NIST GCR 11-941

Continuity of Care (CCD) Suitability Analysis

Lantana Consulting Group

NIST National Institute of Standards and Technology • U.S. Department of Commerce

NIST GCR 11-941

Continuity of Care (CCD) Suitability Analysis

Prepared for
National Institute of Standards and Technology
Gaithersburg Md 20899-8202

By
Lantana Consulting Group

May 2011

U.S. Department of Commerce
Rebecca M. Blank, Acting Secretary

National Institute of Standards and Technology
Patrick D. Gallagher, Under Secretary for Standards and Technology and Director

Lantana

CONSULTING GROUP

Continuity of Care Document (CCD)

Suitability Analysis

May 2011

Prepared by Lantana Consulting Group

for

National Institute of Standards and Technology

Lantana Consulting Group
P.O. Box 177
East Thetford, VT 05043
www.lantanagroup.com

Bob Dolin, MD
President and CMO
bob.dolin@lantanagroup.com

Liora Alschuler
Chief Executive Officer
liora.alschuler@lantanagroup.com

Rick Geimer
Chief Technology Officer
rick.geimer@lantanagroup.com

Gaye Giannone Dolin
Senior Nurse Informaticist
gaye.dolin@lantanagroup.com

Kate Hamilton
Senior Information Analyst
kate.hamilton@lantanagroup.com

Susan Hardy
Senior Technical Editor
susan.hardy@lantangroup.com

Yan Heras
Senior Medical Informaticist
yan.heras@lantanagroup.com

Jingdong Li
Chief Architect
jingdong.li@lantanagroup.com

Brett Marquard
Senior Interoperability Consultant
brett.marquard@lantangroup.com

Sean McIlvenna
Senior Software Architect
sean.mcilvenna@lantangroup.com

Tim Stevens
Technical Writer
tim.stevens@lantanagroup.com

Lauren Wood
Senior Project Manager
lauren.wood@lantanagroup.com

Acknowledgments

This analysis was produced and developed by Lantana Consulting Group for the National Institute of Standards and Technology (NIST) under contract number SB134110CN0039, CCD Definition.

Table of Contents

Table of Figures

Table of Tables

1. Executive Summary

The advent of widespread dependence on electronic exchange of health information has highlighted the importance of interoperability standards and the testability of those standards. Several established and effective test tools and methods are in use; however, the relative merits and, more critically, limitations of different test methods have not been anlaysed or defined. This paper explores test methods for the Health Level Seven (HL7) Continuity of Care Document (CCD); some of our findings and recommendations apply generally to the testing and validation of HL7 Clinical Document Architecture (CDA) instances of all types.

The paper reviews the expression of conformance requirements for Meaningful Use of CCD, categorizes these, and then catalogs the CCD conformance statements. We assess ten methods ranging from fully automated to human inspection for their ability to validate the different categories of conformance requirements.

The analysis indicates that the methods in use are reasonably effective, where combined with a cleaner definition of conformance requirements and descriptions on the limits of each method. Thus, the paper contains recommendations to standards developers on the expression of conformance requirements that could improve the clarity of current test methods and tools.

We give most attention to those methods currently in use: XML Schema, Schematron, and model-driven validation. We also examine potentially useful technologies not widely used for CDA validation at this time, such as RELAX NG and OWL (Web Ontology Language).

2. Introduction

1. Purpose

This document analyzes Health Level Seven's (HL7) Continuity of Care Document (CCD) specification from the perspective of validation, looking at the limits of automated testing and assessing the differential impact of various approaches to it.

This document describes common concepts, structures, and data types within the specification and the extent to which they can be tested through automated processes. It assesses the relevance of testing technologies to CCD and recommends those technologies deemed appropriate for CCD validation. It provides an audit of CCD conformance statements, identifying which test technologies apply. Finally, this document provides recommendations for fully and semi-automated processes to generate test files.

Note that this assessment also provides recommendations to the standards community on construction of conformance requirements to make them more manageable for automated testing; however, the emphasis here is on evaluation of test technologies against an established set of conformance requirements.

Ultimately, this document clearly defines what portions of the HL7 CCD specification are testable and to what extent they will be included in the test data deliverable for this contract.

This report contains the names of commercial products and organizations. Any mention of commercial products or organizations is for information only; it does not imply recommendation or endorsement by Lantana or NIST.

2. Audience

The audience for this document includes NIST staff, standards developers interacting with CCD, and implementers interested in the available means to validate their documents.

3. Approach

Our approach to analyzing CCD coverage consisted of these components:

1. Review of formal conformance requirements
2. Review of validation methods
3. Assessment of validation methods for different types of conformance requirements
4. Analysis of results

We reviewed conformance requirements by describing how these requirements are conveyed and by developing categories of conformance requirements. We then cataloged and categorized all 549 CCD conformance requirements. This review encompassed the conformance requirements of the base standard, CDA, the CCD itself, and the layered constraints added by Integrating the Healthcare Enterprise (IHE), the Health Information Technology Standards Panel (HITSP), and, finally, Meaningful Use.

Independently, we reviewed methods of validating conformance including those in use today and those that might prove useful in the future. In general, we attempted to distinguish between a method of validation and applications of that method; however, where limited tools or experience are available, it is difficult to maintain the distinction between the tool and the method.

We then assessed the fit between these methods and the various types of conformance statements found in CCD. This "coverage analysis" is summarized in tabular form and presented in our conclusions and recommendations to standards developers and to NIST.

The assessment of the testability of CCD conformance statements, presented in the CCD Test Data Coverage appendix, is also the basis for our development of test data presented in a separate document[1].

[1] Lantana Consulting Group. Continuity of Care Document Test Data. May 2011. Separate document prepared for NIST under this contract.

This approach is useful but limited. It is based on broad experience with many of the validation methods and uneven, limited experience applying the tools to CCD. Overall, we have extensive experience with Extensible Markup Language (XML) Schema and Schematron, limited experience with model-driven validation, and no experience applying other methods such as OWL or RELAX NG.

A further limitation is the uneven availability of tools that apply these methods, making it difficult to distinguish between limitations of the method and limitations of the application built to apply the method. This is most apparent for model-driven validation where there is a single, early-stage tool available.

4. *Organization of This Document*

This document contains the following major sections:

- **Introduction**. Provides an overview and scope of the CCD Coverage Report.
- **Healthcare Conformance Requirements and Their Testability**. Identifies testable concepts found in the specifications and describes the challenges presented by those concepts.
- **Test Technologies**. Provides an overview of the tests reviewed.
- **Conclusions and Recommendations.** Recommends methods for creating testable standards and technologies for validating testable conformance statements.
- **Appendices.** Details which CCD conformance statements are testable based on the technologies reviewed, references analysis material, et cetera.

3. Healthcare Conformance Requirements and Their Testability

This section identifies conformance requirements used in healthcare information standards and the challenges presented by those concepts.

The discussion below covers healthcare information standards from several different perspectives, including:

- Datatypes as the building blocks of documents and messages
- Attributes and elements as the building blocks of XML documents
- Standard terminologies as the building blocks of interoperability
- Templates as the building blocks of CDA

1. *Conformance Statements*

Conformance statements define the "enforceable" aspects of an HL7 specification. Failure to follow the rules in the conformance statements means that the artifact in question is invalid.

The majority of conformance statements constrain the CDA base standard. In other words, such conformance statements do not define anything new; rather, they reduce the options provided by the base standard by tightening cardinality, limiting available value sets, et cetera. More rarely, conformance statements extend the base standard through the extension rules of CDA.

1. Conformance Statements vs. Prose Guidance

HL7 specifications contain a mixture of conformance statements and prose guidance. Prose guidance is text that explains how conformance statements can be implemented, sometimes including information about business rules and practices.

Prose guidance in an implmentation guide (IG) should never generate any errors and warnings, thus it should have no impact on validating rule sets. Failure to follow such guidance, however, may result in situations that do not make sense from a clinical or business rule perspective.

Note that CCD was one of the first CDA Release 2 (R2) IGs and, as such, some of the conformance statements blur the line between prose guidance and rigorous, testable conformance criteria. Recent IGs developed with automated template tooling exhibit a much clearer separation between prose guidance and testable conformance statements.

2. Conformance Verbs

The conformance statements in HL7 specifications use conformance verbs to indicate the strength of the constraints. Instructions for using conformance verbs are found in the HL7 Publishing Facilitator Guide[2]. The following table is an excerpt from that guide.

Table 1: Use of Conformance Verbs

To Convey the Sense of:	Use the Following Verb:	
Required/Mandatory	SHALL	SHALL NOT
Best Practice/Recommendation	SHOULD	SHOULD NOT
Acceptable/Permitted	MAY	NEED NOT

From a validation perspective, failure to meet SHALL or SHALL NOT constraints can be considered errors. Failure to meet SHOULD or SHOULD NOT constraints can be considered warnings. Failure to address MAY and NEED NOT constraints does not typically affect validation; thus those constraints are not taken into account when validating rule sets are developed and they need not be included in test suites other than to verify that their presence does not generate an error when it should not.

[2] http://www.hl7.org/v3ballot/html/help/pfg/pfg.html#shall_should_usage

HITSP and IHE use slightly different conformance verbs. The HL7 wiki page on conformance validation[3] shows the current status of work to identify a rough equivalence between these differences; the initial outcome of this work indicates that the more rigorous HL7 language will be adopted. A new CDA Consolidation Project[4] will address these differences.

3. Types of Conformance Statements

This section lists the types of conformance statements typically found in CDA IGs.

1. Parent-to-child Constraints

These constraints indicate a parent-to-child relationship and may contain cardinality defining optionality and repeatability. These conformance statements specify the inclusion and existence of elements or attributes and are easy to validate.

Figure 1: Parent-to-child cardinality constraint example

```
CONF-2: A CCD SHALL contain exactly one ClinicalDocument / documentationOf /
serviceEvent.
```

2. Document Tree Conditional Constraints

Document tree conditional constraints indicate relationships between elements and attributes within the same document tree. These are not simple parent-to-child relationships; instead, they compare two element or attribute values in disparate locations within the same document.

Figure 2: Document tree conditional constraint example

```
CONF-105: The URL of a referenced advance directive document MAY be present,
and SHALL be represented in Observation / reference / ExternalDocument / text /
reference. A <linkHTML> element containing the same URL SHOULD be present in
the associated CDA Narrative Block.
```

3. Template-specific Vocabulary Constraints

There are three kinds of vocabulary constraints: a code can be constrained to equal a specified value, to be drawn from a specified code system, or to be drawn from a specified value set.

A value set may include as few as one value and may be static or dynamic. A static set is fixed as of its publication date. A dynamic set can vary over time and may differ from what is listed at the time the IG is published.

Figure 3: Static value set example

```
CONF-3: The value for "ClinicalDocument / documentationOf / serviceEvent /
@classCode" SHALL be "PCPR" "Care provision" 2.16.840.1.113883.5.6 ActClass
STATIC.
```

Figure 4: Dynamic value set example

```
CONF-6: ClinicalDocument / languageCode SHALL be in the form nn, or nn-CC. The
nn portion SHALL be an ISO-639-1 language code in lower case. The CC portion,
if present, SHALL be an ISO-3166 country code in upper case.
```

4. Non-XML Syntax Constraints

Syntax constraints require document content to conform to a set of syntax requirements, often articulated as a regular expression.

Figure 5: Syntax constraint example

[3] http://wiki.hl7.org/index.php?title=Conformance_validation

[4] http://jira.siframework.org/wiki/display/SIF/CDA+Consolidation+Project

```
CONF-6: ClinicalDocument / languageCode SHALL be in the form nn, or nn-CC. The
nn portion SHALL be an ISO-639-1 language code in lower case. The CC portion,
if present, SHALL be an ISO-3166 country code in upper case.
```

5. Data Integrity Checks

Data integrity checks test whether the data makes clinical sense. These tests often require data manipulation or calculations based on algorithms that are not part of the clinical document. These rules are out of scope for automated validation based on an IG. They can be checked efficiently using standard database functions.

Figure 6: Data integrity check example

```
The admission date must be later than the patient's birth date and equal to or
earlier than the procedure date.
```

6. Business Rules

Business rules are constraints that are useful for the broad use cases often associated with an IG, but do not actually "define" the document type in any way. Business rules are difficult to validate within the context of a CDA document, often requiring access to knowledge that exists outside the document.

Figure 7: Business-rule constraint example

```
CONF-57: A payer in a policy activity SHALL contain one or more performer /
assignedEntity / id, to represent the payer identification number. For pharmacy
benefit programs, this can be valued using the RxBIN and RxPCN numbers assigned
by ANSI and NCPDP respectively. When a nationally recognized payer
identification number is available, it would be placed here.
```

7. Guidance on Data Recording and Accuracy

Some constraints provide guidance to the data recorder. An important subgroup of this class is the "required if known" constraint. Automated validation cannot determine whether the data recorder conformed to the guidance because such validation requires observation of real-world conditions. For example, HITSP requires the work phone for a participant to be coded as "WP"; an automated validation system cannot check whether a phone number is actually a work phone.

Test scripts can be written and applied in the context of application certification with a human monitor to verify the information recorded.

Figure 8: Data recording constraint example

```
CONF-117: The value for "ClinicalDocument / participant / @typeCode" in an
emergency contact participant SHALL be "IND" "Indirect participant"
2.16.840.1.113883.5.90 ParticipationType.
```

8. CDA Extensions

The CDA specification allows users to extend CDA with additional elements and attributes, provided that certain rules are followed. The elements must be in a different namespace from the standard CDA elements, and the recipient must be able to ignore the extensions and still process the document (for example, the extensions cannot change the meaning of standard CDA elements). Elements and attributes from namespaces outside HL7 also may not be used within an element of type ED (encapsulated data; e.g., <text> within <procedure>) or in the human-readable narrative block of a section.

Figure 9: CDA extension example

```
CONF-540: A subject MAY include a deceasedInd element from the urn:hl7-org:sdtc
namespace to indicate whether the person is deceased.
```

9. Compound Conformance Statements

Compound conformance statements combine two or more of any of the types listed earlier in this section in a single conformance statement.

Figure 10: Compound conformance statement example

```
CONF-30: CCD SHOULD contain exactly one and SHALL NOT contain more than one
Payers section (templateId 2.16.840.1.113883.10.20.1.9). The Payers section
SHALL contain a narrative block, and SHOULD contain clinical statements.
Clinical statements SHOULD include one or more coverage activities (templateId
2.16.840.1.113883.10.20.1.20).
```

4. Templates

An HL7 template is a data structure based on the HL7 Reference Information Model (RIM) that expresses the data content needed in a specific clinical or administrative context.

In CDA, templates define reusable sets of constraints that can be identified by template ID. For example, the Health Story Project, HITSP, and IHE have reused the templates defined in CCD.

Two types of templates—open and closed—are found in CDA IGs. Open templates allow all aspects of the base standard unless specifically constrained or prohibited, while closed templates prohibit anything that is not explicitly defined. CCD templates are all open templates.

Many IGs require the use of template IDs. Although CCD does not require them, template IDs are recommended in CCD documents, since many external constraints such as HITSP's C32 specification require them.

Figure 11: CCD template example

```
CONF-82: An advance directive observation (templateId
2.16.840.1.113883.10.20.1.17) SHALL be represented with Observation.
```

Figure 12: C32 template example

```
C32-[CT1-19]: A CDA Document SHALL declare conformance to this specification by
including a <templateID> element with the root attribute set to the value
2.16.840.1.113883.3.88.11.32.1.
```

2. *Base Standard Constraints*

CCD is a CDA IG and, as such, is a constraint on the base CDA standard. The rules of the base CDA standard always apply and have an impact on validation. For example, the CDA requirement that an observation/value be of data type CD (concept descriptor) invokes the rules of the HL7 CD data type; those rules require the presence of a `codeSystem` attribute and that the value of a `codeSystem` attribute follow the rules for object identifier (OID) syntax.

In this section, we describe the most important of the constraints that arise from the base standard.

1. Data Types

CDA R2 is composed of data objects. Each data object contains attributes that are data elements. Every data element has a data type. The data type defines the set of valid values that can be assigned to a data element and the meaning (semantics) of those values. Meaningful data exchange requires the knowledge of the corresponding data type. This is true for complex "values" such as timing specifications as well as for simpler values such as character strings or integer numbers. An instance of a data type is a data value.

Data types are the basic building blocks of a CDA document. A CDA document is composed of many data object instances (i.e., patient, lab results, procedures, administered medications). All

the data object instances must be valid for the assigned data types to make a CDA document syntactically valid.

For more information about HL7 V3 Data Type Release 1, visit: http://www.hl7.org/v3ballot/html/infrastructure/datatypes/datatypes.html.

2. Narrative Block

CDA requires a human-readable narrative block that carries the authenticated content of the document. CDA also defines two relationships that may be asserted between coded entries and the narrative block. First, if all the entries in a section assert the DRIV (derived) relationship, this signals that the narrative block is wholly derived from the entries; e.g., the narrative block contains nothing in addition to what is present in the entries. Second, an entry can reference text content in the narrative block of the current document rather than duplicating the narrative block content in `originalText` elements.

CDA defines a model for the narrative block that is essentially simplified Extensible Hypertext Mark-up Language (XHTML).

3. Identifiers

1. OIDs

An object identifier (OID) is a globally unique string of digits that identifies a path within a tree structure. Numeric digits and the period (.) are the only characters allowed in this string (e.g., "2.16.840.1.113883.6.1"). This format defines a hierarchy, with the left-most number identifying the root and the right-most number identifying the leaf. Registration authorities generate and manage OIDs.

An OID's syntactic validity can be verified by checking its characters. The OID is invalid if any nonnumeric character that is not a period (.) appears.

2. UUIDs

A universally unique identifier (UUID) is a 128-bit, globally unique number. UUIDs may be represented in various ways, but most commonly as a 36-character hexadecimal string of 32 alphanumeric characters separated by four hyphen characters (e.g., "f81d4fae-7dec-11d0-a765-00a0c91e6bf6"). This is the format accepted by HL7. UUIDs may be generated by any authority granted an Institute of Electrical and Electronics Engineers (IEEE) 802 address.

Additionally, UUIDS may be represented as OIDs. In this format, the UUID is converted to numeric format and then appended to the 2.25 OID branch. In this way, the UUID "7b4428f1-de45-40e0-aa9b-2e975f3f6ab5" is represented as "2.25.419453312325339461314294457107592689363 73."

A UUID's syntactic validity can be tested based on its length—36 characters in alphanumeric string representation—and its contents— alphanumeric characters and hyphens.

4. Base Standard Vocabularies

Standard controlled terminologies are building blocks to semantic interoperability. HL7 Version 3 (V3) (including CCD and HITSP/C32) provides a mechanism for communicating codes through coded data types such as CD. HL7 V3 coded data types provide slots for "`code`," "`codeSystem`," "`codeSystemVersion`," "`displayName`," "`originalText`," and "`translations`"; therefore, a concept from any code system can be represented. The coded data types allow a coded concept to be bound to a data element defined in the RIM and CDA framework.

When defining a CDA template, a conformance statement for a coded data element can state whether the data element is bound to a single concept or to a value set. For value-set binding, it can be either dynamic or static. Binding to a single concept and to static value sets are easy to validate because the validation checks a single code or a list of codes that are known to the system at the design time. Dynamic value sets, on the other hand, are not easy to validate because

they are bound to the most current version of the value set, which may be known to the system only at run time.

Vocabularies adopted by the Office of the National Coordinator (ONC) for meaningful use (MU) may be used in CDA implementations. These vocabulary standards include SNOMED CT®, LOINC®, RxNorm, ICD-9-CM, et cetera. CCD specifies which vocabulary standards should be used for coding certain clinical domains—for example, problems and procedures should use SNOMED CT, results and procedures should use LOINC, and products and agents should use RxNorm. Where SNOMED CT is used, it should follow the *Using SNOMED CT in HL7 Version 3; Implementation Guide, Release 1.5*[5]. Because each terminology overlaps with the RIM in different ways, it is critical to select proper concepts that will unambiguously represent the semantic meaning of clinical information when binding to the information model. For instance, it is important to use codes from the correct SNOMED CT hierarchy.

5. XML Structure

1. Syntax Validation

Extensible Mark-up Language (XML) document instances—which include schemas, Extensible Stylesheet Language Transformations (XSLT), and Schematron written in XML syntax—can be tested at several different levels for syntactic correctness. At the base, the document must be well formed (WF). Well-formedness is precisely defined in the XML 1.0 specification[6] and therefore completely testable by an XML parser. Common WF errors include undefined Hypertext Mark-up Language (HTML) character entities (except for those explicitly defined by XML), element nesting errors, and declarations that the document is in a different character encoding than the one it is actually in.

If the XML document instance is not well-formed, then none of the other testing we discuss in this report can be relied upon.

Once the XML document is known to be well-formed, other syntax-level validation can be performed. Schemas for document validation can be written in a number of languages, although the most common is W3C XML Schema (often called XML Schema for short, or XSD). A schema language is a set of rules that dictate which elements are allowed and how they nest, which attributes are allowed on which elements, and which values are allowed for those attributes. These rules form the syntactic grammar that the document must conform to if it is to be called "valid." Different schema languages have different strengths and weaknesses; some of these will be discussed later in this report.

The basic level of schema validation currently for CDA R2 is the CDA.xsd, written in the World Wide Web Consortium (W3C) XML Schema language. This schema supports documents and reports that conform to a large number of IGs. It is very flexible and gives only a small amount of guidance regarding whether a document that it validates will conform to any given IG. For this reason, validating according to the basic CDA XSD can also be considered a necessary baseline test for a document; however, it is not sufficient to determine whether the document actually fits within its context.

2. Attributes and Elements

Attributes and elements form the core building blocks of any XML document. Allowable elements and attributes are defined by CDA and are further constrained by the IG. In CDA, many of the defined elements can be used in several contexts, with different allowed attributes or subelements depending on the context. As an example, in many cases, a given element must contain either attribute A or both attributes B and C, and not a combination. Another example is that the allowed content model of an element may depend on the value of a particular attribute. As we shall see, such requirements are not easily satisfied by all schema languages.

---

[5] Available through HL7 http://www.hl7.org or if an HL7 member with the following link: http://www.hl7.org/v3ballot/html/infrastructure/terminfo/terminfo.htm

[6] http://www.w3.org/TR/REC-xml/

3. Cardinality and Conformance Verbs

Cardinality refers to how many times an element may (or must, or should) be present in a given context. Cardinality is defined as two numbers x..y, where x is the minimum number that must be present, and y is the maximum number (* = unlimited). The cardinality is tested using standard syntactic methods.

In constraints, cardinality is precisely specified for each element, with a mapping to the conformance verbs (see Conformance Verbs). When the matching RIM attribute is "Required" the attribute must be present, and when the RIM attribute is "Mandatory" a nullFlavor value is not allowed.

4. Namespaces

A namespace contains element and attribute definitions (not necessarily defined by a schema). Namespaces allow a document to contain multiple mark-up vocabularies. An XML document can use one or more namespaces. A typical CDA document contains several namespaces.

Figure 13: Namespace declaration example

```
<ClinicalDocument xmlns="urn:hl7-org:v3"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

The first XML namespace (xmlns) attribute defines the default namespace (urn:hl7-org:v3) used by all elements in the CDA base standard. The xmlns:xsi attribute declares the XML schema namespace and allows the use of xsi:type in the document instance.

Implementers and systems may also use other namespaces to support local requirements via CDA's extension mechanism. There are several rules governing the use of other namespaces in CDA; see the section on CDA Extensions for more details.

5. XML Schema and XSI-Type

The data types used in CDA are an XML-specific representation of the data types defined by HL7 in the RIM. Forty such data types specify the building blocks of documents and messages. They include everything from strings (ST) to postal addresses (AD). Some of these data types are represented in the XML using the W3C XML Schema xsi:type attribute. As stated in the W3C XML Schema definition[7], "An element information item in an instance may, however, explicitly assert its type using the attribute xsi:type." To quote CDA R2[8]: "xsi:type is required when:

- The type of the RIM attribute is ANY, RTO (ratio), or QTY (quantity)
- An instance of an SC (character string with code) is being sent as a promotion of an ST"

The xsi:type attribute specifies the restriction of those attributes that are being used in that context in the document instance.

Figure 14: xsi:type example

```
<unitQuantity>
   <numerator value='1' xsi:type='INT'/>
   <denominator value='64' xsi:type='INT'/>
</unitQuantity>
<value xsi:type='IVL_INT'>
   <low value='2'/>
   <high value='4'/>
</value>
```

As part of this restriction, xsi:type can change the definition of a content model. It is the only attribute in W3C XML Schema that can affect the content model of an element[9]. More research is

[7] http://www.w3.org/TR/xmlschema-1/#type
[8] http://www.hl7.org/v3ballot/html/infrastructure/cda/cda.htm
[9] http://www.thaiopensource.com/relaxng/design.html

required to determine the best way to syntactically validate these data types in another schema language in the general case. In practice, this means that XML Schema must be included in any system to validate CDA documents using the current HL7 XML Implementable Technology Specification (ITS).

3. *External Constraints*

MU requires the use of the CCD and the Continuity of Care Record (CCR) for exchange of patient clinical summaries as constrained by C32. It also identifies types of data to be supported in Phase 1, including Medications, Results, Allergies, and Problem Lists.

This section examines testing issues related to HITSP/C32, which identifies CCD as the basic standard for the exchange of a Clinical Summary Document, and other MU testing issues.

1. HITSP/C32

In the HITSP Framework, the HITSP/C32 specification identifies the CCD as the basic standard for the exchange of a Clinical Summary Document. To meet the needs of the US Realm as defined by the American Health Information Community (AHIC) Use Cases, it imposes additional constraints on CCD via the IHE Patient Care Coordination (PCC) Technical Framework, specifically the Exchange of Public Health Record (XPHR) profile. The C32 itself is a minimal document that provides pointers to the actual sections and entry definitions as well as constraints documented in the HITSP/C83 CDA Content Modules and the HITSP/C154 Data Dictionary. Vocabulary and value-set constraints are documented in the HITSP/C80 Clinical Document and Message Terminology Component.

At this level of specification, there may be a significant increase in the number of conformance rules, but there are no additional types of conformance statements that affect testability. Many HITSP constraints appear to be use-case-specific constraints, which are difficult or impossible to test automatically.

It is important to note that C80, C83, and C154 are catalogs of items that are intended to be reused in multiple combinations to support a broad range of exchange scenarios. This is explained in HITSP Publication TN 904 - *Harmonization Framework and Exchange Architecture Technical Note*[10]. This approach fully supports the templated-CDA content, but C80, C83, and C154 do not represent implementable specifications without a document similar to C32 that binds the content modules together for one or more exchange scenarios.

By design, the HITSP/C32 is a generic construct where additional requirements are imposed to support a specific type of exchange. These additional requirements, typically focused on requiring an otherwise optional section, are declared within an Interoperability Specification (IS). Current validation suites do not include any implied or specific constraints from these documents.

For example, IS03 and IS05 describe patient access to health information and patient updates to their own records. Where these are used to update demographics only, other sections, such as medications, are not required. Where used to update medications, the medications section is required.

Currently test gaps exist due to several factors:

1. C32 is generic; the specifics for any given use case are expressed in an IS but there are no IS-specific conformance statements.
2. An exchange scenario sometimes contains implied (or logical) rules only, which rely on common sense.
3. These use-case rules are not currently available as Schematron or any other executable test.
4. There are no document type codes or document-level template IDs that can be scanned to determine if a given instance can support any specific scenario.

[10] http://hitsp.org/ConstructSet_Details.aspx?&PrefixAlpha=5&PrefixNumeric=904

Note that an instance can always be scanned for the section-level template IDs it contains to provide a comprehensive picture of what "standardized" content is available.

Thus, a minimally conforming document can be created and declared valid where the creator of that instance may or may not support all other possible templates of a CCD. Only by voluntary inclusion of all template entries implied by the HITSP/C32 and related documents can a generating system submit an instance to a validation process and be declared capable of creating a "complete" conforming instance.

Given the broad applicability of CCD, this type of testing in support of secondary processing outside the primary intent of the design results in the creation and testing of the most minimal (and simplistic) of documents. These documents will be conforming instances, but have almost no clinical or other transaction support.

2. Meaningful Use

There are no additional testing requirements introduced by the MU rules; however, there are several issues arising from conflicts in the MU/HITSP/IHE/HL7 stack of constraints. These have been resolved in a manner that is critical for implementers to understand:

- MU requires additional vocabularies (such as ICD-9) that are not part of the mandated/allowable vocabularies defined in C83.
  This issue is resolved not by any change or relaxation of the C32 validation rules, but by use of the CDA translation element. This is a best practice that needs to be highlighted for implementers across CDA and related forums and Health Information Technology (HIT) education sites.
- In one instance, the constraints in the IHE XPHR conflict with the CCD and HITSP; in this instance, the current NIST validation tool (http://xreg2.nist.gov/cda-validation/mu.html) follows the HITSP and CCD constraints, and the XPHR constraints are not tested.
- Finally, MU requires four content modules, whereas all modules were optional in the C32 V2.5 specification.

Note that no template ID identifies these constraints or relaxations, so it is currently not possible for a document to assert that it complies with MU constraints. Rather, a user or system must explicitly call an MU validator to test these constraints.

4. Test Technologies

This section describes technologies that are either currently in use for CDA validation or that may be useful in the future. We first provide brief descriptions of the technologies, then document our analysis and ranking of those technologies and how they may be applied to test conformance to CCD.

1. *Ranking Criteria*

The technologies are judged according to their relative strengths and weaknesses against these high-level categories from the section on Healthcare Conformance Requirements and their Testability:

- Conformance Statements: We note any strengths or weaknesses of the technology for validating conformance statements, highlighting any issues with the conformance statement types listed in the previous section. Note that this section does not distinguish between HL7 conformance statements and those of external organizations such as HITSP or IHE.
- Base Standard Constraints: We note any strengths or weaknesses of the technology for validating the base CDA standard and the underlying HL7 V3 standard.

Issues with subcategories of the criteria are listed where appropriate.

Note that all the technologies listed in this document are ranked according to our analysis of their potential based on our knowledge of the technologies and their documented capabilities. We have not done comprehensive testing of each feature to verify all the recommendations.

2. *Currently Used Technologies*

This section briefly describes technologies that are used today to validate CDA documents and addresses the strengths and weaknesses of those technologies for validating various types of conformance statements.

1. CDA.xsd XML Schema

CDA.xsd is the XML Schema file supplied with the base standard. To be compliant with the CDA specification, CDA instances must be validated against the CDA.xsd schema after all extensions have been removed. Note, however, that there are many constraints in the CDA specification that are not represented in the CDA.xsd schema, thus schema validation alone does not guarantee conformance to CDA.

Table 2: CDA.xsd XML Schema Analysis

Criteria	Strengths	Weaknesses	Notes
Conformance Statements	N/A	Unable to validate anything beyond the base standard.	
Base Standard Constraints	XML Structure: Defines the core structure of the base CDA specification, including all allowable elements and attributes. Provides the mimimum standard of conformance according to the CDA spec. Identifiers: Built-in regular expression capabilities sufficient for basic OID/UUID validation.	Data Types: Unable to validate the rules of many data types (i.e., CD needs code and codeSystem attributes unless a nullFlavor is supplied). Narrative Block: Always shows the narrative block as optional, whereas it is only optional for a very small set of use cases, which cannot be differentiated via XML Schema. Base Standard Vocabularies: Applicable for small static value sets only. XML Structure: Does not allow extension elements.	

2. Constrained Schemas

The CDA.xsd schema supplied with the base CDA standard can be modified to validate many aspects of an IG. For example, if `ClinicalDocument/code` is fixed to a single LOINC code in an IG, the constrained schema can limit the code attribute to that code, and the `codeSystem` attribute to the OID for the LOINC coding system. Likewise, if an IG allows an extension, the CDA.xsd schema can be modified to allow the extension.

Constrained CDA schemas have already seen some limited use. Schemas have been created for both CCD (to allow the use of extensions) and C32, though neither one has been widely distributed or adopted to date.

Constrained schemas should be differentiated from use-case specific schemas that are not derived from CDA.xsd, such as those conforming to the HL7 greenCDA methodology. The greenCDA approach involves the creation of custom schemas that focus on variable data, which are then paired with an XSLT transform to generate normative CDA. This approach is useful for generating CDA documents that are valid by design, but the greenSchemas are not themselves useful for validating normative CDA documents.

Table 3: Constrained Schemas Analysis

Criteria	Strengths	Weaknesses	Notes
Conformance Statements	Parent to Child: Can easily validate most parent-to-child constraints. Non-XML Syntax: Regular expressions support non-XML syntax validation. CDA Extensions: Constrained schemas can be modified to allow CDA extentions.	Template Specific Vocabulary: Applicable for small static value sets only. Compound Conformance Statements: Some compound statements may be validated, but will likely need to be broken up into multiple rules, some of which may be untestable.	
Base Standard Constraints	XML Structure: Defines the core structure of the base CDA specification, including all allowable elements and attributes. The mimimum standard of conformance according to the CDA spec. Can allow CDA extensions. Identifiers: Built-in regular expression capabilities sufficient for basic OID/UUID validation.	Data Types: Unable to validate the rules of many data types (i.e., CD needs code and codeSystem attributes unless a nullFlavor is supplied). Narrative Block: Always shows the narrative block as optional, whereas it is only optional for a very small set of use cases, which cannot be differentiated via XML Schema. Base Standard Vocabularies: Applicable for small static value sets only.	

3. Schematron

Schematron provides more comprehensive access to the document tree allowing validation of XML corner cases—including conditional attribute constraints—to a greater degree than with constrained schemas. Schematron tests are often simple to generate because of the ability to specify only the affected area of the document. It is not intended for comprehensive validations, so it is typically paired with other technologies like XML Schema. Schematron leverages XPath, which is described in the next section.

Table 4: Schematron Analysis

Criteria	Strengths	Weaknesses	Notes
Conformance Statements	Parent to Child and	Template Specific	Many software developers

	Document Tree Conditional: Can validate complex conditional constraints using any part of the XML document tree (i.e., not bound to parent/child relationships). Data Integrity Checks: Can use document tree access and XPath comparison functions to perform most document-based integrity checks.	Vocabulary: Generally limited to validating small static value sets, but can perform lookups in other files via the document call (which can also be used to invoke REST-based terminology services that return XML content) Compound Conformance Statements: Can validate some compond conformance statements, but they typically need to be broken up into multiple Schematron rules.	are unfamiliar with Schematron. Can be slow if the reference implementation is run using an XSLT interpreter.
Base Standard Constraints	Data Types: Can validate many aspects of the base standard datatypes that are impractical for XML Schema (i.e., CD needs code and codeSystem attributes unless a nullFlavor is supplied).	Narrative Block: Can only verify that the narrative block is present, but difficult to check that it contains applicable content. Identifiers: Regular expressions are part of XPath 2.0, but current Schematron schemas typically works with XPath 1.0. Base Specification Vocabularies: Most of the base specification value sets are static and already handled more efficiently in the CDA.xsd schema.	

4. XPath

XPath is not a technology to be used on its own. It is an expression language for finding a part of an XML document in terms of its position or hierarchy within the document tree. As such, it is a fundamental part of many XML technologies that can be used to test XML documents for conformance. XPath 1.0 was originally specified as part of XSLT; XPath 2.0 is a superset of XPath 1.0 that incorporates a richer set of data types and works better with XQuery. The result of an XPath expression may be a selection of nodes from the input documents or an atomic value or, more generally, any sequence allowed by the data model[11]. There are some situations in which an XPath 2.0 processor will give different results than an XPath 1.0 processor on the same XPath. These are mostly restricted to datatype-related processing or when the document has been processed against a W3C XML Schema.

XPath uses a non-XML syntax to represent the path through the tree, but the syntax is easy to learn. It is compact enough to fit in a string and similar enough to URL syntax to be recognizable. It is widely implemented, both within XSLT and XQuery processors and in more general-purpose XML libraries in several programming and scripting languages. The language is mature enough that processors have now generally implemented many of the obvious optimizations for speed and efficiency of execution.

XPath is not ranked separately from Schematron for the purposes of this paper; however, use of XPath brings most of the benefits of Schematron to custom validation code (see the section on Custom Validation Code).

5. Model-Driven Validation

[11] http://www.w3.org/TR/xpath20/

Using an underlying Unified Modeling Language (UML) model is an alternative approach to validation that has, to-date, been implemented in a single, open source tool. The Model-driven Health Tools[12] (MDHT) Project develops and promotes model-driven health information standards within the standards community. It provides a unified set of modeling tools for organizations and implementers to design, publish, and implement standards such as CDA, all from a UML model. It uses Object Constraint Language (OCL) as the primary means to realize business conformance constraints.

MDHT provides an Eclipse-based interface already used by many developers. Within Eclipse, it adds a table-based editor for model representation. This enables quick and easy data entry, but the grid UML representation may be initially confusing for many developers, and any constraints below the UML attribute level need to be entered with hand-crafted OCL statements, which are not intuitive for many standards developers.

OCL is quite comprehensive when it comes to constraint representation because it allows the codification of complex relationships. OCL itself, however, is a language that, while using a familiar syntax, is not widely known. The MDHT OCL constraints are hand crafted and they have proven difficult to test and debug.

The tool is under active development. New capabilities such as regular expressions have recently been added.

Table 5: Model-driven Validation Analysis[13]

Criteria	Strengths	Weaknesses	Notes
Conformance Statements	Parent to Child: Expressing parent/child relationships are inherent to UML. Document Tree Conditional: OCL's XPath-like expressive capability makes it possible to test document tree constraints. Non-XML Syntax: MDHT's new regular expression capabilites allow validation of non-XML syntax. Data Integrity Checks: OCL's XPath-like expressive capability should make it possible to execute data integrity checks. CDA Extensions: The model for a particular template can be modified to allow extensions.	Template Specific Vocabulary: Only applicable for small static value sets. Compound Conformance Statements: Some compound statements may be validated, but will likely need to be broken up into multiple rules.	OCL has many of the same strengths as XPath[14]. Once the base standard classes have been created, they can be constrainted in subsequent IGs. Validation in the MDHT implementation is tied to the Eclipse and Java platforms. Future tools may or may not carry this qualification.
Base Standard Constraints	Data Types: OCL has many of the same strengths as XPath, and thus can validate the same kinds of data type rules that Schematron can. Identifiers: Model-driven	Narrative Block: OCL can verify that a narrative block is present, but not that it contains appropriate content. Base standard vocabularies: only	Validation is tied to the Eclipse and Java platforms. Not all base standard constraints have been modeled in MDHT as of this writing. We believe

[12] http://www.openhealthtools.org/projects/charter; http://cdatools.org/; http://sourceforge.net/projects/oht-modeling/
[13] Note that this analysis applies to the methodology, not the capabilities of the current and evolving MDHT application.
[14] See http://en.wikipedia.org/wiki/Object_Constraint_Language - Alternatives

	validation tools that support regular expressions can validate UUID and OID strings. Regular expression support has recently been incorporated into MDHT. XML Structure: MDHT-consrained models can validate most aspects of XML structure. Also, XML Schema validation has recently been incorporated into MHDT, so it will share the same strengths listed for the CDA.xsd XML Schema.	applicable for small static value sets. XML Structure: While model-driven validation alone can validate most aspects of XML structure, in some cases it is necessary to use the XML Schema as well, which requires multi-pass validation. Also, unless that schema is a constrained schema, extensions will still not be supported.	this is a limitation of the currrent implementation, not a general limitation of the approach. Regular expression support is still preliminary, so some datatype validation (such as date or OID patterns) may not be complete at the time of this writing.

6. Terminology Service Validation

Terminology validation is an important piece of CCD validation. When a system receives a CCD instance, it not only needs to validate the structural correctness of the instance, but it also needs to validate whether a correct code is received for a coded data element in CCD. For each coded data element in CCD, the system is able to determine which code or which value set is allowed for a given coded data element based on the CDA and CCD template definitions.

As mentioned previously, static value set binding is easy to validate because the set of allowed concepts in the value set does not change (i.e., a new value set must be created). Because the enumerated list of codes is known to the system at design time, the code can simply be checked against a list of codes in an XML file. Dynamic binding, on the other hand, is difficult to test because the allowed concepts for a coded data element automatically change (expand or contract) as the value set is maintained over time. Terminology services are required to validate a code as a valid member of a dynamic binding value set. These services provide access to terminology content at run time; a function call through terminology services can answer queries such as "is a concept a member of a value set."

Terminology services should be based on the *HL7 Service Functional Model Specification Common Terminology Services Release 2* (CTS 2)[15], which specifies standard interfaces for searching/querying, managing, and accessing terminology content. CTS 2 is an HL7 Draft Standard for Trial Use (DSTU) that specifies the functional requirements for a set of service interfaces to allow the representation, search, access, and maintenance of terminology content, either locally or across a federation of terminology service nodes. The Center for Disease Control and Prevention (CDC) developed the Public Health Information Network Vocabulary Access and Distribution System (PHIN VADS) for accessing, searching, and distributing HL7 messaging value sets used within PHIN and the Nationwide Health Information Network (NHIN). The PHIN VADS vocabulary services are built upon the CTS 2 specification.

Table 6: Terminology Service Validation Analysis

Criteria	Strengths	Weaknesses	Notes
Conformance Statements	Template Specific Vocabularies: Can validate static and dynamic value sets.		
Base Standard Constraints	Base Standard Vocabularies: Can validate static and dynamic value sets.		Base Standard Vocabularies: Most value sets in the base standard are static and already

[15] http://www.hl7.org/documentcenter/public/standards/dstu/2009may/V3_CTS_R2_DSTU_2009OCT.pdf

			present in the CDA XML Schema. For those cases, a terminology service may not be very efficient.

7. Human Validation

Constraints that provide guidance on data recording and accuracy can be tested against a script with a human monitor. These types of constraints are useful for certification, but not for conformance validation. This document addresses automated conformance validation; testing and certification of the accuracy of the data are not considered further here.

Table 7: Human Validation Analysis

Criteria	Strengths	Weaknesses	Notes
Conformance Statements	Often human intervention is the only way to verify the following types of constraints, especially during a certification process: • Data integrity checks • Business rules • Guidance on data recording and accuracy Human intervention is also useful for some compound conformance statements, where only a piece of the statement could otherwise be validated using automated tests.	Human intervention is extremely time consuming and labor intensive, and as such is only recommended for the limited cases listed under Strengths.	
Base Standard Constraints	Narrative block: Humans are best qualified to determine if the narrative block is complete and accurate.	Human intervention is extremely time consuming and labor intensive, and as such is only recommended for the limited cases listed under Strengths.	

3. *Potential Technologies*

The technologies listed below are not widely used for CDA validation at this time, but are worth analyzing for their potential benefits.

1. RELAX NG

RELAX NG[16] is an alternative schema language to W3C XML Schema. Both of those languages grew out of experience with Document Type Definitions (DTD), the schema language that is defined as part of XML 1.0. The RELAX NG language was designed with documents in mind, while W3C XML Schema's design was strongly influenced by database implementers.
RELAX NG is an example of a schema language that specifies a complete document grammar. Documents must meet all constraints defined in that grammar to be considered valid. Strengths of RELAX NG include that it treats attributes and elements as uniformly as possible, which means, for example, that RELAX NG can express more complex attribute alternatives than DTD or W3C XML Schema.

[16] http://www.thaiopensource.com/relaxng/design.html

Table 8: RELAX NG Validation Analysis

Criteria	Strengths	Weaknesses	Notes
Conformance Statements	Parent to Child: Can easily validate parent-to-child constraints. Non-XML Syntax: Has regular expression support for non-XML syntax. CDA Extensions: Can support CDA extensions.	Template-Specific Vocabulary: Appropriate for small static value sets only. Compound Conformance Statements: Most compound conformance statements need to be broken up into separate rules, which can then be evaluated individually to see if they are expressible in RELAX NG.	
Base Standard Constraints	Data Types: Can validate complex attribute alternatives (for example, a code needs code and codeSystem attributes or a nullFlavor). Identifiers: Has built-in support for regular expressions, so can validate OID and UUID syntax. XML Structure: Easily validates basic element, attribute, and cardinality constraints (this is what it was designed for).	Narrative Block: Can validate the presense of the narrative block, but cannot guarantee that it has appropriate content. Base specification vocabularies: Appropriate for small static value sets only.	

2. Web Ontology Language

The Web Ontology Language (OWL)[17] is becoming a popular means of representing ontologies and knowledge. Ontologies and knowledge formalisms have a potential role in validation in that they provide a mechanism for testing whether or not a given instance of a CCD is *subsumed* by a given CCD constraint.

Subsumption testing assesses whether the instance is a proper specialization of the constraint. For example, assume a constraint called “procedure shall be Appendectomy.” One could meet this constraint with an instance that coded the procedure using a CPT4 code ("44950" Appendectomy), a simple SNOMED code ("80146002" Appendectomy), or a more complex SNOMED expression ("65801008" Excision – "363704007" Procedure Site – "181255000" Entire Appendix). In the latter example, determining whether or not the SNOMED expression meets the constraint is a form of subsumption testing. Subsumption rules can be formalized as OWL expressions.

Additional considerations around subsumption testing are described in Dolin RH, Spackman KA, Markwell D. Selective retrieval of pre- and post-coordinated SNOMED concepts. *JAMIA Fall Symposium Supplement* 2002;210-4[18].

Because the use of OWL for constraining CDA documents is still largely theoretical, no analysis tables or formal recommendations are provided here.

3. Custom Validation Code

Custom validation code is the most flexible method for testing complex scenarios, although it is more complex to create than other validation methods. Custom validation code can be most

[17] http://www.w3.org/2004/OWL/
[18] http://www.ncbi.nlm.nih.gov/pubmed/12463817

valuable when developing a system that has several components that must all perform the same validation, or when providing software-as-a-service that may need to validate instances on demand. Custom validation code can be developed using any programming language. Some common programming languages are Java, .NET, and XSLT (non-Schematron-specific).

In general, custom validation code should be developed only when the test scenario is too complex for pre-existing technologies or when the use case requires an enterprise-like architecture/framework for testing. Often, custom validation code is required when test scenarios are overly complex or if they require access to data that resides outside the document (for example, business rule constraints).

No analysis tables or formal recommendations are provided here for custom validation code because its strengths and weaknesses depend on the particulars of the software development language, available libraries, et cetera. Pros and cons are described below.

Pros:

- Can validate any aspect you can imagine writing code for
- With up-front design, tests can be developed in a way that they can be plugged into other applications and easily re-used

Cons:

- Greater up-front and long-term maintenance costs
- Prone to "bugginess"

4. *Test Technology Application*

This section provides examples that show how some (but not all[19]) test technologies may be applied to sample CCD conformance statements, highlighting the strengths of the technology in question.

Table 9: Test Technology Application Examples

Conformance Statement	Constraint Type	Technology Application
Constrained Schemas		
CONF-1: The value for "ClinicalDocument / code" SHALL be "34133-9" "Summarization of episode note" 2.16.840.1.113883.6.1 LOINC STATIC.	Vocabulary	The CDA.xsd schema could be modified, fixing the value of ClinicalDocument/code/@code to 34133-9, and ClinicalDocument/code/@codeSystem to the OID for LOINC.
CONF-5: CCD SHALL contain exactly one ClinicalDocument / languageCode.	Parent-to-child constraint (cardinality example)	The CDA.xsd schema could be modified, changing the cardinality of ClinicalDocument/languageCode from 0..1 to 1..1.
Schematron		
CONF-13: If the author has an associated representedOrganization with no assignedPerson or assignedAuthoringDevice, then the value for "ClinicalDocument / author / assignedAuthor / id / @NullFlavor" SHALL be "NA" "Not applicable" 2.16.840.1.113883.5.1008 NullFlavor STATIC.	Document-tree conditional constraint	XML Schema 1.0 cannot validate this kind of statement; however, it is trivial to implement with Schematron using a single XPath assertion.
Model-Driven Validation		

[19] For instance, the CDA.xsd schema is not useful for CCD conformance statements because CCD conformance statements constrain what is defined in that schema.

CONF-133: If Observation / value in a result observation in the functional status section is of data type CE or CD, then it SHOULD use the same code system used to code the question in Observation / code.	Document-tree conditional constraint	MDHT uses OCL, which has capabilities similar to XPath and which allows an OCL expression to compare the value of two siblings (code and value in this instance) to ensure that they use the same code system when value is of type CE (coded with equivalent) or CD.
Terminology Service Validation		
CONF-55: The value for "Act / code" in a policy activity SHOULD be selected from ValueSet 2.16.840.1.113883.1.11.19832 ActCoverageType DYNAMIC.	Vocabulary	Because ActCoverageType is listed as DYNAMIC, the content of that value set may change over time, so it would be an error to hard code that value set in an XML Schema, Schematron XPath expression, or validation mechanism. However, a call to a terminology service could return the current members of that value set at runtime. The method for calling a terminology service would vary from application to application (Schematron could call a REST-based web service via the document() method, MDHT could invoke a SOAP-based web service via Java code, et cetera).
Human Validation		
CONF-57: A payer in a policy activity SHALL contain one or more performer / assignedEntity / id, to represent the payer identification number. For pharamacy benefit programs this can be valued using the RxBIN and RxPCN numbers assigned by ANSI and NCPDP respectively. When a nationally recognized payer identification number is available, it would be placed here.	Business Rule	A human would be needed to verify that the current use case is a pharmacy benefit program and to determine if RxBIN, RxPCN, or a nationally recognized payer identification number are appropriate for a particular instance.
RELAX NG		
CONF-8: At least one ClinicalDocument / templateId SHALL value ClinicalDocument / templateId / @root with "2.16.840.1.113883.10.20.1," and SHALL NOT contain ClinicalDocument / templateId / @extension.	Parent-to-child constraint	Since RELAX NG treats elements and attributes the same, it is easy to constrain the root.
Custom Validation Code		
CONF-361: If manufacturedMaterial / code contains a precoordinated unit dose (e.g. "metoprolol 25mg tablet"), then SubstanceAdministration / doseQuantity SHALL be a unitless number that indicates the number of products given per administration.	Compound conformance statement (combination of a vocabulary constraint and a document-tree conditional constraint)	This conformance statement would be very difficult to test with any of the recommended technologies in this document; however, with an application programming interface that allows querying the properties of a structured drug coding system like RxNorm, one could check if a drug code contained a pre-coordinated dose, and, if so, generate a validation error if doseQuanity was not a unitless number.

5. Conclusions and Recommendations

1. *Recommendations for Standards Developers*

Currently, conformance statements are often written without testability in mind. Testability comes into consideration only after an IG has been published and there is a desire to validate instances to ensure they comply with the specification.

We recommend considering the testability of conformance statements as they are developed, focusing on writing statements that are fully testable wherever possible. That said, we understand that many real-world conditions are not testable, yet are useful to record in IGs, especially for purposes of certification. Consider the following two fictitious conformance statements:

> **CONF-EXAMPLE-1**: If the phone number in telecom/@value is a work phone, telecom/@use SHALL be "WP."
>
> **CONF-EXAMPLE-2**: If the phone number in telecom/@value is a work phone, telecom/usablePeriod SHALL be present.

A validation system cannot tell that a phone number is a work phone, so neither conformance statement is testable as written. One could argue that neither conformance statement should be present; however, it is useful during certification testing to enter a work phone in an application and verify that it is properly coded as WP when a CDA document is generated.

As such, when otherwise untestable real-world conditions must be represented, we recommend that the IG provide prose guidance describing an explicit method of asserting the condition in a conforming CDA instance.

Using the examples above, CONF-EXAMPLE-1 would be prose guidance: “If the phone number in telecom/@value is a work phone, telecom/@use should be "WP,” or, more generally, “Certified systems must ensure that the user-entered information corresponds to the appropriate value; for example, that a work phone code is coded as ‘WP’”.

CONF-EXAMPLE-2 would be rewritten as follows:

> **CONF-EXAMPLE-2**: If telecom/@use is "WP,” telecom/usablePeriod SHALL be present.

Likewise, any other conformance statements that apply to work phones should be explicitly bound to the value in telecom/@use.

We also recommend discontinuing the practice of coding compound conformance statements. Instead, multiple conformance requirements should be recorded using multiple discrete conformance statements, each of which should be individually testable.

The CDA Consolidation Project[20] is currently addressing the topic of conformance and testability. This may set a positive precedent for development of future IGs. Should future standards adopt this consistent approach, it will simplify the definition of what is testable and the requirements for validation and run-time validation versus certification.

2. *Test Technology Recommendations*

1. General Recommendations to NIST

Of all the technologies listed above, only CDA.xsd and Schematron have been widely adopted. The advances offered by use-case-specific XML schemas and model-driven validation, however, are compelling and deserve further consideration from NIST, especially as they have already seen limited use.

MDHT is an implementation of model-driven validation that is currently being developed with support from ONC via the Standards & Interoperability (S&I) Framework Initiative. One key consideration is that there is only one in-development tool that supports this approach. This in

[20] http://jira.siframework.org/wiki/display/SIF/CDA+Consolidation+Project

itself creates a degree of risk that should be considered when comparing this approach to methods that are supported through a wide range of off-the-shelf tooling.

Other schema languages such as RELAX NG could also prove compelling, but no CDA-based implementations are available to date.

2. Applicable Technologies for Constraint Types

The following tables rank the test technologies from the section on Test Technologies section against the conformance requirements from the Healthcare Conformance Requirements and Their Testability section. The technologies are ranked in the following way:

R: Recommended

CR: Conditionally Recommended

NR: Not Recommended

NA: Not Applicable

For items marked as R, CR, or NR, please refer to the analysis tables in the CCD Test Data Coverage appendix for a description of the strengths or weaknesses supporting that ranking. Note that the recommendations apply to the capabilities inherent in the approach, and assume that any data entry or coding is done correctly. For example, the positive recommendations for Schematron and Model Driven Validation for data types assume that XPath and OCL statements have been entered correctly.

Table 10: Applicable Technologies for Validation

	CDA.xsd XML Schema	Constrained Schemas	Schematron	Model-driven Validation	Terminology Service Validation	Huma Valida
Base Standard Constraints						
Data Types	CR	CR	R	R	NA	NR
Narrative Block	CR	CR	CR	CR	NA	R
Identifiers	R	R	CR	R	NA	NR
Base specification vocabularies	CR	CR	CR	CR	R	NR
XML Structure	CR	R	NA	CR	NA	NR
Conformance Statement Constraints						
Parent to Child	NA	R	R	R	NA	NR
Document Tree Conditional	NA	NA	R	R	NA	NR
Template Specific Vocabulary	NA	CR	CR	CR	R	NR
Non-XML Syntax	NA	R	NA	R	NA	NR
Data Integrity Checks	NA	NA	R	R	NA	R
Business Rules	NA	NA	NA	NA	NA	R
Guidance on Data Recording and Accuracy	NA	NA	NA	NA	NA	R
CDA Extensions	NA	R	NA	R	NA	NR
Compound Conformance Statements	NA	CR	CR	CR	NA	CR

6. References

- CDA Consolidation Project: http://jira.siframework.org/wiki/display/SIF/CDA+Consolidation+Project
- *CDA: Clinical Document Architecture Release 2: Clinical Document Architecture (CDA) Release 2*, May 2005: http://www.hl7.org/v3ballot/html/infrastructure/cda/cda.htm
- Dolin RH, Spackman KA, Markwell D. Selective retrieval of pre- and post-coordinated SNOMED concepts. *JAMIA Fall Symposium Supplement* 2002;210-4: http://www.ncbi.nlm.nih.gov/pubmed/12463817
- *Extensible Markup Language (XML) 1.0 (Fifth Edition)* http://www.w3.org/TR/REC-xml/
- HITSP C154: http://hitsp.org/ConstructSet_Details.aspx?&PrefixAlpha=4&PrefixNumeric=154
- HITSP C32: http://hitsp.org/ConstructSet_Details.aspx?&PrefixAlpha=4&PrefixNumeric=32
- HITSP C80: http://hitsp.org/ConstructSet_Details.aspx?&PrefixAlpha=4&PrefixNumeric=80
- HITSP C83: http://hitsp.org/ConstructSet_Details.aspx?&PrefixAlpha=4&PrefixNumeric=83
- *HITSP Harmonization Framework and Exchange Architecture Technical Note*, TN 904: http://hitsp.org/ConstructSet_Details.aspx?&PrefixAlpha=5&PrefixNumeric=904
- HL7 Conformance Validation: http://wiki.hl7.org/index.php?title=Conformance_validation
- *HL7 Publishing Facilitator Guide*: http://www.hl7.org/v3ballot/html/help/pfg/pfg.html
- *HL7 Service Functional Model Specification Common Terminology Services Release 2* http://www.hl7.org/documentcenter/public/standards/dstu/2009may/V3_CTS_R2_DSTU_2009OCT.pdf
- HL7 V3 Data Type Release 1: http://www.hl7.org/v3ballot/html/infrastructure/datatypes/datatypes.html
- Lantana Consulting Group. Continuity of Care Document Test Data. May 2011. Separate document prepared for NIST under this contract.
- LOINC: Logical Observation Identifiers Names and Codes, Regenstrief Institute. Available at: http://www.regenstrief.org/medinformatics/loinc/
- RELAX NG: http://www.thaiopensource.com/relaxng/design.html
- SNOMED CT: SNOMED Clinical Terms SNOMED International Organization. Available at: http://www.ihtsdo.org/snomed-ct
- *Using SNOMED CT in HL7 Version 3; Implementation Guide, Release 1.5.* Available through HL7 http://www.hl7.org or if an HL7 member with the following link: http://www.hl7.org/v3ballot/html/infrastructure/terminfo/terminfo.htm
- W3C XML Schema definition: http://www.w3.org/TR/xmlschema-1/#type
- Web Ontology Language (OWL): http://www.w3.org/2004/OWL/
- *XML Path Language (XPath) 2.0 (Second Edition)*: http://www.w3.org/TR/xpath20/

A. Acronyms and Abbreviations

AD	Postal Addresses (HL7 V3 data type)
AHIC	American Health Information Community
ASTM	ASTM International (originally known as the American Society for Testing and Materials)
CCD	Continuity of Care Document
CCR	Continuity of Care Record
CD	Concept Descriptor (HL7 V3 data type)
CDA	Clinical Document Architecture
CDC	Center for Disease Control and Prevention
CE	Coded with Equivalents (HL7 V3 data type)
CTS 2	Service Functional Model Specification Common Terminology Services Release 2
DRIV	derived
DSTU	Draft Standard for Trial Use
DTD	Document Type Definition
ED	Encapsulated Data (HL7 V3 data type)
HIT	Health Information Technology
HITSP	Health Information Technology Standards Panel
HL7	Health Level Seven
HTML	Hypertext Mark-up Language
ICD	International Classification of Diseases
IEEE	Institute of Electrical and Electronics Engineers
IG	Implementation Guide
IHE	Integrating the Healthcare Enterprise
IHTSDO	Health Terminology Standard Development Organisation
IS	Interoperability Specification
ITS	Implementable Technology Specification
LOINC	Logical Observation Identifiers Names and Codes
MDHT	Model-driven Health Tools
MU	Meaningful Use
NHIN	Nationwide Health Information Network
OCL	Object Constraint Language
OID	Object Identifier
ONC	Office of the National Coordinator
OWL	Web Ontology Language
PCC	Patient Care Coordination
PHIN VADS	Public Health Information Network Vocabulary Access and Distribution System

QTY	Quantity
REST	Representational State Transfer
R2	Release 2
RIM	Reference Information Model
RTO	Ratio (HL7 V3 data type)
SC	Character String with Code (HL7 V3 data type)
S&I	Standards & Interoperability
SNOMED CT	Systematized Nomenclature of Medicine, Clinical Terms
SOAP	Simple Object Access Protocol
ST	Strings (HL7 V3 data type)
UML	Unified Modeling Language
UUID	Universally Unique Identifier
V3	Version 3
WC3	World Wide Web Consortium
WF	well formed
WP	work phone
XHTML	Extensible Hypertext Mark-up Language
XML	Extensible Markup Language
XPHR	Exchange of Public Health Record
XSD	XML Schema
XSLT	Extensible Stylesheet Language Transformations

B. CCD Test Data Coverage

The following table shows each CCD conformance statement by section and indicates the extent that it has been covered in the test data suite.

In the Status column, “complete” indicates those statements that can be tested with the test data. “Optional” statements are not required by the CCD specification (for example, MAY constraints) and, therefore, cannot be tested; “untestable” statements cannot be tested for the reasons given in the Notes column. “Skipped” statements are also not tested for the reasons given in the Notes. Statements marked with “source-of-info” use information from statements 520 through 533 and thus need not be retested.

Note that the section names in the table below are taken directly from CCD. Thus, “CCR Header Representation” is not a typo; that is the actual section name in CCD (it is a CDA representation of the header from ASTM’s CCR specification).

Table 11: CCD Conformance Statement Coverage

Section	Conformance Statement	Status	Notes
CCR Header Representation			
	001	complete	
	002	complete	
	003	complete	
	004	complete	
	005	complete	
	006	complete	
	007	complete	
	008	complete	
	009	complete	
	010	complete	
	011	complete	
	012	complete	
	013	complete	
	014	optional	
	015	complete	
	016	complete	
	017	complete	
	018	complete	
	019	complete	
	020	complete	
	021	complete	
	022	complete	
	023	complete	
	024	complete	
	025	complete	
	026	complete	
	027	complete	

CCR Body Representation			
	028	optional	
	029	optional	
Payers	030	complete	
Payers	031	complete	
Payers	032	complete	
Payers	033	complete	
Payers	034	complete	
Payers	035	complete	
Payers	036	complete	
Payers	037	complete	
Payers	038	complete	
Payers	039	complete	
Payers	040	complete	
Payers	041	complete	
Payers	042	complete	
Payers	043	complete	
Payers	044	optional	
Payers	045	complete	
Payers	046	complete	
Payers	047	source-of-info	info is reused here; needs to be tested in one place only
Payers	048	complete	
Payers	049	complete	
Payers	050	complete	
Payers	051	complete	
Payers	052	complete	
Payers	053	complete	
Payers	054	complete	
Payers	055	complete	
Payers	056	complete	
Payers	057	complete	
Payers	058	complete	
Payers	059	complete	
Payers	060	complete	
Payers	061	optional	
Payers	062	optional	
Payers	063	optional	
Payers	064	complete	
Payers	065	optional	
Payers	066	complete	
Payers	067	complete	
Payers	068	complete	
Payers	069	complete	
Payers	070	complete	
Payers	071	complete	

Payers	072	complete	
Payers	073	complete	
Payers	074	complete	
Payers	075	complete	
Payers	076	optional	
Advance Directives	077	complete	
Advance Directives	078	complete	
Advance Directives	079	complete	
Advance Directives	080	complete	
Advance Directives	081	complete	
Advance Directives	082	complete	
Advance Directives	083	complete	
Advance Directives	084	complete	
Advance Directives	085	complete	
Advance Directives	086	complete	
Advance Directives	087	complete	
Advance Directives	088	complete	
Advance Directives	089	complete	
Advance Directives	090	optional	
Advance Directives	091	skipped	good files already do not contain an advance directive with 304251008
Advance Directives	092	complete	
Advance Directives	093	optional	
Advance Directives	094	complete	
Advance Directives	095	complete	
Advance Directives	096	optional	
Advance Directives	097	source-of-info	info is reused here; needs to be tested in one place only
Advance Directives	098	complete	
Advance Directives	099	complete	
Advance Directives	100	complete	
Advance Directives	101	complete	
Advance Directives	102	optional	

Advance Directives	103	complete	
Advance Directives	104	complete	
Advance Directives	105	optional	
Advance Directives	106	optional	
Advance Directives	107	untestable	no way to tell if the referenced document is not included
	108	optional	
	109	untestable	no guarantee that it is a guardian
	110	optional	
	111	untestable	no guarantee that it is next of kin
	112	untestable	no guarantee that it is next of kin
	113	untestable	no guarantee that it is next of kin
	114	untestable	no guarantee that it is next of kin
	115	optional	
	116	untestable	no guarantee that it is an emergency contact
	117	untestable	no guarantee that it is an emergency contact
	118	untestable	no guarantee that it is an emergency contact
	119	optional	
	120	untestable	no guarantee that it is a patient caregiver
	121	untestable	no guarantee that it is a patient caregiver
	122	untestable	no guarantee that it is a patient caregiver
Functional Status	123	complete	
Functional Status	124	complete	
Functional Status	125	complete	
Functional Status	126	complete	
Functional Status	127	complete	
Functional Status	128	complete	
Functional Status	129	optional	
Functional Status	130	optional	
Functional Status	131	optional	
Functional Status	132	optional	
Functional Status	133	complete	
Functional Status	134	optional	
Functional Status	135	optional	
Functional Status	136	complete	
Functional Status	137	complete	
Functional Status	138	complete	
Functional Status	139	complete	
Problems	140	complete	
Problems	141	complete	
Problems	142	complete	
Problems	143	complete	
Problems	144	complete	

Problems	145	complete	
Problems	146	complete	
Problems	147	complete	
Problems	148	complete	
Problems	149	complete	
Problems	150	optional	
Problems	151	complete	
Problems	152	optional	
Problems	153	optional	
Problems	154	complete	
Problems	155	complete	
Problems	156	complete	
Problems	157	complete	
Problems	158	complete	
Problems	159	optional	
Problems	160	optional	
Problems	161	source-of-info	info is reused here; needs to be tested in one place only
Problems	162	optional	
Problems	163	complete	
Problems	164	complete	
Problems	165	optional	
Problems	166	complete	
Problems	167	complete	
Problems	168	optional	
Problems	169	complete	
Problems	170	complete	
Problems	171	complete	
Problems	172	complete	
Problems	173	complete	
Problems	174	complete	
Problems	175	complete	
Problems	176	complete	
Problems	177	optional	
Problems	178	complete	
Problems	179	optional	
Problems	180	optional	
Problems	181	complete	
Problems	182	complete	
Problems	183	complete	
Family History	184	complete	
Family History	185	complete	
Family History	186	complete	
Family History	187	complete	
Family History	188	complete	
Family History	189	complete	
Family History	190	complete	

Family History	191	complete	
Family History	192	complete	
Family History	193	complete	
Family History	194	complete	
Family History	195	complete	
Family History	196	complete	
Family History	197	complete	
Family History	198	complete	
Family History	199	source-of-info	info is reused here; needs to be tested in one place only
Family History	200	complete	
Family History	201	complete	
Family History	202	complete	
Family History	203	complete	
Family History	204	complete	
Family History	205	complete	
Family History	206	optional	
Family History	207	optional	
Family History	208	complete	
Family History	209	complete	
Family History	210	optional	
Family History	211	optional	
Family History	212	complete	
Family History	213	complete	
Family History	214	complete	
Family History	215	complete	
Family History	216	optional	
Family History	217	complete	
Family History	218	complete	
Family History	219	complete	
Family History	220	optional	
Family History	221	optional	
Family History	222	optional	
Family History	223	optional	
Family History	224	optional	
Family History	225	complete	
Family History	226	complete	
Family History	227	complete	
Family History	228	complete	
Family History	229	complete	
Family History	230	complete	
Family History	231	complete	
Social History	232	complete	
Social History	233	complete	
Social History	234	complete	
Social History	235	complete	
Social History	236	complete	

Social History	237	complete	
Social History	238	complete	
Social History	239	complete	
Social History	240	complete	
Social History	241	complete	
Social History	242	complete	
Social History	243	complete	
Social History	244	complete	
Social History	245	source-of-info	info is reused here; needs to be tested in one place only
Social History	246	optional	
Social History	247	complete	
Social History	248	complete	
Social History	249	optional	
Social History	250	complete	
Social History	251	optional	
Social History	252	optional	
Social History	253	optional	
Social History	254	optional	
Social History	255	optional	
Alerts	256	complete	
Alerts	257	optional	
Alerts	258	complete	
Alerts	259	complete	
Alerts	260	complete	
Alerts	261	complete	
Alerts	262	complete	
Alerts	263	complete	
Alerts	264	complete	
Alerts	265	complete	
Alerts	266	optional	
Alerts	267	complete	
Alerts	268	optional	
Alerts	269	source-of-info	info is reused here; needs to be tested in one place only
Alerts	270	optional	
Alerts	271	complete	
Alerts	272	complete	
Alerts	273	complete	
Alerts	274	complete	
Alerts	275	complete	
Alerts	276	complete	
Alerts	277	complete	
Alerts	278	complete	
Alerts	279	complete	
Alerts	280	optional	
Alerts	281	complete	

Alerts	282	complete	
Alerts	283	complete	
Alerts	284	complete	
Alerts	285	complete	
Alerts	286	complete	
Alerts	287	complete	
Alerts	288	complete	
Alerts	289	complete	
Alerts	290	complete	
Alerts	291	complete	
Alerts	292	complete	
Alerts	293	complete	
Alerts	294	complete	
Alerts	295	complete	
Alerts	296	complete	
Alerts	297	complete	
Medications	298	complete	
Medications	299	optional	
Medications	300	complete	
Medications	301	complete	
Medications	302	complete	
Medications	303	complete	
Medications	304	complete	
Medications	305	complete	
Medications	306	complete	
Medications	307	complete	
Medications	308	complete	
Medications	309	complete	
Medications	310	complete	
Medications	311	complete	
Medications	312	optional	
Medications	313	optional	
Medications	314	optional	
Medications	315	source-of-info	info is reused here; needs to be tested in one place only
Medications	316	complete	
Medications	317	complete	
Medications	318	complete	
Medications	319	complete	
Medications	320	complete	
Medications	321	optional	
Medications	322	optional	
Medications	323	optional	
Medications	324	optional	
Medications	325	optional	
Medications	326	source-of-info	info is reused here; needs to be tested in one place only

Medications	327	optional	
Medications	328	optional	
Medications	329	complete	
Medications	330	optional	
Medications	331	complete	
Medications	332	complete	
Medications	333	complete	
Medications	334	optional	
Medications	335	complete	
Medications	336	complete	
Medications	337	complete	
Medications	338	optional	
Medications	339	complete	
Medications	340	complete	
Medications	341	complete	
Medications	342	complete	
Medications	343	complete	
Medications	344	complete	
Medications	345	complete	
Medications	346	complete	
Medications	347	complete	
Medications	348	optional	
Medications	349	complete	
Medications	350	optional	
Medications	351	optional	
Medications	352	complete	
Medications	353	complete	
Medications	354	complete	
Medications	355	optional	
Medications	356	complete	
Medications	357	complete	
Medications	358	complete	
Medications	359	complete	
Medications	360	optional	
Medications	361	optional	
Medications	362	optional	
Medications	363	complete	
Medications	364	optional	
Medications	365	optional	
Medications	366	optional	
Medications	367	complete	
Medications	368	optional	
Medications	369	optional	
Medical Equipment	370	complete	
Medical Equipment	371	complete	

Medical Equipment	372	complete	
Medical Equipment	373	complete	
Medical Equipment	374	complete	
Medical Equipment	375	complete	
Immunizations	376	complete	
Immunizations	377	complete	
Immunizations	378	complete	
Immunizations	379	complete	
Immunizations	380	complete	
Vital Signs	381	complete	
Vital Signs	382	complete	
Vital Signs	383	complete	
Vital Signs	384	complete	
Vital Signs	385	complete	
Vital Signs	386	complete	
Vital Signs	387	source-of-info	info is reused here; needs to be tested in one place only
Results	388	complete	
Results	389	complete	
Results	390	complete	
Results	391	complete	
Results	392	complete	
Results	393	complete	
Results	394	complete	
Results	395	complete	
Results	396	complete	
Results	397	complete	
Results	398	complete	
Results	399	optional	
Results	400	optional	
Results	401	complete	
Results	402	complete	
Results	403	optional	
Results	404	optional	
Results	405	complete	
Results	406	source-of-info	info is reused here; needs to be tested in one place only
Results	407	complete	
Results	408	complete	
Results	409	complete	
Results	410	complete	
Results	411	complete	
Results	412	complete	
Results	413	complete	
Results	414	optional	

Results	415	optional	
Results	416	complete	
Results	417	complete	
Results	418	complete	
Results	419	complete	
Results	420	complete	
Results	421	source-of-info	info is reused here; needs to be tested in one place only
Procedures	422	complete	
Procedures	423	complete	
Procedures	424	complete	
Procedures	425	complete	
Procedures	426	complete	
Procedures	427	complete	
Procedures	428	complete	
Procedures	429	complete	
Procedures	430	complete	
Procedures	431	complete	
Procedures	432	complete	
Procedures	433	complete	
Procedures	434	complete	
Procedures	435	optional	
Procedures	436	optional	
Procedures	437	optional	
Procedures	438	optional	
Procedures	439	optional	
Procedures	440	untestable	no statements in the schema could break this CONF
Procedures	441	optional	
Procedures	442	optional	
Procedures	443	optional	
Procedures	444	complete	
Procedures	445	optional	
Procedures	446	optional	
Procedures	447	source-of-info	info is reused here; needs to be tested in one place only
Procedures	448	optional	
Procedures	449	complete	
Procedures	450	complete	
Procedures	451	complete	
Procedures	452	complete	
Encounters	453	complete	
Encounters	454	complete	
Encounters	455	complete	
Encounters	456	complete	
Encounters	457	complete	
Encounters	458	complete	
Encounters	459	complete	

Encounters	460	complete	
Encounters	461	complete	
Encounters	462	complete	
Encounters	463	complete	
Encounters	464	complete	
Encounters	465	optional	
Encounters	466	optional	
Encounters	467	complete	
Encounters	468	optional	
Encounters	469	optional	
Encounters	470	optional	
Encounters	471	optional	
Encounters	472	complete	
Encounters	473	complete	
Encounters	474	complete	
Encounters	475	complete	
Encounters	476	optional	
Encounters	477	complete	
Encounters	478	optional	
Encounters	479	complete	
Plan of Care	480	complete	
Plan of Care	481	complete	
Plan of Care	482	complete	
Plan of Care	483	complete	
Plan of Care	484	complete	
Plan of Care	485	complete	
Plan of Care	486	complete	
Plan of Care	487	complete	
Plan of Care	488	complete	
Plan of Care	489	complete	
Plan of Care	490	complete	
Plan of Care	491	source-of-info	info is reused here; needs to be tested in one place only
Healthcare Providers	492	complete	
Healthcare Providers	493	optional	
Healthcare Providers	494	optional	
CCR Footer Representation			
	495	untestable	not guaranteed to know who all actors are
	496	optional	
	497	untestable	
	498	optional	
	499	optional	
	500	optional	
	501	complete	

	502	optional	
	503	complete	
	504	complete	
	505	complete	
	506	complete	
	507	complete	
General Constraints			
	508	complete	
	509	complete	
	510	complete	
	511	complete	
	512	complete	
	513	complete	
	514	complete	
	515	complete	
	516	complete	
	517	complete	
	518	complete	
	519	complete	
	520	untestable	no guarantee that it is a person source of information
	521	untestable	no guarantee that it is an organization source of information
	522	untestable	no guarantee that reference is for a source of information
	523	untestable	no guarantee that reference is for a source of information
	524	untestable	unable to distinguish difference between source of information observation and a status observation
	525	untestable	unable to distinguish difference between source of information observation and a status observation
	526	untestable	unable to distinguish difference between source of information observation and a status observation
	527	untestable	unable to distinguish difference between source of information observation and a status observation
	528	untestable	unable to distinguish difference between source of information observation and a status observation
	529	untestable	unable to distinguish difference between source of information observation and a status observation
	530	untestable	unable to distinguish difference between source of information observation and a status observation
	531	untestable	unable to distinguish difference between source of information observation and a status observation
	532	untestable	unable to distinguish difference between source of information observation and a status observation
	533	untestable	no way to know for certain that there are missing sources of information
	534	complete	
	535	complete	
	536	untestable	separate specification
Appendix			

	537	optional	
	538	complete	
	539	untestable	invalid CCD conformance statement; base does not always allow ID before NAME.
	540	optional	
	541	complete	
	542	complete	
	543	optional	
	544	complete	
	545	complete	
	546	complete	
	547	complete	
	548	complete	
	549	complete	