5SL – A Language for Declarative Specification and Generation of Digital Libraries

Marcos André Gonçalves and Edward A. Fox
Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061 USA
+1-540-231-5113

{mgoncalv,fox}@vt.edu

ABSTRACT

Digital libraries (DLs) are among the most complex kinds of information systems, due in part to their intrinsic multidisciplinary nature. Nowadays DLs are built within monolithic, tightly integrated, and generally inflexible systems – or by assembling disparate components together in an ad-hoc way, with resulting problems in interoperability and adaptability. More importantly, conceptual modeling, requirements analysis, and software engineering approaches are rarely supported, making it extremely difficult to tailor DL content and behavior to the interests, needs, and preferences of particular communities. In this paper, we address these problems. In particular, we present 5SL, a declarative language for specifying and generating domain-specific digital libraries. 5SL is based on the 5S formal theory for digital libraries and enables high-level specification of DLs in five complementary dimensions, including: the kinds of multimedia information the DL supports (Stream Model); how that information is structured and organized (Structural Model); different logical and presentational properties and operations of DL components (Spatial Model); the behavior of the DL (Scenario Model); and the different societies of actors and managers of services that act together to carry out the DL behavior (Societal Model). The practical feasibility of the approach is demonstrated by the presentation of a 5SL digital library generator for the MARIAN digital library system.

Categories and Subject Descriptors

H.3.7 [**Information Systems**]: Information Storage and Retrieval—*Digital Libraries*; D.2.2 [**Software Engineering**]: Design Tools and Techniques

General Terms

Design, Languages, Theory

JCDL'02, July 13-17, 2002, Portland, Oregon, USA.

Keywords

5S, 5SL, generators, languages, models, design tools

1. INTRODUCTION

Digital libraries (DLs) have emerged as an important research and application area, facilitated by advances in information technology, especially the Internet and the World Wide Web (WWW). There is strong demand for new and varied DL systems capable of dealing with all kinds of mixed-mode, multimodal, and multimedia digital content. As hundreds of DLs are created by colleges, universities, associations, and diverse other organizations to deal with content they create or digitize, and with local collections of information from numerous remote sources, strong pressure will be exerted to tailor each to special requirements of use as well as content.

The process of building a digital library involves specification of the content to be stored; how that content is organized, structured, described, and accessed; which services are offered by the library (e.g., searching, browsing, personalizing, collaborating); and how patrons (and automated agents) ultimately use those services and interact with each other in the DL environment. Thus, by their own nature, DLs are complex and inter-disciplinary information systems.

This complexity makes it difficult and expensive to construct new systems. Nowadays DLs are built within monolithic, tightly integrated, and generally inflexible systems – or by assembling disparate components in an ad-hoc way, with resulting problems in interoperability and adaptability. Ad hoc construction is not sufficient to meet that demand. Designers of DLs often are either library technical staff with little or no formal training in software engineering, or computer scientists who have little background in information science. Many DL systems use homegrown architectures and are built from scratch with little benefit from worldwide experience with DLs and software engineering. More importantly, conceptual modeling, requirements analysis, and methodological approaches are rarely supported, making it extremely difficult and expensive to tailor DL content and behavior to particular communities' interests and needs. The general trend has been to develop tools to solve small parts of the problem, whereas the root of the problem – the lack of specific DL patterns, models, methodologies, formalisms, and languages – is almost completely ignored.

To address these challenges, we propose a novel digital library modeling language for conceptual DL design, called

Models	Primitives	Formalisms	Objectives
Stream Model	Text; video; audio; picture; software program	Sequences; types	Describes properties of the DL content such as encoding and language for textual material or particular forms of multimedia data
Structural Model	Collection; catalog; hypertext; document; metadata; organization tools	Graphs; nodes; links; labels;	Specifies organizational aspects of the DL content
Spatial Model	User interface; index; retrieval model	Sets; operations; spaces; vector space; measure space; probability space	Defines logical and presentational views of several DL components
Scenarios Model	Service; event; condition; action	Sequence diagrams; collaboration diagrams	Details the behavior of DL services
Societies Model	Community; managers; actors; classes; relationships; attributes; operations	Object-oriented modeling constructs; design patterns	Defines managers, responsible for running DL services; actors, that use those services; and relationships among them

Table 1: 5SL Overview

5SL. 5SL is a high-level, domain-specific language, which allows declarative specification of a number of DL features that are often considered in isolation. Domain specific languages are explicitly designed to address a particular class of problems by offering specific abstractions and notations for the domain at hand. Thus, the main contributions of this paper are: 1) the raising of the level of abstraction in digital library specification and modeling, through a DL design methodology which is model-driven and use-case based; 2) an illustration of how the various DL design issues may be combined in a coherent framework that enriches, extends, and customizes classical models for database, hypertext, information retrieval, and software engineering; and 3) the demonstration of the practical feasibility of the approach to effective DL implementation through the presentation of a DL generator that use 5SL to produce DL applications for the MARIAN digital library system.

This paper is organized as follows. Section 2 gives an overview of the language and its foundations. Section 3 details and exemplifies the use of each of the five models (in 5SL). Section 4 presents the specification and implementation of a DL generator. Section 5 compares our approach to related research and section 6 concludes the paper.

2. 5SL GENERAL FRAMEWORK

2.1 5S View of Digital Libraries

Digital libraries are not benefiting from agreement on definitions and underlying theories. Accordingly, we have proposed and formalized the theory of Streams, Structures, Spaces, Scenarios, and Societies (5S) for digital libraries [15]. Streams are sequences of abstract items used to describe static and dynamic content. Structures can be defined as labeled directed graphs, which impose organization. Spaces are sets of abstract items and operations on those sets that obey certain rules. Scenarios consist of sequences of events or actions that modify states of a computation (process) in order to accomplish a functional requirement. Societies comprehend entities and the relationships between and among them. The 5S theory provides a comprehensive and unified view of the many aspects involved in a digital library.

We have used 5S to formally define digital library concepts [15]. Informally, using the 5S concepts, we summarize that a digital library involves managed *collections* of digital information, accessible over a network, and with associated *services* to support the needs of its communities. Information is manifest in terms of *digital objects*, which contain structured textual or multimedia streams (e.g., images, audio, video). *Metadata* describes different properties of digital objects, and is commonly structured into records. *Catalogs* denote sets of metadata records that are sometimes stored in persistent, distributed repositories. In many cases, structures of digital objects and metadata are explicitly represented and explored to improve the quality of services. Basic DL services include indexing, searching, and browsing, and their behavior is described by means of scenarios or use cases (sets of scenarios). Those services can be tailored to the different societies depending on their roles (e.g., creators of material, librarians, patrons, etc.) and particular needs, interests, or preferences.

2.2 Overview of 5SL

5SL is a domain-specific, declarative language with a formal semantics for conceptual modeling of digital libraries. The formal semantics is understood in terms of a translation of language constructs into the 5S-formalized theory. Its formal basis provides an unambiguous and precise DL specification tool, which can facilitate prototyping, allow proofs of assertions and aid validation of implementations.

With 5SL, the specification of a digital library consists of 5 complementary perspectives. Table 1 describes 5SL by detailing its several sub-models in terms of their primitives, underlying formalisms, and objectives. Figure 1 presents a graphical representation of that 5SL metamodel, where relationships crossing borders between models show (some of the many) points where components logically contained in one of the 'S' models are composed or work together to define additional DL constructs.

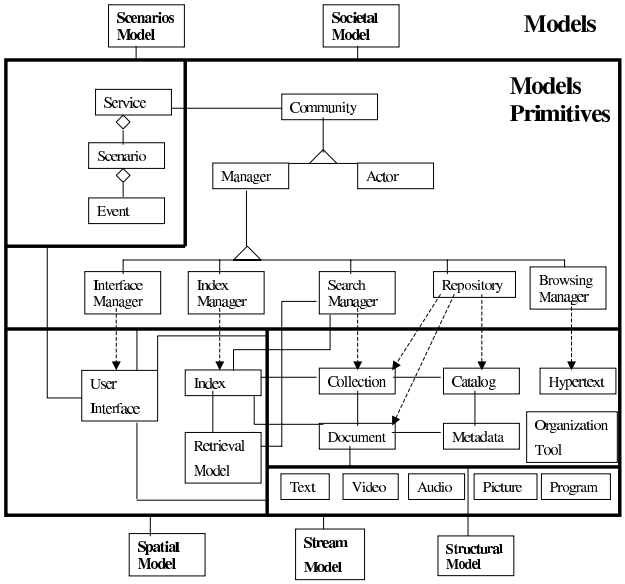

Figure 1: 5SL Metamodel

To improve acceptability and interoperability, 5SL makes extensible use of existing standard specification sublanguages for representing DL concepts, when it turns out to be possible. The need for the integration of multiple languages is a key aspect of the domain-specific language approach [4]. A domain typically consists of multiple subdomains, each of which may require its own particular language. This is particularly true for digital libraries but the aggregative nature of 5S matches this requirement quite well.

5SL utilizes an XML syntax, whose abundance of supporting software tools facilitates the construction of DL generators (see section 4). Most of the 5SL model primitives are defined as XML elements, which can enclose other sublanguages that help to define DL concepts. In more detail, MIME types constitute the basis for encoding streams. XML Schema [26] and/or RDF Schema [8] are the primary tools for describing structures. User Interface Markup Language (UIML) [2] and MathML [18] are used to represent some aspects of spaces. And finally, an adapted and extended version of UXF [25], an XML serialization of UML [7], is used with the Societal and Scenario Models.

2.3 Running Example

Throughout this paper, we employ a simplified running example to illustrate the features and use of 5SL.

A Networked Digital Library of Theses and Dissertations (NDLTD) (www.ndltd.org) digital library manages collections of electronic theses and dissertations (ETDs). Students produce ETDs as a result of their graduate studies and are responsible for submitting their work along with metadata. In the process of creating ETDs, students are encouraged to fully apply new multimedia and hypermedia technologies. The submission service is controlled by a workflow process, which includes a review phase and a cataloguing phase. In the review phase, the university staff checks ETD files, the metadata submitted by the student, and payment of appropriate fees. In the cataloguing phase, MARC, ETD-MS (a metadata standard for ETDs), and other metadata formats are produced from the workflow process, possibly complemented, and distributed for several other catalogs. The DL services for patrons include fulltext and keyword-based searching as well as browsing by author and department. Optionally other services like topical or hierarchical browsing or recommendations can be offered.

3. 5SL MODELS

3.1 Stream Model

The stream model specifies the kinds and formats of multimedia content supported by the digital library. Preservation concerns and technological limitations influence the specification of this model. With the objective of promoting reusability and standardization, we have based this model in the Web standard of MIME types. A MIME type consists of a type (e.g., text), a subtype (e.g., plain, xml) and, in the case of textual data, a character set, which corresponds to the encoding and language. A three-character Z39.53 language code (e.g., ENG, JPN) is used to indicate the language.

Example 1 shows a subset of the kinds of streams supported by an NDLTD DL. It includes English XML texts using UTF-8 encoding, PDF files, and several audio and video formats.

```
<streams>                                              (Ex. 1)
   <text name='ETDText'>
     <content-type>text/xml</content-type>
         <charset>UTF-8</charset>
     <content-type>application/pdf<content-type>
     <lang>ENG</lang>
   </text>
   <audio name='ETDAudio'>
     <content-type='audio/x-aiff'>
   </audio>
   <video name ='ETDVideo'>
     <content-type='video/mpeg'/>
   </video>
    . . .
 </streams>
```

3.2 Structural Model

The structural model considers multiple sources for organization of a digital library. In this model, one describes the internal structure of digital objects (documents), metadata standards, properties of collections and catalogs, as

well as knowledge organization tools, which impose organization upon collections, catalogs, and sets of concepts.

Example 2 presents the portion of the DTD for the structural model, which shows how these different aspects of the DL organization are arranged in a 5SL description.

```
<!ELEMENT Structure (Document, Metadata,      (Ex.2)
                     Organization_Tool?,
                     Collection, Catalog)>
<!ELEMENT Organization_Tool (Authority_File?,
                             Classification_Scheme?,
                             Thesaurus?,
                             Ontology?)>
<!ELEMENT Metadata (Descriptive,
                    Administrative?)>
```

The internal structure of digital objects (or documents) is defined with the use of XML Schema. Similarly, structures for descriptive metadata and knowledge organization tools can be described either with XML or RDF Schema. XML Schema, a standard promoted by the W3C, was conceived to improve on the deficiencies of Document Type Definitions (DTDs), which are context-free grammars that define the logical structure of acceptable XML documents. RDF Schema is a graph and frame-based Schema language for describing RDF, a data model for representing descriptive metadata on the Web. Administrative metadata is described with the use of the Digital Property Rights Language (DPRL) [20], an XML based language developed by Xerox Palo Alto Research Center and used to specify fees, terms, and conditions governing the use of digital content. These choices reflect our policy of extensively using standard sublanguages to promote interoperability when a mapping of the chosen model to the formal 5S definitions exists. It is important to emphasize that the use of those standards does not obligate any of those actual DL structures to be respectively in XML and RDF format, although those are preferred to aid finding components to manipulate them, which could be used by our DL generators.

A typical description of a document is shown in Example 3, which presents a skeleton of an ETD document specification.

```
<document name='ETD'>                          (Ex. 3)
  <stream_enumeration>
    <stream value='ETDText'>
    <stream value='ETDAudio'>
     ...
  </stream_enumeration>
  <structured_stream>
     %XMLSchema%
  <structured_stream>
</document>
```

Documents are defined by imposing structures (possibly including registration and linkage) over (sets of) streams or by using the structure to provide some organization among them. In other words, a document is seen as a structural composition of streams. The set of streams that enter in the composition of the documents (as reflected by the relationship between a document and the entire Stream Model in Figure 1) is specified in the <stream_enumeration> subsection. An XML Schema inside the section <structured_stream> defines the internal organization of the document as a structuring of the streams.

Metadata records are defined in a similar fashion. Figure 2 shows the structure that defines the ETD-MS metadata standard to describe ETDs in an NDLTD DL. It is based on the Dublin Core standard but has special fields to represent information like advisor and committee members, degree and level of education associated with the work, institution granting the degree, and area of study of the intellectual content of the document. A complete XML Schema specification for this is found in [12].

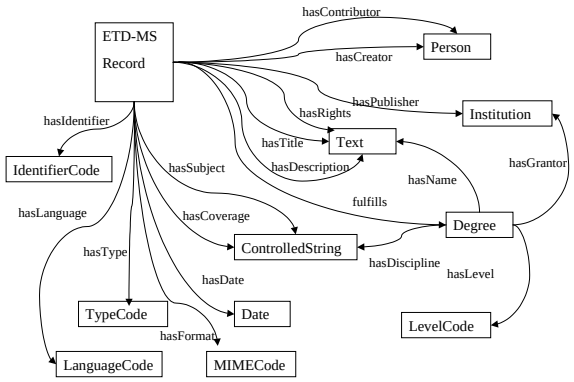

Figure 2: a semantic network representation of the ETD-MS standard

Properties of collections of documents and catalogs of metadata such as name, creator, maintainer, availability, and semantics are similarly defined in the Structural Model. Example 4 shows the 5SL encoding of a NDLTD DL catalog which supports ETD-MS and Dublin Core as metadata formats.

```
<catalog name='VT-ETDCatalog'>                 (Ex. 4)
  <creator='fox@vt.edu' >
  <maintainer='mgoncalv@vt.edu'>
  <public ='true'>
  <metadata_format = 'ETD-MS'
   schema='http://www.ndltd.org/standards/metadata/
          etdms/1.0/etdms.xsd'/>
  <metadata_format = 'dublin_core'
   schema='http://www.openarchives.org/OAI/1.1/dc.xsd'/>
  ....
</catalog>
```

3.2.1 Organization Tools

Organization tools aid structuring in digital libraries. They normally work at the level of collections or catalogs, or over spaces of concepts or features. In 5SL we envision four main types of these tools [17]: 1) authority files, which are lists of terms that are used to control the variant names for an entity or the domain value for a particular field; 2) classification schemes, which provide ways to separate entities into "buckets" or relatively broad topic levels; 3) thesauri, which are based on concepts and relationships between these concepts and terms that represent them; and 4) ontologies,

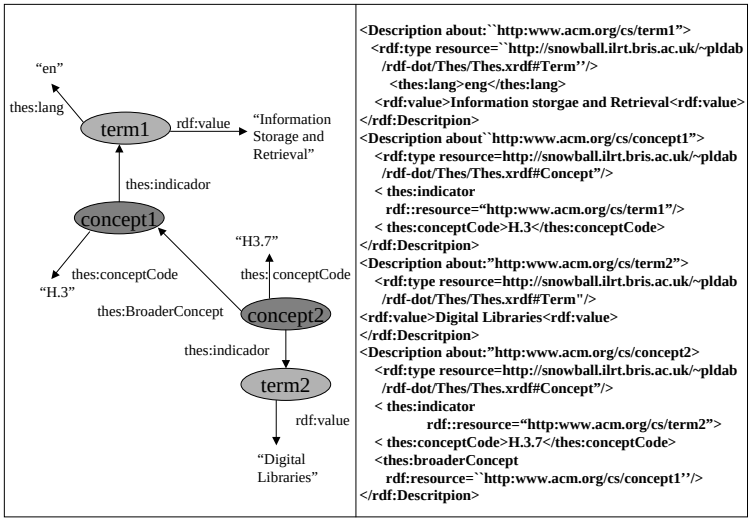

Figure 3: 5SL-RDF representation of a portion of the ACM Classification System

which are models of concepts with rules and axioms for a specific knowledge area.

The representation of these tools in 5SL is based on [10]. This work defines a core RDF vocabulary for representing thesauri and classification schemes. It is based on the ISO2788 guidelines for the establishment and development of monolingual thesaurus. The basic building blocks include the notions of *concepts*, *terms*, and *relationships*. Concepts are indicated by one or more terms. BroaderConcept and relatedConcept denote hierarchical and associative relationships between concepts. Preferred terms are captured by an RDF attribute, termUsage. For thesaurus and classification schemes, two additional properties are described: 1) scope, which includes an optional explanation of the semantics of the concept and/or respective language; and 2) conceptCode, for any code that is assigned to the preferred terms.

Figure 3 illustrates the approach. It shows a RDF graph for a tiny portion of the ACM classification schema with two concepts, indicated respectively by the terms, "Information Storage and Retrieval" and "Digital Libraries", where the former is indicated as being a broaderConcept in relation to the first. The Schema also shows codes and languages for the terms. Part (b) of the figure presents the equivalent RDF code.

3.3 Spatial Model

The Spatial Model specifies a framework for modeling logical representations and operations of several DL components. In particular, this model give details of the underlying DL retrieval models; describes indexes for collections and catalogs (each of which defines a sample space or a multidimensional vector space, depending on the retrieval model), and the user interface appearance (i.e., sets of metric spaces) and behavior.

Properties for the index include types of stemming [5], set of stopwords, if any, etc. Similarly, properties of IR models include types of underlying spaces (e.g., vector, probabilistic), descriptions of corresponding matching operations between query and document abstractions, etc. Currently these properties are described only for documentation purposes and are not being used in the DL generation process, but we have plans to generate complex IR models from those descriptions in the near future (e.g., from equations describing complex graphical belief networks [5]).

3.3.1 The User Interface

The User Interface is an extremely complex component of a digital library and can be connected with all of the 'S's (as shown in Figure 1). We have chosen to describe the UI in the context of the Spatial Model, because we mainly see the modeling of the UI as conceptually defining sets of metric spaces and spatial relationships. In 5SL, the user interface is described by an UIML model [2]. UIML is an XML-based domain specific language for describing user interfaces in a highly device-independent manner. In UIML the interface is described by four elements: structure, style, content, and behavior. A UIML structure enumerates a set of interface parts and their organization corresponds to 5S structures. Similarly, content, which encompasses words, sounds, and images associated with the interface parts, and behavior, which defines UI events and actions, can be described respectively by means of streams and scenarios. Finally, UIML style can be seen as a set of metadata and instructions for the *rendering* of the interface in the actual metric spaces of real UI displayable devices.

3.4 Societal Model

The fundamental concept in the societal model is that of a 'community', a set of entities (human or computer) that share the same characteristics and behavior. In the Societal model we are concerned about identifying the different communities that interact within the DL environment, the main functionalities associated with them, and their semantic relationships. The model is based on the classical object-oriented paradigm, and uses the concepts of at-

tributes, methods, and relationships.

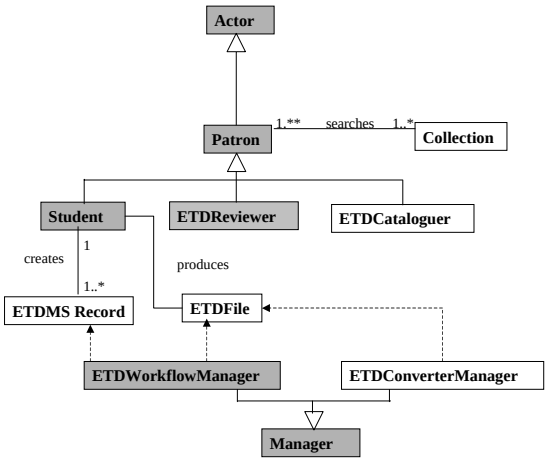

Figure 4: Example of a Societal Schema for an NDLTD site (in UML)

In 5SL, we distinguish two main types of communities: managers and actors. Managers administer DL services while actors explore those services to fulfill their information needs. Managers and actors define and implement basic methods or operations that employ the functionality of the DL services. The exact behavior and functionality of the DL services are described in the Scenario Model with sequences of actions or messages (and corresponding invocations of operations) representing interactions or collaborations between the communities. Note, though, that the functional description of societies as described by their operations/methods could be automatically generated from the Scenarios Model, and their implementation also could be generated by forward engineering [7] or algorithmic model transformations [22]. We have decided to explicitly model at least the interfaces of those operations/methods in the Societal Model as a design choice for consistency purposes, to aid explicit modeling of relationships and constraints, and because of its possible impact in the DL generation process. This decision can be reviewed in the future as we gain more insight with the additional use of the language. Figure 4 shows a simple societal schema for an ETD site that captures the semantics described in the running example (described with UML notation). This schema consists of four actors (Patron, Student, ETDReviewer, and ETDCataloger) and two managers, the ETDWorkflowManager and the ETDConverterManager. In particular, the ETDConverterManager is responsible for converting ETD files into standard formats as specified in the Stream Model.

The XML code of Figure 5 represents part of the 5SL encoding of the societal schema (corresponding to dark boxes in Figure 4).

3.4.1 Core Societal Model

The core societal model is formed by a set of pre-defined managers, whose supporting services constitute the minimal

```
<Society>
   <Actor>
    <Community name=`Patron' IsAbstract=true>
      <Attribute name=`name' type=`String' visibility=`private'/>
      <Attribute name=`ID' type=`Integer' visibility=`private'/>
    </Community>
    <Community name=`Student'/>
      <Operation name=`SubmitETD' visibility=`public' returnType='void'/>
        <Parameter name='ETDFile' type='File'>
    </Community>
    <Community name='ETDReviewer'>
          Operation name=`ReviewETD' visibility=`public' returnType='void'/>
    <\Community>
   </Actor>
    <Generalization>
      <Parent>Patron</Parent>
      <Child>Student</Child>
      <Child>ETDReviewer</Child>
      <Child>ETDCataloger</Child>
    </Generalization>
    <Association>
      <AssociationName name=`Searches'/>
          <Direction source=`Patron' target=`Collection'/>
      </AssociationName>
      <AssociationEnd type=`Patron' multiplicity=`1..*'/>
      <AssociationEnd type='Collection' multiplicity='1..*'/>
    </Association>
  <Manager>
    <Community name='ETDWorkflowManager'>
      <Attribute name=`ETDFiles' type=`set(File)' visibility=`private'/>
      <Attribute name=`MetaRecord' type=`ETDMSRecord' visibility=`private'/>
      <Operation name=`getETDFile' visibility=`public' returnType='void'/>
      <Operation name=`getETDMSRecord' visibility=`public' returnType='void'/>
    </Community>
  </Manager>
</Society>
```

Figure 5: 5SL-XML Societal encoding of the NDLTD Schema

functionality that a digital library should support. There are five of these core managers, as shown in Figure 1. The InterfaceManager is responsible for the active aspects of the user interface. It can be generated from the behavior part of the user interface description. The SearchManager executes the formal search strategy as described in the retrieval model. It normally takes a query representation and a collection, and returns the subset of documents in the collection, along with associated weights, where the weights specify how well the document representation matches with (or implies) the query. The BrowsingManager contains operations to manage hypertexts and to implement run-time navigation activities based on the hypertext topology. The IndexManager is responsible for, given a collection of documents, producing an Index as described in the Spatial Model. The Repository manages collections and catalogs. Basic operations of the repository include adding, deleting, and retrieving documents from a collection.

3.5 Scenario Model

The purpose of the Scenario Model is to describe the behavior of the DL services. The description is realized as a set of sequences of actions that the actor and managers perform to yield an observable result of value to members of the DL society. UML interaction diagrams provide a visual tool to help with this description. 5SL provides a specific way to serialize these graphical representations in XML. A complete specification for a DL should contain at least one scenario for each operation identified in the communities of the Societal Model.

Figure 6 exemplifies the use of a UML sequence diagram for describing a simple scenario of a searching service in an NDLTD DL. A patron searching for ETDs about a particular topic expresses her interests as a fulltext query. The re-

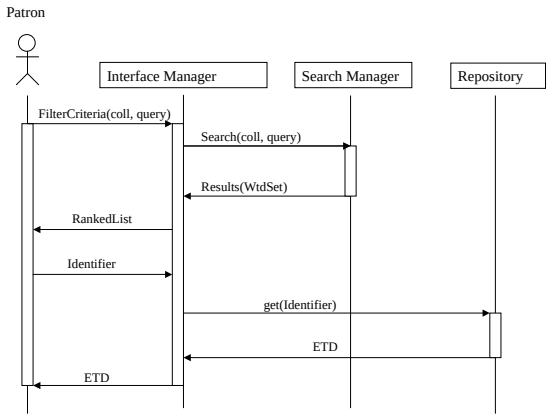

Figure 6: A Simple Search Scenario for ETD site

quest is passed through the InterfaceManager to the SearchManager that executes a search procedure, according with the strategy described in the retrieval model (see Spatial Model). The SearchManager then returns a weighted set of ETD identifiers where weights specify how well the corresponding ETD representations match with the query. The set is presented to the user by the InterfaceManager as an ordered list of titles, which allows her to scan to judge them for relevance. She finds a particular title appealing and requests the ETD. The request is carried out by passing the corresponding identifier to the Repository, which retrieves the particular ETD for the researcher.

```
<SERVICE name ='Searching'>
  <SCENARIO name='SimpleSearching'>
  <NOTE>Simple scenario for an NDLTD
        site searching service</NOTE>
    <EVENT>
     <SENDER>Patron</SENDER>
     <RECEIVER>InterfaceManager</RECEIVER>
     <ACTION name=FilterCriteria/>
       <ARGUMENT>coll</ARGUMENT>
       <ARGUMENT>query</ARGUMENT>
    </EVENT>
    <EVENT>
     <SENDER>InterfaceManager</SENDER>
     <RECEIVER>SearchManager</RECEIVER>
     <ACTION name=Search/>
      <ARGUMENT>coll</ARGUMENT>
      <ARGUMENT>query</ARGUMENT>
    </EVENT>
    <EVENT>
     <SENDER>SearchManager</SENDER>
     <RECEIVER>InterfaceManager</RECEIVER>
     <MESSAGE name=Results/>
      <ARGUMENT>WtdSet</ARGUMENT>
    </EVENT>
    <EVENT>
     <SENDER>InterfaceManager</SENDER>
     <RECEIVER>Patron</RECEIVER>
     <MESSAGE name=RankedList/>
    </EVENT>
    <EVENT>
     <SENDER>Patron</SENDER>
     <RECEIVER>InterfaceManager
     </RECEIVER>
     <MESSAGE name=Identifier/>
    </EVENT>
    <EVENT>
     <SENDER>InterfaceManager
     </SENDER>
     <RECEIVER>Repository
     </RECEIVER>
     <ACTION name=get/>
       <ARGUMENT>Identifier
      </ARGUMENT>
    </EVENT>
    <EVENT>
     <SENDER>Repository</SENDER>
     <RECEIVER>InterfaceManager
     </RECEIVER>
     <MESSAGE name= ETD/>
    </EVENT>
    <EVENT>
     <SENDER>InterfaceManager
     </SENDER>
     <RECEIVER>Patron</RECEIVER>
     <MESSAGE name= ETD/>
    </EVENT>
  </SCENARIO>
</SERVICE>
```

Figure 7: 5SL-XML serialization of the Scenario depicted in Figure 6

The corresponding 5SL code for this scenario is shown in Figure 7. The 5S theory considers scenarios as sets of events, which can be associated with actions and conditions. In 5SL, events are implicitly associated with the receipt and answering of messages. The action associated with the event, which may result in a change of state, is an executable statement that forms an abstraction of a computational procedure as defined in the societal model. The XML textual sequence of event descriptions corresponds to the temporal sequence in the scenario.

A more complex service description is illustrated in Figure 8. For brevity, we omit events related to the InterfaceManager. The scenario depicts the Graduate School perspective of the ETD Submission service. The ETD reviewer logs into the system and checks for the lasted submitted ETDs. For each of those ETDs the reviewer repeats the same process: he chooses an ETD to review; the workflow manager responds with an ETD review page which lists ETD files, metadata, and options; the reviewer downloads ETD files for review and verifies the metadata; and finally he checks if the student has returned all forms and has paid all appropriate fees. The repetition is shown in the picture by an enclosing box with an associated condition; all events in the box are repeated if the reviewer chooses to review another ETD. The reviewer can then choose to accept the submission, so the ETD goes into the library collection. If the reviewer rejects it, an email goes to both student and advisor explaining the reasons for rejection. In the picture, the alternatives are presented by multiple arcs exiting from different boxes representing distinct states; the first arc carries the condition that led to the respective state.

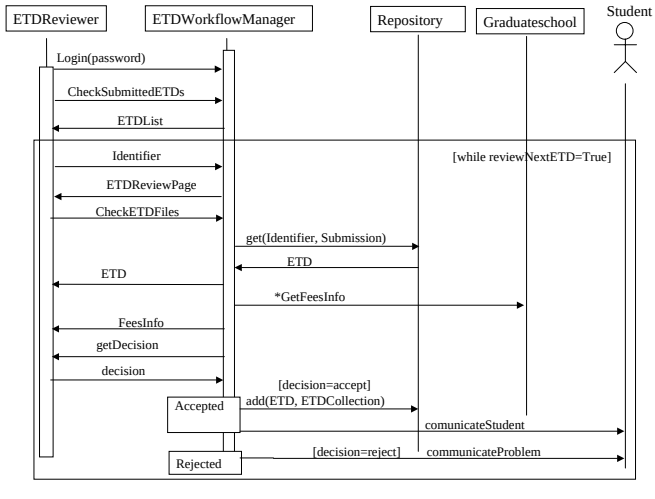

Figure 8: The Review Scenario for the Submission Service

Part of the 5SL encoding is shown in Figure 9. Repetition is captured by the <INTERACTION> element. All events defined inside this element are repeated according to the order of their definition. Alternative are defined inside a <BRANCHING> section. Each <WHEN> subelement with respective condition inside the branching defines an alternative path of execution.

```
<SERVICE name ='Submission'>
  <SCENARIO name='ETDReview'>
   ...
     <INTERATION condition='while reviewNextETD=true'>
        <EVENT>
           <SENDER>ETDReviewer</SENDER>
           <RECEIVER>ETDWorkflowManager </RECEIVER>
           <ACTION name='get'>
             <ARGUMENT>Identifier</ARGUMENT>
              <ARGUMENT>Submission</ARGUMENT>
        </EVENT>
       ...
       <BRANCHING>
          <WHEN condition='decision=accept'>
           <EVENT>
              <SENDER>ETDReviewer</SENDER>
              <RECEIVER>Repository</RECEIVER>
                 <ACTION name='add'/>
                   <ARGUMENT>ETD</ARGUMENT>
                   <ARGUMENT>ETDCollection</ARGUMENT>
                 </ACTION>
           </EVENT>
          ...
          </WHEN>
          <WHEN condition='decision=reject'>
            ...
          </WHEN>
        </BRANCHING>
     </INTERATION>
   </SCENARIO>
</SERVICE>
```

Figure 9: Portion of the 5SL-XML serialization of the Scenario depicted in Figure 8

4. DL GENERATION

The general process of automatic creation of DLs for a particular application is shown in Figure 10. Initially a DL designer is responsible for formalizing a conceptual description of the library using the language concepts. This phase is normally preceded by a 5S analysis of the DL. Declarative specifications in 5SL are then fed into a DL generator, to produce tailored DLs, suitable for specific platforms and requirements. These are built upon a collection of stock parts and configurable components that provide the infrastructure for the new DL. This infrastructure includes the classes of objects and relationships that make up the DL, and processing tools to create the actual library collection from raw documents, as well as services for searching, browsing, and collection maintenance.

5SL is in its infancy but we already have used it to build pilot systems and prototypes. In one of these we built a 5SL generator for the MARIAN digital library system [16] and used 5SL to design a union archive for a federation of ETD sites in NDLTD. In this union archive, metadata in ETD-MS format is periodically harvested from ETD sites using the Open Archives Metadata Harvesting Protocol [24]. The MARIAN system works as a portal for accessing the collection. MARIAN is built around a semantic network model improved with weights and a hierarchy of classes. Any collection of nodes or links in a network can be weighted to represent how well they suit some description or fulfill some role.

The MARIAN DL generator, which is based on a DOM [11] XML parser, automatically generates four kinds of output for the 5SL model of the NDLTD union archive (Figure 10):

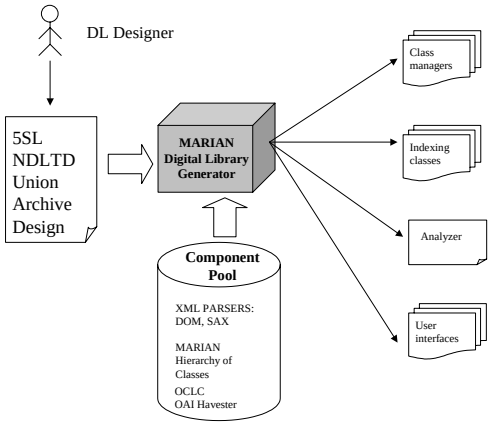

Figure 10: DL Generation Process with 5SL

1. A set of class managers for the NDLTD application

   This include class managers to represent the semantic network view of the ETD-MS standard as represented in Figure 2. Class managers define the logical schema of the DL application, which in MARIAN corresponds to a set of Java classes that represent digital objects, their component parts, and linking information. Class managers also store and maintain instance objects of their class and function as the search engines for the system.

2. Indexing classes

   Represented as sets of semantic bipartite weighted networks between document parts and document features.

3. The analyzer

   The analyzer is an automatically generated SAX event handler that checks incoming XML documents against specifications, extracts structuring and indexing information from valid documents, including controlled authorities like person's names and subject headings, and invokes corresponding class manager methods which materialize and incrementally update structures and indexes, and manage underlying databases.

4. User interfaces

   That includes the search user interface and classes for flat representations of document/metadata with methods for presenting different views of these document abstractions.

The amount of code in lines generated for each of these components is given in Table 2.

Indexing classes	154
Class managers	342
Analyzer	361
Document representation and user interfaces	800

Table 2. MARIAN NDLTD Union Archive generated code

One important feature of this generator is its use of XML namespaces and the MARIAN API. The MARIAN hierarchy of class managers (or API) defines a set of basic types for information retrieval and digital library systems (e.g., controlled strings like personal names and subject headings, English and non-English terms, phrases, etc.). 5SL descriptions use namespaces to import MARIAN types to specify properties of the many different parts of documents and metadata records. These properties include specific matching methods (e.g., controlled strings like personal names and subject heading have different matching methods from English terms and phrases), as well as methods for management of indexes, databases, and sets of instances of the particular class/type. This feature tremendously facilitates the process of DL construction and maintenance.

5. RELATED WORK

Recent research, developed mainly in the database and hypertext communities, has been investigating declarative approaches and representations for specific kinds of information systems, mainly web and e-commerce sites. Strudel [13], Tiramisu [3], and Active Views [1], are examples of systems that have this data-centered perspective of a web site. The common objective is to separate Web site structure, data management, and page presentation, and to provide some query mechanism to allow manipulation of their representations (normally graph-based).

The hypertext/hypermedia community also has a long tradition of developing rich abstraction models and decompositions for hypermedia systems. Examples include OOHDM [23], Web2000 [6], and Autoweb [14]. An interesting approach close to ours is described in [21, 9]. The WebML modeling language and its supporting tool, Torii, provide powerful abstractions to describe and generate the hypertext and navigation structure of Web sites. In comparison, 5SL factors information architectures at a finer granularity, which provides more expressiveness as well as more specific DL constructs and abstractions not present in WebML or any of the cited work.

An even closer approach is described in [28]. The Digital Library Definition Language (DLDL) is focused on describing external behavior of DLs for purposes of supporting interoperability in terms of federated searching. A similar approach is defined in [19]. These approaches however are limited in scope and in their ability to cover all of the challenges in the construction of complex digital libraries.

A remarkable system that shares many objectives with our approach is the New Zealand Greenstone DL system [27] Greenstone allows the construction of complex DLs and tailoring of many parts of DLs to specific domains and needs. However to achieve these goals Greenstone utilizes heterogeneous machinery including Perl modules, proprietary markup languages, and macros, CORBA, Standard Template Library (STL) in C++, etc. In contrast, our approach presents more uniform, high level, and abstract way to deal with all these aspects without committing with any particular implementation or architecture.

In sum, although 5SL shares many of the goals exposed by the research community, none of the implemented methods presents such a comprehensive, homogeneous, and integrated DL oriented approach to cover almost all aspects of digital library design and construction. Moreover, in contrast with most of the related work, 5SL is deeply grounded is a rich formal theory for digital libraries. This, in the future, can facilitate proof of correctness and evaluation of qualitative aspects of DL models such as consistency and management of constraints.

6. CONCLUSIONS AND FUTURE WORK

We have proposed and formalized the theory of Streams, Structures, Spaces, Scenarios, and Societies (5S) for digital libraries [15]. Building on that foundation, we have proposed and specified a modeling language for conceptual DL design, called 5SL. Expressed in XML, and illustrated herein through examples, 5SL shows promise for helping in the design, specification, and construction phases of DLs. Our 5SL-based DL generation process has been demonstrated in conjunction with our MARIAN system and component pool.

Future work will include further application of 5SL in numerous projects of the Digital Library Research Laboratory, such as those connected to the NDLTD and NSDL efforts. To enhance generality, we will construct new DL generators for other types of systems and new DL architectures (e.g., component-based), as well as for more personalized environments (in connection with schemes like PIPE). To validate our approach we will focus on evaluation of the language in terms of usability and complexity of use for intricate DL applications; as well as on evaluation of qualitative aspects of 5SL modeling like consistency among the several 5SL models, complexity of the generation process, and metrics to evaluate 'goodness' of the models. Finally, we hope to advance the broad needs for interoperability through use of 5SL as a canonical DL language in applications that go beyond the current shallow descriptions of DL systems for federated search purposes.

7. ACKNOWLEDGMENTS

Thanks are given for the support of NSF through its grants: IIS-9986089, IIS-0002935, IIS-0080748, IIS-0086227, DUE-0121679, DUE-0121741, and DUE-0136690. The first author also is supported by CAPES, process 1709-98.

8. REFERENCES

[1] S. Abiteboul, B. Amann, S. Cluet, A. Eyal, L. Mignet, and T. Milo. Active views for electronic commerce. In *Proc. of the 21th Int. Conf. on Very Large Databases, Edinburgh, Scotland, 7–10 September, 1999*, pages 138–149, 1999.

[2] M. Abrams, C. Phanouriou, A. L. Batongbacal, S. M. Williams, and J. E. Shuster. UIML: an appliance-independent XML user interface language. *Computer Networks (Amsterdam, Netherlands: 1999)*, 31(11–16):1695–1708, May 1999.

[3] C. R. Anderson, A. Y. Levy, and D. S. Weld. Declarative web site management with Tiramisu. In *WebDB (Informal Proceedings)*, pages 19–24, 1999.

[4] D. L. Atkins, T. Ball, G. Bruns, and K. Cox. Mawl: A domain-specific language for form-based services. *IEEE Transactions on Software Engineering*, 25(3):334–346, May/June 1999.

[5] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley-Longman, 1999.

[6] L. Baresi, F. Garzotto, and P. Paolini. Extending UML for modeling web applications. In Ralph H.

Sprague, Jr., editor, *Proc. 34th Annual Hawaii International Conference on System Sciences (HICSS-34)*. IEEE Computer Society, 2001.

[7] G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Modeling Language User Guide*. Addison-Wesley, Reading, Massachusetts, USA, 1st edition, 1999.

[8] D. Brickley and R. V. Guha (Eds). "Resource Description Framework (RDF) Schema Specification 1.0". W3C Recommendation, March 2000. http://www.w3.org/TR/2000/CR-rdf-schema-20000327/.

[9] S. Ceri, P. Fraternali, and S. Paraboschi. Data-driven, one-to-one web site generation for data-intensive applications. In *Proc. of the 25th Int. Conf. on Very Large Data Bases (VLDB '99)*, pages 615–626, San Francisco, September 1999.

[10] P. Cross, D. Brickley, and T. Koch. Conceptual relationships for encoding thesauri, classification systems and organized metadata collections and a proposal for encoding a core set of thesaurus relationships using an RDF Schema. http://www.desire.org/results/discovery/rdfthesschema.html, June, 2000.

[11] Document object model (DOM). `http://www.w3.org/DOM/`.

[12] ETD-MS: an interoperability metadata standard for electronic theses and dissertations. `http://www.ndltd.org/standards/metadata/`.

[13] M. Fernández, D. Florescu, A. Levy, and D. Suciu. Declarative specification of Web sites with Strudel. *VLDB Journal: Very Large Data Bases*, 9(1):38–55, 2000.

[14] P. Fraternali and P. Paolini. Model-driven development of Web applications: the AutoWeb system. *ACM Transactions on Information Systems*, 18(4):323–382, 2000.

[15] M. A. Gonçalves, E. A. Fox, L. T. Watson, and N. A. Kipp. Streams, structures, spaces, scenarios and societies (5s): A formal model for digital libraries. Technical Report TR-01-12, Virginia Tech, Blacksburg, VA, 2001.

[16] M. A. Gonçalves, R. K. France, and E. A. Fox. Marian: Flexible interoperability for federated digital libraries. In *Proc. of the 5th European Conf. on Research and Advanced Technology for Digital Libraries*, pages 173–186, Darmstadt, Germany, 2001.

[17] G. Hodge. Systems of knowledge organization for digital libraries: Beyond traditional authority files. CLIR Pub91. `http://nkos.slis.kent.edu/KOS_taxonomy.htm`, July, 2000.

[18] P. Ion and R. Miner. Mathematical Markup Language (MathML) 1.0 Specification. W3C Recommendation. `http://www.w3.org/TR/REC-MathML`, Apr 7, 1998.

[19] J. Powell and E. A. Fox. Multilingual federated searching across heterogeneous collections. *D-Lib Magazine*, 5(8), 1998.

[20] A. Ramanujapuram and P. Ram. Digital content and intellectual property rights. *Dr. Dobb's Journal of Software Tools*, 23(12):20–22, 24, 26–27, December 1998.

[21] A. Bongio S. Ceri, P. Fraternali. Web modeling language (WebML): a modeling language for designing web sites. In *Proc. of the 9th Int. World Wide Web Conference*, Amsterdam, 2000.

[22] Siegfried Schönberger, Rudolf K. Keller, and Ismail Khriss. Algorithmic support for model transformation in object-oriented software development. *Concurrency and Computation: Practice and Experience*, 13(5):351–383, April 2001.

[23] D. Schwabe, G. Rossi, and S. D. J. Barbosa. Systematic hypermedia application design with OOHDM. In *Proc. of the 7th ACM Conf. on Hypertext*, pages 116–128, 1996.

[24] H. V. D. Sompel and C. Lagoze. The Open Archives Initiative protocol for metadata harvesting. http://www.openarchives.org, 2001.

[25] J. Suzuki and Y. Yamamoto. Making UML models interoperable with UXF. *Lecture Notes in Computer Science*, 1618:78–87, 1999.

[26] H. S. Thompson, D. Beech, M. Maloney, and N. Mendelsohn (Eds). "XML Schema Part 1: Structures". W3C Recommendation, May 2001. http://www.w3.org/TR/xmlschema-1/.

[27] I. H. Witten, R. J. McNab, S. J. Boddie, and D. Bainbridge. Greenstone: A comprehensive open-source digital library software system. In *Proc. of the 5th ACM Int. Conf. on Digital Libraries*, pages 113–121, San Antonio, Texas, 2000.

[28] M. Zubair, K. Maly, I. Ameerally, and M. Nelson. Dynamic construction of federated digital libraries. In *Proceedings of the Ninth International World Wide Web Conference*, Amsterdam, 2000.