# Extending the Simple Knowledge Organization System for Concept Management in Vocabulary Development Applications

**Joseph T. Tennis and Stuart A. Sutton**
*Box 352840, Mary Gates Hall, Suite 370, Information School of the University of Washington, Seattle, 98195–2840. E-mail: {jtennis, sasutton}@u.washington.edu*

**In this article, we describe the development of an extension to the Simple Knowledge Organization System (SKOS) to accommodate the needs of vocabulary development applications (VDA) managing metadata schemes and requiring close tracking of change to both those schemes and their member concepts. We take a neopragmatic epistemic stance in asserting the need for an entity in SKOS modeling to mediate between the abstract concept and the concrete scheme. While the SKOS model sufficiently describes entities for modeling the current state of a scheme in support of indexing and search on the Semantic Web, it lacks the expressive power to serve the needs of VDA needing to maintain scheme historical continuity. We demonstrate preliminarily that conceptualizations drawn from empirical work in modeling entities in the bibliographic universe, such as works, texts, and exemplars, can provide the basis for SKOS extension in ways that support more rigorous demands of capturing concept evolution in VDA.**

## Introduction

This article explores the conceptual issues around the modeling of concept change in schemes (i.e., controlled vocabularies) using the Simple Knowledge Organization System (SKOS) as those changes occur in a vocabulary development application (VDA). In particular, we report on the need to extend SKOS to provide a full history of a vocabulary's development both within a single VDA and in communication of that encoded history among independent VDAs and other forms of registries.

The issues of managing concept change in schemes and maintaining a history of those changes are not new. Major schemes such as the Universal Decimal Classification (UDC) and the Dewey Decimal Classification have tracked change through records for each class number. These records list previous terms and, in so doing, provide a record of change similar in function to the work we describe here. Examples of these change-management formats can be found for UDC (UDCC, 2003) and for the MARC Classification Format (Library of Congress, 2005). In addition, concept records are used in the creation of thesauri and contain fields to track changes (Anderson & Perez-Carballo, 2005; Soergel, 1974).

While editors of these mature schemes use the various records in managing change, users of the systems that consume these schemes rarely see change information, except, for example, when one term has been replaced by a more current term. We can see this sort of change of terminology with racial groups in the Library of Congress Subject Headings. For example, online public access catalogs can redirect searchers to African Americans because the term "Afro-Americans" is no longer used.[1]

The mature VDAs noted earlier manage their various change records outside the formal modeling of their schemes. As a result, these records of change are not machine actionable or machine communicable in a standard way outside their proprietary domains. The work reported here seeks to address the need to formally model change in the context of a scheme modeling language and to make the histories of schemes amenable to the Semantic Web.

Our specific need to address concept change in VDA emerged through our ongoing work on the National Science Digital Library (NSDL) Metadata Registry (hereafter, Registry). The intent of the Registry is to build on work in the Dublin Core community and elsewhere on metadata registries for the Semantic Web. The goal of the Registry is to enable both collection holders and the various applications that generate, consume, and process metadata to identify, declare, and publish their metadata schemas (element/property sets) and schemes (value spaces/controlled vocabularies) in support of discovery, reuse, standardization, and interoperability within NSDL and globally.

Received March 12, 2007; revised May 26, 2007; accepted May 27, 2007

 ● Published online 18 October 2007 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/asi.20702

[1]See http://tinyurl.com/2sdnq6 for an example of OPAC redirect using change information of the sort described.

The registry framework under development extends the contemporary notions of metadata registry beyond the functions of aggregation of schemes and schemas declared elsewhere and the general Web exposure of those schemes and schemas to include what Miles (2006, p. 58) defined as various levels of functionality including a VDA. Thus, the NSDL registry not only provides scheme and schema aggregation and exposure services but also VDA and scheme/schema namespace services (i.e., scheme/schema hosting) for maintenance agencies unable to manage these functions locally. There also is hope that the registry might provide a means of transitioning orphaned legacy schemes/schemas to the context of the Web. The emerging core functionality of the registry has been described elsewhere by Hillmann, Sutton, Phipps, and Laundry (2006). The concern of the article is with that part of the Registry work focusing on the VDA in general and in conceptual issues in modeling change in scheme concepts and the creation of historical snapshots of those changes in particular.

For scheme modeling within the Registry, we have adopted SKOS as our reference model. SKOS provides the means "for expressing the basic structure and content of concept schemes (thesauri, classification schemes, subject heading lists, taxonomies, terminologies, glossaries and other types of controlled vocabulary)" (Miles & Brickley, 2005a). As an application of Resource Description Framework (RDF), SKOS is well suited to deployment on the semantic web. While schemes expressible in SKOS can be modeled as Resource Description Framework Schema/Web Ontology Language (RDFS/OWL) ontologies, in many circumstances, the simpler SKOS expression is more appropriate because it is a lightweight standard compared to RDFS/OWL and is seen as satisfying most of the requirements set for using schemes in the semantic Web environment (Mikhalenko, 2005).

While SKOS fulfills this need for both simplicity and sufficient power to support search in the context of the Semantic Web while providing a means to "minimize the cost and maximize the utility associated with controlled vocabularies" (Miles, 2006, p. 63), without extension, it lacks the expressive power necessary for the long-term documentation, management, and evolution of a controlled vocabulary in a mature VDA. Therefore, the focus of this article is on the conceptual issues around the extension of SKOS to provide the missing expressive power.

Before setting out the nature of the problem, we must first define the terms used in our discussion. In this article, we avoid use of the terms "version" and "versioning" with regard to changes in scheme concepts, and reserve their use to significant changes to a scheme as a whole that result in a formal edition declaration by the scheme's maintenance agency.[2] We use the term "snapshot" to denote the state of a scheme at any arbitrary, but identifiable, point in time. Snapshots record the then-current state of the scheme's various concepts, relationships among those concepts, and the scheme documentation. Any snapshot may be declared a formal release version or edition by the scheme's maintenance agency; however, it is quite possible in a digital environment for there to be no declared formal versions of a scheme past the scheme's initial release—relying instead on a continuously evolving set of temporal snapshots.

In this article, we are concerned solely with issues of representing change states in a concept as it evolves within a scheme. Thus, the object of our concern is inextricably bound to our notion of the scheme snapshot since the distinction between one snapshot and the next for a given scheme is partially a function of this process of concept evolution.

## Nature of the Problem

The problem we seek to address in modeling concept change can be briefly illustrated by exploring the Dublin Core Metadata Initiative (DCMI) documentation of terms in its DCMI *Resource Type Vocabulary* used in metadata for denoting the genre of a resource. Like concepts in many, if not most mature schemes, DCMI *Type Vocabulary* terms have been refined over time for clarity and other purposes. Those changes have been captured in DCMI documentation. For example, see *DCMI Terms: A Complete Historical Record* (http://dublincore.org/usage/terms/history/) for a listing of all changes that have occurred to DCMI vocabulary terms. Table 1 illustrates three iterations of evolutionary changes in the *DCMI Type Vocabulary* to the "Image" concept denoted in the table as Image-001, Image-002, and Image-003.

These revisions illustrated in Table 1 are the sort of common refinements of concepts that occur in the management of schemes. The first revision (Image-002) occurred in 2003 and represents a structural change in the scheme with a new set of relationships declared between "Image" and two new narrower terms: "Still Image" and "Moving Image." The second revision (Image-003) occurred in 2006 and represents a fine tuning of the text, with explanatory information moved from the concept definition to comments.

Note that in the process of refinement of "Image," the concept Uniform Resource Identifier (URI) does not change. Thus, the core identity of the concept as an abstract notion is maintained.[3] What can be inferred from a comparison of the three instances in Table 1 is that our understanding of the *current state* of the DCMI concept "Image" is operationally a function of the URI for the abstract concept (i.e., http://purl.org/dc/dcmitype/Image) *and* the values used to describe the concept expressed in the most recent instance of the concept, which is identified with its own URI (i.e., http://dublincore.org/usage/terms/history/#Image-003).

[2]Significant changes triggering a new scheme version might include new scheme documentation that expresses a significant shift in the purpose, use, or architecture of the scheme.

[3]DCMI also maintains the abstract nature of the properties in its namespaces in the same manner when it makes refining changes to those properties. The abstract property is consistently and persistently identified through the property URI while historical instantiations of the property are referenced by their own URIs.


DOI: 10.1002/asi

TABLE 1. Concept changes to the *DCMI Resource Type Vocabulary* term "Image."

| | Image-001 | Image-002 | Image-003 |
|---|---|---|---|
| Concept URI | http://purl.org/dc/dcmitype/Image | http://purl.org/dc/dcmitype/Image | http://purl.org/dc/dcmitype/Image |
| Name | Image | Image | Image |
| Label | Image | Image | Image |
| Definition | An image is a primarily symbolic visual representation other than text. For example, images and photographs of physical objects, paintings, prints, drawings, other images and graphics, animations and moving pictures, film, diagrams, maps, musical notation. Note that image may include both electronic and physical representations. | An image is a primarily symbolic visual representation other than text. For example, images and photographs of physical objects, paintings, prints, drawings, other images and graphics, animations and moving pictures, film, diagrams, maps, musical notation. Note that image may include both electronic and physical representations. | A visual representation other than text. |
| Comment | – | – | Examples include images and photographs of physical objects, paintings, prints, drawings, other images and graphics, animations and moving pictures, film, diagrams, maps, musical notation. Note that image may include both electronic and physical representations. |
| Broader Than | – | http://purl.org/dc/dcmitype/StillImage | http://purl.org/dc/dcmitype/StillImage |
| Broader Than | – | http://purl.org/dc/dcmitype/MovingImage | http://purl.org/dc/dcmitype/MovingImage |
| Decision URI | http://dublincore.org/usage/decisions/#Decision-2000-02 | http://dublincore.org/usage/decisions/#Decision-2003-02 | http://dublincore.org/usage/decisions/#Decision-2006-02 |
| Version URI | http://dublincore.org/usage/terms/history/#Image-001 | http://dublincore.org/usage/terms/history/#Image-002 | http://dublincore.org/usage/terms/history/#Image-003 |
| Replaced by URI | http://dublincore.org/usage/terms/history/#Image-002 | http://dublincore.org/usage/terms/history/#Image-003 | – |

We can model the *Resource Type Vocabulary* example as a set of resources with relationships as illustrated in Figure 1 using properties from the SKOS namespace and others from an example registry namespace ("skos" and "ex," respectively).

The DCMI abstract concept "Image" in Figure 1 is instantiated here in the *DCMI Type Vocabulary* namespace by means of the skos:inScheme property. In like fashion, each of the three concept instances in Figure 1 also are declared within the *Resource Type Vocabulary* history through skos:inScheme.

We also can observe in Figure 1 that there is no direct relationship between the concept's abstract identity and its three historical instances stemming from the fact that the instantiation of the abstract concept occurs directly through the skos:inScheme declaration and not by means of the three concept instances. DCMI handles this problem by repeating the values from the most recent historical instance in the namespace declaration for the concept—thus our earlier assertion that we "know" the most recent concept by means of a merger of the concept's abstract identity (i.e., http://purl.org/dc/dcmitype/Image) and the values expressed in the most recent historical instance (i.e., http://dublincore.org/usage/terms/history/#Image-003).

Prior to the first change in the "Image" concept in the DCMI namespace, there was no apparent need to express historical instances, and as a result, the abstract and the concrete were conveniently merged as in Figure 2.

We preliminarily note that the SKOS modeling of its notion of concept follows this same pattern of merger of the abstract and the concrete seen in Figure 2—a topic to which we will return shortly.

While there is much to be said for this simple means of expressing a snapshot of the current state of a scheme (even when laden with historical notes chronicling its evolution), such an expression falls short if one needs to go further by modeling the scheme's evolutionary path as exemplified in the DCMI *Resource Type Vocabulary* example.

In Figure 3, we refine the DCMI example and foreshadow the SKOS extension we propose. The major difference between the modeling here and that in Figure 1 is the reformulation of the relationships between the abstract concept "Image," its three historical instantiations, and the DCMI Type scheme. The resulting assertions are that an abstract concept can be instantiated in a scheme only by means of a concept instance and that we "know" the concept—both now and historically—by means of its concept instances. As we shall demonstrate more fully later, this mediating entity between the abstract concept and the scheme is fundamental to managing change in VDA.

In essence, the fundamental entities and relationships inherent in the *Resource Type Vocabulary* example as


DOI: 10.1002/asi

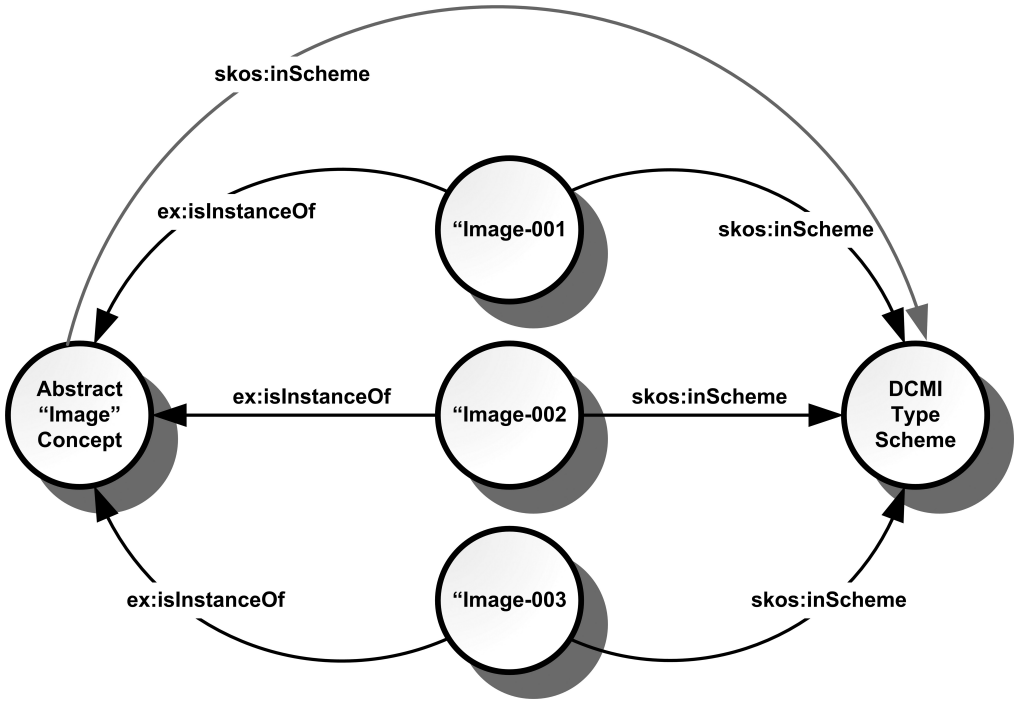



FIG. 1. Relationships among DCMI *Resource Type Vocabulary* concepts, concept instances, and the scheme in which those instances are declared.

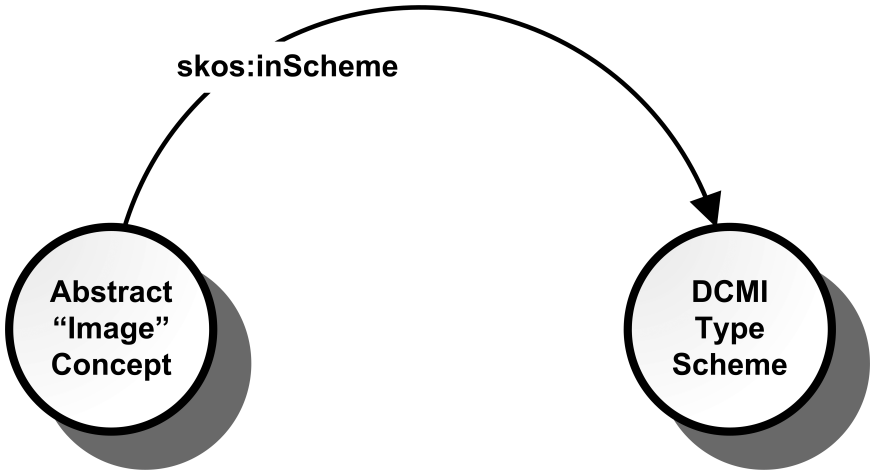



FIG. 2. The abstract concept instantiated in the DCMI *Resource Type Vocabulary* scheme.

extended in Figure 3 can be made explicit by formalizing the notion of the *concept instance* in the context of the following core assertions.

- *Core Assertion 1:* A *concept* is an "abstract idea or notion; a unit of thought" (Miles & Brickley, 2005b) identified by URI;[4]
- *Core Assertion 2:* A *concept instance* is a concrete manifestation of a *concept* within a *scheme* and is identified by URI;
- *Core Assertion 3:* A *scheme* is a collection of *concept instances* and is identified by URI;
- *Core Assertion 4:* A *scheme* may embody more than one *concept instance* of the same *concept* (e.g., a historical sequence of instances reflecting change states).
- *Core Assertion 5:* A scheme snapshot is a point-in-time image of the state of scheme concepts, relationships, and documentation.

It is the conceptualization of the SKOS notion of concept as an abstract entity in Core Assertion 1 that is the crux of the problem to be addressed in making SKOS amenable to representing concept evolution in VDA. Figure 4 illustrates the

[4]Throughout this article, we use URI in the sense defined by W3C at http://www.w3.org/Addressing/


DOI: 10.1002/asi

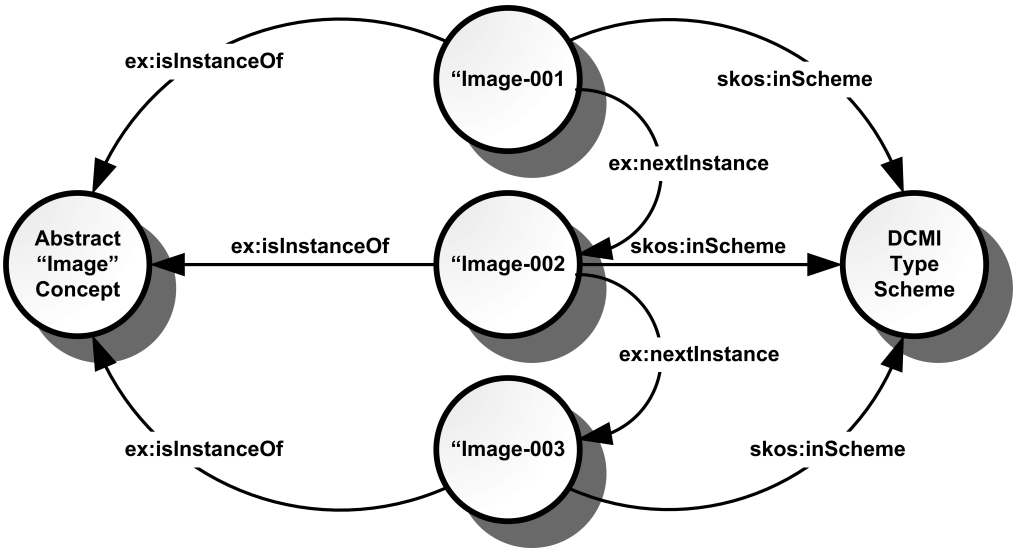



FIG. 3. DCMI *Resource Type Vocabulary* concept relationships revised to integrate concept instances.

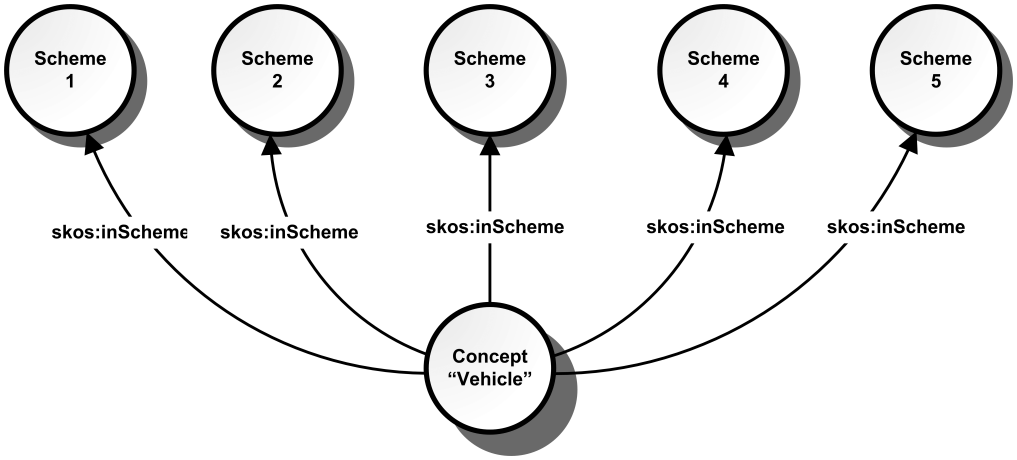



FIG. 4. Example of an abstract concept instantiated in multiple schemes.

case of a single concept's existence in multiple schemes. Note the similarity between the pattern of instantiation denoted here and the DCMI form of instantiation noted in Figure 2.

We can easily assume that over a period of time, a specific concept such as "Vehicle" in Figure 4 might well be instantiated in various concept schemes in domains maintained by agencies other than the maintenance agency of the abstract concept as well as in distinct versions of the concept's original scheme, differentiated by relationship structure, scope notes, and links to collections described. We also may assume that the concept will evolve independently in these various schemes through the processes already noted. Since the SKOS documentation properties have the abstract concept as their subject, we can refine the substance of Figure 4 with various change notes and other documentation, as illustrated in Figure 5.

Figure 5 reflects the current problem in the SKOS model we must face when maintaining a formal representation of incremental change in concept state as reflected in snapshots within a scheme since it is not possible in the figure to ascertain any relationships between any specific change and the scheme in which that change occurred.

In the remainder of this article, we will propose a solution to this problem via a new entity we call the *concept instance*.[5] We will approach this SKOS extension by first briefly

[5]We are aware that our use of the term "concept instance" may be problematic for the knowledge organization and Semantic Web communities, where the term has relatively established meanings that may be seen to be at odds with the definition established here. We easily could have spoken of "concept expressions" instead of "concept instances" with no loss to our intended meaning with, perhaps, a more intuitive framing as concepts and their expressions.


DOI: 10.1002/asi

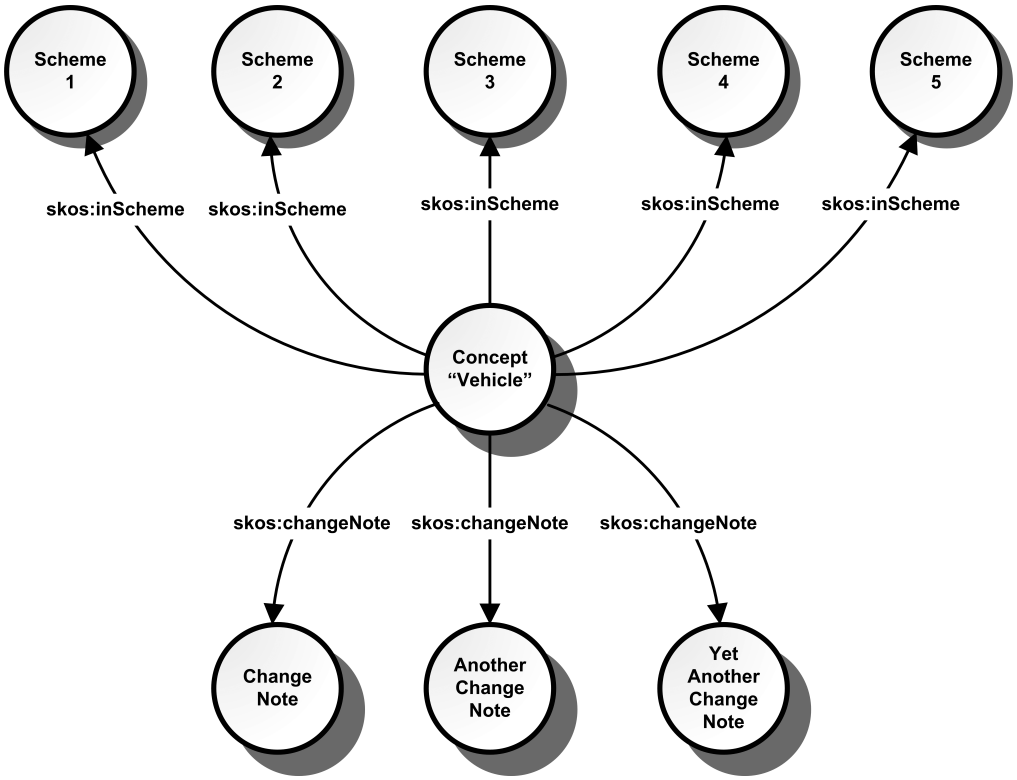



FIG. 5. Concept example with multiple schemes and documentation for multiple changes.

exploring such an entity's associations with abstract models of the bibliographic universe by Patrick Wilson (1968), Richard Smiraglia (2001), and IFLA (1998). We will conclude the article by illustrating the proposed extension of SKOS to include the *concept instance* as a means for modeling select forms of change in the context of VDA.

## A Conceptual Foundation for Concept Instances

Identifying a conceptual foundation for concepts is not without problems. Indexing research has a suite of related constructs: aboutness, topic, concept, and theme, to name a few (Lancaster, 2003). Historically, scholars have swung from highly developed definitions of concept (Dahlberg, 1979) to less-developed definitions (Preschel, 1972; Tinker, 1966). To define and root concept and concept instances, as we propose, in one epistemic stance is to open that stance to criticism from proponents of another. Risking such criticism, we assert here that concepts are useful conventions employed by designers and indexers to facilitate retrieval and display. This is a neopragmatic stance, as articulated by pragmatist Richard Rorty (1982).

Rorty's (1982) stance follows from an interpretation of John Dewey (1930) and Michel Foucault (1972, 1973, 1980). In a discussion comparing two methods of inquiry available to social scientists, Rorty dismissed both the Galilean (scientific and experimental design) approach and the hermeneutical (interpretive) approach of 20th-century social science. In his discussion, Rorty sought to move knowledge creation beyond metaphors of representation—that is, beyond objectively representing an explanation or an understanding—as final products of both Galilean social science and hermeneutic social science, respectively (pp. 195–198). Instead of claiming to represent knowledge of the social world objectively, the neopragmatic approach creates a vocabulary to cope with the social world. That is, social science research should simply aid us in choosing which actions to take. We follow Rorty in this article in terms of his neopragmatic approach. The outcome of this work is a vocabulary that helps us cope with concept change—a vocabulary used to further that basic and applied conversation.

For our purposes, concepts are known through a combination of texts (i.e., literary warrant, scheme warrant, and user warrant) and are made manifest at the hands of the interpreter, whether he or she is the designer, the indexer, or the user of the scheme. This position follows Wilson's (1968) conception of subject matter of documents as being relative to the representation structures used to interpret a text. Though Wilson never provided a definition of concept for his reader, we can use his notion of the subject matter of a document to demonstrate our position. Wilson advocated for the agnostic view of subject matter—that a document does not have a single subject matter. Though our systems of indexing ask us to present documents on a particular subject matter, Wilson posited that our task always is incomplete, and indefinite. He stated that our task is tenable only because we have provided a sense of position—relative to other declarations of subject matter. Here, he invoked a type of


DOI: 10.1002/asi

warrant—the warrant of the scheme. We know where something is in relation to where it is not. This *negative space* construction of subject matter is akin to our notion of concept and concept instance, with one difference. We are concerned with relative position *over time* whereas Wilson is concerned with relative position *at one point in time*. As can be seen from Wilson's argument and from our Rortian stance for a use-based deployment of meaning, we only can do our best at interpreting the notion of a subject as drawn from a range of texts. In the end, the justification for our work is whether it is useful to retrieval.

By means of metaphor, other constructs are useful in this analysis. The bibliographic construct of the *work* can be seen as analogous to our notion of *concept*. *Alice in Wonderland* is a work we know because we have read or are familiar with different instances of it, whether it be *Alice's Adventures Underground*, Walt Disney's *Alice in Wonderland* sound recording, or *Alice's Adventures in Wonderland: the Pop-Up Adaptation*. The work in this case is a construct used in the bibliographic universe to unite these different instances under a single heading. Thus, in *Alice in Wonderland*, we have what Wilson (1968) portrayed as a "family of texts" united by means of the abstract work. We can adapt this practice to the analysis and construction of concepts and concept instances. We assert that just as we know the *work Alice in Wonderland* by means of its expressions and manifestations, we know a *concept* by means of its instances in schemes. Thus, returning to our DCMI example of the concept "Image," we know it by its various historical instances united as a family by means of the abstract concept. In other words, we recognize the DCMI concept by means of its individual instances and that the concept exists in what is common among all of those instances.

We see other analogous constructs in the literature defining the nature of works in the bibliographic universe. For example, Smiraglia (2001) showed that works and instances (i.e., physical manifestations) are the two predominant bibliographic entities. The IFLA (1998) *Functional Requirements for Bibliographic Records* (FRBR) models the bibliographic universe in terms of four entities—works, expressions, manifestations, and items—of which works and manifestations are of particular note to this article.

FRBR defines a work as an abstract entity. "[T]here is no single material object one can point to as the *work*. We recognize the *work* through individual realizations or *expressions* of the *work*, but the *work* itself exists only in the commonality of content between and among the various *expressions* of the *work*" (*IFLA, p. 16*). Thus, while we can say that *Hamlet* and *Alice in Wonderland* existed as works in the minds of their authors, we came to know those works by means of their first manifestations as play and novel, respectively. As the work evolves, we know it by means of what is common across the work's manifestations.

The SKOS notion of a "concept" is similarly defined as an abstract entity: "An abstract idea or notion; a unit of thought" (Miles & Brickley, 2005b). However, unlike FRBR and the modeling suggested by Wilson (1968) and Smiraglia (2001), the SKOS abstract model provides no separate entity to represent the concrete manifestation of the concept. So instead of some *particular* manifestation of the concept existing in a *particular* scheme, the abstract concept is identical to the concrete manifestation. This mixing of abstract and concrete identities may be acceptable in the simplest of cases where no concept changes have occurred; however, we posit that it is the root of the problem we identified in the text accompanying Figures 4 and 5, where we see a single SKOS concept existing in many schemes and also reflecting various changes in relationship to those schemes.

Our proposed concept instance is somewhat analogous to the IFLA manifestation by which a specific expression of a work in the IFLA model is given tangible form. Thus, an abstract SKOS concept is first known for all practical purposes by means of its initial manifestation in a scheme; that is, through its first concept instance. As in the FRBR model, the notion of interest to us—the concept—"exists" pragmatically only in the commonalities among its various instances as it evolves over time.

## Discussion

In this section, we illustrate the core assertions of modeling concepts and concept instances, and their change through VDA. In so doing, we extend the current SKOS model.

### *Core Assertions 1 and 2 Illustrated*

- *Core Assertion 1:* A *concept* is an "abstract idea or notion; a unit of thought" (Miles & Brickley, 2005b) identified by URI.
- *Core Assertion 2:* A *concept instance* is a concrete manifestation of a *concept* within a *scheme* and is identified by URI.

We assert that the modeling solution to the problem of managing concept change in VDA requires the formal entity of *concept instance* in the abstract model for schemes. Through such an entity, we can draw a useful distinction between the abstract and the concrete, the concept, and its instance. By avoiding the conflation of the abstract and concrete, we can represent incremental concept change in VDA as discrete "snapshots" over time. Figure 6 illustrates the concept instance and Core Assertions 1 and 2.

Figure 7 expands and clarifies the proposed modeling by adding a select, nonexhaustive set of possible properties for concept, concept instance, and scheme.

### *Core Assertions 3 and 4 Illustrated*

- *Core Assertion 3:* A *scheme* is a collection of *concept instances* and is identified by URI.
- *Core Assertion 4:* A *scheme* may embody more than one *concept instance* of the same *concept* (e.g., a historical sequence of instances reflecting change states).

In Figure 8, we illustrate the use of the concept instance for modeling change in a VDA. In the figure, we see a single


DOI: 10.1002/asi

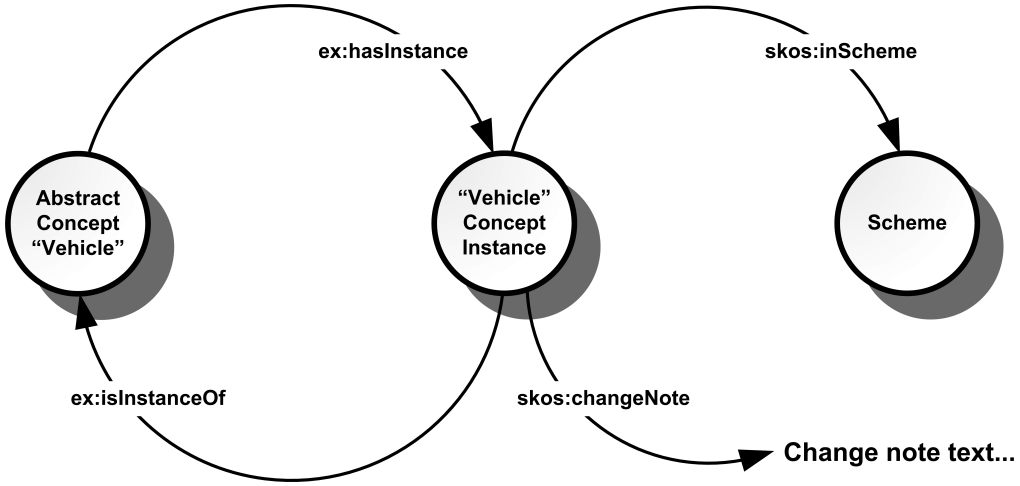



FIG. 6. The concept instance in modeling change.

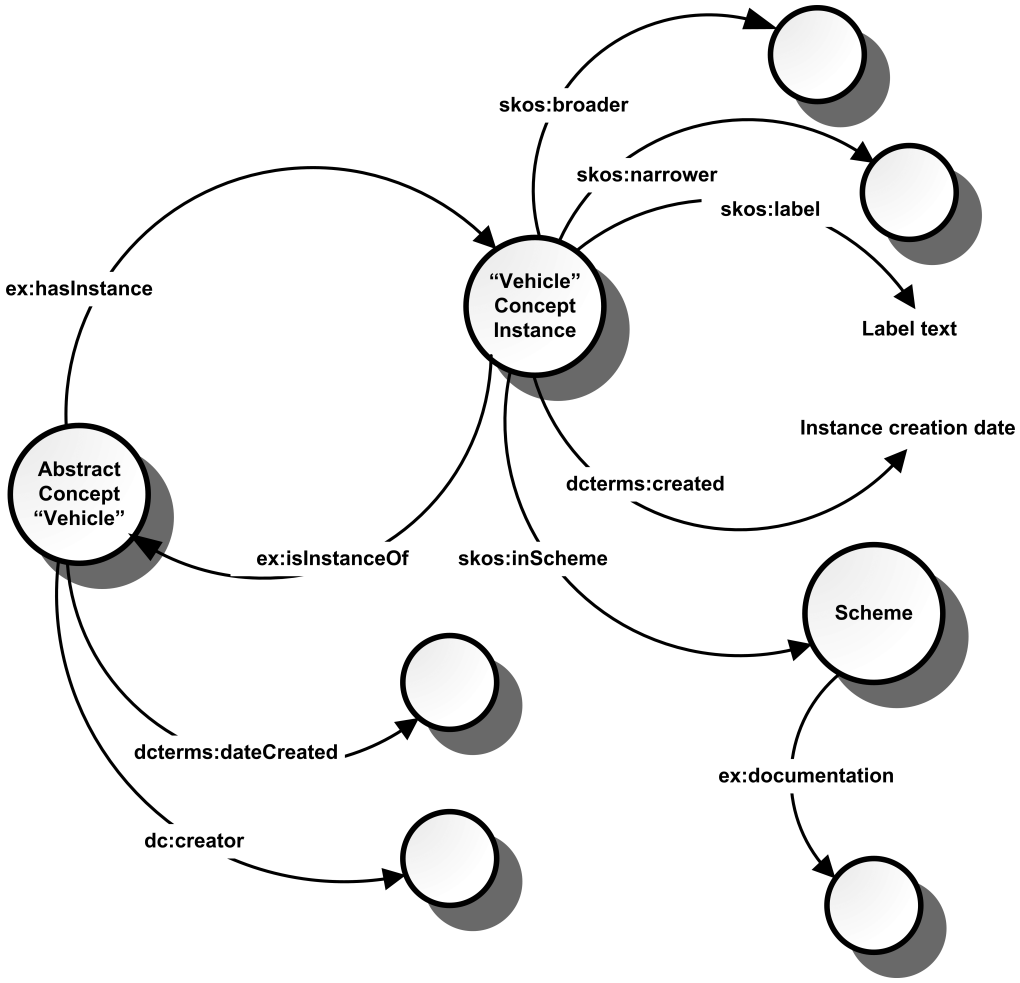



FIG. 7. The concept instance enriched.

abstract concept and two separate schemes. For our purposes, it does not matter whether Schemes 1 and 2 were developed independently of each other or whether Scheme 2 is a new version of Scheme 1. Within each of the two schemes, Concept Vehicle has evolved through two iterations: Instances 1 and 2 in Scheme 1, and Instances 3 and 4 in Scheme 2. Based on SKOS and our earlier discussion, while Concept Vehicle exists as an abstract notion independent of either of the schemes in the figure, it is not *known* until declared in a scheme. This is because we know the concept instance


DOI: 10.1002/asi

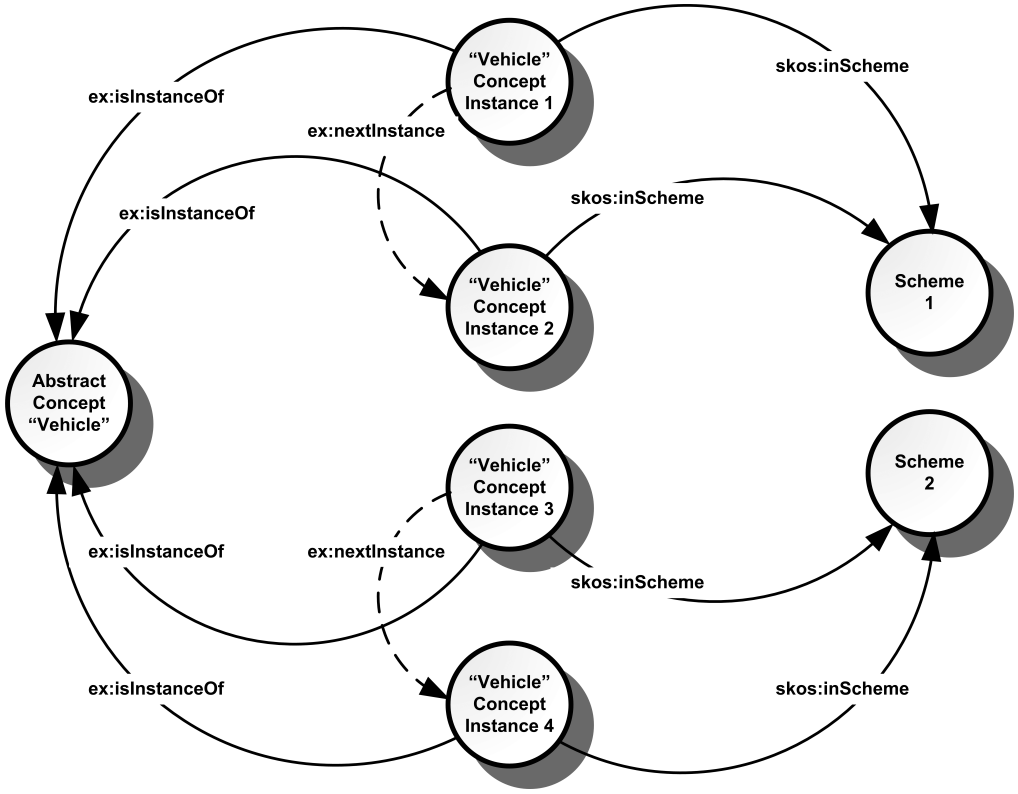



FIG. 8. Concept evolution through concept instances.

through its definition, relationship structure, and deployment. Therefore, Concept Vehicle is *initially known* in Scheme 1 by means of Instance 1 and in Scheme 2 by means of Instance 3. Thus, in a registry we may have *concept precedence* that requires us to examine structural, terminological, and textual aspects of Vehicle Instance 1 before we act in creating Vehicle Instance 2.

We provide a property (ex:nextInstance) to assist in visualizing the sequencing of instances. Depending on other properties used to make statements about the instance (e.g., a date of creation, as illustrated in Figure 7), such an explicit sequencing may not be necessary for the operation of a specific VDA; however, explicit declarations of relationships among instance entities are to be preferred over relationships derived through computations on dates or other values. Scheme snapshots capturing current concept states in Figure 8 are a function of the set of most recent concept instances and the various relationships among them. A snapshot of a particular scheme can be determined at any point in the scheme's history and its concept states revealed—again, including the various relationships among those concepts.

### *Complex Change Illustrated*

- *Core Assertion 5:* A scheme snapshot is a point-in-time image of the state of scheme concepts, relationships, and documentation.

The examples of concept change discussed so far represent relatively simple concept evolution of the sort explored with the DCMI *Resource Type Vocabulary* "Image" concept. Generally, these sorts of change involve simple refinement of the descriptive text used to frame the concept's semantics to achieve greater clarity; however, VDA requires editors to track more complex changes to schemes. For example, we might need to split concepts or lump them together (Tennis, 2005). Modeling more complex forms of change is possible by means of the concept instance, and the snapshot and versioning constructs. In Figure 9, we illustrate one such change: the combining, or *lumping*, of previously separate concepts into a single concept that in this case inherits the semantics of the previously separate concepts.

In Figure 9, we see this more complex example with three abstract concepts shared by Schemes 1 and 2 Version 1 (V.1): Vehicle, Car, and Truck. We also see three snapshots—scheme states as change occurs over time. Let us assume that the maintainers of Scheme 2 have made the decision to use only the concept "Vehicle" to express the meanings previously carried in their indexing by the narrower concepts "Car" and "Truck." The change is accompanied by declaration of a new scheme version: Scheme "2 V.2." Let us assume further that the three abstract concepts of interest have been declared in the context of Scheme 1 and referenced by the maintainers of Scheme 2. The changes made by Scheme 2 have no impact whatsoever on the existence of the three abstract concepts; however, in the context of the declaration


DOI: 10.1002/asi

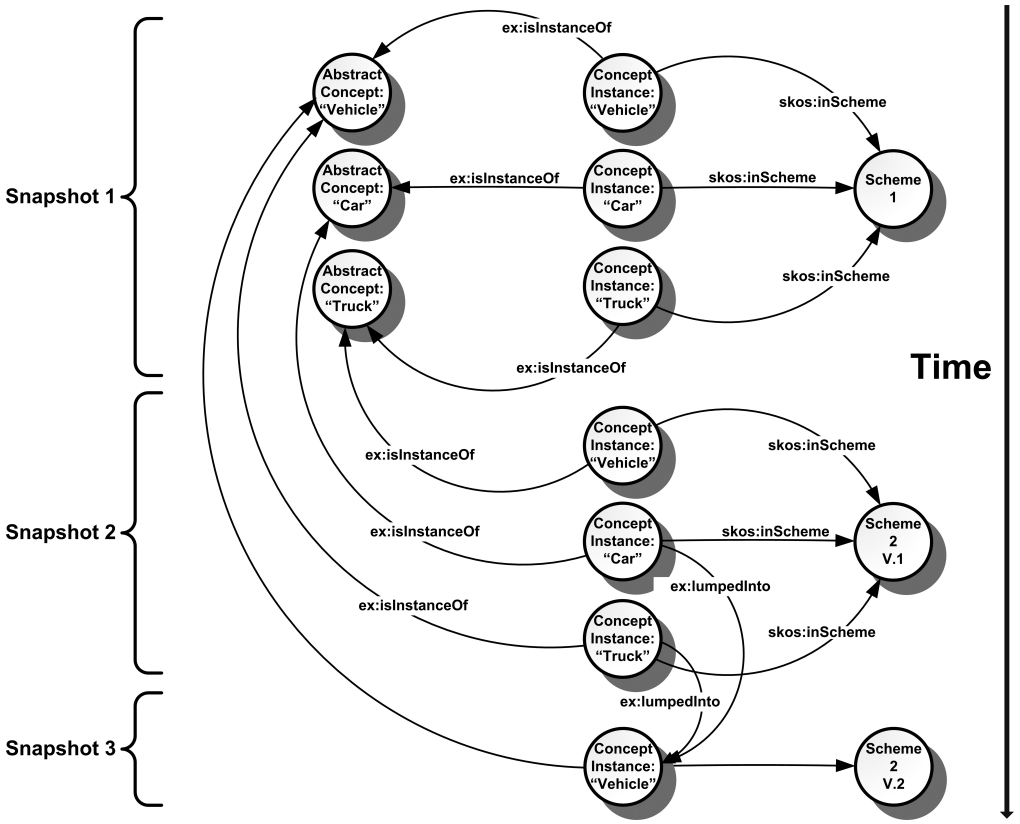



FIG. 9. Concept "lumping" and snapshot references.

of Scheme 2 V.2, the concepts "Car" and "Truck" no longer exist. By means of the concept instances, the history and relationships among the concept instances in Schemes 1, 2 V.1, and 2 V.2 are maintained.

Managing change, whether it is simple or complex, presents a number of system requirements. We have modeled concept change in schemes to elicit those requirements and work toward a specification that extends the current SKOS framework. To do this, we find the need to distinguish between concept and concept instance and between version and snapshot.

## Concept Change in the NSDL Registry

While the conceptualizations described in this article are of a general sort and of potential application in VDA based on reference models other than SKOS, the work described was shaped to no small extent by the actual demands of an emerging system: the NSDL Registry. In Appendixes A and B, we set out the decision points and workflow for the two evolutionary change processes outlined in this article: (a) scheme snapshots and versions/editions in Appendix A and (b) concept change as embodied in concept instances in Appendix B.

In addition to the change processes of concepts as embodied in concept instances, Appendix B also illustrates the decision points and workflow for scheme aspects touched on only tangentially in this article: associative relationships among concepts and scheme documentation. While the diagram in Appendix B represents a single iteration of change to concepts, relationships, and documentation, the living nature of schemes compels an iterative flow such that the snapshots (and scheme versions) found at the bottom of the diagram feed the next iteration of change.

## Conclusion

In this article, a neopragmatic epistemic stance was taken in asserting the need for an entity in SKOS modeling—the concept instance—to mediate between the SKOS abstract concept and the concrete scheme. We demonstrated that this mediating entity, alongside constructs such as version, snapshot, and concept precedence, is necessary to the useful management and application of abstract concepts in indexing processes, vocabulary development, and information retrieval systems. The conceptualization of a concept instance that we described in this article is supported not only by its utility for metadata registry work and management of schemes but also in the analytical work done to describe the life of a work and its instances in the life of users and in the hands of catalogers. We demonstrated that the concept instance provides a means otherwise lacking in


DOI: 10.1002/asi

SKOS for tracing the historical development of concepts in vocabulary development systems.

## Acknowledgments

This research is partially funded by National Science Foundation Grant DUE-0532854. The authors are indebted to Diane Hillmann and Jon Phipps of Cornell University Library and Alistair Miles of the Science and Technology Facilities Council for their contributions to the conceptualizations described herein and for their work to operationalize these conceptualizations in the DCMI Registry. We also thank Melanie Feinberg and Ok Nam Park for their assistance with this research.

## References

Anderson, J.D., & Perez-Carballo, J. (2005). Information retrieval design. St. Petersburg, FL: Ometeca Institute.

Dalhberg (1979). On the theory of the concept. In Neelameghan (Ed.), Ordering systems for global information networks (pp. 54–63). Bangalore, India: International Federation for Documentation.

Dewey, J. (1930). Human nature and conduct. New York: Modern Library.

Foucault, M. (1972). The archaeology of knowledge. New York: Harper & Row.

Foucault, M. (1973). The order of things. New York: Random House.

Foucault, M. (1980). Power/knowledge. Brighton, England: Harvester.

Hillmann, D.I., Sutton, S.A., Phipps, J., & Laundry, R.J. (2006). A metadata registry from vocabularies up: The NSDL Registry Project. DC-2006 Proceedings of the International Conference on Dublin Core and Metadata Applications: Metadata for Knowledge and Learning (pp. 65–75). Manzanillo, Mexico.

International Federation of Library Associations and Institutions (IFLA) Section on Cataloging. (1998). Functional requirements for bibliographic records: Final report. Retrieved November 26, 2006, from http://www.ifla.org/VII/s13/frbr/frbr.pdf

Lancaster, F.W. (2003). Indexing and abstracting in theory and practice. Champaign: University of Illinois, Graduate School of Library and Information Science.

Library of Congress. (2005). MARC 21 concise format for classification data. Retrieved November 26, 2006, from http://www.loc.gov/marc/classification/

Mikhalenko, P. (2005, June). Introducing SKOS. Available at: O'Reilly XML.com. Retrieved November 26, 2006, from http://www.xml.com/pub/a/2005/06/22/skos.html

Miles, A. (2006). SKOS requirements for standardization. DC-2006 Proceedings of the International Conference on Dublin Core and Metadata Applications: Metadata for Knowledge and Learning (pp. 55–64). Manzanillo, Mexico.

Miles, A., & Brickley, D. (Eds.). (2005a, November). SKOS core guide: W3C Working Draft. Retrieved November 26, 2006, from http://www.w3.org/TR/2005/WD-swbp-skos-core-guide-20051102/

Miles, A., & Brickley, D. (Eds.). (2005b, November). SKOS core vocabulary specification W3C Working Draft. Retrieved November 26, 2006, from http://www.w3.org/TR/2005/WD-swbp-skos-core-spec-20051102/

Preschel, B.M. (1972). Indexer consistency in perception of concepts and in choice of terminology. New York: Columbia University School of Library Service.

Rorty, R. (1982). Consequences of pragmatism. Minneapolis: University of Minnesota Press.

Smiraglia, R.P. (2001). Works as signs, symbols, and canons: The epistemology of the work. Knowledge Organization, 28(4), 192–202.

Soergel, D. (1974). Indexing languages and thesauri: Construction and maintenance. Los Angeles: Melvile.

Tennis, J.T. (2005). SKOS and the ontogenesis of vocabularies. Proceedings of the International Conference on Dublin Core and Metadata Applications: Vocabularies in Practice (pp. 275–78). Madrid, Spain.

Tinker, J.F. (1966). Imprecision in indexing. American Documentation, 17, 93–102.

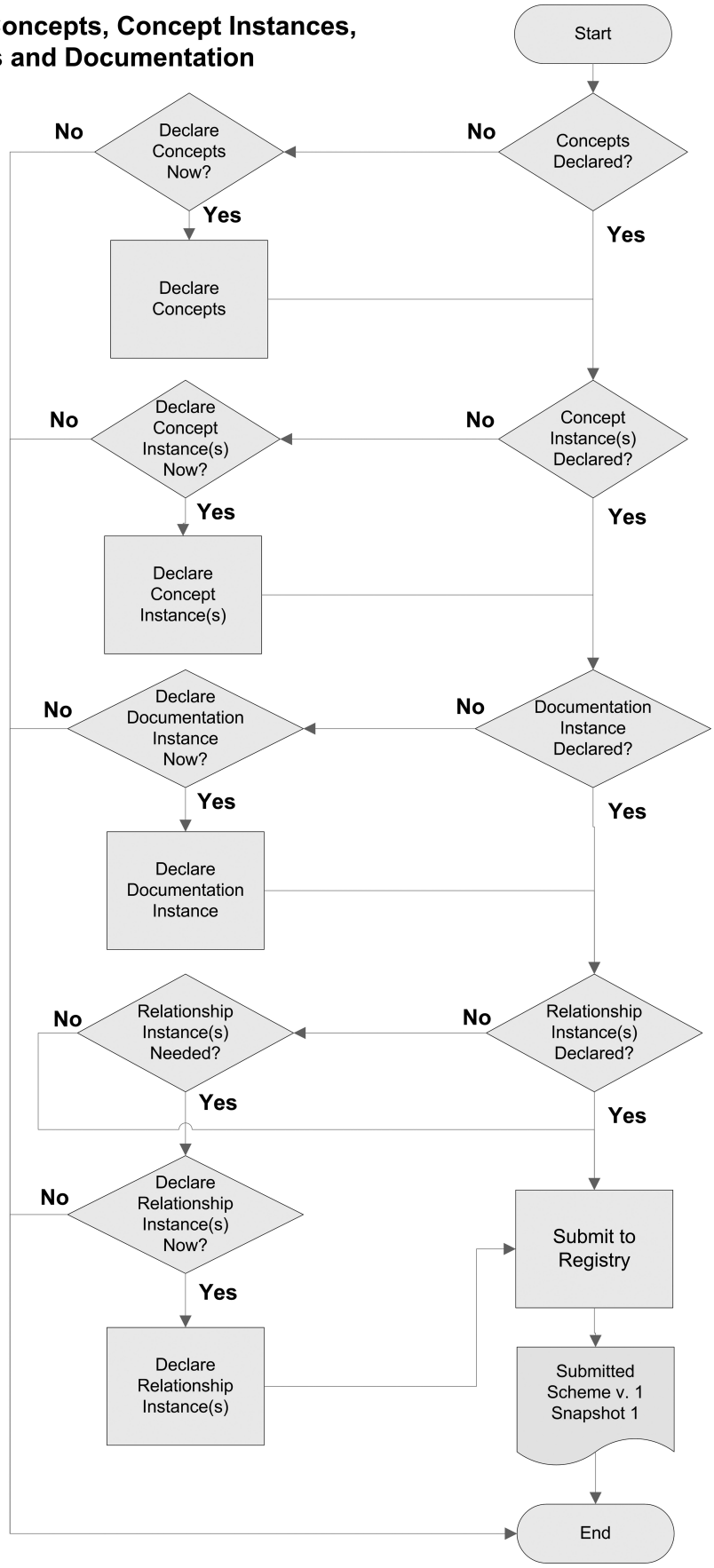
UDCC (2003). Master reference file: Manual. The Hague, The Netherlands: UDCC.

Wilson, P. (1968). Two kinds of power: An essay on bibliographical control. Berkeley: University of California Press.


DOI: 10.1002/asi

# Appendix A:

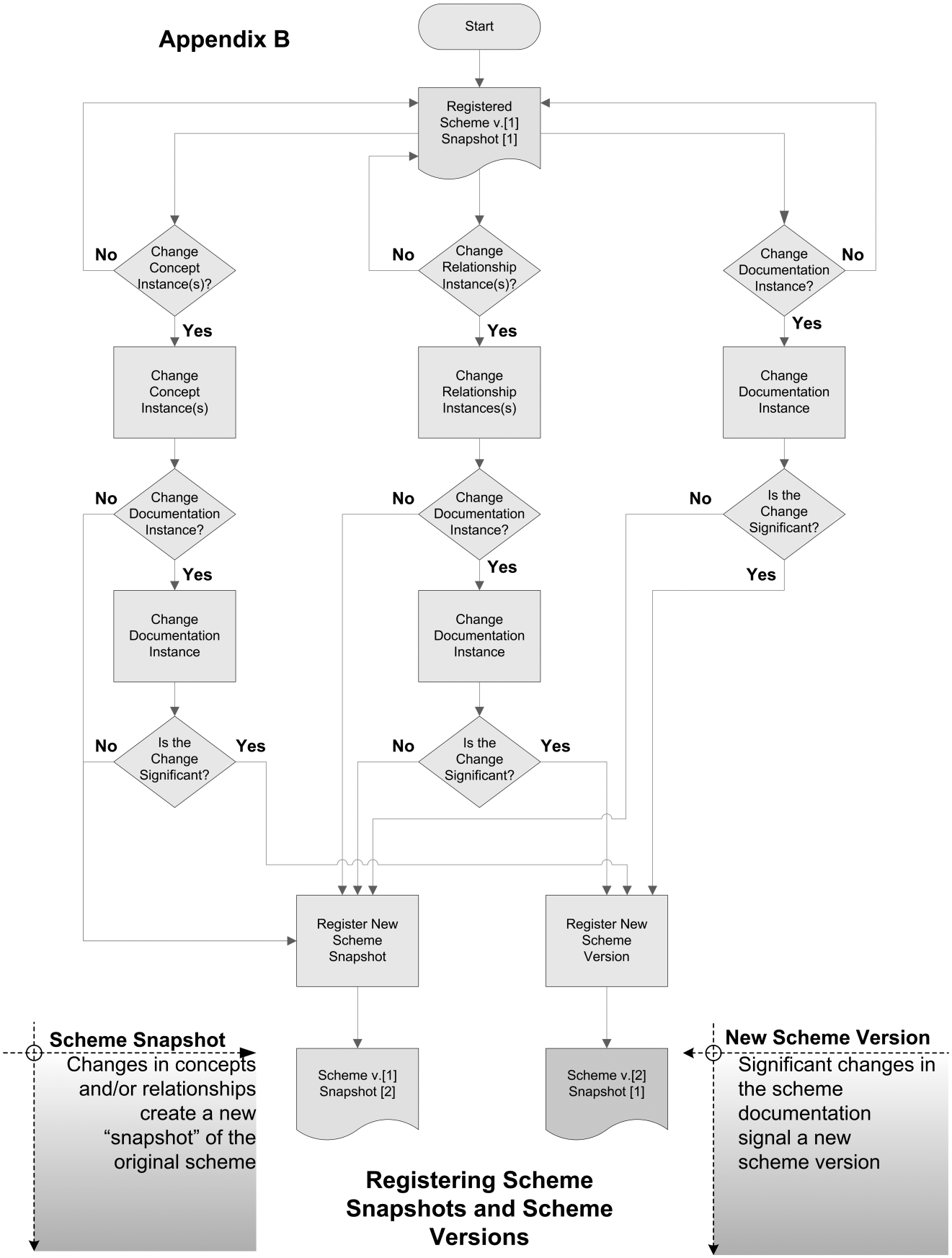
## Registering Concepts, Concept Instances, Relationships and Documentation





DOI: 10.1002/asi

## Appendix B




**Registering Scheme Snapshots and Scheme Versions**


DOI: 10.1002/asi