Generating XML Schemas for DICOM Structured Reporting Templates

LUYIN ZHAO, MS, KWOK PUN LEE, PhD, JINGKUN HU, DPS

Abstract In this paper, the authors describe a methodology to transform programmatically structured reporting (SR) templates defined by the Digital Imaging and Communications for Medicine (DICOM) standard into an XML schema representation. Such schemas can be used in the creation and validation of XML-encoded SR documents that use templates. Templates are a means to put additional constraints on an SR document to promote common formats for specific reporting applications or domains. As the use of templates becomes more widespread in the production of SR documents, it is important to ensure validity of such documents. The work described in this paper is an extension of the authors’ previous work on XML schema representation for DICOM SR. Therefore, this paper inherits and partially modifies the structure defined in the earlier work.

The Digital Imaging and Communications in Medicine (DICOM) standard aims at creating a public and license-free standard method for the transmission of digital medical images and their associated information. The addition of structured reporting (SR) improves the expressiveness, precision, and comparability of not only images and waveforms, but also documentation of diagnostic observations. More specifically, DICOM SR provides a means to encode structured information to enable unambiguous exchange of clinical information and documentation among systems from different vendors. Structured reporting is defined online, and there is a good book on this topic.

As with any structured data in health care, benefits exist in outcome analysis and point-of-care applications. Patient information and data acquisition are typically performed at a location different from that of interpretation and analysis, necessitating exchange of information. Analysis of historical data is facilitated by an accurate representation of the relevant data as well as the interpretation process. Structured reporting is designed to capture unambiguously structured medical data. Structured reporting in its most general form (referred to throughout as general SR) is very flexible, and the same content can be expressed in different forms and structures, hampering interoperability. Templates are a means to put additional constraints on an SR document to promote common formats for specific reporting applications or domains. For example, DICOM standard Part 16 (DICOM Content Mapping Resource) defines reusable SR templates and mammography computer-aided detection SR information object definition (IOD) templates. Additional templates for other specialties are being developed and standardized, such as:

- Supplement 26: Ultrasound OB-GYN Procedure Reports
- Supplement 66: Catheterization Lab SR SOP (Service-Object Pair) Classes
- Supplement 71: Vascular Ultrasound Procedure Reports
- Supplement 72: Echocardiography Procedure Reports

Currently, the DICOM standard is maintained as a set of Word documents. The rich structure of SR in general and templates in particular is described using tables, leading to invariable errors and inconsistencies as more and more templates are developed by different teams of specialists. Such errors and inconsistencies can be typographical errors, inconsistent naming, incorrect structure, and other errors that occur as a consequence of maintaining formal structures using word-processing tables. When the standard is imposed on real-world medical documents, there is no easy way to ensure that the document structure actually conforms to the standard. Therefore, it is highly desirable to capture the structure with a more formal notation that is amenable to machine processing to overcome the above limitations.

In the past few years, we have been focusing our efforts on applying XML technologies to DICOM SR. EXtensible Markup Language (XML) is a standard format for encoding structured data and has been widely adopted by standards and industries. As the Web now has global acceptance for human information access, XML has become an emerging standard for data exchange. Health care standards organizations including HL7 are transitioning to XML. In previous work, we developed XML schemas from the DICOM SR specification that can be used to create and validate general SR documents encoded in XML. As an extension of that work and adaptation to the emerging standards, this paper proposes...
new XML schemas for DICOM SR template specifications and an approach to generate such schemas automatically. These schemas can be used to guide the creation and validation of XML-encoded SR reports created with templates and to promote interoperability of such reports.

The organization of this paper is as follows. In the first section, we discuss the relationship between general SR and templates. In the next section, we briefly describe DICOM SR and its XML encoding using approaches described in our previous work. The following section describes the process of generating XML schemas for SR templates. In the fifth section, we discuss some of the issues encountered when adapting our previous approach to include additional template constraints and our solutions. We then provide our conclusions and suggestions for further work.

The reader is assumed to have some knowledge of the DICOM standard, various SR template supplements, and XML technologies, specifically XML schema. The results described in this paper are extensions of previous work.6,7

### General SR and Templates

General SR specifies three IODs: basic text, enhanced, and comprehensive. They have the same basic data structures but with more advanced features added to the enhanced and comprehensive IODs. Each IOD contains several normalized or composite object classes called information entities (IEs). An IE consists of a number of modules. For instance, the first column in Table 1 (from the DICOM standard) specifies five IEs within a comprehensive IOD. For each IE, the second column lists modules that belong to the specific IE. Each module is defined in another section of the standard. The last column indicates whether the use of the module is mandatory, a user option, or conditional. The subtle difference between user option and conditional is that the presence of optional elements is dependent on user preference, but the presence of conditional elements is dictated by the presence or absence of other elements. These constraints can be used in combination on a single element.

The document IE is SR specific. Within this IE, the SOP common module is common to all SR IODs. The SR document general module contains general information of an SR document. The SR document content module establishes the information model of observation contexts; it is defined in Section C.17.3 of the DICOM standard but is more easily visualized as a tree, as shown in Figure 1. Such a tree is known as an SR content tree, and each node is known as a content item.

General SR basically specifies the data structures and the allowed types of content items. It does have some semantic constraints such as enumerated values for a few attributes. However, different reporting applications have their own specific observation contexts or vocabularies, which are beyond the general SR constraint scope. This issue is handled through the use of SR templates that apply content constraints on the SR document content module and its components. SR templates are used to put additional constraints on an SR content tree to promote common formats and vocabularies for specific reporting applications (e.g., OB-GYN ultrasound9). They are patterns that specify the concept names, relationship with parent, value type, value multiplicity, requirement type, and value set constraint attributes of content items for a particular application. Currently, these templates are represented as tables. For examples, see Tables 2 through 5.9

Figure 2 shows part of the top-level template identifier (TID) 5000 OB-GYN Ultrasound Procedure Report template (each template is identified by a unique TID). To make the relationship between general SR and template clear, compare Figures 1 and 2. Figure 1 indicates that besides the hierarchical relationship, general SR imposes no constraints on, for example, how a content item should be named, what the relationship should be between content items. However, in a template such as the one shown on Figure 2, specific constraints have been imposed: the tree root must be named “OB-GYN Report” and its value type is CONTAINER, the relationship

---

**Table 1** Comprehensive SR IOD Definition (Table A.35.3-1)

<table>
<thead>
<tr>
<th>IE Module</th>
<th>Reference</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patient Patient</td>
<td>C.7.1.1</td>
<td>M</td>
</tr>
<tr>
<td>Specimen identification</td>
<td>C.7.1.2</td>
<td></td>
</tr>
<tr>
<td>Clinical trial subject</td>
<td>C.7.1.3</td>
<td>U</td>
</tr>
<tr>
<td>Study General study</td>
<td>C.7.2.1</td>
<td>M</td>
</tr>
<tr>
<td>Patient study</td>
<td>C.7.2.2</td>
<td>U</td>
</tr>
<tr>
<td>Clinical trial study</td>
<td>C.7.2.3</td>
<td>U</td>
</tr>
<tr>
<td>Series SR document series</td>
<td>C.17.1</td>
<td>M</td>
</tr>
<tr>
<td>Equipment General equipment</td>
<td>C.7.5.1</td>
<td>M</td>
</tr>
<tr>
<td>Document SR document general</td>
<td>C.17.2</td>
<td>M</td>
</tr>
<tr>
<td>SR document content</td>
<td>C.17.3</td>
<td>M</td>
</tr>
<tr>
<td>SOP common</td>
<td>C.12.1</td>
<td>M</td>
</tr>
</tbody>
</table>

C = conditional; IE = information entity; IOD = information object definition; M = mandatory; SOP = service-object pair; SR = structured reporting; U = user option.

---

*Relationships are directional. Thus, the lower left Rel. (Figure 1) indicates a relationship from the lower content item to the upper content item.*
between content items “Image Library” and “Image Entry” must be CONTAINS, etc.

Each reporting application has only one root template. This root template invokes one or more subtemplates. Templates are represented in a tabular format with the same number of columns. Each row represents a content item in the SR content tree. The Rel with Parent column specifies the relationship of the content item with its parent and the Value Type column constrains its content type. If the value of this column is INCLUDE, it means that another template is invoked. The Concept Name column specifies the coded concept of the content item. Table 2 is the root SR template of TID 5000 OB-GYN Ultrasound Procedure Report. Row 1 defines the constraints on the root content item: mandatory (M) without a relationship constraint. The other rows may invoke other subtemplates such as TID 5006 to further constrain subcontent items or impose a single set of constraints on a branch or atomic content items.

The first cell of the Rel with Parent column of Table 3 is blank. Its actual value is CONTAINS passed from its invoker, Row 10 of Table 2. There are three types of constraints for concept name and value set constraint: coded concept, context group reference, and parameter. A coded concept is a coded vocabulary such as defined terms (125005, DCM, “Fetal Long Bones”) in row 1 of Table 3 or enumerated values (EVs) (18185-9, LN, “Number of follicles in left ovary”) in row 2 of Table 4. A context group reference is composed of a context group type and an identifier (ID) number like Define context group identified (DCID) (12006) Fetal Long Bones Biometry Measurements. A parameter declared in a template conveys a value passed from the invoking template, just as a programming language does. The passed value can be a coded concept such as EV (G-A101, SRT, Left) in row 17 of Table 2, a context group reference such as DCID (12006) Fetal Long Bones Biometry Measurements in row 4 of Table 4, or another parameter such as BiometryType in row 2 of Table 4. The sequence from row 3 of Table 3 to row 2

---

**Table 2** TID 5000 OB-GYN Ultrasound Procedure Report

<table>
<thead>
<tr>
<th>NL*</th>
<th>Rel with Parent</th>
<th>Value Type</th>
<th>Concept Name</th>
<th>VM</th>
<th>Req Type</th>
<th>Condition</th>
<th>Value Set Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CONTAINER</td>
<td>EV (125000, DCM, “OB-GYN Ultrasound Procedure Report”)†</td>
<td>1</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>&gt; CONTAINS</td>
<td>INCLUDE</td>
<td>DTID (5006) Fetal Long Bones Section</td>
<td>1-n U</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>&gt; CONTAINS</td>
<td>INCLUDE</td>
<td>DTID (5013) Follicles Section</td>
<td>1</td>
<td>U</td>
<td>$Laterality = EV (G-A101, SRT, “Left”)</td>
<td></td>
</tr>
</tbody>
</table>

DCID = defined context group identifier; DCM = DICOM; DTID = defined template identifier; EV = enumerated values; M = mandatory; MC = mandatory conditional; NL = nested level; SRT = systemized nomenclature of medicine-RT; U = user option; VM = value multiplicity.

*The three values in parentheses are code value, coding scheme designator, and code meaning.

---

**Table 3** TID 5006 Fetal Long Bones Section

<table>
<thead>
<tr>
<th>NL*</th>
<th>Rel with Parent</th>
<th>Value Type</th>
<th>Concept Name</th>
<th>VM</th>
<th>Req Type</th>
<th>Condition</th>
<th>Value Set Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CONTAINER</td>
<td>DT (125003, DCM, “Fetal Long Bones”)‡</td>
<td>1</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>HAS OBS CONTEXT</td>
<td>INCLUDE</td>
<td>DTID (1008) Subject Context, Fetus</td>
<td>1</td>
<td>MC</td>
<td>IF this template is invoked more than once to describe more than one fetus</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>&gt; CONTAINS</td>
<td>INCLUDE</td>
<td>DTID (5008) Fetal Biometry Group</td>
<td>1-n M</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

DCID = defined context group identifier; DCM = DICOM; DT = defined terms; DTID = defined template identifier; M = mandatory; MC = mandatory conditional; NL = nested level; TI = template identifier; VM = value multiplicity.

*The three values in parentheses are code value, coding scheme designator, and code meaning.

---

The three values in parentheses are code value, coding scheme designator, and code meaning.

---

A parameter declared in a template conveys a value passed from the invoking template, just as a programming language does. The passed value can be a coded concept such as EV (G-A101, SRT, Left) in row 17 of Table 2, a context group reference such as DCID (3627) Measurement Type in row 2 of Table 4, or another parameter such as BiometryType in row 2 of Table 4. The sequence from row 3 of Table 3 to row 2
Table 4 ■ TID 5008 Fetal Biometry Group

<table>
<thead>
<tr>
<th>NL</th>
<th>Rel with Parent</th>
<th>Value Type</th>
<th>Concept Name</th>
<th>VM</th>
<th>Req Type</th>
<th>Condition</th>
<th>Value Set Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CONTAINER</td>
<td>DT (125005, DCM, “Biometry Group”)*</td>
<td>1</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>&gt; CONTAINS</td>
<td>INCLUDE</td>
<td>DTID (300) Measurement</td>
<td>1-n</td>
<td>MC</td>
<td>At least one of rows 2 and 3 shall be present</td>
<td>$Measurement = $BiometryType $Derivation = DCID (3627)</td>
</tr>
<tr>
<td>3</td>
<td>&gt; CONTAINS</td>
<td>NUM</td>
<td>EV (18185-9, LN, “Gestational Age”)</td>
<td>1</td>
<td>MC</td>
<td>At least one of rows 2 and 3 shall be present</td>
<td>Units = EV (d, UCUM, days)</td>
</tr>
<tr>
<td>4</td>
<td>&gt;&gt; INFERRED</td>
<td>CODE</td>
<td>DCID (228) Equation or Table</td>
<td>1</td>
<td>U</td>
<td></td>
<td>DCID (12013) Gestational Age Equations and Tables</td>
</tr>
</tbody>
</table>

DCID = defined context group identifier; DCM = DICOM; DT = defined terms; DTID = defined template identifier; EV = enumerated values; LN = logical observation identifier names and codes; M = mandatory; MC = mandatory conditional; NL = nested level; NUM = numeric; UCUM = unified code for units of measure; VM = value multiplicity.
*The three values in parentheses are code value, coding scheme designator, and code meaning.

Table 5 ■ TID 300 Measurement

<table>
<thead>
<tr>
<th>NL</th>
<th>Rel with Parent</th>
<th>Value Type</th>
<th>Concept Name</th>
<th>VM</th>
<th>Req Type</th>
<th>Condition</th>
<th>Value Set Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NUM</td>
<td>$Measurement</td>
<td>1</td>
<td>M</td>
<td></td>
<td>Units = $Units</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>&gt; HAS CONCEPT MOD</td>
<td>CODE</td>
<td>$ModType</td>
<td>1-n</td>
<td>U</td>
<td>$Mod Value</td>
<td></td>
</tr>
</tbody>
</table>

DICOM SR and XML Schema Representation

XML schema for general SR is our previous work, and the basis of the methodology described in this paper. A detailed description has been previously published.7 Here we summarize the framework of generating XML schemas for general SR and demonstrate how to adapt it for templates.

To deploy SR templates, one of the approaches is to transform the template tables into XML schemas as we did to the general SR so that we can use the schemas to generate and validate the corresponding SR documents. The next section discusses the relationship between the XML schemas for general SR and those for the templates.

DICOM SR and XML Schema Representation

XML schema for general SR is our previous work, and the basis of the methodology described in this paper. A detailed description has been previously published.7 Here we summarize the framework of generating XML schemas for general SR and demonstrate how to adapt it for templates.

XML Schemas for General SR

The XML schemas for general SR are the XML representation of SR IODs. Figure 3 is a high-level structural diagram of the SR schema. The schema definitions patient_ie, study_ie, series_ie, equipment_ie, and sop_common_module are for the common modules. The schema sr_document_general_module represents general information of SR. Since our focus is on SR templates, we concentrate on the schema sr_document_content_module, which is used to represent content trees. We use XML Spy10 to display the details of the schema for the content tree as illustrated in Figures 4 through 8. This is a useful way of visualizing schema fragments: The type of

Figure 2. Part of the top-level OB-GYN Ultrasound Procedure Report template.
Figure 3. XML schema structure for representing general SR.

Figure 4. XML schema for DICOM general SR.

Figure 5. sr_document_content_module element.

Figure 6. document_relationship_macro element.

Figure 7. content_sequence element.

Figure 8. content_sequence_item element.
Table 6 ▪ TID 5002 OB-GYN Procedure Summary Section

<table>
<thead>
<tr>
<th>NL</th>
<th>Rel with Parent</th>
<th>Value Type</th>
<th>Concept Name</th>
<th>VM</th>
<th>Req Type</th>
<th>Condition</th>
<th>Value Set Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CONTAINER</td>
<td>DT (121111, DCM, &quot;Summary&quot;)*</td>
<td>1</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>CONTAINS DATE</td>
<td>DCID (12003) OB-GYN Dates</td>
<td>1-n</td>
<td>U</td>
<td></td>
<td></td>
<td>$Measurement = BCID (12001) OB-GYN Summary</td>
</tr>
<tr>
<td>3</td>
<td>CONTAINS INCLUDE</td>
<td>DTID (300) Measurement</td>
<td>1-n</td>
<td>U</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>CONTAINS TEXT</td>
<td>EV (121066, DCM, &quot;Comment&quot;)</td>
<td>1-n</td>
<td>U</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>INCLUDE</td>
<td>DTID (320) Image or Spatial Coordinates</td>
<td>1-n</td>
<td>U</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>CONTAINS INCLUDE</td>
<td>BTID (5003) OB-GYN Fetus Summary</td>
<td>1-n</td>
<td>UC</td>
<td>No more than 1 inclusion per fetus</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

BTID = baseline template identifier; DCID = defined context group identifier; DCM = DICOM; DT = defined terms; DTID = defined template identifier; EV = enumerated values; M = mandatory; NL = nested level; TID = template identifier; U = user option; UC = user option conditional; VM = value multiplicity.

*The three values in parentheses are code value, coding scheme designator, and code meaning.

**Figure 9.** XML schema for TID 5002.

an element is shown under the name and other unnecessary details are suppressed. The schema sr_document_content_module and its children serve as the meta-model of the schemas of the templates. Later we refer back to Figure 3 to show how it is modified by templates.

The content_sequence element is a nonempty sequence of content_sequence_item elements, which correspond to content items mentioned before. For this case, we show both the graphic and the schema fragment because we refer back to this in the next section.
Note the recursive reference to document_relationship_macro. This is how the SR content tree is built up.

**XML Schemas for Templates**

DICOM SR templates define domain-specific frameworks to allow uniform expression of structured data for various medical specialties. Templates are (additional) constraints on a general SR content tree and can specify, for example, what values are permissible as the value type of a given content item, what relationships with the parent are allowed, what concept names can appear, etc.

In general, one can specify with templates the shape of the subtree hierarchy as well as the allowed values and types at each node. Our approach in generating XML schemas for templates is to modify our methodology for schemas for general SR to take into account these additional constraints.

If we treat the schema definition of sr_document_content_module and its children as a meta-model of the SR content item tree, then the schema definitions of templates are semi-instances of this meta-model. To differentiate between them, we use template-specific names for those complexTypes or

---

**Table 7** - TID 1000 Quotation

<table>
<thead>
<tr>
<th>NL</th>
<th>Rel with Parent</th>
<th>Value Type</th>
<th>Concept Name</th>
<th>VM</th>
<th>Req Type</th>
<th>Condition</th>
<th>Value Set Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>HAS OBS CONTEXT</td>
<td>CODE</td>
<td>EV (121001, DCM, “Quotation Mode”)</td>
<td>1</td>
<td>M</td>
<td></td>
<td>EV (121003, DCM, “Document”)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>EV (121004, DCM, “Verbal”)</td>
</tr>
<tr>
<td>2</td>
<td>HAS OBS CONTEXT</td>
<td>COMPOSITE</td>
<td>EV(121002, DCM, “Quoted Source”)</td>
<td>1</td>
<td>MC</td>
<td>Required if quoted material source is a DICOM composite object</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>HAS OBS CONTEXT</td>
<td>INCLUDE</td>
<td>DTID (1001)</td>
<td>1</td>
<td>M</td>
<td>“Observation Context”</td>
<td></td>
</tr>
</tbody>
</table>

DCM = DICOM; DTID = defined template identifier; EV = enumerated values; M = mandatory; MC = mandatory conditional; NL = nested list; TID = template identifier; VM = value multiplicity.

*The three values in parentheses are code value, coding scheme designator, and code meaning.
groups having the same structures and elements but with specified values. To illustrate the relationship between the schemas of general SR and those of the templates, let us take template TID 5002 OB-GYN Procedure Summary Section shown in Table 6 as an example.

The graphic representation of the generated XML schema for template TID 5002 is given in Figure 9. The schema has more constraints on elements. For example, the relationship_type element in procedure_summary_5002_1_content carries a fixed value "CONTAINS". A clearer example is shown in Figure 10. Compare it with Figure 3, and one can see the additional constraints that the templates put on a general SR content tree (highlighted in shaded boxes).

The generation process is described in the following section.

### Generating XML Schemas for SR Templates

The discussion in the last section constitutes the basis for programmatic XML schema for SR template generation. The process consists of two steps (Fig. 11):

1. **Conversion** of template tables (which have a flat structure) to an intermediate XML representation that tries to recover some of the structural information inherent in templates. We use a tool called Majix for this purpose. The conversion is not perfect, and some cleaning is needed before the next step. Occasionally, the conversion detects typographical and structural errors in the tables. These, as well as mistakes introduced by Majix, are corrected manually. This step is necessary because at present templates are defined with text and tables in a Word document. DICOM is in the process of defining an XML representation of the specification. When this representation is completed, this step will become unnecessary.

2. **Transformation of XML representations to XML schemas.** This is accomplished with a Java program using the Xerces Document Object Model Parser from Apache to generate the schemas. This can also be done with XSLT transformations. The general approach has been described in the last section. More details are discussed in the next section. The schema generated for template TID 5002 OB-GYN Procedure Summary Section is included in Appendix 1.

### Issues in Defining Schema for SR Templates

In this section, we discuss issues encountered as we refine the methodology for general SR to accommodate templates and how they are dealt with. Consider the utility TID 1000 Quotation shown in Table 7:

A template can refer to another template by inclusion, e.g., row 3 indicates a reference to TID 1001 Observation Context. A domain-specific template set such as the OB-GYN Ultrasound Procedure Reports consists of a set of cross-referenced tables.

At the subtree where TID 1000 is used, the table above indicates that the content sequence is such that the first item must have a value type of CODE, and the second item (if present) a value type of COMPOSITE. The concept name of the first item must be "Quotation Mode" (with a well-defined coding scheme designator and code value), and the concept name of the second item is "Quoted Source." The remaining items are defined in another template (TID 1001).

A first attempt to design a schema for this template is to treat it as a content sequence where the content items have further constraints (Fig. 12).

Here the element tid_1000 takes the place of the content_sequence element in Figure 7, and the sequence is constrained to possess specific kinds of content items. (The last element in the diagram, tid_1001, is a reference to the schema for TID 1001.) Constraints on the individual content items can be specified in the respective types as shown in Figure 13.

The new types are specializations of the type for the content_sequence_item element (Fig. 8) and constrain their components with values from the corresponding rows of the template table. For example, the first element in Figure 13 will have a relationship_type element with a fixed value of "HAS OBS CONTEXT" as given by row 1 of the template table. Of course this means that we must have unique names for the type corresponding to each row. We adopt the convention that these types are named with a string, which is a concatenation of the value from the concept name column, the...
template ID, the row number, and a suffix indicating the role of the type. Thus, the first content item above has a type called quotation_mode_1000_1_content (Fig. 14).

There is an additional problem: Since the content_sequence_item elements in Figure 13 are children of the same content_sequence element, they cannot have different types according to the “Element Declarations Consistent” constraint of the XML Schema Recommendation. We resolve this by introducing group references, as shown in the schema fragment below and in Figure 15.

\[
\begin{align*}
\text{xsd:group name} & \text{=} \text{tid}_1000 \\
\text{xsd:sequence} & \text{=} \\
\text{xsd:element name} & \text{=} \text{content_sequence_item} \\
\text{type} & \text{=} \text{pms:quotation_mode_1000_1_content} \\
\end{align*}
\]

This allows us to generate a valid schema for TID 1000. Although the resulting schema is larger in size, this is not an issue since it is generated programmatically.

For another example template with a more complicated structure, refer to TID 5002 OB-GYN Procedure Summary Section (Table 8). It illustrates another issue that must be addressed.

Row 2 of TID 5002 refers to context group 12003 in the Concept Name column. As explained previously, a context group refers to a list of codes from which one code may be chosen in a given context and may appear in the concept name or value set constraint columns of templates tables. Context groups are one of the most important structures used by SR templates. A context group is specified as a table (Table 9). Figure 16 is a graphic representation of TID 5002.

A context group is incorporated into template schemas by defining a type for each group. The type specifies a choice of the elements listed in a given table. The type for context group 12003 is as follows:

\[
\begin{align*}
\text{xsd:complexType name} & \text{=} \text{cid}_12003 \\
\text{xsd:choice} & \text{=} \\
\text{xsd:element name} & \text{=} \text{content_sequence_item} \\
\text{type} & \text{=} \text{pms:quotation_mode_1000_1_content} \\
\end{align*}
\]

This allows us to generate a valid schema for TID 1000. Although the resulting schema is larger in size, this is not an issue since it is generated programmatically.

For another example template with a more complicated structure, refer to TID 5002 OB-GYN Procedure Summary Section (Table 8). It illustrates another issue that must be addressed.

Row 2 of TID 5002 refers to context group 12003 in the Concept Name column. As explained previously, a context group refers to a list of codes from which one code may be chosen in a given context and may appear in the concept name or value set constraint columns of templates tables. Context groups are one of the most important structures used by SR templates. A context group is specified as a table (Table 9). Figure 16 is a graphic representation of TID 5002.

A context group is incorporated into template schemas by defining a type for each group. The type specifies a choice of the elements listed in a given table. The type for context group 12003 is as follows:

\[
\begin{align*}
\text{xsd:complexType name} & \text{=} \text{cid}_12003 \\
\text{xsd:choice} & \text{=} \\
\text{xsd:element name} & \text{=} \text{content_sequence_item} \\
\text{type} & \text{=} \text{pms:quotation_mode_1000_1_content} \\
\end{align*}
\]

This allows us to generate a valid schema for TID 1000. Although the resulting schema is larger in size, this is not an issue since it is generated programmatically.
templates, without which more widespread use of templates will be hindered. We believe that an XML-based framework is appropriate, and our work is a good starting point for the development of a rigorous and machine-processable mechanism for templates. As an extension of this, other highly structured portions of the DICOM standard should likewise be made available in a form that can be processed and validated by software. Such efforts have already begun in the DICOM committee but will take time to complete. In the meantime, the only normative references are the published documents (which can be freely downloaded). We strongly believe that the complexity of handling DICOM standards illustrated by this paper will not be reduced significantly as long as natural language is used as the primary mechanism for defining structured information and suggest to the DICOM community that a more formal approach be adopted as soon as possible.

References


APPENDIX 1

XML Schema Fragment for Template TID 5002
Included here are portions of the generated schema for TID 5002, illustrating the solutions to the issues discussed in the section “Issues in Defining Schema for SR Templates.”

```xml
<xs:element name="code_value" type="pms:code_value" fixed="11778-8"/>
<xs:element name="coding_scheme_designator" type="coding_scheme_designator" fixed="LN"/>
<xs:element name="code_meaning" type="pms:code_meaning" fixed="EDD"/>
<xs:sequence/>
</xs:group>
...
```

Subsequently, we can refer to this context group simply as <xs:group ref="pms:cid_12003"/>

Conclusion and Future Work

Creating and validating SR documents based on the existing standard description is a labor-intensive process and is subject to misinterpretations and errors. Templates for various application domains are typically developed by different domain specialists, and the resulting templates may have inconsistencies and may be difficult to maintain and update. We have described in this paper a methodology that generates XML schemas for DICOM SR templates from the existing specification. These schemas can be used in the creation and validation of SR documents that use templates and have been encoded in XML. The method is a refinement of our earlier work on generating schemas for general SR. A substantial collection of SR templates is being developed in the DICOM community, and it is expected that, in the future, most SR documents will use templates. It is therefore necessary to extend our earlier work to handle templates. We see two main directions for future work.

1. Constraints: Templates themselves can have constraints on usage. Constraints are currently expressed mostly in natural language. The semantics of such constraints need to be defined more precisely before they can be adequately represented in schemas.

2. Templates with Parameters: Templates with parameters have been defined and will be incorporated into the next version of the DICOM standard. Such templates are useful, but, as the current version of XML schema does not allow type definitions with parameters, it is not straightforward to generate schemas for templates with parameters.

At the same time, easy-to-use ancillary tools for validating and verifying the use of templates in SR documents need to be developed.

The complexity of this process as illustrated in this paper underscores the need for a more formal representation for

```xml
<xs:element name="code_value" type="pms:code_value" fixed="11778-8"/>
<xs:element name="coding_scheme_designator" type="coding_scheme_designator" fixed="LN"/>
<xs:element name="code_meaning" type="pms:code_meaning" fixed="EDD"/>
<xs:sequence/>
</xs:group>
```
TID 5002 OB-GYN PROCEDURE SUMMARY SECTION
</xsd:documentation>
<!Start of external schemas->
<!-needed external schemas are specified here, including tid_300.xsd, tid_320.xsd and tid_5003.xsd->
<!-End of external schemas->
<!-This allows this template to be referenced from elsewhere->
<xsd:group name="tid_5002">
<xsd:sequence>
<xsd:element name="content_sequence_item" type="pms:summary_5002_1_content"/>
</xsd:sequence>
</xsd:group>
<xsd:complexType name="summary_5002_1_content">
<xsd:sequence>
<xsd:element name="relationship_type" type="pms:relationship_type" fixed="CONTAINS"/>
<xsd:element name="document_relationship_macro" type="pms:document_relationship_summary_5002_1"/>
<xsd:element name="document_content_macro" type="pms:document_content_summary_5002_1"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="document_content_summary_5002_1">
<xsd:sequence>
<xsd:element name="value_type" type="pms:value_type" fixed="CONTAINER"/>
<xsd:element name="concept_name_code_sequence" type="pms:concept_name_code_sequence"/>
<xsd:element name="continuity_of_content" type="pms:continuity_of_content"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="document_relationship_summary_5002_1">
<xsd:sequence>
<xsd:element name="observation_datetime" type="pms:observation_datetime" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element name="content_template_sequence" type="pms:content_template_sequence_summary_5002_1"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="content_sequence_summary_5002_1">
<xsd:sequence>
<xsd:element name="value_type" type="pms:value_type" fixed="CONTAINER"/>
<xsd:element name="concept_name_code_sequence" type="pms:concept_name_code_sequence"/>
<xsd:element name="continuity_of_content" type="pms:continuity_of_content"/>
</xsd:sequence>
</xsd:complexType>
<xsd:sequence>
  <xsd:attributeGroup ref="pms:concept_name_code_sequence_attributes"/>
</xsd:complexType>
<xsd:complexType name="date_5002_2_code">
  <xsd:choice>
    <xsd:group ref="pms:cid_12003"/>
  </xsd:choice>
</xsd:complexType>
<!-[EV (121106, DCM, "Comment")]->
<xsd:group name="group_comment_5002_4_content">
  <xsd:sequence>
    <xsd:element name="content_sequence_item" type="pms:comment_5002_4_content"/>
  </xsd:sequence>
</xsd:group>
<xsd:complexType name="comment_5002_4_content">
  <xsd:sequence>
    <xsd:element name="relationship_type" type="pms:relationship_type" fixed="CONTAINS"/>
    <xsd:element name="document_relationship_macro" type="pms:document_relationship_comment_5002_4"/>
    <xsd:element name="document_content_macro" type="pms:document_content_comment_5002_4"/>
  </xsd:sequence>
</xsd:complexType>
<!-[omitted->
<xsd:complexType name="document_relationship_comment_5002_4">
  <xsd:sequence>
    <xsd:element name="observation_datetime" type="pms:observation_datetime" minOccurs="0"/>
    <xsd:element name="content_template_sequence" type="pms:content_template_sequence_comment_5002_4"/>
    <xsd:element name="content_sequence" type="pms:content_sequence_comment_5002_4"/>
  </xsd:sequence>
</xsd:complexType>
<!-[omitted->
<xsd:complexType name="content_sequence_comment_5002_4">
  <xsd:sequence>
    <xsd:group ref="pms:tid_320" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
</xsd:schema>