CDA standard.

3.3 Benefits

Coupling advantages for Gazelle ObjectsChecker

This module allows eliminating the manual part of requirements formalization into OCL constraints, and allows going directly from Templates Design to the generation of Java validation code. This process was packaged into one executable that takes as input the URL to the Building Block Repository, and generates a ZIP file containing an executable, which takes as input a CDA documents, and generates as output an XML documents containing checks results. There is no more manual intervention for the creation of new validators using Gazelle ObjectsChecker, only a Template Design is sufficient to generate the validation tool. The gain on time in validation tools creation was huge; we go from some weeks to a few minutes.

Another advantage is the robustness of the tool. All the OCL constraints are generated automatically based on the requirements included in the Templates Design. The misunderstanding of specifications requirements is not possible any more. There is no need to heavily test the behavior of each OCL constraint, only acceptance tests are needed in order to verify that the HL7 templates were well written into the ART-DECOR® tool, and the exported Templates Design is in concordance with the original CDA specification.

Coupling advantages for ART-DECOR®

This coupling allows ART-DECOR® to move rigor at point of content profiles and implementation guides documentation and avoid discovery of issues/gaps at the time ObjectsChecker input is created. Also, it allows to reduce gaps and misunderstanding of CDA specifications, first because the use of the generated validation tool with acceptance tests will provide a feedback about the conformity of the HL7 Templates Design with the original specification, and second because the HL7 Templates Converter provides a report about the conformance of the Templates Design with the CDA standard, and such information is valuable for writers of implementation guides.

Coupling advantages for Implementation Guides Authorities

The import of requirements from Templates Design to Gazelle ObjectsChecker provides a way to validate the implementation guide itself. First, by having a validation tool we have a way to test CDA samples and to check if the output is conform to the implementation guide, that there are no conflicts with the specification. Second, Gazelle ObjectsChecker provides during the processing of the requirement a validation of the Templates Design regarding the CDA standard, which raises the reliability on implementation guides.

4 Applications and Illustration

4.1 Customer Templates Design validation

Using samples from diverse projects

Based on the CDA requirements in [20], and based on the HL7 Templates Converter module, a model of validation of Templates Design was created, and OCL constraints were included based on these requirements. Gazelle ObjectsChecker was used to validate the Customer Templates Design. Reference [20] can be interpreted as a restriction of the HL7 Templates Standard when used to describe CDA implementation guides.

From the online ART-DECOR® instance and Gazelle instance, we chose a number of projects, and we executed the validation of their Templates Design. Figure 3 de-

scribes the number of errors found during this manipulation.

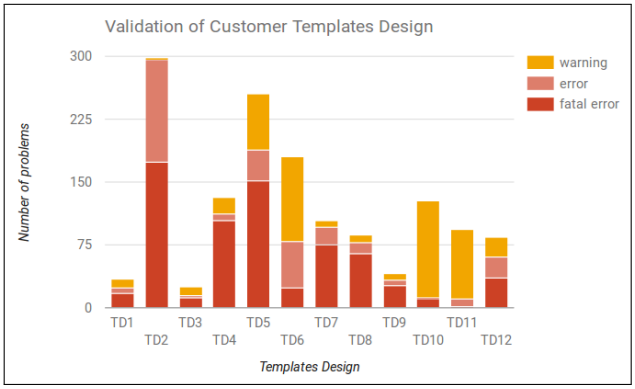

Figure 3: Validation of Customer Templates Design.

The analysis of figure 3 proves that there is a lack of validation regarding the CDA standard in the implementation guides. This validation is helpful to improve the quality of these specifications.

The percentage of the errors found is between 0.01% and 0.1% regarding the complete number of checks performed, which proves the strength of ART-DECOR in the detection of requirements errors in an early stage.

Errors found, lesson learned

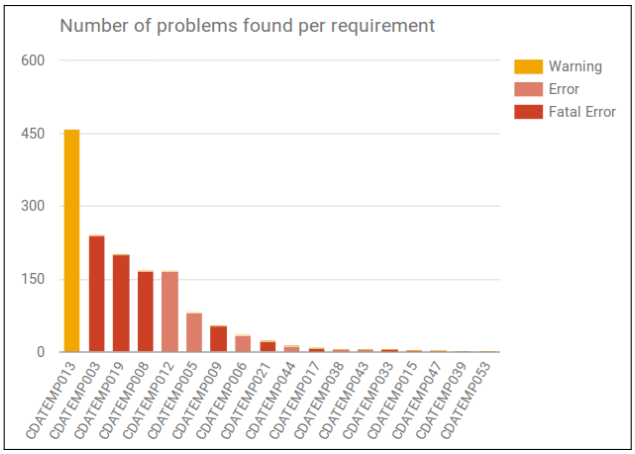

Figure 4: Number of problems found per requirement.

Figure 4 describes the list of most frequently found errors, based on the validation of a set of Customer Templates Design. The list of the complete requirements checked are in reference [20], where each requirement is identified by a unique identifier taking the form of CDATEMP-YYY. The most found errors are related to requirements:

- CDATEMP-013: a warning about the use of an attribute having default value. It is not mandatory to specify it in CDA documents as specified by the CDA standard; however a good number of templates force to provide attributes with a default value as a mandatory element. Example: AssignedAuthor/@classCode has the value 'ASSIGNED' by default, CDA templates does not need to make it mandatory.
- CDATEMP-003: fatal error: an element SHALL be from CDA model. This error occurs for example when the author specifies elements in a component that do not belong to it. Example: to specify ¡addr¿ element as a child element of ¡author¿.
- CDATEMP-019: fatal error: related to a misuse of a code or a valueSet. This error occurs when Templates Design allows having a value for an attribute; however this value is not permitted in CDA standard. Example: Participant/@type SHALL not be described by a valueSet containing codes out of the valueSet ParticipationType.
- CDATEMP-012: error: If isOptional attribute is specified with the value 'true', the original CDA attribute SHALL not be mandatory. This error occurs when the author is relaxing the CDA requirement, making an attribute optional when CDA requires it. Example: specify Act/@classCode as isOptional='true'; this is not permitted because this attribute is mandatory in CDA.

These requirements are not easily checked manually, the automation of these checks allows the consultants and implementation guides writer to move forward from fixing technical minor errors to the core of CDA elements specification.

4.2 Validation of CDA documents

IHE CDA pharmacy samples validation

In order to test the import module from ART-DECOR® to Gazelle ObjectsChecker, we created three BBR (Building Block Repositories):

- IHE-PRE: Templates related to IHE Prescription from pharmacy domain [23]
- IHE-DIS: Templates related to IHE Dispensation from pharmacy domain [24]
- IHE-PADV: Templates related to IHE Pharmaceutical Advice [25]

From these HL7 templates, we created schematrons validators using DECOR module, and model-based validator using Gazelle ObjectsChecker. The aim of this application is to compare the generated schematrons and Gazelle ObjectsChecker validators to the existing validation tools for IHE pharmacy: handwritten schematrons and handwritten model-based validators based on Gazelle ObjectsChecker. The indicators of comparison are:

- The number of checks tested

- The average of the number of errors found for a set of CDA samples

The most checks done are by the schematrons generated from DECOR tool, then the OCL constraints generated by Gazelle ObjectsChecker. The difference between the number of DECOR schematrons checks and Gazelle ObjectsChecker constraints is due to the fact that Gazelle ObjectsChecker filters out the requirements already tested by the schema, as it is redundant to test them again. This filtering is based on the knowledge of the CDA model. Such filtering allows improving validation time.

For each kind of validator, we selected a set of CDA documents coming from multiple vendors using Gazelle platform, and we executed the validation tool on it. Gazelle ObjectsChecker coupled with ART-DECOR® and the schematrons generated from ART-DECOR® generate nearly the same number of errors, far from handwritten schematrons. But, even if the number of the errors is not the same we suppose that the output of the validation is the same, and all the requirements are tested by all the tools. We are not proving that the validator with the most number of errors is the better one, and the others are missing some rules; but it is only a comparison between the granularity of the requirements. Validation tools generated from ART-DECOR® contains a greater granularity than the other tools; hand written requirements may combine some requirements in the same check, which explain why they have less number of errors found. Gazelle ObjectsChecker coupled with ART-DECOR® has in one hand the advantage over the generated schematrons regarding the filtering and the optimization of the requirements executed, and in the other hand Gazelle ObjectsChecker offers the possibility to validate the model coming from HL7 Templates Standard before generating the OCL constraints.

Comparison to schematron validation

Based on the study of the last paragraph, we noted that the negative constraints are better supported by handwritten rules, even if HL7 templates standard support it using XPath rules. The granularity is better in validation tools based on ART-DECOR® export. The use of ART-DECOR® as a modeling tool of constraints simplifies the creation and the deployment of validation tools, even consultant that does not know schematrons and OCL language can create and generate their own validation tools. Gazelle ObjectsChecker allows optimizing the checks based on the knowledge of the CDA standard, and allows providing a pre-validation and testing before the generation of validation tools; from the testing in the last paragraph, for a less number of checks Gazelle ObjectsChecker has the same granularity of errors as the schematrons generated by DECOR. However, the generated schematrons are easier to deploy, even if Gazelle ObjectsChecker provides standalone validation tools using jars of validation. Finally, another advantage of Gazelle ObjectsChecker is the strength of the tool in the validation of basic requirements of the CDA standard [2].

5 Conclusion

HL7 Templates Standard provides the possibility to exchange templates description for reusability and exchange purpose. The designing of thousands of HL7 templates using ART-DECOR proved the strength of HL7 Templates Standard as a templates exchange format. The HL7 templates available from the ART-DECOR® site are valuable resources for templates editors.

HL7 Templates standard allows also normalizing the input for validation tools by improving the interoperability between HL7 CDA templates editor tools and HL7 CDA validation tools. Combining Gazelle ObjectsChecker and ART-DECOR® improves the validation process, and also brings added value to templates definitions by identifying requirements issues. This coupling improves the quality of validation tools based on Gazelle ObjectsChecker, reducing the time of validation tools creation, adding robustness to the constraints because they are generated and not handwritten. Finally this process improves the quality of the CDA specifications by bringing further checks regarding the CDA requirements, which reduce the gaps and misunderstanding of CDA specifications.

We applied this process for the definition of IHE pharmacy profile templates edition using ART-DECOR®, for the generation of validation tools using Gazelle ObjectsChecker and ART-DECOR® schematrons and proved the strength of HL7 Templates Standard and its ability to describe all needed requirements. The validation output proved that the results of validation have a better level of granularity and specialization than handwritten validation tools, the number of checks is bigger than other tools, leading to better identification of the errors.

Many perspectives can follow this paper, a complete testing process for generators of validation tools can be developed, and a harmonization between outputs of templates editor tools can improve the validation process of existing validation tools consequently.

References

[1] HL7 Templates Standard: Specification and Use of Reusable Information Constraint Templates, Release 1

[2] Abderrazek Boufahja and Eric Poiseau, Model-based Validation of XML Requirements, Applied in Healthcare IT Standards, HEALTHINF 2014

[3] Abderrazek B., Eric P., Guillaume T., Anne-Gaelle B., Model-based Analysis of HL7 CDA R2 Conformance and Requirements Coverage, IHIC 2015

[4] Rene Spronk, Analysis of CDA R2 testing tools, Feb 13, 2015, ringholm, `http://www.ringholm.com/column/HL7_CDA_Conformance_testing_tools_analysis.htm`

[5] Trifolia Workbench, available from https://www.lantanagroup.com/resources/trifolia/

[6] Software Implementation of CDA, http://wiki.hl7.org/index.php?title=Software_Implementation_of_CDA

[7] Sondra Renly, Rita Altamore, L. N. A new model for collaboration: Building CDA documents in MDHT. In AMIA Annual Symposium Proceeding, November, 2012

[8] MDHT Modeling Style Guide for CDA, MDHT 1.1.0 May 2012, v2

[9] Andrea Ribick, HL7 Releases New Tool for Capturing, Storing and Managing CDA Templates, July 19, 2011, HL7 Press Releases

[10] Eric Poiseau, D3.1: Testing tools overview: Testing tools gap analysis with description of required new tools (ANTILOPE project), version 1.1, Feb 2015

[11] Hans-Erik Erikson, Magnus Penker, B. L. D. F. (2004). UML2 Toolkit. Wiley Publishing, Inc

[12] OMG Object Constraint Language specification (OCL). Object Management Group, v2.3.1 edition, January 2012

[13] Thorp, J. Europe's E-Health Initiatives, AHIMA 2010

[14] IHE-Europe, The IHE Connectathon: what it is and how it is done, November 2015

[15] Zoi Kolitsi, D3.1 EXPAND Vision, WP 3.1 eHealth interoperability assets, June 2014

[16] HL7, Data Types - Abstract Specification, Release 1, 2004

[17] Mitch Amiano, XML: Problem - Design - Solution, Wrox edition, August 2007

[18] R. Allen Wyken, XML Schema Essentials, Woley edition, April 2002

[19] Eric van der Vlist, Schematron, O'Reilly edition, March 2007

[20] Abderrazek Boufahja, HL7 CDA Requirements for HL7 Templates Standard, November 05, 2015, http://gazelle.ihe.net/files/cdatemplates_requirements_restriction.pdf

[21] Abderrazek Boufahja, HL7 CDA R2 Basic Requirements, gazelle team /IHE-Europe, V0.1, April 14, 2014. available from: http://gazelle.ihe.net/cda/cda-basic-req.pdf

[22] OASIS (November 2011), Test Assertions Markup Language (TAML). Advancing Open Standards For the Information Society (OASIS), v1.0 edition

[23] IHE Pharmacy Technical Framework Supplement, Pharmacy Prescription (PRE), October 23, 2015

[24] IHE Pharmacy Technical Framework Supplement, Pharmacy Dispense (DIS), October 23, 2015

[25] IHE Pharmacy Technical Framework Supplement, Pharmacy Pharmaceutical Advice(PADV), October 23, 2015

[26] Abderrazek Boufahja, Specifications for HL7 Templates Converter module, November 01, 2015

