HHS Public Access

Author manuscript

Proc IEEE Int Conf Inf Reuse Integr. Author manuscript; available in PMC 2017 April 12.

Published in final edited form as:

Proc IEEE Int Conf Inf Reuse Integr. 2016 July ; 2016: 58–66. doi:10.1109/IRI.2016.16.

Modeling Integration and Reuse of Heterogeneous Terminologies in Faceted Browsing Systems

Daniel R. Harris

Center for Clinical and Translational Science, University of Kentucky, Lexington, KY, USA

Abstract

We integrate heterogeneous terminologies into our category-theoretic model of faceted browsing and show that existing terminologies and vocabularies can be reused as facets in a cohesive, interactive system. Commonly found in online search engines and digital libraries, faceted browsing systems depend upon one or more taxonomies which outline the structure and content of the facets available for user interaction. Controlled vocabularies or terminologies are often externally curated and are available as a reusable resource across systems. We demonstrated previously that category theory can abstractly model faceted browsing in a way that supports the development of interfaces capable of reusing and integrating multiple models of faceted browsing. We extend this model by illustrating that terminologies can be reused and integrated as facets across systems with examples from the biomedical domain.

I. Introduction

Faceted classification is the process of assigning facets to resources in a way that enables intelligent exploratory search aided by an interactive faceted taxonomy [1]. Exploratory search using a faceted taxonomy is often called faceted browsing (or faceted navigation or faceted search) [2] and is commonly found in digital libraries or online search engines. Facets are the individual elements of the faceted taxonomy and are simply attributes known to describe an object being cataloged; these collections of facets are often organized as sets, hierarchies, lattices, or graphs. Facets are usually shown alongside a list of other related, relevant facets that aid in interactive filtering and expansion of search results [3]. A simple example of facets for a digital library of books would be genre or publication date. The taxonomy behind the interface is either custom to the search needs of the interface or bootstrapped by a terminology familiar to those with working knowledge of the domain. In the biomedical domain for example, patients are often classified according to ICD10 diagnosis codes [4] in their electronic health record; as seen in Fig. 1, the i2b2 query tool is capable of searching for patients using ICD10 codes [5] as well as other common biomedical terminologies. We will discuss i2b2 and another biomedical application in Section IV.

Facet models formalize faceted data representations and the interactive operations that follow for exploratory search tasks. Wei et al. observed three major theoretical foundations behind current research of facet models: set theory, formal concept analysis, and lightweight ontologies [1]. In our previous work, we demonstrated that category theory can act as a theoretical foundation for faceted browsing that encourages reuse and interoperability by

uniting different facet models together under a common framework [6], [7]. We also established facets and faceted taxonomies as categories and have demonstrated how the computational elements of category theory, such as products and functors, extend the utility of our model [6]. The usefulness of faceted browsing systems is well-established in the digital libraries research community [8], [9], but reuse and interoperability is typically not a major design consideration [6]. Our goal is to create a rich environment for faceted browsing where reuse and interoperability are primary design considerations.

In this paper, we integrate heterogeneous terminologies as facets into the category-theoretic model of faceted browsing so that existing and well-known terminologies can be reused in an intelligent manner. These terminologies themselves can act as a faceted taxonomy, but we also demonstrate the usefulness of modeling a terminology as a facet type. We discuss how to create instances of facets and faceted taxonomies so that our model can interact with multiple, heterogeneous sources. We present and compare two considerations for modeling faceted browsing interfaces that utilize multiple terminologies: the need to merge facets together and the need for multiple focuses from different terminologies.

II. Background

We must discuss faceted taxonomies and introduce concepts from category theory before discussing our category-theoretic model of faceted browsing and its extensions.

A. Faceted Taxonomies

At the heart of faceted browsing, regardless of the facet model chosen for a particular interface, there lies a taxonomy which organizes and gives structure to the facets that describe the resources to be explored. Faceted taxonomies can aid in the construction of information models or aid in the construction of a larger ontology [10], [11]. If facet browsing is truly a pivotal element to modern information retrieval [12], then great care must be taken to abstractly model and fully integrate the taxonomies behind the interface. A facet browsing interface may depend upon one or many faceted taxonomies to drive exploration and discovery, depending upon the needs and complexity of its design.

B. Category Theory

Category theory has been useful in modeling problems from multiple science domains [13], including physics [14], cognitive science [15], and computational biology [16]. Categories can also model databases [17], [13] where migration between schemas can be represented elegantly [18]. We will demonstrate that facets and schemas are structurally related in Section III-B2.

In this section, we introduce a few concepts from category theory that are necessary for understanding our model. Informally, a category $\mathscr{C}$ is defined by stating a few facts about the proposed category (specifying its objects, morphisms, identities, and compositions) and demonstrating that they obey identity and associativity laws [13].

Definition 1—A category $\mathscr{C}$ consists of the following:

1. A collection of objects, $Ob(\mathscr{C})$.
2. A collection of morphisms (also called arrows). For every pair $x, y \in Ob(\mathscr{C})$, there exists a set $Hom_{\mathscr{C}}(x, y)$ that contains morphisms from x to y; a morphism $f \in Hom_{\mathscr{C}}(x, y)$ is of the form $f: x \rightarrow y$, where x is the domain and y is the codomain of f.
3. For every object $x \in Ob(\mathscr{C})$, the identity morphism, $id_x \in Hom_{\mathscr{C}}(x, x)$, exists.
4. For $x, y, z \in Ob(\mathscr{C})$, the composition function is defined as follows: $\circ: Hom_{\mathscr{C}}(y, z) \times Hom_{\mathscr{C}}(x, y) \rightarrow Hom_{\mathscr{C}}(x, z)$

Given 1–4, the following laws hold:

1. identity: for every $x, y \in Ob(\mathscr{C})$ and every morphism $f: x \rightarrow y$, $f \circ id_x = f$ and $id_y \circ f = f$.
2. associativity: if $w, x, y, z \in Ob(\mathscr{C})$ and $f: w \rightarrow x$, $g: x \rightarrow y$, $h: y \rightarrow z$, then $(h \circ g) \circ f = h \circ (g \circ f) \in Hom_{\mathscr{C}}(w, z)$

Our model of faceted browsing leverages two well-known categories: **Rel** and **Cat.** We leverage these as building blocks in our model by creating subcategories: categories constructed from other categories by taking only a subset of their objects and the necessary corresponding morphisms.

Definition 2—Rel is the category of sets as objects and relations as morphisms [19], where we define relation arrows $f: X \rightarrow Y \in Hom_{\mathbf{Rel}}(X, Y)$ to be a subset of $X \times Y$.

Definition 3—Cat is the category of categories. The objects of **Cat** are categories and the morphisms are functors (mappings between categories).

Functors can informally be thought of as mappings between categories, but additional conditions are required:

Definition 4—A functor F from category $\mathscr{C}_1$ to $\mathscr{C}_2$ is denoted $F: \mathscr{C}_1 \rightarrow \mathscr{C}_2$, where $F: Ob(\mathscr{C}_1) \rightarrow Ob(\mathscr{C}_2)$ and for every $x, y \in Ob(\mathscr{C}_1)$, $F: Hom_{\mathscr{C}_1}(x, y) \rightarrow Hom_{\mathscr{C}_2}(F(x), F(y))$. Additionally, the following must be preserved:

1. identity: for any object $x \in Ob(\mathscr{C}_1)$, $F(id_{\mathscr{C}_1}) = id_{F(\mathscr{C}_1)}$.
2. composition: for any $x, y, z \in Ob(\mathscr{C}_1)$ with $f: x \rightarrow y$ and $g: y \rightarrow z$, then $F(g \circ f) = F(g) \circ F(f)$.

In this section, we describe our category-theoretic model of faceted browsing. We demonstrated previously that our model encourages and facilitates reuse and interoperability within and across faceted browsing systems; we describe only the key elements and leave the minor details available in our prior work [6].

Definition 5—Let **Tax** be a sub-category of **Rel**, the category of sets as objects and relations as morphisms where *Ob*(**Tax**) = *Ob*(**Rel**) and let the morphisms be the relations

that correspond only to the ⊆ relations. The identity and composition definitions are simply copied from **Rel**.

Tax is simply a slimmer version of **Rel**, where we know exactly what binary relation is being used to order the objects. In our previous work, we did not apply a name to **Tax** and left this category described as **Rel** restricted to inclusion mappings [6]; applying a name allows us to be concise in our discussions, which is important because **Tax** will be the building block that will allow us to apply the additional structure and granularity needed to support faceted browsing. We can refer to an independent facet, such as genre, language, or price-range, as a *facet type*.

Definition 6—A facet type (a facet i and its related sub-facets) of a faceted taxonomy is a sub-category of **Tax**, the category of sets as objects and inclusion relations as morphisms. Let us call this sub-category $\mathbf{Facet}_i$ and let $Ob(\mathbf{Facet}_i) \subseteq Ob(\mathbf{Tax})$ with the morphisms being the corresponding ⊆ relations for those objects. The relevant identity and composition definitions are also copied from **Tax**.

From this facet type, users make focused selections when drilling down into faceted data. This selection pinpoints a subset of the facets within this type and by proxy, it pinpoints a subset of the resources classified.

Definition 7—We can define a subcategory of $\mathbf{Facet}_i$, called $\mathbf{Focus}_i$, to represent a focused selection of objects from $\mathbf{Facet}_i$ having $Ob(\mathbf{Focus}_i) \subseteq Ob(\mathbf{Facet}_i)$ and the necessary corresponding morphisms, identity, and composition definitions for those objects.

Each individual facet category belongs to a larger taxonomy that collectively represents the structure of information within a facet browsing system.

Definition 8—Let **FacetTax** be a category that represents a faceted taxonomy, whose objects are the disjoint union of $\mathbf{Facet}_i$ categories. In other words, let $Ob(\mathbf{FaceTax}) = \bigsqcup_{i=1}^{n} \mathbf{Facet}_i$ and $n = |Ob(\mathbf{FacetTax})|$. The morphisms of **FacetTax** are functors (mappings between categories) of the form $Hom_{\mathbf{FacetTax}}(\mathscr{C}, \mathscr{D}) = \{F : \mathscr{C} \to \mathscr{D}\}$.

Once you have a faceted taxonomy constructed, interactivity and engagement with it follows; a natural task for users of a faceted system is to perform queries that focus and filter objects being explored.

Definition 9—A facet universe, U, is the n-ary product [19] within the **FacetTax** category, defined as $\prod_{i=1}^{n} \mathbf{Facet}_i$, where $n = |Ob(\mathbf{FacetTax})|$. The n coordinates of U are projection functors $P_j : \Pi\mathbf{Facet}_i \to \mathbf{Facet}_j$, where $j = 1, \ldots, n$ is the jth projection of the n-ary product.

Note that since $\mathbf{Focus}_i$ is a subcategory of $\mathbf{Facet}_i$, there exists a restricted universe $U_{\subseteq} \subseteq U$ where every facet is potentially reduced to a focused subset. The act of querying the universe is essentially constructing this restricted universe $U_{\subseteq}$.

Definition 10—A faceted query, Q, is the modified n-ary product[19] within the **FacetTax** category, defined as $\prod_{i=1}^{n} \mathbf{Focus}_i$, where $n = |Ob(\mathbf{FacetTax})|$. The n coordinates of Q are similarly defined as projection functors $P_j : \mathbf{Focus}_i \rightarrow \mathbf{Focus}_j$.

C. A Category-theoretic Model

We visually summarize the key containers and products in Fig. 2. We will later demonstrate that this same faceted taxonomy can be represented as a graph. The objects of each $\mathbf{Facet}_i$ are sets of resources that have been classified as belonging to that facet type; our model can reuse the facets and adjust the surrounding structure to fit our needs: if we wish to arrange the facets as graphs, we can do so without bothering the resource and facet linkages. Fig. 3 shows a sample piece of a medication taxonomy; each resource is classified using the taxonomy. In our model, we refer to resources in the general sense. The type of resource depends upon the interface: resources could be books in a digital library system, documents in a electronic health system, and so on. Note that the taxonomy in Fig. 3 could easily be considered the facet type *medications*, which belongs to a large taxonomy (not pictured) instead of a complete faceted taxonomy to itself; either scenario are acceptable as this will depend upon the design of the faceted browsing system, which can vary.

III. Leveraging Multiple Terminologies

The category-theoretic model is perfectly capable of representing basic faceted interfaces in its current form, but the ability to model and interact with multiple heterogeneous sources is needed to support more intricate interfaces. The ability to integrate multiple terminologies rests largely upon our ability to model *instances* of our facet categories. Understanding the relationship between schemas and facets will be key to understanding the process for creating instances.

In our previous work on modeling faceted browsing for reusability, we demonstrated the importance that graphs play in reusing and integrating models [6]. We confirm this importance in the following sub-sections.

A. Underlying Graphs

The ability to transform into other structures enables the category theoretic model of faceted browsing to consume other models. We show that graphs underlie categories and that a graph-based representation of a facet can be used as input in modeling taxonomies.

Definition 11—Grph is the category with graphs as objects. A graph G is a sequence where $G := (V, A, src, tgt)$ with the following:

1. a set V of vertices of G
2. a set A of edges of G
3. a source function $src : A \rightarrow V$ that maps arrows to their source vertex
4. a target function $tgt : A \rightarrow V$ that maps arrows to their target vertex

Definition 12—The graph underlying a category $\mathscr{C}$ is defined as a sequence $U(\mathscr{C})=(Ob(\mathscr{C}), Hom_{\mathscr{C}}, dom, cod)$, [13].

We previously demonstrated given that there exists a functor U: **Cat** → **Grph**, so **FacetTax** can produce graphs of **Facet**$_i$ categories for i = (1,…, /*Ob*(**FacetTax**)|) [6].

Definition 13—Let U(**Facet**$_i$) be the underlying graph of an individual facet and let U(**FacetTax**) be the underlying graph of the faceted taxonomy at large, as constructed and detailed above.

This underlying graph will be important in discussing the relationship between schemas and faceted taxonomies, which will allow us to create instances of facets and faceted taxomonies.

B. Facet and Schema

In this section, we describe how to create instances of facets and faceted taxonomies with a method and rationale that is inspired by Spivak's database schemas [13]. In fact, we discover that facets are equivalent to database schemas. Although this equivalence is strange at first, conceptually the idea of a database schema is not unlike facets when viewed from a category theory perspective: both describe the conceptual layout that organizes information (rows/ entities in the case of databases and resources in the case of facets). Fig. 4 shows the same faceted information found in Fig. 3, but within a schema. Note that parts of the table are abbreviated with ellipses in order to save space. We will discuss these tables and their relationship with faceted browsing in detail in the next section.

1) Preliminary Definitions—Spivak's definition of schemas depends upon the idea of congruence, which in turn depends on defining paths, path concatenation, and path equivalence declarations [13].

Definition 14: If G := (V, A, src, tgt) is a graph, then a path of length n in G is a sequence of arrows denoted $p \in Path_G^{(n)}$, where $Path_G$ is the set of paths in G [13].

Definition 15: Given a path $p: v \rightarrow w$ and $q: q \rightarrow x$, $p++q: v \rightarrow x$ is the concatenation of the two paths [13].

Definition 16: A path equivalence declaration (abbreviated by Spivak as PED) is an expression of the form $p \simeq q$, where $p, q \in Path_G$ have the same source and target, e.g., $src(p) = src(q)$ and $tgt(p) = tgt(q)$ [13].

Definition 17: A congruence on G is a relation $\simeq$ on $Path_G$ with the following [13]:

1. The relation $\simeq$ is an equivalence relation.
2. If $p \simeq q$, then $src(p) = src(q)$ and $tgt(p) = tgt(q)$.
3. If given paths $p, p' : a \rightarrow b$ and $q, q' : b \rightarrow c$, and if $p \simeq p'$ and $q \simeq q'$, then $(p ++ q) \simeq (p' ++ q')$.

Informally, a congruence is an enhanced equivalence relation that marks how different paths in G relate to one another by enforcing additional constraints; pairing a graph with a congruence forms a schema [13].

2) Categorical View of Schemas—We give Spivak's definition of a schema below; this definition is generic enough to also apply to faceted browsing when looking at the underlying graph of the facet categories. Fig. 4 contains a schema corresponding to the medications example from Fig. 3.

Definition 18: A schema S is a named pair $S = (G, \simeq)$, where G is a graph and $\simeq$ is a congruence on G [13].

Note that the keys in Fig. 4 would normally be integer keys, but here text labels are applied to increase readability and to improve the ease of understanding the example. The resource table in this schema contains a generic list of resources (for example, documents or library items) where each resource has a foreign key indicating how it is classified. The medications table contains a list of classes and sub-classes for medications, as well as a self-referential foreign key pointing back at itself; this foreign key indicates this particular medication's ancestor. The self-referential key gives additional structure to the medication classes and sub-classes found within the table without the need for additional relationship tables; this method of storing a taxonomy is similar to closure tables [20].

In Fig. 4, the entry with *Medication* as its key has no foreign key; this null relationship indicates that is the root of this particular facet graph; with respect to the category-theoretic model, it implies there are no morphisms having this object in its domain.

C. Instances of Facets and Faceted Taxonomies

An instance of a facet is a collection of objects whose data are classified according to specific relationships, such as the one illustrated in Fig. 3. We formalize this below using Spivak's instances of schemas as inspiration [13].

Definition 19—Let $F = (U(\mathbf{Facet}_i), \simeq)$, where the graph underlying a facet type is denoted $U(\mathbf{Facet}_i)$ for some $\mathbf{Facet}_i \in Ob(\mathbf{FacetTax})$ and where $\simeq$ is a congruence on $U(\mathbf{Facet}_i)$. An instance on F is denoted $(Facet, Ancestor) : F \rightarrow \mathbf{Set}$ where:

1. Facet is a function defined as $Facet : V \rightarrow \mathbf{Set}$, so for each vertex $v \in V$ we can recover a set of facets denoted $Facet(v)$ within this facet type.
2. for every arrow $a \in A$ having $v = src(a)$ and $w = tgt(a)$, a function $Ancestor(a) : Facet(v) \rightarrow Facet(w)$.
3. congruence is preserved: for any $v, v' \in V$ and paths p, p' from v to v' where $p = v[f_0, f_1 f_2, \ldots, f_m]$ and $p' = [f_0', f_1', f_2', \ldots, f_n']$, if $p \simeq p'$, for all $x \in Facet(v)$, $ancestor(f_m) \bigcirc \ldots \bigcirc ancestor(f_1) \bigcirc ancestor(f_0)(x) = ancestor(f_n') \bigcirc \ldots \bigcirc ancestor(f_1') \bigcirc ancestor(f_0')(x) \in Facet(v')$

To create instances of **FacetTax**, the logic remains the same from **Facet**: take the underlying graph and a congruence; we omit this definition due to redundancy and space considerations.

We will use instances in the next section to model the integration and reuse multiple heterogeneous sources of information.

IV. Bootstrapping Faceted Taxonomies

Faceted taxonomies are common in the biomedical domain where controlled vocabularies are curated and integrated into interfaces in order to assist in the exploration and interaction required by the system. We present two different use cases for faceted taxonomies with different requirements: one where merging heterogeneous terminologies into a single taxonomy fits the design of the interface (for example, i2b2) and one where having control over multiple independent instances of facets is desired (for example, DELVE).

A. i2b2

The i2b2 (Informatics for Integrating Biology and the Bedside) query tool allows researchers to locate patient cohorts for clinical research and clinical trial recruitment [5]; the tool itself provides a drag-and-drop method of creating Boolean queries of inclusion and exclusion criteria from a hierarchical list of facets. For example, if someone wanted to search for only female patients, they would click into the *Demographics* facet, into the *Gender* facet, and drag *Female* to the first query panel. In addition, if they wanted female diabetics, they would also navigate into the *Diagnoses* facet and drag the desired type of diabetes into the second panel. i2b2's boolean queries are formed from having logical *or*-statements across panels and *and*-statements within a panel. With respect to the example above, if the user wanted female diabetic and hypertensive patients, they would also find the hypertension facet and drag it into the same panel having diabetes, so that the panel represents patients having either diabetes or hypertension. This boolean construction can be continued with any number of facets from any number of terminologies.

The biomedical domain has a long history of curating and maintaining controlled vocabularies and terminologies, such as those found in the Unified Medical Language System (UMLS) [21]. The structure behind these terminologies is a rich source for building faceted browsing systems that explore resources having been classified with these standards.

In Fig 5, the taxonomy of a local implementation of i2b2 is partially shown; note that every facet type of a patient is compiled into a central taxonomy as part of the meta-data cell for i2b2 [5]. This means that the central taxonomy has very different concepts, such as diagnoses and laboratory procedures, residing in the same table. Our local implementation of i2b2 uses ICD10 codes [4] for diagnoses and HCPCs codes [22] for procedures; these terminologies are externally and independently curated and made available by their creators. To i2b2, diagnosis is a facet type and ICD10 provides the organizational structure behind diagnoses, but ICD10 is a full terminology and one can consider ICD10 itself to be a facted taxonomy for diagnoses; the use of large-scale existing terminologies in faceted browsing system blurs the line between facet types and facet taxonomies, similar to our example and discussion of Fig. 3. Our modeling technique needs to abstractly and consistently be able to model both of these cases. In either case, the goal is encourage reuse of existing terminologies so that our faceted taxonomies contain accepted interoperable standards. An extension of i2b2 allows networking queries between institutions, so that one boolean query

can return counts of patients from multiple clinical sites; this would be impossible without integration of accepted biomedical terminologies into the faceted backbone of i2b2.

B. Merge Operations

Suppose we have multiple instances of facets, $I_0, I_1, \ldots, I_N$, how do we satisfy the requirements of an application such as i2b2 that expects a single instance to act as a master? For example, I_0 could be medications, while I_1 could be procedures, and so on.

Each **Facet**$_i$ category is disjoint and contains no linkage to another **Facet**$_j$ where $i \neq j$, so we must manufacture a link. This is a meta-facet, an organizational tool that typically aids in drawing the faceted taxonomy [6]. By design, the meta-facet must connect to the root of each facet; we can easily identify the root in our facet graph because it is the only entry with a null ancestor. Given an instance, such as I_0 above, we know that the root of I_0 is the source of an arrow $a \in A$ from U(**Facet**$_0$) where $Ancestor(a)$ is the empty set; we shall call this function that returns the root object $root(I_i) : A \rightarrow$ **Set** for some instance I_i.

Definition 20—Let **Facet**$_M$ be a meta-facet category for categories **Facet**$_0$,…, **Facet**$_N$, containing a meta-object and the roots of the others:

$$Ob(\mathbf{Facet}_M) = M \cup root(I_0) \cup \ldots \cup root(I_N)$$

M is a meta-object sharing a relationship with every object: $Hom_{\mathbf{Facet}_M}(M, x)$ for each $x \in Ob$(**Facet**$_M$).

Fig. 6 illustrates adding a meta-facet to join together a collection of facets; each black subtree represents a particular facet type. M is a new meta-object that must be created as well as the gray and dotted arrows that link this meta-object and the roots of the other facet graphs.

Let us define the union of two underlying graphs, U(**Facet**$_i$) and U(**Facet**$_j$), as the union of its constituent parts. By definition, the sets of vertices and arrows for graphs underlying two **Facet** categories, **Facet**$_i$ and **Facet**$_j$, are disjoint and can be merged with the union of corresponding vertices and arrows; this leaves the graph disconnected, since no object of **Facet**$_i$ and **Facet**$_j$ is in common.

Using the root of each instance and a meta-facet, we can create a new instance connecting every other underlying graph to our meta-facet:

Definition 21—The merger of instances $I_0, I_1, \ldots, I_N$ of categories **Facet**$_0$,…, **Facet**$_N$ is a new instance I_M on $(G_U, \simeq_U)$ where:

1. $G_U = U(\mathbf{Facet}_0) \cup \ldots \cup U(\mathbf{Facet}_N) \cup U(\mathbf{Facet}_M)$. This is the union of the underlying graphs of the meta-data facet and the facets that are merging.
2. $\simeq_U$ is a congruence on G_U. We define this the same as in Section III-C but do note that the collection of paths have grown. No two paths in the merging categories conflict because the facets are disjoint by definition.

The merged instance I_M is not defined much differently than $I_0,\ldots, I_N$ in that it still maintains (*Facet, Ancestor*) : $F \rightarrow$ **Set** function mappings; the only difference is that the underlying graph has changed with additional path considerations. The merge operation is simply a transformation: we are manipulating the facets into a graph and symbolically merging graphs to suit our needs. The information regarding classified resources that is embedded into each facet gets reused; only the surrounding structure changes.

C. Implementation

If we connect this back to our notion that schemas are not structurally different than facets, it is clear that I_M is simply another table containing $N+1$ relationships with entries from the **Facet**$_0$,…, **Facet**$_N$ categories sharing a relationship with the meta-facet. The foreign keys of these meta-relationships would simply point back to the roots of the other facets; this enables reuse in-place without needlessly copying data. Furthermore, this gives a clear implementation path for enabling reusable terminologies in a standard relational database, where tables help structure facets and the resources that have been classified accordingly. If a relational database is not possible for the application, then an equivalent scheme can be mimicked in other environments. For example, a web-application could use JSON (Javascript Object Notation) data interchange format [23] to store the taxonomy and links to resources.

D. DELVE

DELVE (Document ExpLoration and Visualization Engine) is our framework and application for browsing biomedical literature through heavy use of visualizations [24]. In fact, our motivation for choosing category theory began when first designing DELVE, due to the difficulty in modeling facets that are controlled by visualizations or found within a visualization. In the case of i2b2, the design of the interface insists on merging terminologies together into a master taxonomy that directs exploration within the interface. With DELVE supporting multiple visualizations, a master taxonomy is unrealistic as each visualization potentially requires a different set of facets altogether.

1) Understanding DELVE—In Fig. 7, a query for fibromyalgia is shown. The screen is split into two parts for this example; the abbreviated left-hand side contains a cloud and the right-hand side contains a list of relevant biomedical publications. The default cloud shows the frequency of terms using the MeSH (Medical Subject Headings) vocabulary; librarians at the National Library of Medicine manually review journal articles and tag them with appropriate MeSH terms [25]. MeSH terms are hierarchically organized and are typically accurate reflections of the article's contents since they are manually assigned, making them great facet candidates.

DELVE also provides other collections of terms as facets for two reasons: 1) interdisciplinary collaboration typically creates researchers interested in biomedical literature who are not familiar with MeSH terms and 2) granularity and phrasing of terms can be an issue. For example, a researcher using DELVE queries for fibromyalgia as seen in Fig. 7; they are also interested in *functional somatic syndromes* but this term is not directly available as a MeSH term. Instead, articles covering *functional somatic syndromes* are

typically tagged *somatoform disorders*; without this knowledge, a researcher could miss desired articles. DELVE resolves this issue by providing a list of biomedical trigrams as a facet, which was compiled by analyzing all trigrams found within Pubmed's library of biomedical articles; the phrase *functional somatic syndromes* occurs in great frequency. From a modeling perspective, there are natural differences in the structure of the MeSH hierarchy and the collection of anchoring trigrams, but our categorical model naturally accounts for this by allowing objects to have any inclusive relationship within **Facet** categories: including those who have many (MeSH terms) and those who have none (DELVE's trigrams). In DELVE's case, instances of facets play a role when creating focused collections of documents based on what the user has selected through the interface, which could potentially span one or more facets.

2) Focusing Considerations—The annotated screen-shot in Fig. 8 demonstrates DELVE's ability to use a facet to focus. In this example, a search for fibromyalgia is focused on the MeSH term *analgesics*, which causes the documents viewer to show only those documents that are classified as belonging to the MeSH term *analgesics*. Multiple points of focus are supported in the subsequent version of DELVE, such as focusing using different word clouds and word trees. If the user also selects the MeSH term *female*, the document viewer would only show those documents tagged with both MeSH terms *analgesics* and *female*. Color is used to visually offset the facets that being focused upon. The document viewer ranks according to how many occurrences of the focus terms can be found within the abstract of the article.

Within one faceted taxonomy, aggregating focuses becomes a focused version of the queries discussed in Section II. Suppose the user also wishes to focus on the trigram *functional somatic disorders*. If we have created instances of **Facet** categories as discussed in Section III-C, we can also create instances of focused subcategories by taking a subgraph of the graph underlying **Facet**:

Definition 22: Given instances $I_0, I_1, \ldots, I_N$ of categories $\mathbf{Facet}_0, \ldots, \mathbf{Facet}_N$, let $I_{F0}, I_{F1}, \ldots, I_{FN}$ be focused instances created by replacing $U((\mathbf{Facet}_i))$ with $U(\mathbf{Focus}_i)$ for $i = 0, 1, \ldots, N$.

3) Recalling Resources—At some point during a user's interactive session in a faceted browsing system, it is advantageous or desirable to recall and list all resources that were classified according to a focused selection of facets. When creating instances of our facet categories, we defined a function capable of returning the ancestor of the facet type for a given facet. We can similarly define a function capable of returning focused resources.

Definition 23: Let R be a function defined as $R(Focus, Resource) : \mathbf{Focus} \rightarrow \mathbf{Set}$, where:

1. *Focus* is a function similar to the *Facet* defined in Section III-C: $Focus : V \rightarrow \mathbf{Set}$, so for each vertex $v \in V$ we can recover a set of focused facets denoted **Focus**(v)
2. Resource is a function defined for every focused facet $f \in Focus(v)$ above as $Resource(f) : Focus(v) \rightarrow Resource(f)$.

In other words, similar to how we defined a function *Ancestor* in Section III-C as a self-referential link back to facets, we now define a function that unrolls the foreign relationship between facets and resources. An example of this is seen in Fig 4: the resource with *resource 2* as its key holds a foreign relationship with the medication that has *anti-diabetic* as its primary key. Relating this back the definition above, we rephrase this as: for every facet in the graph, collect their primary keys (PKs) and from the resource table, collect any primary keys where any foreign keys matched the original keys (PKs). At this point, the interface is free to present the resources as needed, which consequentially allows us to model ranking and sorting schemes for resources; we leave these discussions as future work.

V. Future Work

As mentioned previously, a natural consequence of modeling facets, faceted taxonomies, and faceted browsing systems is that resources ultimately get retrieved. This opens the door to abstractly modeling and developing deeper manipulations of faceted data in a way that is transparent and reusable across systems. For example, categorical constructions such as pullbacks and pushouts can help dynamically organize and reorganize faceted data. These types of operations could potentially lead to creating facets dynamically, where new facets are created on the fly from computations involving existing ones.

We are developing an application programming interface (API) for faceted browsing and wish to include support for interfaces that require multiple heterogeneous terminologies. The mapping between schemas and facets clears the path to implementation with a database containing faceted data and taxonomies. Support for functional databases is growing [17], [18], but a traditional relational database is adequate. An API for faceted browsing can bridge the gap between a categorical model for faceted browsing and databases, allowing us to start with a traditional relational databases and migrate towards functional databases as they mature.

The impact that visualizations play in faceted browsing systems deserves to be explored further. In systems such as DELVE, one interaction can have consequences in many parts of the interface. Ultimately, with a categorical model, one will be able to mathematically prove something is possible before implementation; the relationships and road maps between proof and implementation paths need to be researched further.

VI. Conclusions

We extended our category-theoretic model of faceted browsing to support multiple heterogeneous terminologies as facets, which are needed in interfaces where more than one source of information controls the exploration of the data. Two use-cases emerged from our discussions of integrating multiple terminologies: merging instances into a single master and operation considerations when managing multiple facets.

We also showed that facets are categorically similar to database schemas, which allowed us to create instances of facets and faceted taxonomies, and in turn support modeling heterogeneous terminologies as facets. Our model had already been demonstrated to encourage the reuse and interoperability of existing facet models [6], but the extensions

presented today encourage the reuse of existing terminologies and provides a clear path to integrating them as controllable facets within a faceted browsing system.

Acknowledgments

The project described was supported by the National Center for Research Resources and the National Center for Advancing Translational Sciences, National Institutes of Health, through Grant UL1TR000117. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

References

1. Wei B, Liu J, Zheng Q, Zhang W, Fu X, Feng B. A survey of faceted search. Journal of Web engineering. 2013; 12(1–2):41–64.
2. Hearst MA. Clustering versus faceted categories for information exploration. Communications of the ACM. 2006; 49(4):59–61.
3. Hearst, MA. SIGIR workshop on faceted search. ACM; 2006. Design recommendations for hierarchical faceted search interfaces; p. 1-5.
4. W. H. Organization. et al. The ICD-10 Classification of Mental and Behavioural Disorders: Clinical descriptions and diagnostic guidelines. World Health Organization; 1992. The icd-10 classification of mental and behavioural disorders: clinical descriptions and diagnostic guidelines.
5. Murphy SN, Weber G, Mendis M, Gainer V, Chueh HC, Churchill S, Kohane I. Serving the enterprise and beyond with informatics for integrating biology and the bedside (i2b2). Journal of the American Medical Informatics Association. 2010; 17(2):124–130. [PubMed: 20190053]
6. Harris, DR. Information Reuse and Integration (IRI), 2015 IEEE International Conference on. IEEE; 2015. Modeling reusable and interoperable faceted browsing systems with category theory; p. 388-395.
7. Harris, DR. Foundations of reusable and interoperable facet models using category theory; Information Systems Frontiers. 2016. p. 1-13.[Online]. Available: http://dx.doi.org/10.1007/s10796-016-9658-6
8. Fagan JC. Usability studies of faceted browsing: a literature review. Information Technology and Libraries. 2013; 29(2):58–66.
9. Niu, X., Hemminger, B. Analyzing the interaction patterns in a faceted search interface. Journal of the Association for Information Science and Technology. 2014. [Online]. Available: http://dx.doi.org/10.1002/asi.23227
10. Chu, H-J., Chow, R-C. Information Reuse and Integration (IRI), 2010 IEEE International Conference on. IEEE; 2010. An information model for managing domain knowledge via faceted taxonomies; p. 378-379.
11. Prieto-Díaz, R. Information Reuse and Integration, 2003 IRI 2003 IEEE International Conference on. IEEE; 2003. A faceted approach to building ontologies; p. 458-465.
12. Dawson, A., Brown, D., Broughton, V. Aslib proceedings: new information perspectives. Vol. 58. Emerald Group Publishing Limited; 2006. The need for a faceted classification as the basis of all methods of information retrieval; p. 49-72.
13. Spivak, DI. Category Theory for the Sciences. MIT Press; 2014.
14. Coecke, B., Paquette, ÉO. New Structures for Physics. Springer; 2011. Categories for the practising physicist; p. 173-286.
15. Phillips S, Wilson WH. Categorial compositionality: A category theory explanation for the systematicity of human cognition. PLoS computational biology. 2010; 6(7):e1000858. [PubMed: 20661306]
16. Spivak DI, Giesa T, Wood E, Buehler MJ. Category theoretic analysis of hierarchical protein materials and social networks. PLoS One. 2011; 6(9):e23911. [PubMed: 21931622]
17. Spivak DI. Simplicial databases. arXiv preprint arXiv:0904.2012. 2009
18. Spivak DI. Functorial data migration. Information and Computation. 2012; 217:31–51.
19. Barr, M., Wells, C. Category theory for computing science. Prentice Hall; New York: 1990.

20. Karwin, B. SQL antipatterns: avoiding the pitfalls of database programming. 1st. Pragmatic Bookshelf; 2010.
21. Bodenreider O. The unified medical language system (umls): integrating biomedical terminology. Nucleic acids research. 2004; 32(suppl 1):D267–D270. [PubMed: 14681409]
22. C. for Medicare & Medicaid Services. Healthcare Common Procedure Coding System (HCPCS). Centers for Medicare & Medicaid Services; 2003.
23. Bray, T. The javascript object notation (json) data interchange format. Mar. 2014 retrieved June 15, 2016 from https://tools.ietf.org/html/rfc7159/
24. Harris, DR., Kavuluru, R., Yu, S., Theakston, R., Jaromczyk, JW., Johnson, TR. Proceedings of the summit on clinical research informatics. AMIA; 2014. Delve: A document exploration and visualization engine; p. 179
25. Lowe HJ, Barnett GO. Understanding and using the medical subject headings (mesh) vocabulary to perform literature searches. Journal of the American Medical Association. 1994; 271(14):1103–1108. [PubMed: 8151853]

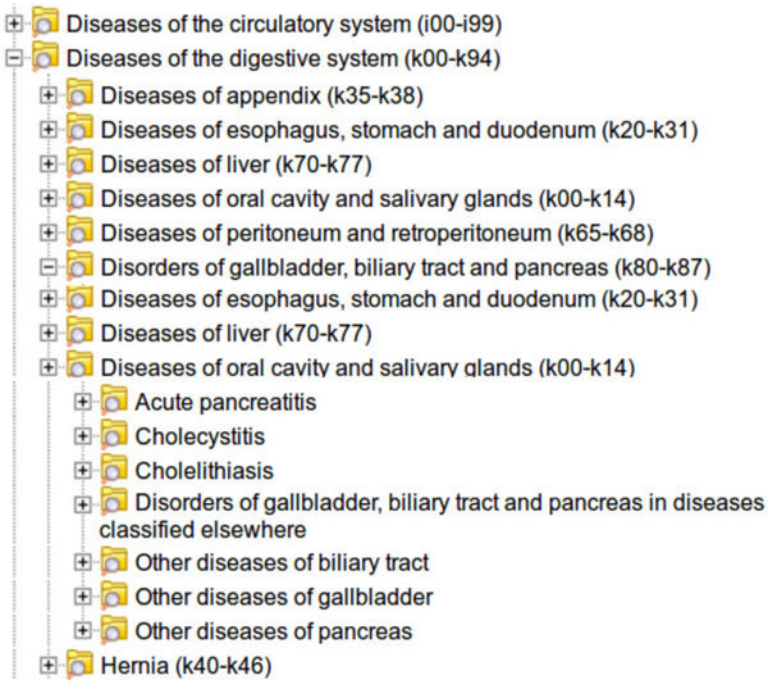

Fig. 1.

Users can select from a variety of biomedical facets within i2b2, including those from existing and well-known terminologies; a subset of the ICD10 terminology as viewed through the i2b2 query tool is shown here.

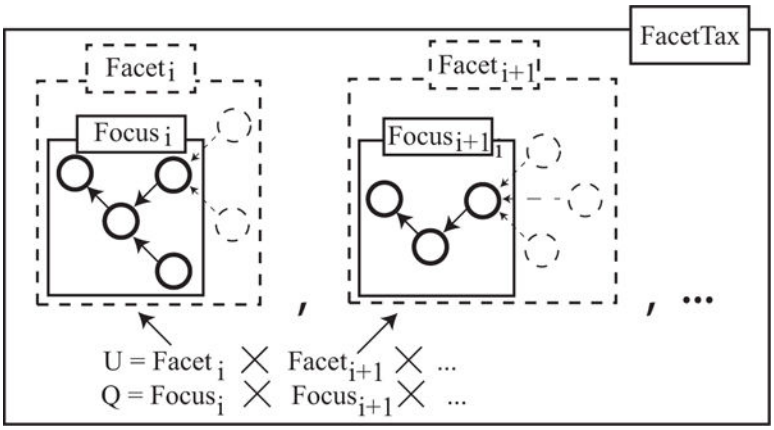

Fig. 2.

The structure of facet, focus, and taxonomy are easy to visualize due to their natural hierarchical relationships. Universes and queries are products utilizing this structure.

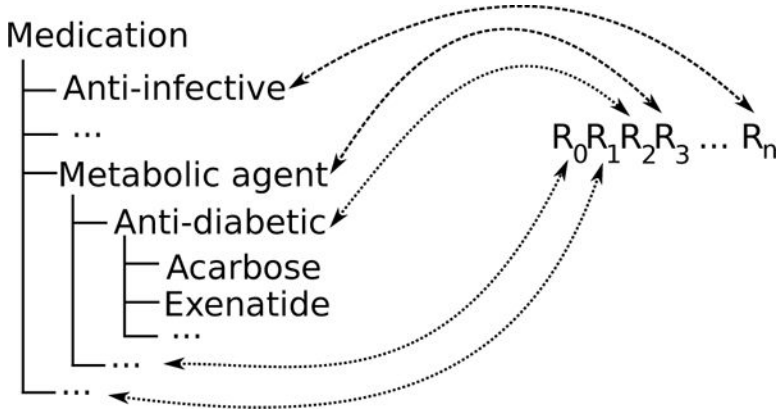

Fig. 3.

We show a sample faceted taxonomy for medications. The objects of each **Facet** are pointers to a resource that has been classified as belonging to that particular facet type.

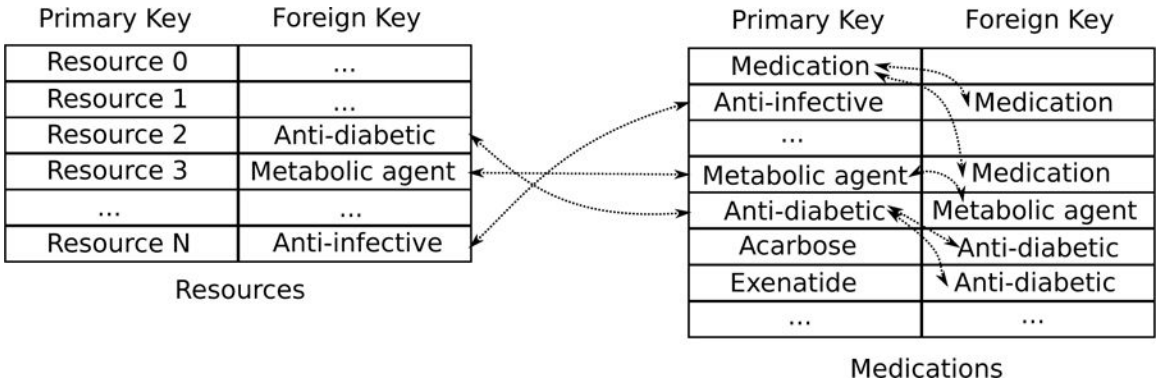

Fig. 4.

A resource table and a medications table using example data from Fig. 3 shows the role that primary and foreign keys play in modeling faceted browsing.

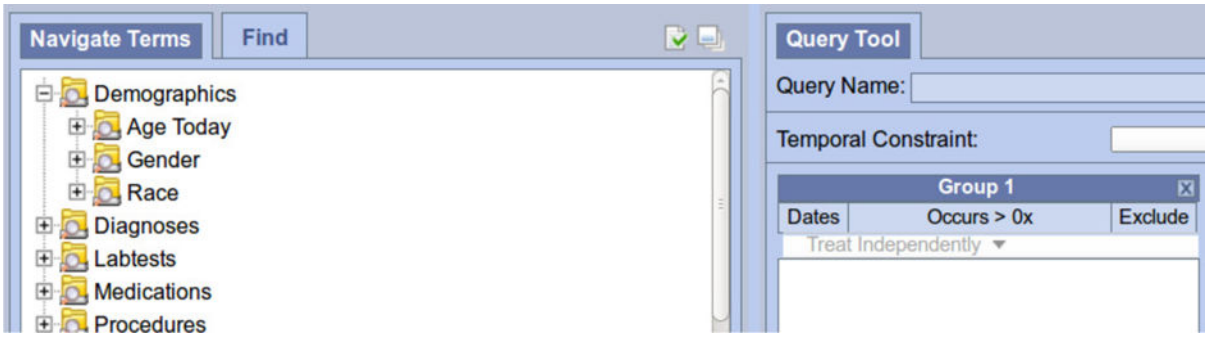

Fig. 5.
The i2b2 query tool uses drag-and-drop interaction to construct queries to find patients.

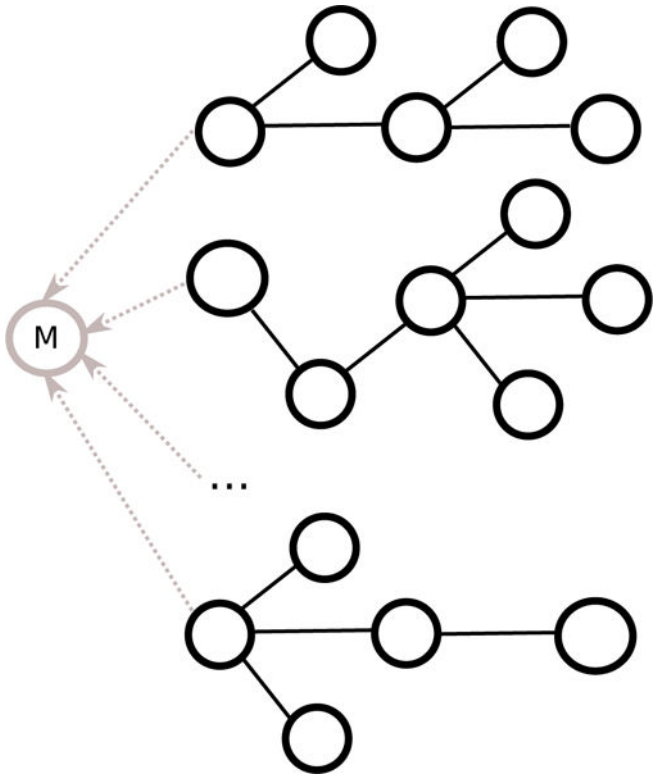

Fig. 6.
A meta-facet can assist in merging facets together by providing a common anchor point.

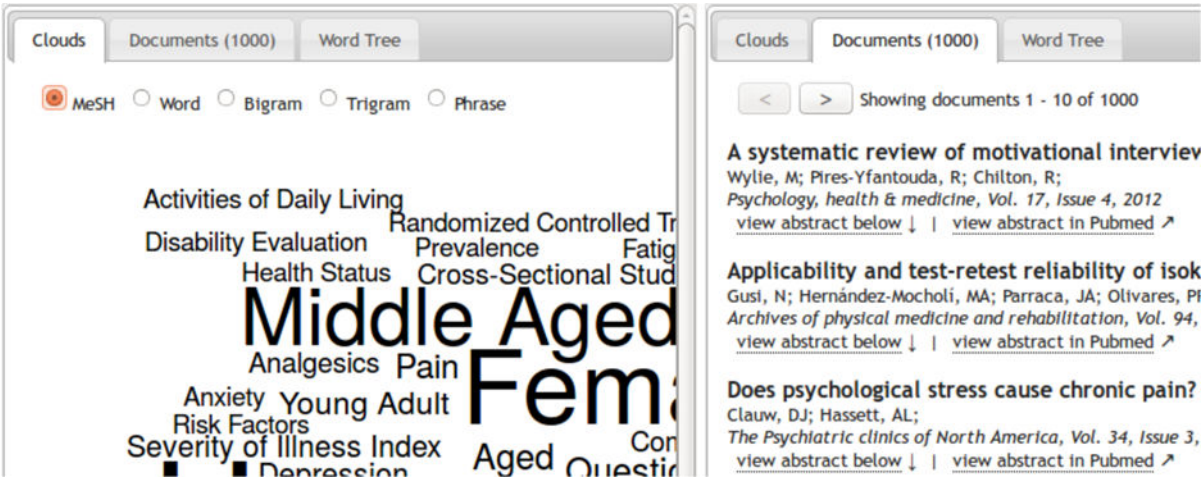

Fig. 7.

DELVE contains visualizations controlled by facets as well as visualizations that contain facets.

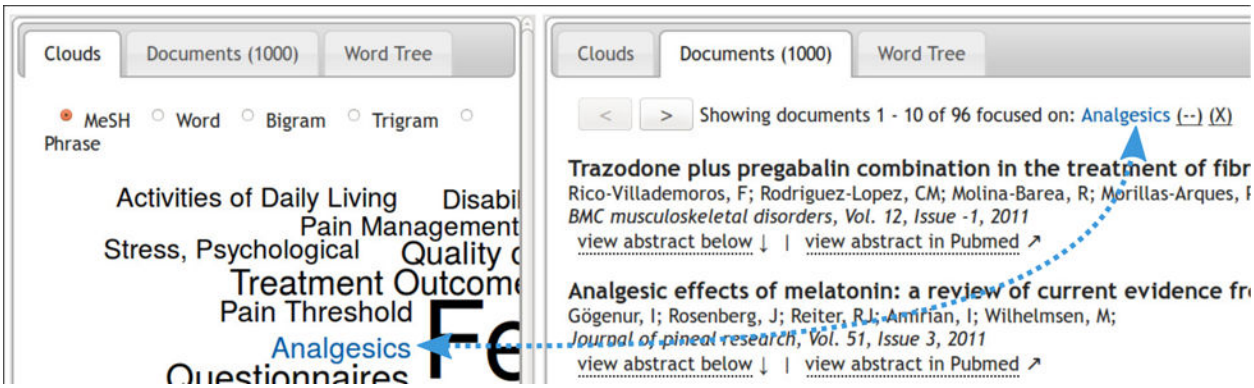

Fig. 8.

A DELVE search for fibromyalgia publications focusing on analgesics