Streams, Structures, Spaces, Scenarios, Societies (5S): A Formal Model for Digital Libraries

Marcos André Gonçalves, Edward A. Fox
Layne T. Watson, Neill A. Kipp
Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061 USA
{*mgoncalv,fox,ltw,kipp*}*@cs.vt.edu*

Abstract

Digital libraries (DLs) are complex information systems and therefore demand formal foundations lest development efforts diverge and interoperability suffers. In this paper, we propose the fundamental abstractions of Streams, Structures, Spaces, Scenarios, and Societies (5S), which contribute to define digital libraries rigorously and usefully. 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 in order to accomplish a functional requirement. Societies comprehend entities and the relationships between and among them. Together these abstractions relate and unify concepts, among others, of digital objects, metadata, collections, and services required to formalize and elucidate "digital libraries". The applicability, versatility and unifying power of the theory is demonstrated through its use in three distinct applications: building and interpretation of a DL taxonomy, analysis of case studies of digital libraries, and utilization as a formal basis for a DL description language.

Keywords: digital libraries, theory, foundations, definitions, applications

1 Motivation

Digital libraries are extremely complex information systems. The proper concept of a digital library seems hard to completely understand and evades definitional consensus. Different views (*e.g.*, historical, technological) and perspectives (*e.g.*, from the library and information science, information retrieval, or human-computer interaction communities) have led to a myriad of differing definitions. Licklider, in his seminal work [68, pp.36–39], visualized a collection of digital versions of the worldwide corpus of published literature and its availability through interconnected computers. More recently, Levy and Marshall gave a view of digital libraries as a polygamy of documents, technology, and work [67]. Lesk analyzed the relative weights of the words *digital* and *library* in recent efforts in the field, and concluded that those efforts are dissociated from an understanding of users' needs and their use of the resources being provided [66]. Borgman explicitly explored the competing visions of the digital library field, both from the research and from the practitioner communities, and showed the difficulty that this conflict imposes on activities like defining terms, characterizing terminologies, and establishing contexts [14]. A Delphi study of digital libraries coalesced a broad definition: organized collection of resources, mechanisms for browsing and searching, distributed networked environments, and sets of services objectified to meet users' needs [58]. The President's Information Technology Advisory Committee (PITAC) Panel on Digital Libraries discusses "digital libraries – the networked collections of digital text, documents, images, sounds, scientific data, and software that are the core of today's Internet and tomorrow's universally accessible digital repositories of all human knowledge" [91]. Underlying all of these is the consensus agreement that digital libraries are fundamentally complex.

Such complexity most probably is due to the inherently interdisciplinary nature of this kind of system. Digital libraries integrate findings from disciplines such as hypertext, information retrieval, multimedia services, database management, and human-computer interaction [29]. The need to accommodate all these characteristics complicates the understanding of the underlying concepts and functionalities of digital libraries, thus making it difficult and expensive to construct new digital library systems. Designers of digital libraries are most often library technical staff, with little to no formal training in software engineering, or computer scientists with little background in the research findings about information retrieval or hypertext. Thus, digital library systems are usually built from scratch using home-grown architectures that do not benefit from digital library and software design experience. Wasted effort and poor interoperability can therefore ensue, raising the costs of digital libraries and risking the fluidity of information assets in the future.

The broad and deep requirements of digital libraries demand new models and theories in order to understand better the complex interactions among its several components [40]. As evidence of this claim, the summary report of the Joint NSF-European Union (EU) Working Groups on Future Directions of Digital Libraries Research recommended that "new models and theories be developed in order to understand the complex interactions between the various components in a globally distributed digital library" [101]. However, though the necessity for such an underlying theory has long been perceived and advocated, little if any progress has been made towards a formal model or theory for digital libraries. Formal mathematical models strengthen common practice. Their lack leads to diverging efforts and has made interoperability one of the most important problems faced by the field. As a matter of fact, it is not surprising that most of the disciplines related to digital libraries have formal models that have steered them well: programming languages, relational databases, hypertext, multimedia, and information retrieval.

In this paper we introduce five formalisms—streams, structures, spaces, scenarios, and societies (5S)— as a framework for providing theoretical and practical unification of digital libraries. These formalisms are important for making sense of complexity and can ultimately serve as an aid for designers, implementers, and evaluators of digital libraries. These abstractions work with other known and derived definitions to yield a formal, rigorous model of digital libraries.

This paper is organized as follows. Section 2 presents several formal models that have been developed for different kinds of information systems, setting a context for section 3 which really presents the five "S" abstractions and defines them formally. Section 4 then builds on this framework to formally describe digital libraries concepts. Section 5 discusses applications of the model and section 6 concludes the paper.

2 Formal Models for Information Systems

Formal models and theories are crucial to specify and understand clearly and unambiguously the characteristics, structure, and behavior of complex information systems. A formal model abstracts the general characteristics and common features of a set of systems developed for similar problems, and explains their structures and processes. Furthermore, formal models for information systems can be used as a tool for the design of a real system while providing a precise specification of requirements against which the implementation can be compared for correctness. Thus, most of the current classes of information systems have some underlying formal model.

2.1 Databases

The relational model is the most established formal model for databases. In this model, entities of the real world are modeled as fixed sequences of attribute values, called relations, which are actually subsets of the Cartesian product of the domain set of each attribute [20]. Normalization encompasses a set of operations designed to divide complex relations into ones with simple domains. A specific algebra (called relational algebra [20]) describes operations in this model; the relational calculus provides an alternative formulation [112].

Object-oriented databases have become the focus of much current database research and development efforts due to the limitations of the relational model regarding more complex applications, like multimedia [18] and geographic information systems [84, 83]. One of the first developments of a formal framework for object databases explicitly divided them into a structural object model and a behavioral layer [10]. The structural model is described as a directed graph whose nodes can be simple values or other abstract objects. Higher-order logic constructs describe the behaviors of the classes, methods, and inheritance [10]. The ODMG [17] specifications represent the current standard for this kind of database. It includes a system of types for tuples and arbitrary collections (like sets, lists, and arrays) as well as persistent roots or OIDs to the objects in the database.

Self-describing data formats, which are the basis for semi-structured database models, also have taken the stage. An example is the Object Exchange Model (OEM), developed in the context of the TSIMMIS project at Stanford for integrating heterogeneous data sources, an extension of the ODMG model [2]. Objects in OEM are represented by a finite labeled graph whose edges connect atomic and complex objects that are uniquely identified. The value of a complex object is a set of identifiers (oids) of other objects whereas the value of atomic objects is an atomic value of its type.

Essential to database architectures is the definition of a view mechanism, which provides logical independence. In the relational world, those are formalized as virtual relations defined by queries. In the object oriented context, very recently Guerrini *et al.* [49] provided the first formalization of a view model for object oriented databases.

2.2 Information Retrieval

Formal models for information retrieval (IR) have arisen since the mid-sixties to undergird a primarily empirical approach [54]. The three classic models in information retrieval are called boolean, vector, and probabilistic.

The boolean model is set theoretic and is the basis for many IR systems. Documents are seen as sets of terms (e.g., keywords, phrases). A query is represented as a logical expression built on terms and some combination of the logical operators AND, OR, and NOT. Searching is carried out by returning documents that have combinations of terms satisfying the logical constraints of the query, with supplementary facilities that allow proximity and truncation searching. The boolean model, in its pure form, has inherent limitations for searching textual documents: boolean queries are notoriously difficult to write and modify; allow little control over the size of the result set; and more importantly disallow the production of ranked output [97].

In the vector space model, terms (or concepts or features) used to index documents are considered as coordinates of a multi-dimensional space [98]. Documents and queries are represented by vectors in that space where the i-th element of the vector denotes the value of the i-th term, with the precise value of each such element being determined by some term weighting scheme. Retrieval involves ranking the document vector space with respect to the query (which has been located in the space as a result of indexing) based on some similarity function such as the cosine measure between vectors.

Probabilistic methods [92, 111, 124] attempt to model the IR universe within a probabilistic framework based on the following assumption: given a query q and a document d in the collection, estimate the probability that the user will find the document d relevant. This estimation assumes that: 1) the probability of relevance depends only on the query and the document representation and 2) there is a well-defined partition of the collection between relevant and non-relevant documents, where the relevant are those documents which the user prefers as the answer set to the query q. That ideal answer set of relevant documents should maximize the overall probability of relevance to the user [93]. These models rely on probabilistic and statistical notions over user's provided relevance information as well as on inference network concepts to combine different kinds of information to derive that ranking.

In a previous attempt to provide a unified theory for some of these approaches, Tague *et al.* presented a formal model for IR systems which took into account most of the aspects of the previous models and several other components derivated from a categorization of IR systems.

The model was based on context-free grammars and hypergraphs with extended representations for indexing, ranking, and navigating [110]. More recently, Baeza-Yates and Ribeiro-Neto [7] gave a formal characterization of IR models in terms of logical views (or representations) of documents and user queries, a framework for modeling these documents, queries and their relationships, and a ranking function which associates real numbers with queries and documents. Such a general model is used to explain the other various IR models that are seen as instantiations or specializations of the general model.

2.3 Hypertext and Multimedia

Hypertext systems provide the foundations for a complementary activity of retrieval called browsing, characterized by the lack of clearly defined objectives and whose purpose can change during the interaction with the system. Hypertext systems have increasingly incorporated multimedia content, leading thus to the concept of "hypermedia systems".

Multimedia and hypertext/hypermedia information systems also have received formal treatments. Lucarella and Zanzi present a formal graph-based object model of a multimedia database and describe both the underlying theory and the design and implementation of the Multimedia Object Retrieval Environment (MORE) system, illustrating the approach [70].

The Dexter model for hypertext abstracts the dynamic and run-time aspects of hypermedia systems as well as the data storage layer [50]. Nevertheless the Dexter model has been shown not to be sufficient to handle the so-called "hypermedia-in-the-large," required when systems are integrated and expanded through time [65]. The Dexter model also is not sufficient for multimedia, and was extended and implemented as the Amsterdam model by adding temporal logics, events, and activations [51]. These models are important to digital libraries because the DL itself, especially in "browse" mode, can be presented as a large hypertext of nodes that are the documents contained by the library.

2.4 Digital Libraries

Surprisingly, formal models for digital libraries are missed in the literature and one could conjecture that is due to the previously argued complexity of the field. Wang [116] provides one first attempt to fulfill this gap. His so-called "hybrid approach" defines a digital library as a combination of a special purpose database and a hypermedia-based user interface and builds upon this combination to formalize digital libraries in terms of the formal language Z [106]. Kalinichenko *et. al.* [56] presented a canonical model for information systems and a compositional approach which they applied to provide a partial solution for interoperability in DLs. These approaches, clearly incomplete, are, as far as we know, the only attempts to provide some comprehensive formalization for the digital libraries field.

3 The 5S Formal Framework

3.1 Mathematical Preliminaries

In this section, we briefly review the mathematical foundations necessary for the development of the following discussion. These concepts include sets, relations, functions, sequences, tuples, strings, graphs, and grammars [23].

Definition 1 *A **set** is an unordered collection of distinguishable entities, called its "elements" or "members."*

Formally, *set* and $\in$ ("element of") are taken as undefined terms in the axioms of set theory. The subtleties of these axioms are not important for the present development. We remark that a set cannot contain itself and the "set of all sets" does not exist. That x is an element of S is denoted $x \in S$, and there is an "empty" set ($\emptyset$).

The notation $S = \{x|P(x)\}$ defines a set S of precisely those objects x for which the logical proposition $P(x)$ is true. Standard operations between sets A and B include union: $A \cup B = \{x|x \in A \text{ or } x \in B\}$; intersection: $A \cap B = \{x|x \in A \text{ and } x \in B\}$; and Cartesian product: $A \times B = \{(a,b)|a \in A \text{ and } b \in B\}$ where (a,b) is called an *ordered pair.* A is called a *subset* of B, denoted by $A \subset B$, if $x \in A$ implies $x \in B$. The set of all subsets of set S (including $\emptyset$) exists, is called the *power set* of S, and is denoted 2^S.

Definition 2 *A binary **relation** R on sets A and B is a subset of $A \times B$. We sometimes write $(a,b) \in R$ as aRb. An n-ary relation R on sets $A_1, A_2, ..., A_n$ is a subset of the Cartesian product $A_1 \times A_2 \times ... \times A_n$.*

Definition 3 *Given two sets A and B, a **function** f is a binary relation on $A \times B$ such that for each $a \in A$ there exists $b \in B$ such that $(a,b) \in f$, and if $(a,b) \in f$ and $(a,c) \in f$ then*

$b = c$. The set A is called the domain of f and the set B is called the codomain of f. We write $f : A \rightarrow B$ and $b = f(a)$ as a common notation for $(a, b) \in f$. The set $\{f(a)|a \in A\}$ is called the range of f.

Definition 4 *A **sequence** is a function f whose domain is the set of natural numbers or some initial subset $\{1, 2, ..., n\}$ of the natural numbers and whose codomain is any set.*

Definition 5 *A **tuple** is a finite sequence that is often denoted by listing the range values of the function as $\langle f(1), f(2), ..., f(n)\rangle$.*

Definition 6 *A **string** is a finite sequence of characters or symbols drawn from a finite set with at least two elements, called an **alphabet**. A string is often denoted by concatenating range values without punctuation.*

Let Σ be an alphabet. Σ^* denotes the set of all strings from Σ, including the empty string (an empty sequence ϵ). A language is a subset of Σ^*.

Definition 7 *A **graph** G is a pair (V, E), where V is a nonempty set of vertices and E is a set of two-item sets of vertices, $\{u, v\}$, $u, v \in V$, called edges. A **directed graph** (or digraph) G is a pair (V, E), where V is a nonempty set of vertices (nodes) and E is a set of edges (arcs) where each edge is an ordered pair of distinct vertices (v_i, v_j), with $v_i, v_j \in V$ and $v_i \neq v_j$. The edge (v_i, v_j) is said to be **incident** on vertices v_i and v_j, in which case v_i is **adjacent to** v_j, and v_j is **adjacent from** v_i.*

Several additional concepts are associated with graphs. A **walk** in graph G is a sequence of not-necessarily distinct vertices such that for every adjacent pair v_i, v_{i+1}, $1 \leq i < n$, in the sequence, $(v_i, v_{i+1}) \in E$. We call v_1 the origin of the walk and v_n the terminus. If the edges of the walk are distinct, the walk is a **trail**. The **length** of the walk is the number of edges that it contains. If the vertices are distinct, the walk is a **path**. A walk is **closed** if $v_1 = v_n$ and the walk has positive length. A **cycle** is a closed walk where the origin and non-terminal vertices are distinct. A graph is **connected** if there is a path from any vertex to any other vertex in the graph. A graph is **acyclic** if it has no cycles. A **tree** is a connected, acyclic graph. A **directed tree** or (DAG) is a connected, directed graph where one vertex - called the root - is adjacent from no vertices and all other vertices are adjacent from exactly one vertex.

Definition 8 *A **context-free grammar** is a quadruple* (V, Σ, R, s_0) *where* V *is a finite set of variables called non-terminals,* Σ *is an alphabet of terminal symbols,* R *is a finite set of rules and* s_0 *is a distinguished element of* V *called the **start** symbol.*

A **rule**, also called a production, is an element of the set $V \times (V \cup \Sigma)^*$. Each production is of the form $A \to \alpha$ where A is a non-terminal and α is a string of symbols (terminals and/or non-terminals).

3.2 Streams

Streams are sequences of elements of an arbitrary type. In this sense, they can model both static content, as textual material, and dynamic content, as in a temporal presentation of a digital video or time and positional data (e.g., from a GPS) for a moving object.

A dynamic stream represents an information flow—a sequence of messages encoded by the sender and communicated using a transmission channel possibly distorted with noise, to a receiver whose goal is to reconstruct the sender's messages and interpret message semantics [103]. Dynamic streams facilitate communication in digital libraries, and are thus important for representing whatever communications take place in the digital library. Examples of dynamic streams and their applications include video-on-demand, filtering and routing of streams of news, and transmission of messages. Typically, a dynamic stream is understood through its temporal nature. A dynamic stream can then be interpreted as an finite sequence of clock times and values that can be used to define a stream algebra, allowing operations on diverse kinds of multimedia streams [71]. The synchronization of streams can be specified with Petri Nets [81] or other approaches.

In the static case, a stream corresponds to the information content of an entity and is interpreted as a sequence of basic elements, probably of a same type. Types of stream include text, video, and audio. The type of the stream defines its semantics and area of application. For example, statically thinking, any text representation can be seen as a stream of characters, so that text documents, like scientific articles and books can be considered as structured streams. Streams can be formally defined as below.

Definition 9 *A **stream** is a sequence whose codomain is a nonempty set.*

3.3 Structures

A structure specifies the way in which parts of a whole are arranged or organized. In digital libraries, structures can represent hypertexts, taxonomies, system connections, user relationships, containment, dataflows, and workflows, to cite a few. Books, for example can be structured logically into chapters, sections, subsections, and paragraphs; or physically into cover, pages, line groups (paragraphs), and lines [37]. Structuring orients readers within a document's information space. Indeed structured documents often rely on markup languages [36, 22, 43].

Relational and object-oriented databases impose strict structures (called *schemas*) on data, typically using tables or graphs as units of structuring [10]. Indexing in information retrieval systems by a manual or automated process serves not only to improve performance but also to cluster and/or classify documents to support future requests, generating an organizational structure for the document space.

With the increase of heterogeneity of material continually being added to digital libraries, we find that much of this material is "semistructured" or "unstructured". Such "semistructured data" refers to data that may have some structure, where the structure is not as rigid, regular, or complete as the structure used by structured documents or traditional database management systems [1]. Query languages and algorithms can extract structure from these data [2, 78, 73, 85, 44, 21]. Although most of those efforts have a "data-centric" view of semi-structured data, recent work with a more "document-centric view" have emerged [6, 35]. In general, human and natural language processing routines can expend considerable effort to unlock the interwoven structures found in texts at syntactic, semantic, pragmatic, and discourse levels. Here our definition of structure extends from that of graph or tree.

Definition 10 *A **structure** is a tuple $(G, L, \mathcal{F})$, where $G = (V, E)$ is a directed graph with vertex set V and edge set E, L is a set of label values, and $\mathcal{F}$ is a labeling function $\mathcal{F} : (V \cup E) \rightarrow L$.*

As a derivative of this definition, the next one follows.

Definition 11 *A **substructure** of a structure $(G, L, \mathcal{F})$ is another structure $(G', L', \mathcal{F}')$ where $G' = (V', E')$ is a subgraph of G, $L' \subseteq L$ and $\mathcal{F}' : (V' \cup E') \rightarrow L'$.*

3.4 Spaces

A space is any set of objects together with operations on those objects that obey certain rules. Despite the generality of this definition, spaces are extremely important mathematical constructs. The operations and rules associated with a space define its properties. For example, in mathematics, affine, linear, metric, and topological spaces define the basis for algebra and analysis [42]. In the context of digital libraries, Licklider discusses spaces for information [68, p.62]. In the information retrieval discipline, Salton and Lesk formulated an algebraic theory based on vector spaces and implemented it in the SMART System [98]. Spaces can be generalized into "feature spaces," sometimes used with image as well as document collections and suitable for clustering or probabilistic retrieval [93]. Document spaces are a key concept in those theories.

Human understanding is captured in conceptual spaces. Various spaces or subspaces can handle metadata like author and date, or relationships like citation-based links [30, 64, 16]. Multimedia systems must represent real as well as synthetic spaces in one or several dimensions, limited by some presentational space (windows, views, projections) and transformed to other spaces to facilitate processing such as compression [104, 128]. Many of the synthetic spaces represented in virtual reality systems are analogs to real spaces, or to information spaces of various types. Digital libraries may model traditional libraries by using virtual reality spaces or environments [9, 79]. Also spaces for computer-supported cooperative work provide a context for virtual meetings and collaborations [24, 87].

Again, spaces are distinguished by the operations on their elements. Digital libraries can use many types of spaces for indexing, visualizing, and other services that they perform. The most prominent of these for digital libraries are measurable spaces, measure spaces, probability spaces, vector spaces, and topological spaces. In the following we define formally these concepts of space.

Definition 12 *Let X be a set. A σ-**algebra** is a collection $\mathbb{B}$ of subsets of X that satisfies the following conditions:*

1. *every union of a countable collection of sets in $\mathbb{B}$ is again in $\mathbb{B}$, i.e., if $A_i \in \mathbb{B}$ ($i = 1, 2, 3, \ldots$), then $\bigcup_{i=1}^{\infty} A_i \in \mathbb{B}$;*
2. *if $A \in \mathbb{B}$, then $\tilde{A} \in \mathbb{B}$, where $\tilde{A}$ is the complement of A with respect to X.*

One consequence of the definition of σ-algebra is that the intersection of a countable collection of sets in $\mathbb{B}$ is again in $\mathbb{B}$.

Definition 13 *A **measurable space** is a tuple $(X, \mathbb{B})$ consisting of a set X and a σ-algebra $\mathbb{B}$ of subsets of X.*

A subset A of X is called *measurable* (or *measurable with respect to* $\mathbb{B}$) if $A \in \mathbb{B}$. A *measure* μ on measurable space $(X, \mathbb{B})$ is a nonnegative extended real-valued function defined for all sets of $\mathbb{B}$ such that the following conditions are satisfied:

1. $\mu(\emptyset) = 0$ where $\emptyset$ is the empty set, and
2. $\mu\left(\bigcup_{i=1}^{\infty} A_i\right) = \Sigma_{i=1}^{\infty}\mu(A_i)$ for any sequence A_i of pairwise disjoint measurable sets.

Definition 14 *A **measure space** $(X, \mathbb{B}, \mu)$ is a measurable space $(X, \mathbb{B})$, with measure μ defined on $\mathbb{B}$.*

Definition 15 *A **probability space** is a measure space $(X, \mathbb{B}, \mu)$, such that measure $\mu(X) = 1$.*

Definition 16 *A **vector space** is a set V of objects (vectors) together with a field S of "scalars" with an addition operation $+ : V \times V \to V$ and a multiplication operation $* : S \times V \to V$ such that if x, y, z are in V and α and β are in S then:*

1. *there is a unique vector $0 \in V$ such that $x + 0 = x$ for all $x \in V$ (additive identity);*
2. *for each vector $x \in V$ there exists a vector $-x \in V$ such that $x + (-x) = 0$ (additive inverse);*
3. *$(x + y) + z = x + (y + z)$ (associativity of $+$);*
4. *$x + y = y + x$ (commutativity of $+$);*
5. *$1 * x = x$ (identity);*
6. *$(\alpha * \beta) * x = \alpha * (\beta * x)$ (associativity of $*$);*
7. *$(\alpha + \beta) * x = \alpha * x + \beta * x$ (distributivity of $*$ over $+$, right); and*
8. *$\alpha * (x + y) = \alpha * x + \alpha * y$ (distributivity of $*$ over $+$, left).*

Vector spaces are the basis for a widely used information retrieval model, the Vector Space Model (VSM) [96]. In this model, a document space or set D consists of documents d_i, each identified by one or more index terms. In such a space, each document d_i is then represented by a t-dimensional vector $d_i = (d_{i1}, d_{i2}, ..., d_{it})$ where d_{ij} is the weight of the jth index term t_j. Furthermore, let d_i and d_j be any two documents in D. It is possible to use their representation vectors to compute the degree of similarity for the corresponding terms and term weights, that is, the similarity coefficient between the two documents, denoted as $s(d_i, d_j)$. One could use the inner product of the two vectors, or functions of the angle between the vector pairs. Also, by normalizing the vectors d_i and d_j, and projecting the vectors onto the unit sphere, the distance between two document points on the sphere can be inversely correlated with the similarity coefficient between two documents.

Definition 17 *A **topological space** is a pair $(X, \mathcal{T})$ consisting of a set X and a family $\mathcal{T} \subset 2^X$ of subsets of X such that:*

1. *$\emptyset \in \mathcal{T}$ and $X \in \mathcal{T}$;*
2. *for any collection of sets in $\mathcal{T}$, $\{A_i \in \mathcal{T} | i \in I\}$, $\cup_{i \in I} A_i$ is also in $\mathcal{T}$, and if I is finite, $\cap_{i \in I} A_i$ is in $\mathcal{T}$.*

$\mathcal{T}$ is said to be a topology for X, and elements of $\mathcal{T}$ are called **open** sets. The complement of an open set is called a **closed** set.

Vector spaces and measure spaces are often built on top of topological spaces, the latter being the more basic concept. Any use of the concept of distance implies an underlying metric space, which is a topological space whose open sets are defined by $\{y \mid d(x, y) < r\}$, where $d(x, y)$ is the distance between x and y.

Definition 18 *A **space** is a measurable space, measure space, probability space, vector space or a topological space.*

3.5 Scenarios

A scenario is a story that describes possible ways to use a system to accomplish some function that the user desires. Scenarios are useful as part of the process of designing information systems. Scenarios can be used to describe external system behavior from the user's point of view [59]; provide guidelines to build a cost-effective prototype [108]; or help to validate, infer and support requirements specifications and provide acceptance criteria for testing [53, 109, 63]. Developers can quickly grasp the potentials and complexities of digital libraries through scenarios. Scenarios tell what happens to the streams, in the spaces, and through the structures. Scenarios help us visualize the spaces, by setting up streams from views of structures. Thus, taken together the scenarios describe services, activities, tasks and operations and those ultimately specify the functionalities of a digital library.

For example, user scenarios describe one or more users engaged in some meaningful activity with an existing or envisioned system. This approach has been used as a design model for hypermedia applications [82]. Human information needs, and the processes of satisfying them in the context of digital libraries, are well suited to description with scenarios, including these key types: fact-finding, learning, gathering, and exploring [120]. Additionally, scenarios can aid understanding of how digital libraries affect organizations and societies, and how challenges to support social needs relate to underlying assumptions of digital libraries [67]. Scenarios also help us consider the complexities of current publishing methods, as well as how they may be reshaped in the era of digital libraries, considering publishing paths, associated participants, and publication functions [119].

The concepts of state and event are fundamental to understanding scenarios. Informally, a state is determined by what contents are in specified locations, as, for example, in a computer memory, disk storage, visualization, or the real world. An event denotes a transition or change between states, for example, executing a command in a program. Scenarios specify sequences of events, which involve actions that modify states of a computation and influence the occurrence and outcome of future events. From this it is easy to see how dataflow and workflow in digital libraries and elsewhere can be modeled using scenarios. Following, we formally define all those concepts.

Definition 19 *A **state** is a function $s : L \rightarrow V$, from labels L to values V. A **state set** S consists of a set of state functions $s : L \rightarrow V$.*

Thus $s_i(X)$ is the value, or the contents, of location X in state $s_i \in S$.

Definition 20 *A **transition event** (or simply **event**) on a state set S is an element $e = (s_i, s_j) \in (S \times S)$ of a binary relation on state set S that signifies the transition from one state to another. An event e is defined by a condition function $c(s_i)$ which evaluates a Boolean function in state s_i, and by an action function p.*

This transition event is not a *probabilistic* event [23]. Rather, it is more like the events in networked operating systems theory [105], transitions in finite state machines [25], those modeled by the Unified Modeling Language (UML) [13], or transitions between places in Petri Nets [81].

The condition is used to describe circumstances under which a state transition can take place. An action models a reference to an operator, command, subprogram or method, responsible to perform the actual state transition. Events and actions can have parameters that abstract data items associated with attributes (labels) of a state.

Definition 21 *A **scenario** is a sequence of related transition events $\langle e_1, e_2, ..., e_n \rangle$ on state set S such that $e_k = (s_k, s_{k+1})$, for $1 \leq k \leq n$.*

We also can interpret a scenario as a path in a directed graph $G = (S, \Sigma_e)$, where vertices correspond to states in the state set S and directed edges are equivalent to events in a set of events Σ_e (and correspond to transitions between states). (Technically, G must be a pseudodigraph [1], since loops (s_i, s_i) are possible as events).

Definition 22 *A **service**, **activity**, **task**, **procedure**, or **operation** is a set of scenarios.*

Note that the scenarios defining a service can have shared states. Such a set of related scenarios has been called a "scenario view" [53] and a "use case" in the UML [13]. In this framework, a simple transmission service of streams can be formally specified as:

Definition 23 *Let $T = \langle t_1, t_2, ..., t_n \rangle$ be a stream. Let S be the initial state set of the source before the transmission. Let D be the final state set of the destination after the transmission. Let s_{t_i} be the state in $S \cup D$ that indicates that the source is ready to transmit stream item t_i. Let d_{t_i} be the state in $S \cup D$ that indicates that the destination has just received stream item t_i. Let event $e_{t_i} = \langle s_{t_i}, d_{t_i} \rangle$. A **transmission** of stream T is the scenario (sequence of events) $e_T = \langle e_{t_1}, e_{t_2}, ...e_{t_n} \rangle$. (See Figure 1.)*

Scenarios are *implemented* to make a working system, and the so-called "specification-implementation" gap must be overcome [94]. Formally the implementation of scenarios can be seen in two complementary ways. It's easy to see how to map the definition of scenario as stated to an abstract machine represented by a deterministic finite automaton (DFA). This automaton $M = (Q, \Sigma_e, \delta, q_0, F)$ is such that M is the user-perceived conceptual state machine of the system and accepts a language $L(M)$ over the set of events Σ_e. A grammar $G = (V, \Sigma_e, R, s_0)$ for the language $L(M)$ is such that the non-terminals set V corresponds to the state set S, the terminals

[1] A digraph which permits both loops and multiple edges between nodes.

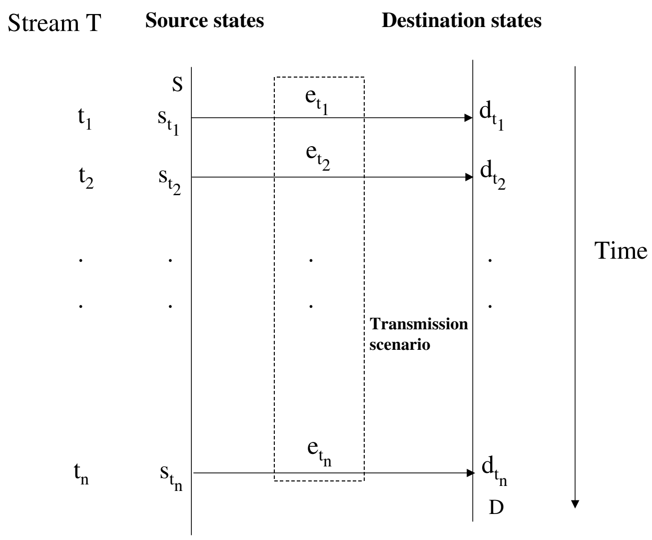

Figure 1: A scenario for the transmission of a stream

are the finite set of events Σ_e, s_0 is a distinguished initial state initializing all locations X, and R is a finite set of rules. Each rule in R is of the form $s_i \rightarrow es_j$ and conveys the system from state s_i to s_j as a consequence of event e, or is of the form $s_i \rightarrow e$ when $s_j \in F$ is a final state. The grammar and the corresponding conceptual state machine make up the abstract formal model which the analyst uses to capture, represent, and display system behavior in terms of scenarios.

Denotational semantics [121] offer a programming language perspective for the question. The implementation of a scenario can be seen as the specification of an ideal computer program. The program consists of expressions (e.g., Boolean, arithmetic) that will be evaluated and commands to be executed [121]. Expressions are evaluated with respect to a particular state producing values according to its type. The resulting values can influence the execution of commands, which will lead to change in state (or according to our terminology, produce events).

More formally, we can represent the situation of an arithmetic expression a waiting to be evaluated in a state s_i by the pair $\langle a, s_i \rangle$. We shall define an evaluation relation between such pairs and numbers $\langle a, s_i \rangle \rightarrow n$ meaning: expression a in state s_i evaluates to number n. The role of a command is to execute, to change the state. A pair $\langle c, s_i \rangle$ represents the (command) configuration from which it remains to execute command c from state s_i. The relation $\langle c, s_i \rangle \rightarrow s_j$ represents the full execution of command c in state s_i which terminates in state s_j.

In denotational semantics the *denotation* of a command or expression is defined as a partial function on states. Thus, for example, an arithmetic expression a denotes a function $A[[a]] : S \rightarrow R$, from a set of states to the set of real numbers. Similarly a boolean expression b will denote a function $B[[b]] : S \rightarrow T$ from the state set to the set of truth values. A command c denotes a partial function $C[[c]] : S \rightarrow S$. The brackets [[]] are traditional in denotational semantics. A is really a function from an arithmetic expression exp to the function $A[[exp]] : S \rightarrow R$. $A[[a]]$ is said to denote the expression a. The semantics of those functions is defined by structural induction. The denotation of an arithmetic sum, by structural induction, as a relation between states and numbers is as follows:

$A[[n]] = \{(s, n)|s \in S\}$
$A[[X]] = \{(s, s(X))|s \in S\}$
$A[[a_0 + a_1]] = \{(s, n_0 + n_1)|(s, n_0) \in A[[a_0]] \text{ and } (s, a_1) \in A[[a_1]]\}$

This structural induction on arithmetic sum shows that "+" is a function where the left-hand side represent syntactic signs whereas the signs on the right represent operations on numbers, e.g., for any state s,
$A[[3 + 5]]s = A[[3]]s + A[[5]]s = 3 + 5 = 8$.
For commands c we define a partial function $C[[c]]$ mapping c to a function $C[[c] : S \rightarrow S$ assuming that the definition of subcommands of c has been provided. The formal specification of a denotation for commands is outside of the scope of this paper, but the interested reader can find detailed examples in [121].

In sum, explicitly enumerating expressions and commands that will be associated with changes of state (or events) and defining a denotation for each one, we can formally express the implementation of scenarios. Those specifications can be formally validated and mapped to some programming language in a real computer. A similar approach which abstracts the process specification of a program behavior has been proposed in [69] to allow long term preservation of digital information.

3.6 Societies

A society is a set of entities and activities and the relationships between them. The entities are hardware, software, and wetware (humans) that are somehow related to the digital library. The activities are what the entities have done, do, and expect to do with each other. The relationships make connections between and among the entities and activities of the society.

Societies are necessary to describe the context of use of a digital library, since societies are the reason why libraries are built and maintained. In this sense, digital libraries are used for collecting, preserving and sharing information artifacts between society members. For example, digital libraries help to grow the relationship between library patrons (society members) and the information they seek.

A society is the highest-level component of the library, as a digital library exists to serve the information needs of its societies. *Cognitive Models* for Information Retrieval [11, 27, 15], for example, focus on user's information-seeking behavior (i.e., formation, nature, and properties of a user's information need) and on the ways in which IR systems are used in operational environments.

In digital libraries, specific human societies include patrons, authors, publishers, editors, maintainers, developers, and the library staff. There are also societies of learners and teachers. In a human society, people have roles, purposes, and relationships. Societies follow certain rules and their members play different roles—participants, managers, leaders, contributors, or users. Members have activities and relationships. During their activities, society members have created information artifacts—art, history, images, data—that can be managed by the library. Societies are holistic—substantially more than the sums of their constituents and the relationships between them.

Several societal issues arise when we consider them in the digital library context. These include policies for information use, reuse, privacy, ownership, intellectual property rights, access management, security, etc. [91]. Therefore, societal governance (law and its enforcement) is a fundamental concern in digital libraries. Language barriers are also an essential concern in information systems and internationalization of online materials is a big issue in digital libraries, given their globally distributed nature [80].

Economics, a critical societal concern, is also key for digital libraries [55]. Collections that were "born electronic" are cheaper to house and maintain, while scanning paper documents to be used online can be prohibitively expensive. Internet access is widely available and is inexpensive. Online materials are seeing more use, including from exceedingly remote locations. With circulation costs of electronic materials very low, digital delivery makes sense. However, it brings the problem of long-term storage and preservation such that the myriad of information now being produced can be accessible to future generations [69].

Modeling a society to the extent that it can be predicted reliably is not possible, due to its sensitivity to its inputs that gives it its chaotic nature. However, understanding certain aspects of societies as sets of entities, activities, and relationships can be beneficial.

In sum, a society is composed of individuals. Individuals have an intrinsic identity and are grouped into communities indirectly by way of descriptions that apply to all members of a community. Individuals are related to each other through relationships. Relationships specify interconnections and communications among individuals. We formalize those concepts below.

Definition 24 *A **society** is a tuple (C, R), where*

1. $C = \{c_1, c_2, ..., c_n\}$ *is a set of conceptual communities, each community refering to a set of individuals of the same class or type (e.g., actors, activities, components, hardware, software, data);*
2. $R = \{r_1, r_2, ..., r_m\}$ *is a set of relationships, each relationship being a tuple* $r_j = (e_j, i_j)$, *where* e_j *is a Cartesian product* $c_{k_1} \times c_{k_2} \times \cdots \times c_{k_{n_j}}$, $1 \leq k_1 < k_2 < \cdots < k_{n_j} \leq n$, *which specifies the communities involved in the relationship and* i_j *is an activity (cf. Definition 22) that describes the interactions or communications among individuals.*

4 Using 5S to formally define Digital Library

As pointed out in previous sections, there is no consensual definition of a digital library. This makes the task of formally defining this kind of application and its components extremely difficult. In this section, we approach this problem by constructively defining a "core" or a "minimal" digital library, i.e., the minimal set of components that make a digital library, without which, in our view, a system/application cannot be considered a digital library. Each component (e.g., collections, services) is formally defined in terms of a S-based construct or as combinations or compositions of two or more of them. The set-oriented and functional mathematical formal basis of 5S allows us to precisely define those components as functional compositions or set-based combinations of the formal Ss.

Informally, a digital library is a managed *collection* of information with associated *services* involving communities where information is stored in digital formats and accessible over a network [3]. Information in digital libraries is manifest in terms of *digital objects*, which can contain textual or multimedia content (e.g., images, audio, video), and metadata. Metadata have been informally defined as data about other data. Although the distinction between data and metadata often depends on the context, metadata commonly appears in a structured way and covering different categories of information about a digital object. The most common kind of metadata is *descriptive metadata*, which include catalogs, indexes and other summary information used to describe objects in a digital library. Another common characteristic of digital objects and metadata is the presence of some internal structure, which can be explicitly represented and explored to provide better DL services. Basic services provided by digital libraries are indexing, searching, and browsing. Those services can be tailored to the different communities depending on their roles, for example, creators of material, librarians, patrons, etc.

In the following we formally define those concepts of *metadata (structural and descriptive), digital objects, collection, catalog, repository, indexing, searching, and browsing services*, and finally *digital library.*

Definition 25 ***Structural metadata*** *is a structure.*

This simple definition emphasizes the role of structural metadata as a representation or abstraction of relationships between digital objects and their components' parts. The graph-based representation of this type of metadata can be explicitly expressed, as in the case of markup [22], or implicitly computed [74, 19].

Definition 26 *Let $\mathcal{R}$,$\mathcal{L}$, and $\mathcal{P}$ represent sets of labels for Resources, Literals, and Properties respectively.* ***Descriptive metadata*** *is a structure $(G, \mathcal{R} \cup \mathcal{L} \cup \mathcal{P}, \mathcal{F})$, where for each directed edge $e = (v_i, v_j)$ of G, $\mathcal{F}(v_i) \in \mathcal{R} \cup \mathcal{L}$, $\mathcal{F}(v_j) \in \mathcal{R} \cup \mathcal{L}$ and $\mathcal{F}(e) \in \mathcal{P}$. $\mathcal{F}(v_k) \in \mathcal{L}$ if and only if node v_k has outdegree 0. The triple $st = (\mathcal{F}(v_i), \mathcal{F}(e), \mathcal{F}(v_j))$ is called a* ***statement,*** *meaning that the resource or literal labeled $\mathcal{F}(v_i)$ has property $\mathcal{F}(e)$ with value $\mathcal{F}(v_j)$ (which can be designated as another resource or a literal).*

This definition, inspired by new developments in the metadata area [107, 118], emphasizes the semantic relationships implied by the labeling function in the structure.

Definition 27 *Given a structure $(G, L, \mathcal{F})$, $G = (V, E)$ and a stream S, a* ***StructuredStream*** *is a function $V \rightarrow (\mathbb{N} \times \mathbb{N})$ that associates each node $v_k \in V$ with a pair of natural numbers (a, b), $a < b$, corresponding to a contiguous subsequence $[S_a, S_b]$ (segment) of the Stream S.*

Therefore, a StructuredStream defines a mapping from nodes of a structure to segments of a stream. An example in a textual stream can be seen in Figure 2. From the example, it can be

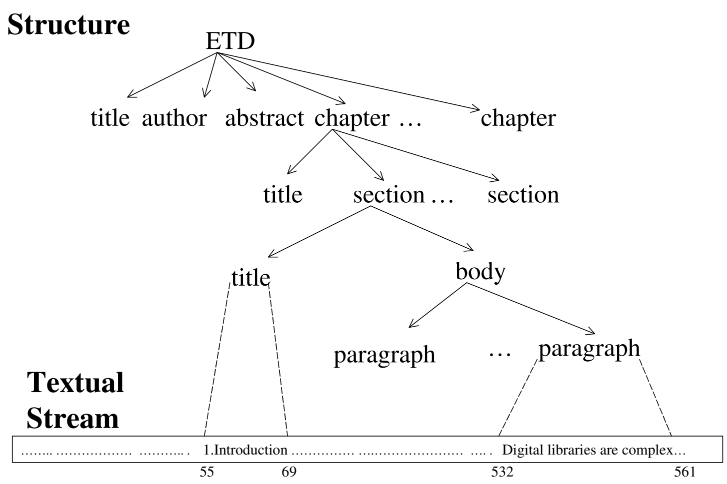

Figure 2: A StructuredStream for a Electronic Dissertation (adapted from [74])

deduced that several structures can be imposed over one stream and vice-versa. Also, it can be seen that segments associated with a node should include the segments of its children (in case of a hierarchical tree), although it is not equal to the union of those as "gaps" or "holes" can occur between child segments [74]. Finally, it should be noted that this definition works also for multimedia streams like audio, video, and images.

Definition 28 *A **digital object** is a tuple* $do = (h, SM, ST, StructuredStreams)$*, where*

1. $h \in H$*, where* H *is a set of universally unique handles (labels);*
2. $SM = \{sm_1, sm_2, \ldots, sm_n\}$ *is a set of streams;*
3. $ST = \{st_1, st_2, \ldots, st_m\}$ *is a set of structural metadata;*
4. $StructuredStreams = \{stsm_1, stsm_2, \ldots, stsm_p\}$ *is a set of StructuredStream functions defined from the streams in the* SM *set (the second component) of the digital object and from the structures in the* ST *set (the third component).*

Figure 3 shows an example of an oversimplified digital object with one structure and several streams. Two important aspects must be pointed out about this formal definition of a digital object:

1. Any real implementation does not need to enforce physical containment of the several

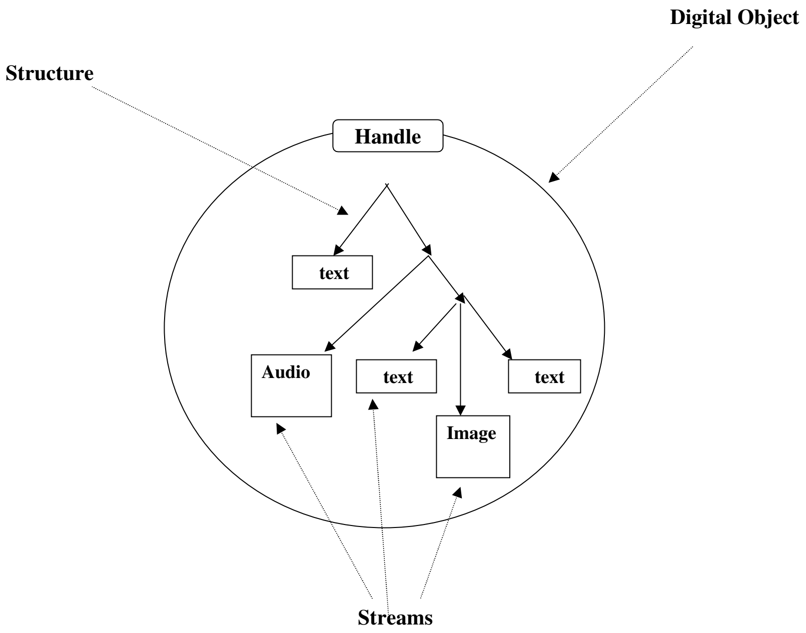

Figure 3: A simple digital object

component parts of a digital object; for example, we could have pointers to external streams.

2. The definition does not consider active behavior of a digital object (e.g., [62, 76, 77]) where operations, like different disseminations or exporting of subparts, are performed by external entities, like the user interface or the repository (see below). In spite of the fact that there is no explicit restriction regarding this, it does conform with our minimalist approach.

Definition 29 *A* ***collection*** $C = \{do_1, do_2, \ldots, do_k\}$ *is a set of digital objects.*

Definition 30 *Let C be a collection with handles H. A* ***metadata catalog*** *DM for C is a set of pairs* $\{(h, \{dm_1, \ldots, dm_{k_h}\})\}$*, where* $h \in H$ *and the* dm_i *are descriptive metadata.*

Definition 31 *Let C be a collection with handles H. A* ***repository*** *is a tuple* $(R, get, store, del)$*, where* $R \subset 2^C$ *is a family of collections (including* $\tilde{C}$*) and the functions "get", "store," and "del" satisfy:*

1. $get : H \to C$ *maps a handle h to a digital object get(h).*

2. $store : C \times R \rightarrow R$ maps $(do, \tilde{C})$ to the augmented collection $\{do\} \cup \tilde{C}$.

3. $del : H \times R \rightarrow R$ maps $(h, \tilde{C})$ to the smaller collection $\tilde{C} - \{get(h)\}$.

Thus a repository encapsulates a collection and specific services to manage and access the collection.

Definition 32 *An **index** $I : \mathcal{T} \rightarrow 2^H$ is a function where $\mathcal{T}$ is an indexing space over a set of indexing terms and H is a set of handles. An **indexing service** implements an index.*

The interpretation of indexing space is dependent upon which underlying space it is based. Terms of a indexing space can be words, phrases, concepts, or features appearing or associated with the content of a digital object (in their descriptive and structural metadata or streams). Normally, if a vector space is considered, terms are treated as unrelated, therefore defining orthogonal vectors that span a space with $rank = m$, where $m = |I|$. If a probabilistic space $p = (X, \mathbb{B}, \mu)$ is used, X is the set of distinct terms and is called a *sample space.* Also an index can be thought of as a mapping from an indexing space to a *document (digital object) space* defined by the collection, this latter being a topological space.

Definition 33 *Let Q be a set of logical representations for the user information needs, collectively called queries. Let $M_I : Q \times C \rightarrow \mathbb{R}$ be a matching function, defined by an index I, that associates a real number with a query $q \in Q$ and a digital object $do \in C$, indicating how well the query representation matches with the digital object, both structurally and by content. A **searching service** is a set of search scenarios $\{sc_1, sc_2, \ldots, sc_t\}$, where for each query $q \in Q$ there is a searching scenario $sc_k = \langle e_0, \ldots, e_n \rangle$ such that e_0 is the start event triggered by a query q and event e_n is the final event of returning the matching function values $M_I(q, d)$ for all $d \in C$.*

As event e_n, most probably just the documents with higher scores will be presented to the user in a list ordered by value.

Let $do_k(2)$ denote the stream set component of a digital object do_k, $do_k(3)$ its structural metadata set component, and $do_k(4)$ its set of StructuredStreams functions. Let also $G[v]$ denote the subgraph of a directed graph G containing v and all points and edges starting from the node v. Finally, let $f : A \rightarrow B$ and let $\mathcal{D}$ be any non-empty subset of A. The **restriction** of f to $\mathcal{D}$, denoted by $f|_{\mathcal{D}}$, is a subset of f and is a function from $\mathcal{D}$ to B.

Thus, for a collection C:

1. AllStreams $= (\cup_{do_k \in C} do_k(2))$ and *AllSubStreams* $= \cup_{sm_t \in AllStreams}\{sm_t[i,j] \mid sm_t = \langle a_0, a_1, \ldots, a_n\rangle, 0 \le i \le j \le n)\}$ will be the set of all streams and substreams (segments of streams) of all digital objects in the collection C;

2. *AllSubStructuredStreams* $= \bigcup_{k,j}(SubStructuredStream_{k_j})$ where:

   (a) $d_k \in C$;

   (b) $G_{k_j} = (V_{k_j}, E_{k_j})$ is the first component of some structure $st_{k_j} \in d_k(3)$;

   (c) $\mathcal{H}_{k_j} = \{G_{k_j}[v_t]\}$ is the set of all substructures of st_{k_j} where $v_t \in V_{k_j}$;

   (d) $SubStructuredStream_{k_j} = \{\mathcal{S}|_{V'} \mid (V', E') \in \mathcal{H}_{k_j}, \mathcal{S} \in do_k(4)$ is a StructuredStream function defined from the structure $st_{k_j}\}$.

   Therefore, *AllSubStructuredStreams* corresponds to the set of all possible substructures and their corresponding connections to streams inside digital objects of the collection.

Definition 34 *Let $H = ((V_H, E_H), L_H, \mathcal{F}_H)$ be a structure and C be a collection. A **hypertext** $HT = (H, Contents, \mathcal{P})$ is a triple such that:*

1. *$Contents \subseteq C \cup AllSubStreams \cup AllSubStructuredStreams$ is a set of contents that can include digital objects of a collection C, all of their streams (and substreams) and all possible restrictions of the StructuredStream functions of digital objects.*

2. *$\mathcal{P} : V_H \rightarrow Contents$ is a function which associates a node of the hypertext with the node content.*

A hyperlink is an edge in the hypertext graph. Source nodes of a hyperlink are called "anchors" and are generally associated via function $\mathcal{P}$ with segments of streams. Also, in this definition, two basic types of hyperlinks can be identified: *structural* and *referential* [117]. Structural hyperlinks allow navigation inside internal structures and across streams of digital objects. Referential hyperlinks usually have their target nodes associated with external digital objects or their subcomponents.

Figure 4 illustrates the definition. The hypertext is made by structural hyperlinks that follow the structural metadata and external referential links. Links originate from segments of streams. Link targets for, respectively, links 1, 2, and 3, are an entire digital object, a portion of its StructuredStream function and one of its streams, in this case an image. An instance of this model is the Web. The Web is a structure where hypertext links connect nodes that can be associated with: 1) complete HTML pages that can be considered digital objects; 2) substructures of a

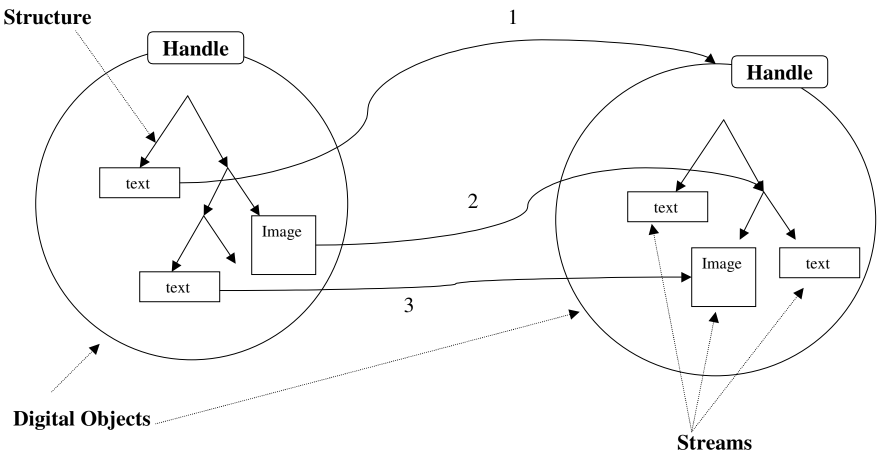

Figure 4: A simple hypertext

HTML page, for example, for a section of the page; and 3) links to streams, e.g., images, audios, or text. It should be noted that for the sake of brevity we are not describing here links to services, for example, external plugins that can be invoked by browsers or Web forms.

Definition 35 *A **browsing service** is a set of scenarios* $\{sc_1, \ldots, sc_n\}$ *over a hypertext (meaning events are defined by edges of the hypertext graph* (V_H, E_H)*), such that traverse link events* e_i *are associated with a function* $TraverseLink : V_H \times E_H \rightarrow Contents$*, which given a node and a link retrieves the content of the target node, i.e.,* $TraverseLink(v_k, e_{k_i}) = \mathcal{P}(v_t)$ *for* $e_{k_i} = (v_k, v_t) \in E_H$.

Therefore, by this definition, every browsing service is associated with an underlying hypertext construct. This view can for example unify the three modes of browsing defined by Baeza-Yates and Ribeiro-Neto [7]: flat browsing, structured guided, and navigational mode. The third one is the most general case and fits exactly our model. The first two can be considered special cases. In flat browsing the hypertext has a flat organization, for example, an ordered list of documents or a set of points in an image and the graph structure of the hypertext corresponds to a disconnected bipartite graph. In the second one, which includes classification hierarchies and directories, the hypertext graph is a tree. It is, for example, the work of many semi-structured wrapper algorithms to disclose this hypertext "hidden" structure in the Web. Once revealed, this structure can be stored in databases or represented in other semi-structured models to allow queries or transformations. Methodologies like PIPE [88] make use of this information to personalize Web sites. Note also that more sophisticated kinds of hypertext can be defined by extending the current definition. For example, we could relax the function $\mathcal{P}$ to be a relation and associate different contents with the same node, which could be achieved by having different modes of traversing the same link in an extension of the *TraverseLink* function. However, the present definition is simpler and serves well our minimalist approach.

Definition 36 *A* ***digital library*** *is a 4-tuple* $(R, DM, Serv, Soc)$*, where*

- *R is a repository;*
- *DM is a metadata catalog;*
- *Serv is a set of services containing at least services for indexing, searching, and browsing;*
- *Soc is a society of users of the digital library.*

5 Applications of 5S

In this section, we illustrate the expressiveness and unifying power of 5S as a theory for digital libraries through its use in three different kinds of applications. In the first one, we build a taxonomy of DL concepts derived from the literature and characterize the result in the light of the theory. The second kind of application uses 5S as an analytic tool to understand and dissect a DL instance and a DL protocol for interoperability. And finally, we present a brief description of a declarative language based on 5S for the specification and automatic generation of DL applications.

5.1 Digital Library Taxonomy

A taxonomy is a classification system of empirical entities with the goal of classifying cases according to their measured similarity on several variables [8]. Classifications are a premier descriptive tool and as such, they give a foundation towards an explanation for a phenomena. Classifications provide a terminology and vocabulary for the field and help to reduce complexity and achieve parsimony by logically arranging concepts through the identification of similarities and differences. We have built a taxonomy for digital libraries as a classification system of terms involved with the field. Our taxonomy describes the digital library field in conceptual terms and therefore its organization is amenable to be interpreted in the light of our 5S theory. This interpretation takes a more informal conceptual understanding of the 'Ss' and corresponding DL components to understand the resulting agglomerations of common concepts in the taxonomy.

In the process of building such a taxonomy, we have considered the principles of taxonomies in social sciences, notably cluster analysis, and the faceted classification schemes [113]. The presentation of the taxonomy also was influenced by the work of Saracevic and Kantor [99] in their taxonomy of *value* in libraries and information services. In particular we were guided by the idea that topics written about a subject unequivocally reveal the appropriate facets for that subject [28], and that those facets are enough to describe the phenomenon [89]. We followed an agglomerative strategy using subjective relational concepts like association and correlation. During the construction of the taxonomy we tried to accommodate all the terms found in the literature and marginal fields, guarantee mutual exclusivity, and ensure consistency and clarity.

To collect the unstructured list of concepts, we went through this literature to find all features, issues, and roles utilized and identified specific terms. In particular, we explored relevant contributions from the following literature sources:

- ACM DL conferences (1995-2000),
- ACM Transactions on Information System,
- Communications of the ACM (particularly 4/95, 4/98, 5/2001),
- D-Lib Magazine,
- European Conference on Digital Libraries (1997-2000),
- IEEE Computer DL Issue (4/97),
- IEEE-CS International Conference - Advances in Digital Libraries (1996-2000),
- Independent (Texas) DL Conferences 93, 94,
- International Journal on Digital Libraries (Springer),
- Journal of the American Society for Information Science (and Technology),
- Web in general.

As a starting point, we used a initial set of terms and phrases listed alphabetically in [32]. To this list we added other terms from the various articles. When this was reasonably voluminous, we produced a grouping of terms of similar or related meaning into "notational families" known as facets. Each group was given a label that described the idea behind the homogeneity of the group or the main variable considered. From there, we grouped the clusters, and so on, until we achieved convergence into one unique facet called "digital library."

Once the initial taxonomy was complete, we noticed certain terms were missing or ambiguous, so we added terms and qualified them in each context. After several iterations of successive clustering, declustering, and reclustering, we released a more concrete and agreed working set for peer review. The resulting taxonomy is shown in Figure 5.

We must point out that, as with any classification system, our taxonomy must evolve to accommodate changes in the digital library field. However, two factors contribute to the stability of the taxonomy, and therefore to its relative longevity. First the taxonomy was derived from a significant corpus of digital library literature; therefore it is more stable than personal opinions, for example. Second, the higher-level groupings are significantly abstract so that they may be applied to many fields, with possible additions or changes probably necessary only in the level of specific categories. As an undermining factor, it weights the youth and extremely fast development of the field. In the following we describe the main facets and sub-facets of the taxonomy, making use of 5S as an analytical tool. In particular, we discuss the key parts of Figure 5 informally in terms of the five "S"s and their combinations.

5.1.1 Actors: Who interacts with/within DLs?

Sets of *actors* that share a common behavior in terms of services and interactions constitute a community, the building blocks of a society. Communities—of students, teachers, librarians—interact with digital libraries and use digital libraries to interact, following predicted scenarios. Communities can act as a query-generator service, from the point of view of the library, and as a teaching, learning, and working service, from the point of view of other humans and organizations. Communications between actors and among the same and different communities occurs by sending and receiving streams. Communities of autonomous agents and computers also play roles in digital libraries. They act on the part of humans in the information society, performing scenarios upon our request. To operate, they need structures of vocabulary and protocols to confront various information and negotiation spaces. They act by sending (possibly structured) streams of queries and retrieving streams of results.

Actors
- *Humans*
  - Patrons
  - Learners
  - Readers
  - Annotators
  - Librarians
  - Authors
  - Publishers
  - Editors
  - Reviewers
  - Designers
  - Implementers
  - Tool builders
  - Evaluators
  - Maintainers
  - Managers
  - Funders
- *Electronic Agents*
  - Crawlers
  - Knowbots
  - Mediator agents
  - Collection agents
  - User agents
- *Distributed Computers*
  - Clients
  - Servers

Activities
- *Creating*
- *Collecting*
  - Submitting
  - Acquiring
    - Digitizing/OCRing
    - Purchasing/Licensing
- *Organizing*
  - Analyzing
  - Classifying (metadata)
  - Indexing
  - Annotating
    - Rating
    - Narrating
  - Clustering
- *Selecting*
  - Searching
  - Federating
  - Harvesting
  - Browsing
  - Navigating
  - Visualizing
- *Disseminating*
  - Providing access
  - Publicizing
    - Marketing
    - Advertising
  - Filtering
  - Translating
  - Collaborating
  - Using Email/Listservs
- *Requesting*
- *Preserving*
  - Renewing
  - Translating
  - Converting
  - Emulating
  - Conserving
- *Evaluating*
  - Leading focus groups
  - Doing sociological study
  - Doing action research
  - Surveying
  - Doing ethnographic study
  - Analyzing logs
  - Doing laboratory study
  - Measuring
- *Abstracting*
  - Cogitating (reflecting)
  - Relating
  - Comparing
  - Trusting
- *Modeling*
- *Personalizing*
  - Customizing

Components
- *Documents*
  - Structured documents
  - Semi-structured documents
  - E-Artifacts
  - Books, digital
  - Multimedia documents
  - Hypertext documents
  - Hypermedia documents
  - Diagrams
  - Photographs, digital
  - Maps, digital
  - Music, digital
  - Video, digital
  - Speech, digital
  - Database
  - Courseware
  - Versions
- *Repositories*
  - Knowledge
  - Information
  - Data
  - Facts
  - Logs
  - Relationships
    - Hyperlinks
  - Annotations
  - User profiles
- *Knowledge organization sources*
  - Term lists
    - Authority files
    - Glossaries
    - Gazetteers
    - Dictionaries
  - Classification/Categorization tools
    - Subject headings
    - Classification schemes
    - Taxonomies
  - Relationship groups
    - Thesauri
    - Semantic networks
    - Ontologies
  - Catalogs
    - Metadata records
    - Attributes
- *Clusters*
- *Handles*
  - OIDs
  - URI/URLs, PURLs, URNs
- *Substrate*
  - Communications
    - Protocols
    - Networks
    - Internet
    - Web
  - Modules
    - Relational databases
    - Object databases
    - Information retrieval systems
    - File Systems
    - Servers (e.g., audio, video)
    - Operating systems
    - Distributed systems
    - Multimedia systems
    - User interfaces

Socio-economic/Legal Issues
- *Policies*
  - Rights management
  - Intellectual property
  - Copyright clearances
  - Subscriptions
  - Privacy
  - Security
  - Billing/Charging
  - Pay-per-view
  - Multilingual access
  - Special-needs access
- *Standards*
  - Access
  - Description
  - Operation
  - Storage
  - Transmission
- *Qualities*
  - Accessibility
  - Interoperability
  - Maintainability
  - Scalability
  - Sustainability
  - Usability

Environment
- *Academic disciplines*
  - Anthropology
  - Business
    - Economics
    - Management
    - Marketing
    - Publishing
    - Copyright law
  - Computer science
    - Database theory
    - HCI
    - Hypertext
    - Information retrieval
    - Multimedia
    - Software engineering
  - Education
  - Ethnography
  - Engineering
  - Humanities
    - Museum science
  - Information science
  - Library science
  - Philosophy
  - Sociology
- *Purposes*
  - National library
  - Commercial service
  - Public services
  - Education support
  - Training
- *Scope*
  - Personal
  - Group
  - Department
  - Company/discipline
  - State
  - National
  - Worldwide

Figure 5: Taxonomy of Digital Libraries Terms

5.1.2 Activities: What happens in DLs?

Activities of digital libraries— abstracting, creating, collecting, disseminating, evaluating, modeling, organizing, preserving, personalizing, requesting, and selecting — all are services that follow scenarios. Furthermore, these activities make and characterize relationships within and between societies, streams, and structures. Each activity happens in a setting, arena, or space. The relationships developed can be seen under the optics of larger structures (e.g., social networks [102, 57]).

5.1.3 Components: What constitutes DLs?

Digital libraries can contain repositories of knowledge, information, data, metadata, relationships, logs, annotations, user profiles, and documents, all which can be interpreted as distinct forms of digital objects, according to their particular structures, metadata, and streams. They can contain structuring and organizational materials: term lists (e.g., authority files, dictionaries), classification tools (e.g., subject headings and taxonomies), thesauri, ontologies and catalogs. DLs are served by a substrate—a foundational complex amalgamation of different combinations of Ss that involves computers, network connections, file and operating systems, communications links, and protocols.

5.1.4 Socio-economic, Legal Aspects: What surrounds the DL?

This facet is mainly related to the societal aspects of the DL and their relationships and interactions, including regulations, measures and derivatives. It abstracts aspects surrounding the other DL issues and involves policies, economic issues, standards, and qualities. Most of those are generally established by normative structured documents. Policies and quality control can be enforced by specific services, for example, authentication, authorization [41], cryptography and specific practices (scenarios) or protocols, which can involve other communication services and serialized streams.

5.1.5 Environment: In what contexts are DLs embedded?

The environment is the space that defines the use and the context of a DL. It involves again the society that set up the DL, uses it, and keeps it going. But it is also how the DL fits into the structure of community and its organization and dictates the scenarios by which its activities are performed.

Academic Disciplines define a problem area "per se" and build a rational consensus of ideas and information about the problem that leads to a solution [100]. Thus they carve out a space for their approaches, structure some subject knowledge jointly with specific scenarios that define the methods or activities used to solve their specific problems. *Purposes* and *Scope* define the societies which the DL must serve and determine a specific structure of libraries that gives particular scenarios for those users.

5.2 DL Case Studies with 5S

In the last section, 5S was used to provide a better understanding of the DL field as a whole. The goals of this section are threefold: 1) to show the use of 5S as an analytical tool helpful to better comprehend specific DL phenomena; 2) to present the complex interplays that occur among 5S components and DL concepts in real DL application and that go beyond those described in our minimalist formal DL specification; and 3) to illustrate the possibility of using 5S as a instrument for requirements analysis in DL development.

5.2.1 The Networked Digital Library of Theses and Dissertations (NDLTD)

The Networked Digital Library of Theses and Dissertations (NDLTD) [86, 75, 33] is an international federation of universities, libraries, and other supporting institutions focused on efforts related to electronic theses and dissertations (ETDs). Many libraries and universities run their own programs and services, but there also are consortia activities at the state (e.g., OhioLINK), regional (e.g., Catalunya, Spain), and national (Australia, Germany, India, Portugal) levels. NDLTD allows institutions to cooperate and collaborate in a federated fashion, in a scalable and sustainable effort, especially since automation affords savings to both students and their universities relative to old paper-based approaches. As the distributed collection grows, and ultimately achieves critical mass, NDLTD has the potential to become one of the largest and most active digital libraries supporting education and research.

NDLTD Societies The primary society addressed through NDLTD is graduate students. The project aims to enhance graduate education, particularly of those students who prepare either a thesis or dissertation. Consequently, a second society is implicated, namely those involved in administering graduate programs. Those who are deans or associate deans of graduate schools, and their supervisors (e.g., associate provosts or associate chancellors) and staff, as well as the members of related associations (e.g., Council of Graduate Schools in USA, or the Canadian Association of Graduate Schools), are key members of this important society, that often decides if a university will join NDLTD. Because some universities have distributed these responsibilities to colleges or faculties, or because some involved in graduate program administration are too busy to carefully study NDLTD, we expanded this second society to include those in colleges or departments that administer graduate programs, allowing them to have their respective units join NDLTD prior to an action by the entire university. The third society related to NDLTD includes those involved in related activities in university libraries. This often involves the director or dean of the university library, as well as those involved in automation, support of multimedia development, training, cataloging, preservation, or other similar roles.

A fourth society involved in NDLTD is that of faculty. They may encourage students to start early to experiment with electronic theses and dissertations (ETDs), and to prepare expressive works, using multimedia. They may assist by providing tools in their laboratories that help with production of an ETD. They may guide students to produce high-quality works; that, in turn, may encourage and help large numbers of potentially interested readers. Faculty also assist students to grasp key issues regarding intellectual property and copyright, and to make their research results available to the widest community of readers possible given constraints relating to patents or publishers (see next paragraph).

The fifth, whose importance to the project became obvious early in 1997, is that of publishers. Though NDLTD was developed as a university effort, there is linkage with scholarly publishers because thesis and dissertation work often relates to other writings involving those students, such as conference papers, journal articles, and monographs. Because of copyright laws, and because of publisher policies that may force editors to make judgements regarding prior publication, this important society must be considered. In cases like ACM, IEEE-CS, and Elsevier, there is strong support, which has been highly beneficial.

NDLTD Scenarios/Services Each of the societies involved in NDLTD needs particular services from the digital library. They engage in various tasks and activities related to ETDs - each with corresponding scenarios. The NDLTD team has focused on training (through workshops, online materials, and help in media centers or library sites) to help students with the authoring or creation of ETDs. Next, there is the process of submission, supported by workflow software to help students enter and edit the metadata (including abstracts) about their ETDs. Staff in the graduate school and library also use other parts of the workflow software as they check, approve, archive, and catalog new ETDs. Library staff ensure that new works are added to the collection, and that the system affords access almost all the time. In terms of volume, the most active scenarios relate to use of the digital library. First, there are simple (running) and advanced (prototype) interfaces that support accessing individual university sites (searching or browsing), federated search across multiple sites and access to a union archive collection through the MARIAN [34, 46, 45] and the VTLS [114] digital library systems. There is experimental software to add annotation capabilities (the service selected as most important to add, based on focus groups to determine what other scenarios apply) [72]. There is also experimental software, extending the SIFT package [126] from Stanford University and a prototype in the MARIAN system, to provide filtering and routing services based on stored user profiles, for those who wish to be notified whenever an interesting ETD arrives. As time proceeds, our work in interoperability with other digital library software like Greenstone [122, 123], Phronesis [39], and Emerge [38] may allow us to support other universities that choose to use those packages to provide access services for their local ETDs.

NDLTD Spaces One space-related aspect of NDLTD is the physical location of members (a metric space) - now spread over parts of Africa, Asia, Australia, and Europe, as well as North, Central and South America. The Internet provides the name space of machines, while the WWW provides the name space of servers. Vocabulary used in different NDLTD services like searching relates to the conceptual space used in indexing. This will become more disciplined, as members use both some version of MARC, Dublin Core, or the new developed ETD-MS thesis and dissertations metadata standard [4], which is likely to provide the basic conceptual space for accessing the NDLTD collection. In addition, manual, semi-automatic, and automatic indexing and classification methods can be applied to place ETDs into conceptual spaces that relate to the Library of Congress or Dewey classifications, as well as discipline-specific thesauri (e.g., ACM's category system for computing) [48]. Another major space-related aspect of NDLTD deals with user interfaces. There are multiple graphical user interfaces that relate to our various software routines, including the ENVISION interface [52]. In addition, ongoing experimentation is investigating how the library metaphor applies to using our collection in our 10x10x10' CAVE (virtual reality environment) [79].

NDLTD Streams NDLTD deals with a variety of streams. At the simplest level are streams of characters for text, and streams of pixels for images. Some students have included audio files, or digital video, with their ETDs, which must be rendered as streams. These present challenges regarding quality of service if played back in real time, or alternative storage problems if downloaded and then played back from a local system. On the one hand, using standards like MPEG will make it easier to prolong the useful life of multimedia-rich ETDs, but on the other hand the representations that allow streaming of audio and video tend to be proprietary. This suggests that students probably should store both types of representation. The other class of streams related to NDLTD is that of network protocols. Those involve transmissions of serialized streams over the network. Efforts on federated search, harvesting and hybrid services, using a number of protocols, like Dienst, Z39.50, the Harvest system, and the Open Archives Metadata Harvesting Protocol have been developed in the context of NDLTD [46, 47, 45].

NDLTD Structures Structure plays many roles in NDLTD. A database management system is at the heart of the software for submission and workflow management developed at Virginia Tech. XML and SGML are ways to describe the structure of metadata, or of ETDs themselves. While only a small number of submissions at Virginia Tech have used such markup approaches, larger numbers are being collected in Germany. Structures in the form of *semantic networks* are used inside MARIAN to represent ETD collections and metadata and are explored in the provided services.

5.2.2 Open Archives Initiative

The Open Archives Initiative (OAI) [61, 26] is not a digital library by itself but a multi-institutional project to address interoperability of archives and digital libraries by defining simple protocols for the exchange of metadata. The current OAI technical infrastructure is defined by the Open Archives Metadata Harvesting Protocol, which defines mechanisms for archives to expose and export their metadata. In the following, this technical infrastructure is analyzed from the 5S point of view.

Open Archives Societies The main societies for which the OAI is designed are electronic, namely active agents called harvesters and repositories, which interact through the Open Archives Metadata Harvesting Protocol. The other two kinds of societies emphasized by the initiative are the so-called *data providers* and *service providers.* The former may be the manager of an e-print archive, acting on behalf of the authors submitting documents to the archive. The latter is a third party, creating end-user services based on data harvested from archives. At last, we have those communities constituted by the final users of the services and those involved with administrative aspects of repositories/archives.

Open Archives Streams The main streams associated to the OAI are dynamic and include communications between harvester agents and the repository server. Those communications are organized as *requests* from the agent to the server, which occur through specific verbs (see Open Archives Scenarios) embedded in HTTP requests, and *responses* that are textual metadata, which must be encoded and serialized in XML streams. The Open Archives Initiative so far has not considered multimedia streams, except when they are encoded in XML as part of the metadata.

Open Archives Structures Major structures of OAI are involved with *records*, *sets*, and *metadata formats*. OAI records can be considered containers [60], which encapsulate several kinds of descriptive metadata. Thus, OAI records obey a structure organized into:

- *Header*, which corresponds to information that is common to all records and includes a unique identifier and a datestamp – the date of creation, deletion, or latest date of modification of an item, the effect of which is a change in the metadata of a record disseminated from that item.
- A single manifestation of the metadata from an item. The OAI protocol supports multiple manifestations (structures) of metadata for any single item. At a minimum, repositories must be able to return records with metadata expressed in the Dublin Core format, without any qualification. Optionally, a repository also may be capable of disseminating other formats of metadata.
- *About*, an optional container to hold data about the metadata record itself, as opposed to the digital object associated with the metadata. Typically, this container is used to hold rights information regarding the metadata record, terms and conditions for usage, etc.

Sets are optional hierarchical structures for grouping items in a repository for the purpose of selective harvesting of records. Memberships of records in *sets* are not mandatory, but *sets* can share common records.

Registries, with data about various OAI-compliant repositories, also are provided. This allows users or harvesters or service providers to find suitable collections.

Open Archives Scenarios Regarding OAI repositories and the harvesting protocol, there is a fixed set of scenarios, namely those involved with requests and responses in the protocol conversations between harvesters and OAI archives. In a 5S analysis, we can associate each pair request-response with a different scenario, involving an interaction between harvester/repository. Thus, in the OAI Harvesting protocol there are scenarios for retrieving the identifiers of records in the repository restricted to specific Sets (ListIdentifiers verb); to retrieve a particular record given an identifier and metadata format (GetRecord verb); to retrieve information about the repository, including administrative information (Identify verb); and to list all supported metadata formats, records and sets in the repository (respectively, ListMetadataFormats, ListRecords, and ListSets verbs)

Another extremely important set of services, which is not part of the OAI technical specifications itself, but is essential to its functionality, is provided by a **mediation middleware**. This layer, which is placed between the repository and the OAI protocol itself, provides vertical communications, conversions, and translations from the OAI verbs and metadata organization to specific internal queries and operations on the underlying data representations of the repository. For example, if the repository is built upon a relational database, the mediation middleware is responsible for translating OAI requests to corresponding SQL queries.

Open Archives Spaces The OAI framework is naturally distributed along the physical space. Service providers can build indexing spaces on the top of metadata spaces, a kind of document space, and make use of vector or probabilistic spaces for building services like searching and filtering.

5.3 Declarative Generation of DLs

As a third application of the 5S framework, we have designed 5SL, 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 aids validation of implementations.

The structural organization and the semantics of 5SL constructs mimic directly the formalisms. Table 1 shows the basic XML syntax of the language while Table 2 details a portion of the DTD for the <structures> section.

```
<?xml version=‘‘1.0">
<!DOCTYPE 5SL PUBLIC>
<5SL xmlns='http://www.dlib.vt.edu/5SL/5SL.xsd'>
<streams>      ... </streams>
<structures>   ... </structures>
<spaces>      ... </spaces>
<services>    ... </services>
<societies>    ... </societies>
</5SL>
```

Table 1: 5SL basic XML syntax

```
<!ELEMENT structures (document, metadata,
                      classification?,relationship_groups?)>
<!ELEMENT metadata (descriptive,administrative?)>
<!ELEMENT classification(subject_headings?,classification_scheme?,taxonomy?>
<!ELEMENT relationship_group (thesaurus?,ontology?)>
```

Table 2: Portion of the DTD for the <structures> section

To improve acceptability and interoperability, 5SL tries to make an extensible use of existing standard specification sublanguages for representing DL concepts, when it turns out to be possible. That possibility is defined by the ability to formally map those standards and sublanguages to 5S formal specifications. Moreover, the need for the integration of multiple languages is a key aspect of the domain-specific language approach [5]. 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 pretty well. Therefore, we are using XML as a basic syntax, MIME types to encode streams, XML Schema [125] and RDF Schema [90] for describing respectively structural and descriptive metadata, RuleML [12] for representing ECA events in scenarios, and MathML [115] for spaces. The mapping of these sublanguages to our formal definitions is direct.

The general process of automatic creation of DLs and a particular application is shown in Figure 6. 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.

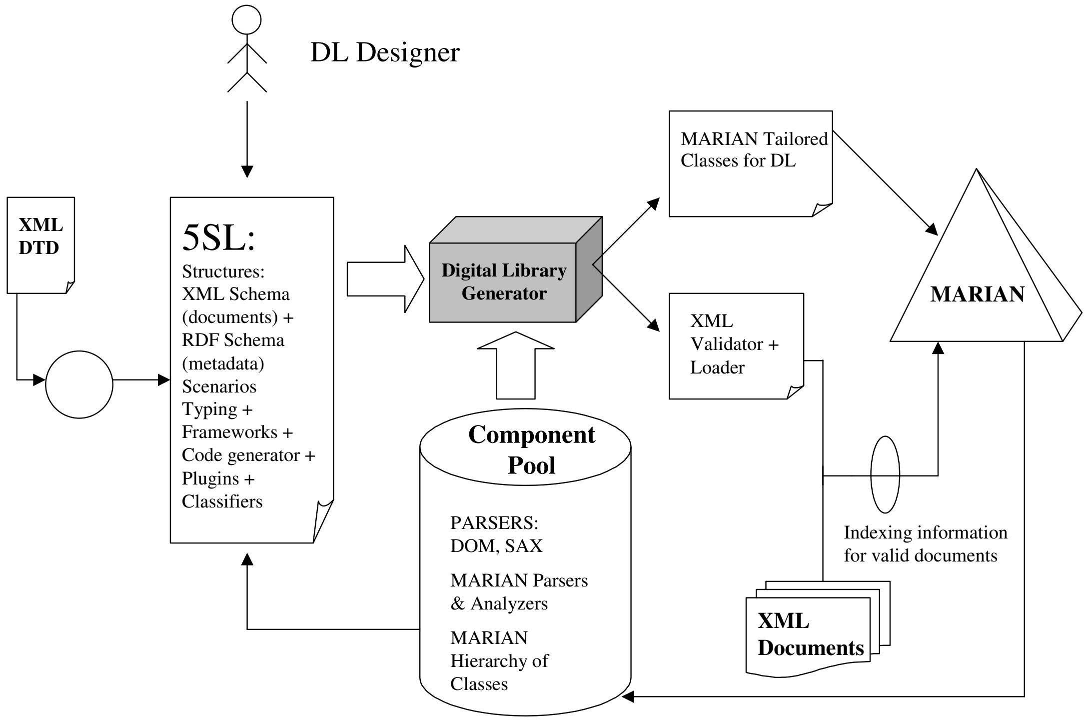

Figure 6: DL Generation process with 5S

We already have used 5SL for building XML-based digital libraries with IR-based services [48]. In one of those applications, a 5SL description for a NDLTD node was formulated according to the analysis performed in section 5.2.1. A specific DL generator targeted to the MARIAN digital library system also was developed. MARIAN is a multi-user information system developed at

Virginia Tech over the last decade, building upon our work in the 1980s with SMART [95] and CODER [31]. It is designed to support large numbers of simultaneous sessions of the sort commonly encountered in (digital) library environments: short sequences of often unrelated queries punctuated by browsing and examination of documents. MARIAN is built around a semantic network model improved with a hierarchy of classes, and a weighting schema, through which 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 is able to read 5SL specifications and generates two kinds of output: 1) a logical schema for the DL application, which in MARIAN corresponds to a set of Java classes that represent digital objects, their component parts, their metadata, and linking information for both structuring and indexing information and 2) an automatically generated Validator/Loader module responsible for checking incoming XML documents against specifications, extracting structuring and indexing information from valid documents, and invoking the corresponding classes methods in MARIAN that will materialize the structures and indexes defined in the 5SL schemata.

6 Conclusions

Motivated by the challenge of Licklider [68] to develop a theory for digital libraries, we have developed 5S. We show that formal definitions allow the 5S framework to be fully described and make it possible to clearly and formally define a minimal digital library. Using that framework we demonstrate its utility: to discuss the terminology found in the digital library literature, to describe a representative digital library and the Open Archives Initiative, and to construct 5SL – a declarative specification language from which digital libraries can be generated.

Future work with 5S framework will proceed in several directions. We will use our framework to help guide further development of the OAI and the NDLTD, as well as other digital library applications such as NSDL [127]. We will extend 5SL to be more complete, and to enable generation of personalized digital libraries in connection with PIPE [88, 48]. Further, we will encourage and assist others to adopt and adapt 5S and 5SL.

Finally, we plan to continue our work on the theory of digital libraries. We intend to use 5S to help with formal analysis of interoperability issues in digital libraries. The formal definitions given here can be used to prove helpful lemmas and theorems, and to guide future work in the field.

Acknowledgements

We are indebted to Robert K. France for the innumerous constructive discussions. The authors wish to thank the DLRL team members that assisted in this process. Anonymous reviewers of earlier drafts helped shape this work significantly. Errors and omissions are the sole responsibility of the authors.

References

[1] Serge Abiteboul, P. Buneman, and Dan Suciu. *Data on the Web - From Relations to Semi-structured Data and XML.* Morgan Kaufmann Publishers, San Francisco, 1999.

[2] Serge Abiteboul, D. Quass, J. McHugh, Jennifer Widom, and J. L. Wiener. The Lorel Query Language for Semistructured Data. *International Journal on Digital Libraries*, 1(1):5–19, April 1997.

[3] W. Arms. *Digital Libraries.* MIT Press, Cambridge, Massachussetts, 2000.

[4] Anthony Atkins. Interoperability metadata standard for electronic theses and dissertations. http://www.ndltd.org/standards/metadata/current.html, 2001.

[5] 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.

[6] R. Baeza-Yates and G. Navarro. XQL and Proximal Nodes. In *Proceedings of the ACM SIGIR 200 Workshop on XML and Information Retrieval*, Athens, Greece, 2000. ACM Press.

[7] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval.* Addison-Wesley-Longman, May 1999.

[8] K. D. Bayley. *Typologies and Taxonomies – An Introduction to Classification Techniques.* SAGE Publications, Thousand Oaks, California, 1994.

[9] Murat Bayraktar, Chang Zhang, Bharadwaj Vadapalli, Neill A. Kipp, and Edward A. Fox. A web art gallery. In *DL'98: Proceedings of the 3rd ACM International Conference on Digital Libraries*, pages 277–278, Pittsburgh, PA, 1998.

[10] Catriel Beeri. A formal approach to object-oriented databases. *IEEE Data and Knowledge Engineering*, 5:353–382, December 1990.

[11] N. J. Belkin, R. N. Oddy, and H. M. Brooks. ASK for information retrieval. *Journal of Documentation*, 33(2):61–71, June 1982.

[12] H. Boley. Rule markup language. http://www.dfki.uni-kl.de/ruleml/, 2001.

[13] Grady Booch. UML in action. *Communications of the ACM*, 42(10):26–28, October 1999.

[14] C. L. Borgman. What are digital libraries? competing visions. *Information Processing and Management*, 35(3):227–243, January 1999.

[15] P. Borlund and P. Ingwersen. The development of a method for the evaluation of interactive information retrieval systems. *Journal of Documentation*, 53(3):225–250, June 1997.

[16] Robert D. Cameron. A universal citation database as a catalyst for reform in scholarly communication. *First Monday*, 2(4), 1997.

[17] R. G. G. Cattell, T. Atwood, J. Dubl, G. Ferran, M. Loomis, and D. Wade. *The Object Database Standard: ODMG*. Morgan Kaufmann Publishers, Los Altos (CA), USA, 1994.

[18] Stavros Christodoulakis and Leonidas Koveos. Multimedia information systems: Issues and approaches. In *Modern Database Systems: The Object Model, Interoperability, and Beyond*, pages 318–337. ACM Press, New York, 1995.

[19] Charles L. Clarke, Gordon V. Cormack, and Forbes J. Burkowski. An algebra for structured text search and a framework for its implementation. *The Computer Journal*, 38:43–56, 1995.

[20] E. F. Codd. A relational model for large shared data banks. *Communications of the ACM*, 13(6):377–387, June 1970.

[21] World Wide Web Consortium. XML Query. http://www.w3.org/XML/Query, 2001.

[22] J. H. Coombs and A. H. Renear S. J. DeRose. Markup systems and the future of scholarly text processing. *Communications of the ACM*, 30(11):933–947, November 1988.

[23] Thomas H. Cormen, Charles Eric Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. MIT Press/McGraw-Hill, Cambridge, Massachusetts, 1990.

[24] Andy Crabtree, Michael B. Twidale, Jon O'Brien, and David M. Nichols. Talking in the library: implications for the design of digital libraries. In Robert B. Allen and Edie Rasmussen, editors, *Proceedings of the 2nd ACM International Conference on Digital Libraries*, pages 221–229, New York, July 23–26 1997. ACM Press.

[25] M. D. Davis, R. Sigal, and E. J. Weyuker. *Computation, Complexity, and Languages (second edition)*. Academic Press, 1994.

[26] Herbert Van de Sompel and Carl Lagoze. The Santa Fe Convention of the Open Archives Initiative. *D-Lib Magazine*, 6(2), February 15, 2000.

[27] D. Ellis. The physical and cognitive paradigms in information retrieval research. *Journal of Documentation*, 48:45–64, 1992.

[28] D. J. Foskett. Thesaurus. In *Encyclopedia of Library and Information Science - Volume 30*, pages 416–462. Marcel Dekker, New York, 1980.

[29] E. A. Fox and G. Marchionini. Toward a worldwide digital library. *Communications of the ACM*, 41(4):22–28, April 1998.

[30] Edward A. Fox. *Extending the Boolean and Vector Space Models of Information Retrieval with P-Norm Queries and Multiple Concept Types.* PhD thesis, Cornell University Department of Computer Science, 1983.

[31] Edward A. Fox. Development of the CODER system: a testbed for artificial intelligence methods in information retrieval. *Information Processing and Management*, 23(4):341–366, 1987.

[32] Edward A. Fox, Robert M. Akscyn, Richard K. Furuta, and John J. Leggett. Digital libraries. *Communications of the ACM*, 22-28(4):1022–1036, 1995.

[33] Edward A. Fox, John L. Eaton, Gail McMillan, Neill A. Kipp, Paul Mather, Tim McGonigle, William Schweiker, and Brian DeVane. Networked digital library of theses and dissertations: An international effort unlocking university resources. *D-Lib Magazine*, 3(9), 1997.

[34] Edward A. Fox, Robert K. France, Eskinder Sahle, Amjad Daoud, and Ben E. Cline. Development of a modern OPAC: From REVTOLC to MARIAN. In *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Interface Issues, pages 248–259, 1993.

[35] N. Fuhr. XIRQL - An Extension of XQL for Information Retrieval. In *Proceedings of the ACM SIGIR 200 Workshop on XML and Information Retrieval*, Athens, Greece, 2000. ACM Press.

[36] R. Furuta, V. Quint, and J. Andre. Interactively editing structured documents. *Electronic Publishing—Origination, Dissemination, and Design*, 1(1):19–44, April 1989.

[37] Richard Furuta. Defining and using structure in digital documents. In *Proceedings of the First Annual Conference on the Theory and Practice of Digital Libraries*, 1994.

[38] Joe Futrelle, Su-Shing Chen, and Kevin C. Chang. NBDL: A CIS Framework for NSDL. In *Proceedings of the First ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'2001)*, pages 124–125, Roanoke, Virginia, June 24-28 2001.

[39] David A. Garza-Salazar. Phronesis. http://copernico.mty.itesm.mx/ tempo/Proyectos/, 2001.

[40] H. Gladney, E. A. Fox, Z. Ahmed, R. Ashany, N. J. Belkin, and Maria Zemankova. Digital library: Gross Structure and Requirements: Report from a March 1994 Workshop. In *Proceedings of Digital Libraries '94: the First Annual Conference On the Theory and Practice of Digital Libraries*, pages 101–107, College Station, Texas, 1994.

[41] H. M. Gladney and A. Cantu. Authorization management for digital libraries. *Communications of the ACM*, 44(5):63–65, May 2001.

[42] R. Godement. *Algebra*. Kershaw Publ. Co. Ltd, London, 1969.

[43] Charles F. Goldfarb and Paul Prescod. *The XML Handbook*. Prentice-Hall PTR, Upper Saddle River, NJ 07458, USA, 1998.

[44] R. Goldman and J. Widom. Interactive query and search in semistructured databases. *Lecture Notes in Computer Science*, 1590:52–63, 1999.

[45] Marcos André Gonçalves, Robert K. France, and Edward A. Fox. MARIAN: Flexible Interoperability for Federated Digital Libraries. In *Proceedings of the 5th European Conference on Research and Advanced Technology for Digital Libraries*, Darmsdadt, Germany, 2001. Springer (to appear).

[46] Marcos André Gonçalves, Robert K. France, Edward A. Fox, and Tamas E. Doszkocs. MARIAN Searching and Querying across Heterogeneous Federated Digital Libraries. In *First DELOS Workshop: Information Seeking, Searching and Querying in Digital Libraries*, 2000.

[47] Marcos André Gonçalves, Robert K. France, Edward A. Fox, Eberhard R. Hilf, Michael Hohlfeld, Kerstin Zimmermann, and Thomas Severiens. Flexible interoperability in a federated digital library of theses and dissertations. In *Proceedings of the 20th World Conference on Open Learning and Distance Education, The Future of Learning - Learning for the Future: Shaping the Transition, ICDE2001*, April, 01–05 2001.

[48] Marcos André Gonçalves, Ali A. Zafer, Naren Ramakrishnan, and Edward A. Fox. Modeling and Building Personalized Digital Libraries with PIPE and 5SL. In *Proceedings of the*

Joint DELOS-NSF Workshop on Personalization and Recommender Systems in Digital Libraries, Dublin, Ireland, 2001.

[49] Giovanna Guerrini, Elisa Bertino, Barbara Catania, and Jesus Garcia-Molina. A formal model of views for object-oriented database systems. *Theory and Practice of Object Systems*, 3(3):157–183, 1997.

[50] F. Halasz and M. Schwartz. The Dexter Hypertext Reference Model. *Communications of the ACM*, 37(2):30, February 1994.

[51] Lynda Hardman, Dick C. A. Bulterman, and Guido van Rossum. The Amsterdam hypermedia model: adding time and context to the Dexter model. *Communications of the ACM*, 37(2):50–62, February 1994.

[52] Lenwood S. Heath, Deborah Hix, Lucy T. Nowell, William C. Wake, Guillermo A. Averboch, Eric Labow, Scott A. Guyer, Denis J. Brueni, Robert K. France, Kaushai Dalal, and Edward A. Fox. Envision: A user-centered database of computer science literature. *Communications of the ACM*, 38(4):52–53, April 1995.

[53] Pei Hsia, Jayarajan Samuel, Jerry Gao, David Kung, Yasufumi Toyoshima, and Cris Chen. Formal approach to scenario analysis. *IEEE Software*, 11(2):33–41, March 1994.

[54] Karen Sparck Jones and Peter Willett, editors. *Readings in Information Retrieval*. Multimedia Information and Systems. Morgan Kaufmann Publishers, 1997.

[55] B. Kahin and H. R. Varian. *Internet Publishing and Beyond: The Economics of Digital Information and Intellectual Property*. MIT Press, Cambridge, Massachusetts, 2000.

[56] L. A. Kalinichenko, D. O. Briukhov, N. A. Skvortsov, and V. N. Zakharov. Infrastructure of the subject mediating environment aiming at semantic interoperability of heterogeneous digital library collections. In *Proceedings of the 2nd Russian Scientific Conference on DIGITAL LIBRARIES: Advanced Methods and Technologies*, 2000.

[57] Henry Kautz, Bart Selman, and Mehul Shah. Referral Web: Combining social networks and collaborative filtering. *Communications of the ACM*, 40(3):63–65, March 1997.

[58] T. Kochtanek and K. K. Hein. Delphi study of digital libraries. *Information Processing and Management*, 35(3):245–254, 1999.

[59] M. Kying. Creating contexts for design. In *Scenario-Based Design: Envisioning Work and Technology in System Development*. John Wiley & Sons, New York , NY , USA, 1995.

[60] Carl Lagoze. The Warwick framework: A container architecture for diverse sets of metadata. *D-Lib Magazine*, 2(7), July 15, 1996.

[61] Carl Lagoze and Herbert Van de Sompel. The Open Archives Initiative. In *Proceedings of the First ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'2001)*, pages 54–62, Roanoke, Virginia, June 24-28 2001.

[62] Carl Lagoze, David Fielding, and Sandra Payette. Making global digital libraries work: Collection services, connectivity regions, and collection views. In Ian Witten, Rob Akscyn, and Frank M. Shipman, editors, *Proceedings of the 3rd ACM Conference on Digital Libraries (DL-98)*, pages 134–143, New York, June23–26 1998. ACM.

[63] A. V. Lamsweerde and L. Willemet. Inferring declarative requirements specifications from operational scenarios. *IEEE Transactions on Software Engineering*, 24(12):1089–1114, December 1998.

[64] Steve Lawrence, C. Lee Giles, and Kurt D. Bollacker. Digital libraries and autonomous citation indexing. *IEEE Computer*, 32(6):67–71, 1999.

[65] John J. Leggett and John L. Schnase. Viewing Dexter with open eyes. *Communications of the ACM*, 37(2):76–86, February 1994.

[66] M. Lesk. Expanding digital library research: Media, genre, place and subjects. In *Proceedings of the International Symposium on Digital Libraries 1999: ISDL'99*, Tsukuba, Ibaraki, Japan, September, 28-29 1991.

[67] D. M. Levy and C. C. Marshall. Going digital: a look at assumptions underlying digital libraries. *Communications of the ACM*, 38(8):77–84, April 1995.

[68] J. C. R. Licklider. *Libraries of the Future.* MIT Press, Cambridge, Massachusetts, 1965.

[69] Raymond A. Lorie. Long Term Preservation of Digital Information. In *Proceedings of the First ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'2001)*, pages 346–352, Roanoke, Virginia, June 24-28 2001.

[70] Dario Lucarella and Antonella Zanzi. A visual retrieval environment for hypermedia information systems. *ACM Transactions on Information Systems*, 14(1):3–29, January 1996.

[71] Wendy E. Mackay and Michel Beaudouin-Lafon. DIVA exploratory data analysis with multimedia streams. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI-98) : Making the Impossible Possible*, pages 416–423, New York, April 18–23 1998. ACM Press.

[72] Todd Miller. Annotation system for a collection of ETDs. http://www.ndltd.org/ndltd-sc/990416/annsystem.pdf, 1999.

[73] Ion Muslea, Steve Minton, and Craig Knoblock. A hierarchical approach to wrapper induction. In Oren Etzioni, Jörg P. Müller, and Jeffrey M. Bradshaw, editors, *Proceedings of the Third International Conference on Autonomous Agents (Agents'99)*, pages 190–197, Seattle, WA, USA, 1999. ACM Press.

[74] Gonzalo Navarro and Ricardo Baeza-Yates. Proximal nodes: A model to query document databases by content and structure. *ACM Transactions on Information Systems*, 15(4):400–435, 1997.

[75] NDLTD. Networked digital library of theses and dissertations. http://www.ndltd.org, 2001.

[76] Michael L. Nelson, Kurt Maly, Mohammad Zubair, and Stewart N. T. Shen. SODA: Smart objects, dumb archives. In S. Abiteboul and A.-M. Vercoustre, editors, *Proc. 3rd European Conf. Research and Advanced Technology for Digital Libraries, ECDL*, number 1696 in Lecture Notes in Computer Science, LNCS, Paris, France, September 1999. Springer-Verlag.

[77] Micheal L. Nelson and Kurt Maly. Buckets: Smart objects for digital libraries. *Communications of the ACM*, 44(5):60–61, May 2001.

[78] S. Nestorov, S. Abiteboul, and R. Motwani. Inferring structure in semistructured data. In *Proceedings of the Workshop on Management of Semi-structured Data*, 1997.

[79] Fernando Das Neves and E. A. Fox. A study of user behavior in an immersive virtual environment for digital libraries. In *Proceedings of the Fifth ACM Conference on Digital Libraries (ACM DL'2000)*, pages 103–112, San Antonio, TX, 2000.

[80] Douglas Oard, Carol Peters, Miguel Ruiz, Robert Frederking, Judith Klavans, and Praic Sheridan. Multilingual information discovery and accesS (MIDAS): A joint ACM DL'99 / ACM SIGIR'99 workshop. *D-Lib Magazine*, 5(10), October 15, 1999.

[81] Andreas Oberweis and Peter Sander. Information system behavior specification by high-level Petri nets. *ACM Transactions on Information Systems*, 14(4):380–420, October 1996.

[82] Ryuichi Ogawa, Hiroaki Harada, and Asao Kaneko. Scenario-based hypermedia: A model and a system. In *Proceedings of the ECHT'90 European Conference on Hypertext*, Toolkits for Hypermedia Applications, pages 38–51, 1990.

[83] J.L.d. Oliveira, Marcos André Gonçalves, and C.B. Medeiros. A framework for designing and implementing the user interface of a geographic digital library. *International Journal on Digital Libraries*, 2(3):190–206, 1999.

[84] Juliano Lopes Oliveira, Fatima Pires, and Claudia Bauzer Medeiros. An environment for modeling and design of geographic applications. *GeoInformatica*, 1(1):29–58, 1997.

[85] Yannis Papakonstantinou and Vasilis Vassalos. Query rewriting for semistructured data. In Alex Delis, Christos Faloutsos, and Shahram Ghandeharizadeh, editors, *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data: SIGMOD '99, Philadelphia, PA, USA, June 1–3, 1999*, volume 28(2), pages 455–466, New York, NY 10036, USA, 1999. ACM Press.

[86] C. Phanouriou, N. A. Kipp, O. Sornil, P. Mather, and E. A. Fox. A Digital Library for Authors: Recent Progress of the Networked Digital Library of Theses and Dissertations. In *Proceedings ACM Digital Libraries '99*, pages 20–27, Berkeley, CA, Aug 11-14 1999.

[87] Robert Prince, Jianwen Su, Hong Tang, and Yonggang Zhao. The design of an interactive online help desk in the Alexandria Digital Library. In Dimitrios Georgakopoulos, Wolfgang Prinz, and Alexander L. Wolf, editors, *Proceedings of the International Joint Conference on Work Activities and Collaboration: WACC '99*, pages 217–226, San Francisco, CA, 1999. ACM Press.

[88] N. Ramakrishnan. PIPE: Web Personalization By Partial Evaluation. *IEEE Internet Computing*, 4(6):21–31, 2000.

[89] S. R. Ranganathan. *A Descriptive Account of Colon Classification*. Bangalore: Sarada Ranganathan Endowment for Library Science, 1965.

[90] Resource Description Framework (RDF) Schema Specification 1.0. http://www.w3.org/TR/2000/CR-rdf-schema-20000327/, 2000.

[91] R. Reddy and I. Wladawsky-Berger. Digital Libraries: Universal Access to Human Knowledge - A Report to the President. President's Information Technology Advisory Committee (PITAC), Panel on Digital Libraries. http://www.itrd.gov/pubs/pitac/pitac-dl-9feb01.pdf, 2001.

[92] S. E. Robertson and K. Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27(3):129–146, May-June 1976.

[93] Stephen E. Robertson. The probability ranking principle in IR. *Journal of Documentation*, 33:294–304, 1977.

[94] Mary Beth Rosson and John M. Carroll. Object-oriented design from user scenarios. In *Proceedings of ACM CHI 96 Conference on Human Factors in Computing Systems*, pages 342–343, 1996.

[95] G. Salton and M. J. McGill. *The SMART and SIRE Experimental Retrieval Systems*. McGraw-Hill, New York, 1983.

[96] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, November 1975.

[97] Gerard Salton, Edward A. Fox, and Harry Wu. Extended Boolean information retrieval. *Communications of the ACM*, 26(12):1022–1036, 1983.

[98] Gerard Salton and Micheal E. Lesk. The SMART automatic document retrieval system—an illustration. *Communications of the ACM*, 8(6):391–398, 1965.

[99] T. Saracevic and P. B. Kantor. Studying the value of library and information services. Part II. Methodology and Taxonomy. *Journal of the American Society for Information Science*, 48(6):543–563, 1997.

[100] T. Sarasevic. Relevance: a review and a framework for thinking on the notion in information science. *Journal of the American Society for Information Science*, 26:321–343, 1975.

[101] P. Schauble and A. F. Smeaton. Summary report of the series of joint NSF-EU working groups on future directions for digital library research: An international research agenda for digital libraries. http://www.ercim.org/publication/ws-proceedings, October, 1998.

[102] M. F. Schwartz and D. C. M. Wood. Discovering shared interests using graph analysis. *Communications of the ACM*, 36(8):78, Aug. 1993.

[103] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, July, October 1948.

[104] Edleno Silva de Moura, Gonzalo Navarro, Nivio Ziviani, and Ricardo Baeza-Yates. Fast and flexible word searching on compressed text. *ACM Transactions on Information Systems*, 18(2):113–139, April 2000.

[105] M. Singhal and N. Shivaratri. *Advanced Concepts in Operating Systems: Distributed, Database, and Multiprocessor Operating Systems*. McGraw-Hill, New York, 1994.

[106] J. Spivey. *Introducing Z: A Specification Language and its Formal Semantics*. Cambridge University Press, 1988.

[107] W3C Standards. *Resource Description Framework (RDF) Model and Syntax Specification*, 1998. Available at http://www.w3.org/TR/WD-rdf-syntax/.

[108] A. Sutcliffe. A technique combination approach to requirements engineering. In *Proceedings of the Third International Symposium on Requirements Engineering*, pages 65–77, Annapolis, 1997. IEEE.

[109] A. G. Sutcliffe, N. A. M. Maiden, S. Minocha, and D. Manuel. Supporting scenario-based requirements engineering. *IEEE Transactions on Software Engineering*, 24(12):1072–1088, December 1998.

[110] J. Tague, A. Salminen, and C. McClellan. Complete formal model for information systems. In *Proceedings of the fourteenth annual international ACM/SIGIR conference on Research and development in information retrieval*, pages 14–20, Chicago, IL USA, October 13-16 1991.

[111] H. Turtle and W. B. Croft. Evaluation of an inference network-based retrieval model. *ACM Trans. on Inf. Sys.*, 9(3):187, 1991.

[112] Jeffrey D. Ullman. *Principles of Database and Knowledge-Bade Systems. Volume I: Classical Database Systems.* Computer Science Press, 1988.

[113] B. C. Vickery. Faceted classification schemes. In *Rutgers Series for the Intelectual Organization of Information – Volume 5.* Rutgers University Press, New Brunswick, NJ, USA, 1965.

[114] VTLS. VTLS. http://www.vtls.com, 2001.

[115] W3C. Mathematical Markup Language (MathML). http://www.w3.org/Math/, 2001.

[116] Bing Wang. A hybrid system approach for supporting digital libraries. *International Journal on Digital Libraries*, 2(2-3):91–110, 1999.

[117] W. Wang and R. Rada. Structured hypertext with domain semantics. *ACM Transactions on Information Systems*, 16(4):372–412, October 1998.

[118] Stuart Weibel. The Dublin Core Metadata Initiative. http://purl.oclc.org/dc/, 1998.

[119] Gio Wiederhold. Digital libraries, value, and productivity. *Communications of the ACM*, 38(4):85–96, April 1995.

[120] R. Wilkison and M. Fuller. Integration of information retrieval and hypertext via structure. In *Information Retrieval and Hypertext*, pages 257–271, Boston, USA, 1996. Kluwer Academic Publishers.

[121] Glynn Winskel. *The Formal Semantics of Programming Languages: An Introduction.* Foundations of Computing series. MIT Press, Cambridge, MA, USA, February 1993.

[122] Ian H. Witten, David Bainbridge, and Stefan Boddie. Greenstone: Open-source DL software. *Communications of the ACM*, 44(5):47, 2001.

[123] Ian H. Witten, Rodger J. McNab, Stefan J. Boddie, and David Bainbridge. Greenstone: A comprehensive open-source digital library software system. In *Proceedings of the Fifth ACM International Conference on Digital Libraries (ACM DL'2000)*, pages 113–121, San Antonio, TX, June 2-7 2000.

[124] S. K. M. Wong and Y. Y. Yao. On modeling information retrieval with probabilistic inference. *ACM Transactions on Information Systems*, 13(1):38–68, January 1995.

[125] XML Schema Part 0: Primer, W3C Working Draft. http://www.w3.org/TR/xmlschema-0, 2000.

[126] Tak W. Yan and Hector Garcia-Molina. The SIFT information dissemination system. *ACM Transactions on Database Systems*, 24(4):529–565, 1999.

[127] Lee L. Zia. The NSF National Science, Mathematics, Enginnering, and Technology Education Digital Library Program. *Communications of the ACM*, 44(5):83, May 2001.

[128] Nivio Ziviani, Edleno Silva de Moura, Gonzalo Navarro, and Ricardo Baeza-Yates. Compression: A key for next-generation text retrieval systems. *IEEE Computer*, 33(11):37–44, November 2000.