IU Digital Music Library Data Model Specification V2

FINAL RELEASE — September 18, 2003

Purpose

This document is intended to provide a shared understanding of the conceptual requirements and the data model as they exist in Variations2, version 2.

Overview

This document is broken into two major sections. The first section establishes the conceptual requirements for modeling data in Variations2. That is, the types of entities that need to be represented and the types of relationships between those entities. The second section of the document then elaborates these concepts, providing a precise list of records and fields in the data model.

Motivating Forces

In defining a model for any type of data, it is useful to keep in mind the forces that shaped the model.

Flexibility: It should be possible to quickly and easily locate all records that are related to a given record.

Collocation: The model should naturally place related records together as the result of searches.

Independence: Individual records viewed out of context should make as much sense as possible.

Minimization of Redundancy: Information should not be duplicated across multiple records.

Note that it is not possible to fully satisfy all of these forces at once. For example, completely independent records would contain much redundant information. While these tradeoffs are inevitable, each force has been taken into consideration in development of the model.

Conceptual Entities

At the highest level we deal with conceptual entities. The first cluster deals with the representation of music and media. Here we explore the abstract notions of a musical work as well as the concrete items on which it is manifested, such as CDs or MP3 files. The creation and manifestation of music, however, is not just at random, so we carefully provide for identifying the contributors associated with creating music as well its concrete containers.

---

[2] These are included to handle the situation where sources do not provide more precise date information. If a more precise range is given in the source, that should be used over the less precise form.

The second cluster of entities represents our need to provide and track access to interact with the music metadata and media as well as our need to restrict access. Users and groups of users are provided access to a service or sets of services. These services provide the means for interacting with the musical metadata and media. By identifying users and the services with which they are allowed to interact we provide a framework for the necessary control of access.

Music, Media, and Contributors

The following top-level entities represent the abstract and concrete forms of music.

Work: Represents the abstract concept of a musical piece or set of pieces.

Instantiation: Represents a manifestation of a work as a performance or a score.

Container: Represents the physical item or set of item(s) on which instantiations of works can be found.

Media Object: Represents a piece of digital media content, such as a sound file or score image.

As noted earlier, an important part of any of these concepts is attribution of the person/groups responsible.

Contributor: Represents persons or groups that contribute to a Work, Instantiation, or Container.

Descriptive Concerns

In order for the entities in the system to have any real meaning, there must be a mechanism for adequately describing an entity. Without adequate description the user will not be able to efficiently identify entities of interest when searching and browsing. That is, appropriate metadata is necessary for all entities to be distinguished from one another as well as to group them by their similarities. Common means of describing entities would include names or titles (e.g. Grand Funk Railroad), as well as dates associated with the creation or modification of the entity (e.g. date of composition for a musical work). Within the context of music we also have an entire array of musical style, genre, form, and period descriptors that can be applied.

Vocabularies

In order to deal with potential ambiguity (terms having multiple possible interpretations) and redundancy (multiple terms with the same interpretation), controlled vocabularies will be employed. These vocabularies constrain the set of possible descriptive values to remove ambiguity and redundancy that will in turn allow us to more easily deal with the data. These controlled vocabularies come in two basic forms: those that are sponsored/controlled by outside agencies (*e.g.* Uniform Names/Titles from Library of Congress Name Authority Records), and those that are created internally for the specific purpose of supporting our needs.

Work

- **Type (required):** Specifies the type of work represented (i.e. Collective or Single)
- **Uniform Title (required):** Specifies the uniform title taken from existing authority records or generated according to AACR2 rules.
- **Variant Title(s):** Alternative titles of the work.
- **Date of Composition**
- **Place of Composition**
- **Date of First Performance**
- **Place of First Performance**
- **Date of First Publication**
- **Original Text Title:** Complete title of the non-musical work text used in the musical work.
- **Language(s):** Language used in the work or associated creative materials (e.g. programs written by the composer).
- **Related Resource(s):** Resources that have content relevant to the study of the work.
- **Note:** Additional information about the work.

Within the Work entity there is also a need to categorize and describe the musical properties of the Work. It is important to keep in mind that actual manifestations of music are represented by Instantiations, and these manifestations may deviate somewhat from properties generally associated with the abstract Work. Descriptions of Work properties should be thought of as guidelines or defaults that may be deviated from in any particular Instantiation of a Work.

- **Class Number:** Appropriate class number for this work based on Library of Congress guidelines.
- **Subject Heading(s):** Library of Congress subject headings appropriate for this work.
- **Genre, Form and Style:** Descriptors identifying the work's genre, form and style attributes controlled via the Music Thesaurus.
- **Instrumentation:** Quantity and type of instrumentation/voices utilized.
- **Key**

Instantiation

- **Title (required):** Title as presented on the parent container
- **Language(s):** Language(s) utilized in this instantiation
- **Place of Performance:** For performed music, specifies the location of performance.
- **Date of Performance:** For performed music, specifies the date of performance.
- **Extent (required):** Extent (duration/length) of the instantiation.
- **Document Description (required):** Identifies the type and format of the document from which the instantiation is taken.
- **Notation:** For printed music, specifies the notation in which the instantiation appears. Must be one of the following values: Tablature, Modern Staff, or …
- **Note:** Additional information about the instantiation or instantiation record.

The Instantiation is the authority on the properties of the manifested musical piece. When creating Instantiation metadata, the metadata stored in the Work will be consulted, but portions may be removed, replaced, and/or augmented with properties particular to the Instantiation.

- **Class Number:** Appropriate class number for this instantiation based on Library of Congress guidelines.
- **Subject Heading(s):** Library of Congress subject headings appropriate for this work.
- **Genre, Form and Style:** Descriptors identifying the work's genre, form and style attributes controlled via the Music Thesaurus.
- **Instrumentation:** Quantity and type of instrumentation/voices utilized.

Container

- **Display Title (required):** Title presented on container to be used in displays.
- **Additional Title(s):** Additional title(s) presented on the container.
- **Date of Publication**
- **Place of Publication**
- **Language(s):** Language(s) utilized in the container.
- **Document Description(s) (required):** Identifies the document types and formats found in this container. As well as documents' physical description and any additional notes about the document.
- **Publisher (required):** Name of the publisher.
- **Publisher Label/ Plate Number(s):** Publisher label, plate numbers as they appear on the container.
- **Edition:** Number or other Identifier representing the particular edition of this container.
- **Uniform Series Title**
- **Provenance:** Known history of ownership for the container.
- **Note:** Additional information about the container.

Contributor

- **Type (required):** We identify two types of contributor: person or group.
- **Uniform/Preferred Name (required):** Full uniform name of the contributor. Formatting controlled by AACR2.
- **Variant Name(s):** Alternate names for the contributor.
- **Contributor Date(s):** Date range associated with the contributor (birth-death, flourished, etc.). End points of the range may not always be known, however. E.g. Date of death is known, but date of birth is not
- **Variant Date(s):** For some contributor dates there may be multiple opinions on the "true" value. This element stores alternative dates associated with the contributor.
- **Place of Origin:** Location of birth (for persons) or foundation (for groups).
- **Related Resource(s):** Resources that have content relevant to the study of the contributor.
- **Note:** Additional information about the contributor.

Structural Concerns

Both the Work and Container can be further described by specifying their structural make-up. Each can be thought of as having a hierarchical breakdown that can be useful both for navigation within that particular entity as well as allowing finer grained search than would be provided using entity level descriptive metadata alone.

Work

The structural declaration of a musical work involves identifying and properly nesting the sections found within the Work. Examples of sections might be

movements found in symphonies or acts and scenes in an opera. Work structure should allow for labeling of structural elements for display as well as optionally associating a searchable value.

Container

Within a particular Container there may be one or more physical items (e.g. CDs, LPs, score volumes) and within these physical items there may be additional sections (e.g. sides, tracks, pages). The structural elements can be labeled as in the Work structure.

Relationships

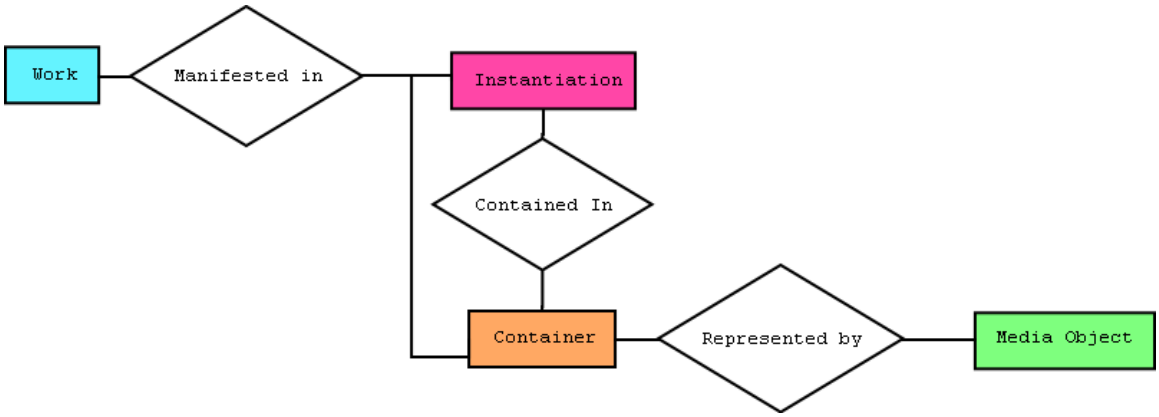
In addition to describing the entities themselves we also need to describe their relationship(s) to one another. We need to first indicate how entities will be identified and then the mechanisms by which those IDs can be referenced.

IDs

In order to locate and manipulate records stored within the system it is necessary to uniquely identify them to prevent ambiguity. The ID serves no other purpose than to provide a robust mechanism for identifying entities that is completely independent of the metadata that describes the entity.

Simple Relationships

The basic relationships create links from the fully abstract work to the actual entities stored in the digital library, digital media objects.



Contributions

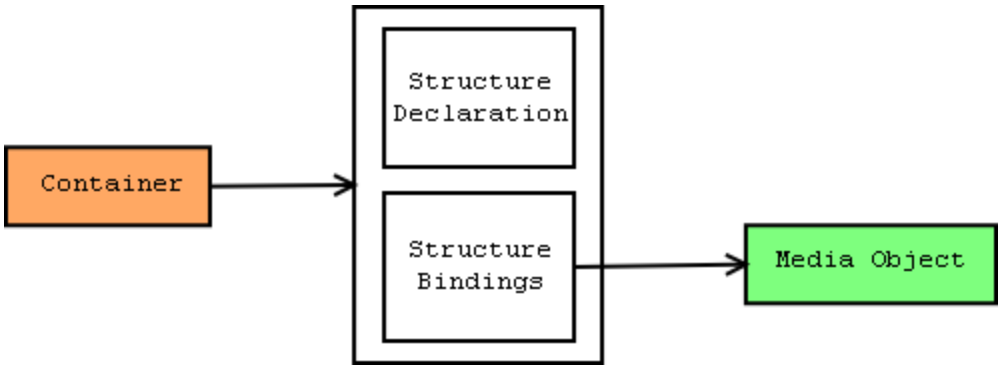
One of the most important relationships in our model is the relationship between a contributor and the target of his/her contribution. For our purposes it is insufficient to simply relate the Contributor and Work/Instantiation/Container entities. We require the ability to further describe the nature of the contribution that was made. To handle this we provide for the notion of Contribution, which can embody both the relationship as well as information about the type of the relationship.

Related Works

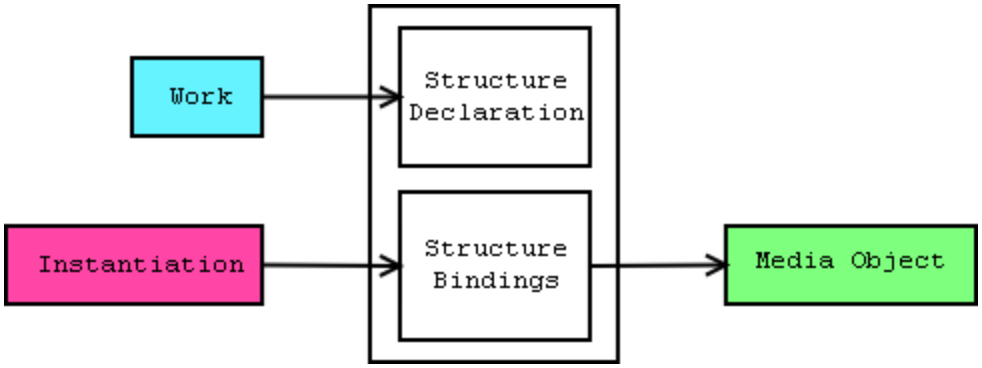
In order to model the relationships between works, we again need something slightly more sophisticated than a simple relationship tying the works together. The nature of the relationship between works may be of many types (e.g. derivative/original, collective/member, etc.) and this type needs to be stored with the relationship.

Structural Relationships

Both Container and Work entities have mechanisms to describe their own structure. In addition, however, there must be a binding or linking made between these structural descriptions and the media objects that contain relevant content. These bindings specify a mapping between the structure and particular offsets into relevant media objects. Container structure and bindings are located within the Container entity itself.



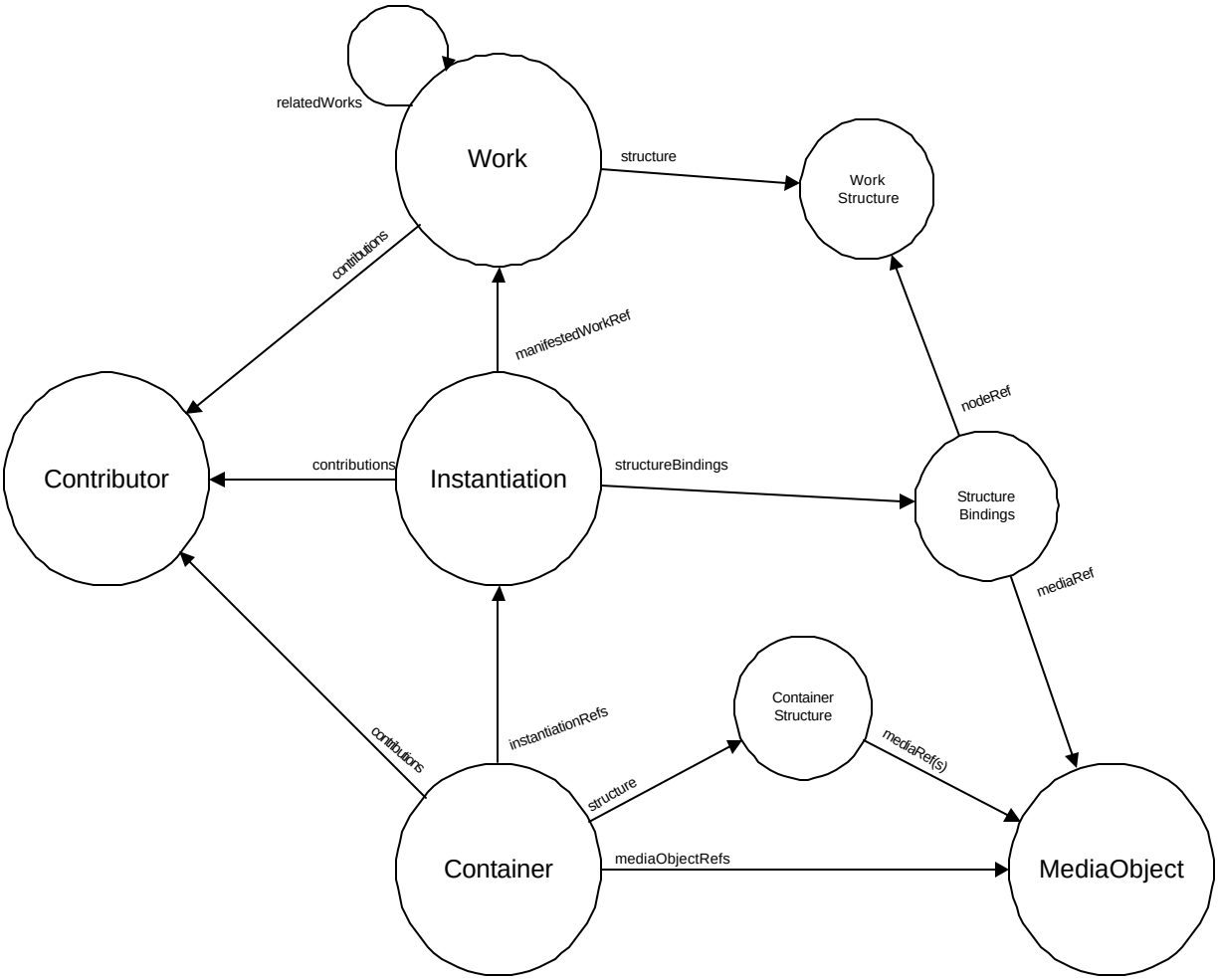
Work structure and bindings work in a similar way, but because there are many potential instantiations of a single work, the Work structure bindings must be stored in the Instantiation.



All Relationships

All relationships in the model are shown below. Large circles represent top-level records. Smaller circles represent records that are stored within the top-level records. Labels on the connection indicate the names of fields in which the

relationships are stored. When these field names are singular, they indicate a one-to-one relationship. Plural field names indicate a one-to-many relationship.



Administrative Concerns

Source and Vocabulary Tracking

In the process of cataloging entities, external sources will invariably be used for decisions on formatting (e.g. different types of subject headings may have different forms) as well as to indicate the original source of the data. The original source might be the physical item(s) represented by the container itself (e.g. for instantiation titles) or a more formal source such a particular authority record (local or otherwise). In any case, it is desirable to track these sources as the cataloging is carried out so there is always a trail back to the original source.

Digitized Media Information

In addition to just providing digital content, it is useful to store information about the nature of the digitized media as well as the process in which it was

generated. This area of metadata may become more sophisticated as we add additional versions of content.

- **File format:** MIME type for the file
- **Compression:** Was compression of any sort used? If so, what was the nature of the compression?
- **Bit Depth:** (images)
- **Resolution:** (images)
- **Number of Channels:** (sound)
- **Sample Size:** (sound)
- **Sample Rate:** (sound)
- **Digitization Information:** Date(s) and Names/IDs for digitization technicians responsible for creating the digital content.
- **Hardware/Software used:** Description of the hardware and software employed in the digitization process.

Each media object has the potential to store more than one actual media file. This is most commonly the case with images stored to represent scores. In any case we need a mechanism to identify and verify the integrity of the files associated with the media object.

- **Sequence Number:** Indicates where this file falls within the sequencing of the media object.
- **File location:** URL to locate the file.
- **File Size:** File size in kilobytes.
- **Checksum:** 128 bit MD5 checksum.

Linkage to Existing System(s)

There is a natural relationship between our Container entity and the MARC records stored in IUCAT or OCLC. Given that as much content as possible will be imported from current MARC records, its useful to keep references back to these original sources.

- **Other system IDs (control numbers):** OCLC and IUCAT.
- **Physical Identification:** Location, call number, and copy information associated with the original physical item(s) represented by the Container.

Workflow/Status Information

As the metadata representing entities is actually entered in records, it is important to keep track of the status of the record. In the simplest case the status possibilities would be “available for public access” and “not available for public access”. The system does however allow for a mechanism to add more fine grained status descriptors should these prove necessary/useful.

Interaction and Access Control

This category might be rightly called even more administrative issues. It is distinguished by its focus purely upon users, their interaction with the system, and how that interaction is regulated.

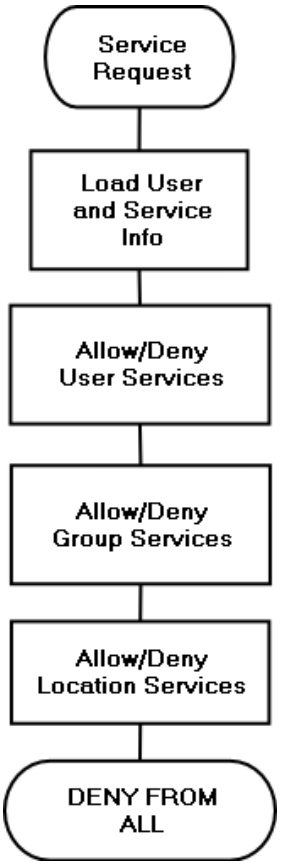
Users and Groups

Service: A record representing an access-controlled resource. Services include streamed audio content and images (controlled to the Container level), the metadata repository, directory services, and content management. Services regulate access by specifying which Groups are allowed and which Groups are denied access privileges.

User: A record representing a user of the DML system. User records contain references to Groups they are members of. Additionally Users may contain references to services to which they are allowed or denied access.

Group: Groups aggregate Users to make Service access control specifications easier to maintain.

IP List: Stores information to allow access based on the IP address of the client.



The authorization process is (conceptually) a three-step process. It is initiated with a request for access to a service in the system. The service request is a duple made up of a User and Service. A request to listen to a Container might look like:

{mjadud, “They Might Be Giants”},

Whereas a request to update the metadata repository with a new Work record might look like:

{mjadud, “Metadata Repository”}

The authorization process begins by locating the records for both the user (the User record) and the content the user is interested in (the Service record). Three steps follow to determine whether access will be granted.

1. If the User is specifically denied or allowed access to the Service in question, they are either permitted access or denied, and the process ends.
2. The group membership of both the User and Service are checked; if they are both members of the same group, access is permitted; otherwise, we progress on to step three.
3. If the IP address the user is connecting from is considered “privileged,” access is granted. Otherwise, access is denied, and the process ends.

Step three is primarily intended to give full access to all machines in the Music Library. While other uses might be found, it is not intended for broad application

Access Information

For each type of record that is stored within the system we wish to track activity using that record. The creation and last update are considered special activities that we especially wish to track so they are stored independently as well as being reflected in the generic, last access.

- **Creation Information:** Date, time, and ID of user that created the record.
- **Last Update Information:** Date, time, and ID of user that last updated the record.

Additionally for media objects useful information can be obtained by keeping track of the last access and number of accesses. This information can be an indicator of the utility/usage/popularity of the media object and may allow for optimization in later versions (e.g. store less used media objects on “slow”, cheap storage devices).

- **Last Access Information:** Date, time and ID of user that last accessed the record.
- **Number of Accesses:** Number of times the record has been accessed.

Copyright

As has often come up in discussions about restricting access to content, the desire to be able to make these access decisions by drawing upon information known about the copyright ownership suggests that we need to store information that can assist in making this determination.

- **Copyright Declaration(s):** Copyright date(s) and holder(s) associated with the work/instantiation/container. Also indicates the type or domain of the copyright declaration (e.g. Music, Text, Recording, etc.)
- **Public Domain Status:** Indicates whether this work is known to be in the public domain, known NOT to be in the public domain, or is unknown.

External Unique Identifiers

- **ISMN:** International Standard Music Number. References: http://www.ismn.spk-berlin.de/
- **MRC/EAN/UPC:** Media Catalog Number, European Article Number, and Universal Product Code. MRC and EAN have 13 numeric digits. UPC only has 12 and a leading zero. These numbers provide unique identifiers for the physical item(s) represented by the Container. References: http://www.ean-int.org/, http://www.uc-council.org/
- **ISBN:** For printed sources, specifies the International Standard Book Number provides a unique identifier.
- **Performing Rights Society Work Number:** By identifying the performing rights society and work number associated with a particular work we may be able to more easily track current copyright owners.
- **International Standard Recording Code:** The ISRC provides a mechanism for tracking particular recordings of works across publications.

Data Model

The discussion of the actual data model which will support the conceptual requirements stated above starts with a discussion of the common element types found in entities and how these may be implemented in a uniform way. By doing so we reduce the amount of work and confusion necessary to support the conceptual elements.

Next, we integrate these data-types into records that will represent the high-level conceptual entities with which we began. Finally, we list all other record types needed to support the top-level records.

This listing provides a conceptual overview of the data types and their relationships. In implementation, the actual field names may be slightly different. Please refer to the XML Schema for the definitive format of the data.

Basic Data types

Descriptor

A descriptor represents a type of textual data that has an internally defined, controlled vocabulary. An example would be the role descriptor used to define the type of contribution that is made to a work/instantiation/container. The descriptors in a vocabulary will be fixed in general (i.e. new descriptors shouldn't be created on a whim), but additions and deletions will be possible if necessary. When a field is filled with a Descriptor, the vocabulary itself will be stored separately, and is not detailed in this document, though suggested values are sometimes given to illustrate the nature of the field.

Date

In current practice, a lot of information is associated with "dates". Certainty and precision qualifiers (e.g.? and circa) are used to further indicate the nature of the date. Multiple possible dates may be listed where the exact date is not known. Less precise units may be used (centuries, decades) where exact years are not known. Finally, dates that are incorrect (e.g. due to publishing error) are annotated along with the correct date.

Clearly we need quite a bit of flexibility to support all these activities, but we cannot simply allow the date to be free form if we wish to normalize (descriptions for the same date should be the same) and convert the date to an internal numeric form for sorting/searching purposes. To support these activities as much of the date specification must be as regular as possible.

The representation described below is based on ISO 8601:1988 Date/Time Representations, with extensions from common library practice:

Valid Date Base forms:

YY*uu* -- Century
or **YYY*u*** -- Decade
or **YYYY** -- Year
or **YYYY**-**MM** -- Year/Month
or **YYYY**-**MM**-**DD** -- Year/month/Day

Valid Date Base form connectors and qualfiers:

? -- Date may be incorrect, follows date
ca. --Actual date is near or equal to this date, precedes date

early (approx. YY00-YY33)
or **middle** (approx. YY34-YY66)
or **late** (approx. YY67-YY99) – additional precision for century-based dates[2]

/ --OR, used provide multiple **possible** dates
to – range, specifies begin and end dates

Additionally we provide a mechanism for specifying date related descriptive content that will not be convertible or should not be used when sorting using dates.

Valid date comment (follows date base): **[*comment*]**

A few examples:

- 1977-02-25, 1977-02-27 [container also specifies 1977-02-29]
- early 12uu
- ca. 197u
- 1978/1979
- 1954-12 to 1955-01

It is worth noting that this formatting does NOT imply a particular format when presenting data to the user (e.g. in a result set display). By creating a format that can easily be disassembled, we can ensure that it is possible to translate into any presentation format we desire.

Timestamp

Precise date/time combinations created by the system are stored in the software platform's native timestamp objects, which are usually accurate to the level of milliseconds. Timestamps are used to track the exact times at which records are created or modified.

Typed Date

In some situations (namely Contributor dates), it is more convenient to have a single field capable of storing different types of dates rather than attempting to specify and allocate space for all the different types of dates up front. To support

this we create a typed date type that stores information on the type as well as the date.

Field	Mult.	Type
Date	1	Date
Type	1	Date type descriptor

Text

In its most simple form this data type allows the storage of textual characters. Any valid Unicode character can be used, including the ANSI/ANSEL character sets that are currently used by Indiana University's current MARC system.

Some text also includes a description of the source for the text. In some cases, the text field itself contains a source directly, and is denoted as type Sourced Text. For other fields, source information may be stored in Note fields with the type Source Data Found.

Title

From outward appearances titles are just another piece of text, but they have in current practice an additional "hidden" functionality. When titles are sorted, articles at the beginning of the title are not taken into account. In order to support this, we provide for an indicator to be specified that provides an offset that specifies the number of characters to skip from the beginning of the title[3].

Field	Mult.	Type
Title	1	Text
Non-filing indicator	1	Number

Like plain text, titles can also contain source information.

Note

For each entity type we have the ability to describe further any details that don't "fit" into any other place. Notes can be categorized so that, for example, administrative notes intended for library staff and administrators are not displayed in typical user interfaces.

Field	Mult.	Type
Note Content	1	Text
Note Type	1	Note Type Descriptor

In practice, we will have two distinct classes of notes, each with its own vocabulary of types. This allows bibliographic records (Containers and

[3] White space before the first non-article character should also be included in this offset.

Instantiations) to have differing note information from authority records (Works and Creators).

Contribution

The contribution data type is used in the representation of the typed relationship between a contributor and the target of their contribution (e.g. a Container, Work or Instantiation). In addition, the contribution can store a name to be associated with the contributor that is specific to that contribution. In general, these will be contributor names as indicated on the Container materials.

Field	Mult.	Type
Contributor	1	ID
Name Used	0…1	Text
Role Type	1	Role Descriptor

Work Relation

Used to represent types of relationships between Works. Examples being collective work-member work and original work-derived work relationships.

Field	Mult.	Type
Related Work	1	ID
Relation Type	1	Relationship Descriptor

Document Info

The document description data type is used in the representation of the Container in order to describe its associated document(s)

Field	Mult.	Type
Document Type	1	Document Type Descriptor (Sound, Score)
Document Format	1	Document Format Descriptor (LP, CD, Tape, Vocal Score, Full Score, Parts)
Physical Description	1	Text

Language Use

Description of language(s) used occurs in Work, Instantiation, and Containers. In each case there may be desire to indicate the nature or context in which the particular language is being used.

Field	Mult.	Type
Language	1	Language Descriptor
Location/Context of Use	0…1	Text

Instrumentation Info

Field	Mult.	Type
Quantity	1	Number
Instrumentation Descriptor	1	Instrumentation Descriptor

Resource

Field	Mult.	Type
Title	1	Text
Description	1	Text
Location	0…1	URL

Access Info

Typically automatically generated by the system, this data-type records the date and time of a particular access as well as the ID of the user involved.

Field	Mult.	Type
User	1	User ID
Access Timestamp	1	Timestamp

File Info

File Info provides a mechanism for storing the administrative metadata to be associated with each file found within a media object.

Field	Mult.	Type
File Name	1	Text
File Size	1	Number
File Extent	0…1	Number (size of the stored media, in its units)
Checksum	1	Text
Creation	1	Timestamp
Last Update	1	Timestamp

Physical ID Info

Information that identifies the physical items that make up a given Container.

Field	Mult.	Type
Location	1	Location Descriptor
Call Number	1	Text
Copy Number	1	Number

Copyright Declaration

Work, Instantiation, and Container each could indicate various copyright declarations that we need to store for later review. We will store this information

by identifying the copyright holder, the copyright domain, and the copyright date if available.

Field	Mult.	Type
Copyright Owner	1	Text
Copyright Domain	1	Text
Copyright Date	0…1	Date

Top-Level Records

Yellow	Descriptive
Tan	Structural
Blue	Relational/Connective
Green	Administrative

Contributor Record

Field	Mult.	Type
Type	1	Type Descriptor (Person or Group)
Uniform Name	1	Sourced Text
Variant Name(s)	0…Many	Text
Date(s)	0…Many	Typed Date
Variant Date(s)	0…Many	Typed Date
Place of Origin	0…1	Text
Note(s)	0…Many	Note
Related Resource(s)	0…Many	Resource
ID	1	ID
Record Creation	1	Access Info
Last Record Update	1	Access Info
Record Status	1	Status Descriptor

Work Record

Field	Mult.	Type
Type	1	Descriptor (Single or Collective)
Uniform Title	1	Sourced Title
Variant Title(s)	0…Many	Title
Date of Composition	0…1	Date
Place of Composition	0…1	Text
Date of First Performance	0…1	Date
Date of First Publication	0…1	Date
Other Dates	0…Many	Typed Date
Original Text Title	0…1	Text
Language(s)	0...Many	Language Use
Subject Heading(s)	0…Many	Sourced Text
Class Number	0…1	Sourced Text

Instrumentation	0…Many	Instrument Info
Genre, Form, and Style	0…Many	Genre, Form, Style Descriptor
Key	0…1	Key Descriptor
Note(s)	0…Many	Note
Related resource(s)	0…Many	Resource
Work Structure	0…1	Structure Declaration
ID	1	ID
Contribution(s)	0…Many	Contribution
Related Work(s)	0…Many	Work Relation
Copyright Declaration(s)	0…Many	Copyright Declaration
Performing Rights Society Work Number	0…1	Text
Public Domain	1	Descriptor (Unknown, Known Public Domain, Known NOT Public Domain)
Other System ID's	0…Many	Text
Work Structure Source	0…1	Text
Record Creation	1	Access Info
Last Record Update	1	Access Info
Record Status	1	Status Descriptor

Instantiation Record

Field	Mult.	Type
Title	1	Title
Document Description	1	Document Info
Subject Heading(s)	0…Many	Sourced Text
Class Number	0…1	Sourced Text
Instrumentation	0…Many	Instrument Info,
Genre, Form, and Style	0…Many	Genre, Form, and Style Descriptor
Language(s)	0…Many	Language Use
Notation	0…1	Notation Descriptor
Date(s) of Performance	0…Many	Date
Place of Performance	0…1	Text
Note(s)	0…Many	Note
ID	1	ID
Contribution(s)	0…Many	Contribution
Manifested Work	1	ID
Completeness Indicator	1	Descriptor (Complete or Incomplete)
Structural Binding	0…1	Structural Binding
Binding Level	1	Binding Level Descriptor (Outline, Measure)
Copyright Declaration(s)	0…Many	Copyright Declaration
Public Domain	1	Descriptor (Unknown, Known Public Domain, Known NOT Public Domain)
Record Creation	1	Access Info

Last Record Update	1	Access Info
Record Status	1	Status Descriptor

Container Record

Field	Mult.	Type
Display Title	1	Title
Additional Titles	0…Many	Title
Uniform Series Title	0…Many	Title
Publisher	0…1	Text
Edition	0…1	Text
Publisher/Plate Numbers	0...Many	Text
Document Description(s)	1…Many	Document Info
Language(s)	0…Many	Language Use
Date of Publication	0…1	Date
Date of Copyright	0…1	Date
Place of Publication	0…1	Text
Note(s)	0…Many	Note
Container Structure	0…1	Structural Declaration
ID	1	ID
Contribution(s)	0…Many	Contribution
Contained Media Objects	0…Many	ID
Contained Instantiations	0…Many	ID
Physical Identification	1	Physical ID Info
Condition	0…1	Condition Descriptor
Holding Status	0…1	Holding Status Descriptor (Public, Reserve)
ISBN	0…1	Text
MRC/EAN/UPC	0…1	Text
ISMN	0…1	Text
Copyright Declaration(s)	0…Many	Copyright Declaration
Public Domain	1	Descriptor (Unknown, Known Public Domain, Known NOT Public Domain)
Other System ID's	0…Many	Text
Record Creation	1	Access Info
Last Record Update	1	Access Info
Record Status	1	Status Descriptor

Media Object Record

Field	Mult.	Type
Label	1	Text
ID	1	ID
File Format	1	Text
Compression	1	Text
Bit Depth	1	Text

Resolution	1	Text
Number of Channels	1	Number
Sample Size	1	Number
Sample Rate	1	Number
Digitization Envirionment	1	Text (Hardware and software description)
Date Digitized	1	Date
Digitized By	1	Text
Record Creation	1	Access Info
Last Record Update	1	Access Info
Record Status	1	Status Descriptor
Number of Accesses	1	Number
Contained File(s)	1...Many	File Info

Use of XML

Before moving on to the more complex data types that enable connections between the top-level records, it is useful to note that the entire data model can be stored in XML format. For some of the data types described below, XML examples are given. This format follows a consistent style, using the following guidelines:

- Capitalization must be consistent. XML attributes are named using camelCase, which has words run together, the first word in lower-case, and succeeding words having an initial capital. XML elements should be UpperCamelCase, with an initial capital.
- Underscores are not allowed in XML attribute/element names.
- When deciding whether a field should be stored as an XML element or attribute, try to make the "data" elements, and the "metadata" attributes, while realizing that there is not always a clear distinction between the two.
- When possible, the name for a list of items should be the plural of the name for an individual item. For example, a list of <VariantName> elements is enclosed in a <VariantNames> element.

Structure and Bindings

Container Structure

Every container has a structure that allows navigation. The detail represented in this structure may vary widely, from a score volume that only has a table of contents section and the main section, to a multiple-item boxed set that contains books, CD's, etc., where the books have chapters and subsections, while the CD's may have groups of tracks along with the tracks themselves.

Field	Mult.	Type
Label	1	Text
Item	0...Many	Item

Item

An Item is a particular piece of a container, such as a CD. Many containers will only have one item.

Field	Mult.	Type
Label	1	Text
Div	0...Many	Div
Chunk	0...Many	Chunk

Div

A Div is used to indicate sections of a container that need a particular grouping, such as the table of contents in a score.

Field	Mult.	Type
Label	1	Text
ISRC	0...1	Text (International Standard Recording Code)
Div	0...Many	Div
Chunk	0...Many	Chunk

Chunk

A chunk is the smallest portion of a container that can be represented, typically a single page of a score or single track of a CD. The main purpose of Chunk is to put a label on a Content Interval.

Field	Mult.	Type
Label	1	Text
ISRC	0...1	Text (International Standard Recording Code)
Content Interval	1	Content Interval

Content Interval

Field	Mult.	Type
Media Ref	1	ID (of a MediaObject)
Begin	1	Number
End	1	Number

Work Structure

Each node within a work structure has an ID number, which is used by items that bind to this structure. Since sections of a work are completely abstract, and are not constrained by physical/digital media, they are all held in a single type of node, which can have a descriptive type identifier.

Field	Mult.	Type
Label	1	Text
ID	1	Number
Detail Note	0...1	Sourced Text

Section	0...Many	Section

Section

Sections can contain other sections, to provide hierarchical work structures. The title of a section is optional. While every section must have a label, a title is generally used for portions of a work that can stand on their own (like an aria). Titles are considered searchable information, while labels are not.

Field	Mult.	Type
Title	0...1	Text
Type	1	Section Type Descriptor (Act, Movement, Section, etc.)
Label	1	Text
ID	1	Number
Section	0...Many	Section
Detail Note	0...1	Sourced Text

Work Structure Bindings

Bindings to the work structure are stored with each instantiation. These bindings connect the instantiation's associated media objects to the relevant nodes of the work structure, using the ID numbers in the structure nodes.

Field	Mult.	Type
Binding	1...Many	Binding

Binding

Each binding can contain a label. This indicates the labels that actually appear as part of the instantiation, which may be different from the labels in the work structure.

Field	Mult.	Type
Node Ref	1	Number (ID of a section in the work structure)
Label	0...1	Text
Content Interval	1	Content Interval

Sample Structures

```
<ContainerStructure label="Symphony no. 7 in A major, op. 92 / Ludwig
 van Beethoven">
   <Item label="">
      <Div label="Title Pages">
         <Chunk label="[i]">
            <ContentInterval end="1" begin="0"
             mediaRef="IU/MediaObject/4031"/>
         </Chunk>
         <Chunk label="[ii]">
```

```
            <ContentInterval end="2" begin="1"
             mediaRef="IU/MediaObject/4031"/>
         </Chunk>
      </Div>
      <Div label="Table of Contents">
         <Chunk label="[iii]">
            <ContentInterval end="3" begin="2"
             mediaRef="IU/MediaObject/4031"/>
         </Chunk>
      </Div>
      <Div label="Instrumentation">
         <Chunk label="[iv]">
            <ContentInterval end="4" begin="3"
             mediaRef="IU/MediaObject/4031"/>
         </Chunk>
      </Div>
      <Div label="Symphony No. 7 in A major, op. 92">
         <Chunk label="[1]">
            <ContentInterval end="5" begin="4"
             mediaRef="IU/MediaObject/4031"/>
         </Chunk>
         <Chunk label="2">
            <ContentInterval end="6" begin="5"
             mediaRef="IU/MediaObject/4031"/>
         </Chunk>
      </Div>
   </Item>
</ContainerStructure>

<WorkStructure label="Symphonies, no. 7, op. 92, A major (Ludwig van
 Beethoven)" id="1">
   <Section title="" type="Movement" label="I. Poco sostenuto - Vivace"
    id="2">
      <Section title="" type="Section" label="Introduction" id="3"/>
      <Section title="" type="Section" label="Exposition" id="4"/>
      <Section title="" type="Section" label="Development" id="8"/>
      <Section title="" type="Section" label="Recapitulation" id="9"/>
      <Section title="" type="Section" label="Coda" id="10"/>
   </Section>
   <Section title="" type="Movement" label="II. Allegretto" id="5"/>
   <Section title="" type="Movement" label="III. Presto-Assai meno
    presto-Presto" id="6"/>
   <Section title="" type="Movement" label="IV. Allegro con brio"
    id="7"/>
</WorkStructure>

<StructureBindings>
   <Binding nodeRef="1">
      <ContentInterval end="554707" begin="0"
       mediaRef="IU/MediaObject/5809"/>
   </Binding>
   <Binding nodeRef="2">
      <ContentInterval end="1043000" begin="554707"
       mediaRef="IU/MediaObject/5809"/>
   </Binding>
   <Binding nodeRef="3">
```

```
      <ContentInterval end="1709000" begin="1043000"
       mediaRef="IU/MediaObject/5809"/>
   </Binding>
</StructureBindings>
```

User, Group, and Service Records

User

Field	Mult.	Type
ID (Username)	1	Text
First Name	1	Text
Last Name	1	Text
Expiration Date	1	Date
Allowed Services	0…Many	ID
Disallowed Services	0…Many	ID
Group Membership	0…Many	Group ID
Record Creation	1	Access Info
Last Record Update	1	Access Info

Group

Field	Mult.	Type
ID	1	Group ID
Name	1	Text
Expiration Date	1	Date
Record Creation	1	Access Info
Last Record Update	1	Access Info

IP List

Field	Mult.	Type
ID	1	IP List ID
Name	1	Text
IP Address(es)	0...Many	IP Address (IP descriptor or range descriptor)
Record Creation	1	Access Info
Last Record Update	1	Access Info

Service

Field	Mult.	Type
ID	1	ID

Name	1	Text
Location	1	URI
Expiration	1	Date
Allowed Groups	0…Many	Group ID
Disallowed Groups	0…Many	Group ID
IP List	0…1	IP List ID
Record Creation	1	Access Info
Last Record Update	1	Access Info

Bookmarks and References

Each user has their own set of bookmarks which is stored on the server, and may be exported to an XML or HTML file. Bookmarks stored in XML form may also be imported into the system.

Bookmarks are "owned" by a particular window. This means that bookmarks can only be opened in the window where they were created. They only need to store the minimal amount of information needed for the owning window and find a particular place in the media files it uses.

Bookmark Tree

The bookmark tree holds all of a user's bookmarks and folders. The annotation is typically used only in exported bookmarks, to indicate the user's name and the time of export.

Field	Mult.	Type
Folder	0...Many	Folder
Bookmark	0...Many	Bookmark
Annotation	0…1	Text

Folder

Users may organize their bookmarks into various folders. Folders may be nested to provide a hierarchical organization of bookmarks.

Field	Mult.	Type
Name	1	Text
Annotation	0...1	Text
Folder	0...Many	Folder
Bookmark	0...Many	Bookmark

Bookmark

Different types of bookmarks will have slightly different structures. In particular, the player and viewer bookmarks each have a single container reference, while opus bookmarks have multiple instantiation references instead. Note that for an opus bookmark, an offset should only be present for the recording instantiation. If an offset is present for the score instantiation, it will be ignored.

Field	Mult.	Type
Name	1	Text
FullName	0...1	Text (a more detailed name, often a container title)
Type	1	Bookmark Type Descriptor (Player, Viewer)
Annotation	0...1	Text
ContainerRef	0...1	Container Reference
InstantiationRef	0...Many	Instantiation Reference
WorkRef	0...1	Work Reference
URL	1	URL

Container Reference

The container referenced by a bookmark. The offset is a number of milliseconds for recordings, or a number of pages for scores. For bookmarks that represent a range instead of a point, the end offset is also used.

Field	Mult.	Type
ID	1	ID
Offset	1	Number
EndOffset	0...1	Number

Instantiation Reference

The instantiation referenced by a bookmark. The offset is a number of milliseconds for recordings, or a number of pages for scores. For bookmarks that represent a range instead of a point, the end offset is also used.

Field	Mult.	Type
ID	1	ID
Offset	1	Number
EndOffset	0...1	Number

Work Reference

A work referenced by a bookmark. The start and end nodes are references to portions of the work structure.

Field	Mult.	Type
ID	1	ID
StartNode	1	Number
EndNode	1	Number

Sample Bookmarks

```
<Bookmark name="Track 5. I Poco sostenuto-Vivace [0:00]" type="player">
   <Annotation/>
   <FullName>Symphonies 5 &amp;amp; 7 [sound recording] : Egmont overture;
    Track 5. I Poco sostenuto-Vivace</FullName>
   <ContainerRef offset="1831121">IU/Container/6</ContainerRef>
   <Url>
     http://purl.dlib.indiana.edu/iudl/variations2/access/type=player&
       container_id=IU/Container/6&start_offset=1831121
```

```
   </Url>
</Bookmark>

<Bookmark name="Page 4" type="viewer">
   <Annotation>My favorite page</Annotation>
   <FullName>Symphony no. 7 in A major, op. 92; 4</FullName>
   <ContainerRef offset="7">IU/Container/3</ContainerRef>
   <Url>
     http://purl.dlib.indiana.edu/iudl/variations2/access/type=viewer&
        container_id=IU/Container/3&start_offset=7
   </Url>
</Bookmark>

<Bookmark name="Allegro con brio (M 32)" type="opus">
   <Annotation>My favorite section</Annotation>
   <FullName>
     Symphonies 5 &amp;amp; 7 [sound recording] : Egmont overture;Symphony
     no. 7 in A major, op. 92
   </FullName>
   <InstantiationRef offset="183">
       IU/Instantiation/3061
   </InstantiationRef>
   <InstantiationRef>IU/Instantiation/3031</InstantiationRef>
   <WorkRef startNode="3" endNode="4">IU/Work/2001</WorkRef>
   <Url>
      http://purl.dlib.indiana.edu/iudl/variations2/access/type=opus&
      instantiation_id=IU/Instantiation/3061,IU/Instantiation/3031&
      start_offset=183&work_id=IU/Work/2001&work_start=3&work_end=4
   </Url>
</Bookmark>
```

Timelines

Users may create timelines to describe the content of recordings. These are similar to container structures, but with added information related to the graphical display of the structure. Timelines do not necessarily cover the entire extent of the associated container. All colors are stored as RGB values.

Timeline

A timeline holds information about the line itself, as well as the tree of bubbles.

Field	Mult.	Type
Auto-scaling on	1	Boolean
Background Color	1	Color
Black and White Bubbles	1	Boolean
Bubble Height	1	Number
Bubble Type	1	Number
Editable	1	Boolean
Media Offset	1	Number
Media Length	1	Number
Media Content	1	Container ID

Play when Bubble Clicked	1	Boolean
Resizable	1	Boolean
Stop Play at End of Selection	1	Boolean
Times Visible	1	Boolean
Title	1	Text
Timepoint List	1	List of Timepoints
Bubble Tree	1	Bubble
Bubble Level Colors	0...1	List of Colors

Bubble

The bubbles are arranged in a tree. A single root bubble encloses all other bubbles. The root bubble does not get displayed by the timeliner interface. Even though it is possible to calculate a bubble's level from the tree, we will explicitly store the levels. This makes tree processing easier, since a bubble's parent is not necessarily one level up. It also allows for the future possibility of users manually adjusting levels (ie. setting levels differently than the hierarchy would imply).

Field	Mult.	Type
Color	0...1	Color
Level	1	Number
Label	0...1	Text
Annotation	0...1	Text
Bubble	0...Many	Bubble

Timepoint

The time points are stored separately from the bubbles. Each point can have an annotation, and serves as the left edge of one or more bubbles.

Field	Mult.	Type
Label	0...1	Text
Offset	1	Number
Annotation	0...1	Text

Sample Timeline

Here is a sample timeline that has four bubbles across the base, with the last two grouped together:

```
<Timeline BWBubbles="true" autoscale="true" bgColor="255,255,204"
 bubbleHeight="40" bubbleType="1" clickPlay="false" editable="true"
 mediaContent="IU/Container/4888" mediaLength="119000"
 mediaOffset="469893" resizable="true" stopPlay="false"
 title="Timeline: Schubert, Die Forelle" visibleTimes="false">
   <LevelColors>
     <LevelColor>166,202,240</LevelColor>
     <LevelColor>255,255,120</LevelColor>
     <LevelColor>255,200,100</LevelColor>
   </LevelColors>
```

```
   <Timepoints>
      <Timepoint offset="0"/>
      <Timepoint offset="10233" label="X">
         <Annotation>my favorite spot</Annotation>
      </Timepoint>
      <Timepoint offset="33949"/>
      <Timepoint offset="44000"/>
      <Timepoint offset="49000"/>
      <Timepoint offset="119000"/>
   </Timepoints>
   <Bubble color="166,202,240" level="3">
      <Bubble color="100,100,0" label="hello" level="1">
         <Annotation>blah</Annotation>
      </Bubble>
      <Bubble level="1"/>
      <Bubble level="2">
         <Bubble level="1"/>
         <Bubble level="1"/>
      </Bubble>
   </Bubble>
</Timeline>
```

Supporting the Model

This section describes additional portions of the Variations2 system that, while not part of the data model, work directly with the data model and are integral to its usage.

Indices

The Variations2 server builds indices to support searching. These are described briefly here, and in more detail in the *Data Model Specification Format and Code Generation* document.

Text Indices

When records are created or updated, any data in an indexed field is copied to an index table in the database. During search, the index table is used to find matches and reference to the appropriate record.

NOTE: When work structures are indexed, only the title attributes of section elements go into the index.

Record Type	Indexed Fields
Contributor	Uniform Name, Variant Names
Work	Uniform Title, Variant Titles, Structure
Container	Display Title, Additional Titles

Keywords

Keyword search works a bit differently than the basic search, since keyword searching only applies to containers, and fields associated with other record types (eg. works, contributors) must be used in the keyword index.

When a new container is created, its entry in the keyword index is initialized with the display title, so minimal searching can be performed. However, when entities are updated, the keyword index is **not** changed. This prevents the entry in the keyword index from being reset to just a title after it has been filled with more useful information.

Periodically, a separate program runs to calculate the keyword index. For each container record, it collects all related works, instantiations, and creators. The contents of these records are then placed into the keyword index for the container. The contents of all fields are added to the keyword index, except the authoritative/administrative notes (which are not publicly viewable).

Search Results

When a search is performed, the main result will be a recordset object that contains records representing top-level data model objects. In addition to these records, the recordset can also contain meta-information about the search.

Matched Terms

During indexing, when the server places an entry in the text index, it also stores the name of the field that generated the entry. When a record is looked up using the index, the name of this field and the contents is added to the resultant recordset, and this information can be used by clients to highlight information in the result listing. For example, if the user searches for creator "Bach", the following would be added to the result set:

```
<TextMatches>
   <TextMatch>
      <Id>IU/Contributor/4870</Id>
      <FieldName>uniformName</FieldName>
      <Text>Bach, Carl Philipp Emanuel</Text>
   </TextMatch>
   <TextMatch>
      <Id>IU/Contributor/1015</Id>
      <FieldName>uniformName</FieldName>
      <Text>Bach, Johann Sebastian</Text>
   </TextMatch>
   <TextMatch>
      <Id>IU/Contributor/1015</Id>
      <FieldName>variantName</FieldName>
      <Text>Bach, Juan S</Text>
   </TextMatch>
</TextMatches>
```

Note that the example above is incomplete. Johann Sebastian Bach has many variant names, and all variants that match the search criteria would be returned.

Relevance/Similarity Scores

Result sets may also contain a list of relevance/similarity scores for ordering result lists. This looks like:

```
<RelItems>
   <RelItem value="0.3">IU/Work/1114</RelItem>
   <RelItem value="0.2">IU/Work/8036</RelItem>
</RelItems>
```

This list of values may be generated by the Variations 2 server, or it may come from some external source. The client will use this list (if available) to sort result sets.

Command Files

Variations2 can accept command files that will tell it which windows to open, and in what state. There are currently two types of command files: timeline files, and V2X files. The timeline files simply contain a single timeline object (in XML format), and open the timeliner to display this timeline. V2X files can hold both timelines and references that are similar to bookmarks.

V2X Files

A V2X file consists of valid XML. The root element of the file must be V2X, and this element can contain any number of commands. Each command opens exactly one Variations2 window.

Below is a sample command to open a search window. The Search tag may be used alone, or with relevance scores, as described above.

```
<V2X version="2.0">
   <Search>
      <RelItems>
         <RelItem value="0.3">IU/Work/1114</RelItem>
         <RelItem value="0.2">IU/Work/8036</RelItem>
      </RelItems>
   </Search>
</V2X>
```

Player, Viwer, and Opus commands may be used to open a window of the appropriate type. These commands contain the same reference information as the associated bookmarks, but descriptive information (FullName, URL, Annotation) is not necessary, and will be ignored. Sample commands for these windows are shown below.

```
<V2X version="2.0">
   <Player>
      <ContainerRef offset="1831001">IU/Container/6</ContainerRef>
   </Player>
   <Viewer>
```

```
      <ContainerRef offset="3">IU/Container/4</ContainerRef>
   </Viewer>
   <Opus>
      <InstantiationRef offset="183">
         IU/Instantiation/3061
      </InstantiationRef>
      <InstantiationRef>IU/Instantiation/3031</InstantiationRef>
      <WorkRef startNode="3" endNode="4">IU/Work/2001</WorkRef>
   </Opus>
</V2X>
```

Old V2X Format

The old format of V2X files is deprecated, as it violates some of our design guidelines. However, it is still accepted by version 2 clients. It will be removed in version 3. Examples of player and viewer commands are shown below.

```
<v2x version="1.0">
   <player>
      <container_id offset="1831001">IU/Container/6</container_id>
   </player>
</v2x>

<v2x version="1.0">
   <viewer>
      <container_id offset="3">IU/Container/4</container_id>
   </viewer>
</v2x>
```