Evolution Relations for Knowledge Organization Systems: The case of MDR named authority lists

Author: Eugeniu Costetchi

Date: 14/12/2015

Version: 0.7

Contents

[Introduction 2](#_Toc437876651)

[The broad context: Open Issues for Knowledge Organization Systems 2](#_Toc437876652)

[The context of MDR tables 3](#_Toc437876653)

[Concept examples from NALs 4](#_Toc437876654)

[Derived requirements 6](#_Toc437876655)

[Current proposal: pseudo-versioning via reified properties (SKOS-AP) 7](#_Toc437876656)

[Towards the new proposal: addressing the limitations and the NAL requirements 8](#_Toc437876657)

[Distinguishing Abstract Concept and Concept Instance 8](#_Toc437876658)

[Distinguishing Schemes for Concepts and Schemes for instances 10](#_Toc437876659)

[Backward compatibility with SKOS-AP-EU 10](#_Toc437876660)

[Distinguishing the scope of relations: concept meaning, action result and time interval 11](#_Toc437876661)

[Allen's interval algebra relations 12](#_Toc437876662)

[MARC 21 from Library of Congress 12](#_Toc437876663)

[The minimal set of relations (to cover the examples) 13](#_Toc437876664)

[Examples representations of NAL concepts 13](#_Toc437876665)

[Implementation proposal 17](#_Toc437876666)

[Conclusion 18](#_Toc437876667)

[Bibliography 19](#_Toc437876668)

# 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[[1]](#footnote-1) (Congress 2000). In addition, concept records are used in the creation of thesauri and contain fields to track changes (Soergel 1974, Anderson and Perez-Carballo 2005)

Tennis and Sutton have published a solution to address concept change through their 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.

Metadata Registry unit of the European Publication Office maintains multiple Controlled authority lists (aka Named authority lists). The edition and maintenance of these lists is done in XML format with a internally defined XSD schema which supports temporal evolution of authority codes. Now, the general tendency at the level of European institutions is to move towards Semantic Web standards. SKOS has been chosen as suitable standard for representing controlled authority lists. However SKOS, because of the lack of support, raises concerns with regard to temporal evolution/versioning of the authority codes (concepts) which are set to be explored in current paper.

But before going specifically into the issue of versioning, I present the short overview of open issues in SKOS community.

## The broad context: Open Issues for Knowledge Organization Systems

Originally the SKOS W3C Workgroup needed 35 months to complete work on SKOS core specifications. In order to focus its efforts and keep the specification as short and simple as possible, the group declared several topics to be out of scope (Baker, et al. 2013):

* Concept Coordination
  + Many KOSs are intended to be used as building blocks for constructing ‘‘coordinated’’ concepts, for example to aggregate the ‘‘simple’’ concepts ‘‘aspirin’’ and ‘‘side effect’’ into a ‘‘compound’’ concept ‘‘aspirin—side-effects’’. Compound concepts can be created on a one-off basis by catalogers, as they are needed in resource description, or they can be added as concepts to the KOS itself by its maintainers (which is known as ‘‘pre-coordination’’, as with the Library of Congress subject heading ‘‘China—history’’). (Baker, et al. 2013)
* Subject indexing
  + As defined by Leonard Will, subject indexing involves ‘‘intellectual analysis of the subject matter of a document to identify the concepts represented in it, and allocation of the corresponding preferred terms to allow the information to be retrieved’’.41 The working group recognized as a candidate requirement the ‘‘ability to represent the indexing relationship between a resource and a concept that indexes it’’, whereby the SKOS model would include ‘‘mechanisms to attach a given resource (e.g. corresponding to a document) to a concept the resource is about, e.g. to query for the resources described by a given concept’’ (Baker, et al. 2013)
* Provenance information (about mappings)
  + The ability ‘‘to record provenance information on mappings between concepts in different concept schemes’’ was recognized as a candidate requirement for SKOS (Baker, et al. 2013).
* Describing concept schemes
  + Concept schemes have authors, titles, publishers, dates issued, subject coverage, and the like. The working group felt that the question of what properties to use in describing a concept scheme was an issue best left to communities of practice (Baker, et al. 2013).
* Concept evolution
  + The working group acknowledged the importance of mechanisms for representing the temporal evolution of concept schemes—an issue that raises questions of granularity (whether to version individual statements, concept descriptions, or entire concept schemes) and of how to represent such versioning information in interoperably machine-readable ways. The group considered this topic best left to the community for research and testing (Baker, et al. 2013).

This paper mainly focuses on concept evolution relations however because of the nature of MDR examples it covers marginally the issue of concept coordination as well. Evolutionary relations in SKOS are not a new topic. A set of important discussions and proposals is listed on W3C SKOS-Issues page[[2]](#footnote-2).

## The context of MDR tables

Temporal evolution for Concepts has always been an important aspect for Authority lists (NALs) at MDR. Currently NALS are edited and managed as Excel worksheets. The first two columns express the identity of a unique record Id and an (possibly repeated) authority code. The authority code bears the identity of the described concept while the record id uniquely identifies the version of the concept description. Hence a concept may have several version descriptions.

The concept description is essentially the rest of the columns that carry various properties semi formally defined by the column name/type. Each new version of the concept is a record in the worksheet with all properties distributed linearly: properties of the concept, of the concept labels (various types), relations to successor or predecessor version of the concept (referenced by record id) and relations to concepts or concept versions in other tables.

The challenge now is to express the NALs in RDF form by adhering to the SKOS principles and definitions because NALs are thesauri to an extent. Ideally the semantics of the resulting KOS should allow for the same level of expressivity maintain and detail as in the Excel worksheets.

One successful attempt to achieve this was expression as simple SKOS. Because NALs are more than simple thesauri, the resulting file expresses the latest version for each concept and the set of afferent labels in various languages. The simple SKOS approach does not permit:

1. Expressing additional specifications for labels (such as context, script, creation, IMMC approval, start use dates etc.)
2. Expressing relations to other tables such as location, label type, etc.
3. Expressing concept versions.

To mend the above constraints, an application profile (AP) has been created titled SKOS-AP. It is based on SKOS-XL which (among others) defines a new class: Label to allow for reified label relations. The SKOS-AP also defines NAL specific properties at the concept and label levels and at the additional two classes XlNote and XlNotation.

SKOS-AP enables expressing label properties, relations to other tables and also pseudo-versions of the concept. The versioning of the concept can be expressed through instances of Label, XlNote and XlNotation classes each having assigned a version number and/or start and end use dates. In fact the concept itself always stays the same, whereas it is ascribed various versions of labels and notes (and any other reified property) over the time.

The pseudo-versioning approach proposed in SKOS-AP works for relatively simple NALs that focus mainly on label definitions. Some other NALS are more complex in their descriptions (beyond label evolution). NALs such as Countries, Corporate Bodies, Places etc. make multiple references to concepts in other tables. The references to concepts in other tables may differ by concept version and/or may make a reference to a specific version of the foreign concept and not its label (see examples at the page 4).

Expressing the relation between a concept version and a version of a foreign concept is not possible in SKOS-AP as it lacks the distinction between a concept (in abstract sense) and a specific version of that concept (in specific sense or as an instance of the abstract concept). This limitation of SKOS-AP in expressing concept versioning constitutes the main focus of the current paper.

Next are presented a set of examples occurring in MDR NAL tables. A richer set of examples is explored in the last section of this paper.

## Concept examples from NALs

Below are presented several examples of different phenomena occurring in the authority tables. These examples are taken from Corporate Body authority table.

Table Example of concept successive versions

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rec Id | Authority  Code | Label | Start Date | End Date | Pred | Succ |
| COB094 | MARE | DG XIV – Fisheries | 01/01/1992 | 30/09/1999 |  | COB095 |
| COB095 | MARE | Directorate-General for Fisheries | 01/10/1999 | 12/01/2005 | COB094 | COB096 |

Table 1 presents two simplified records for two successive versions of MARE authority code. This example is useful for deciding on two important questions:

* What is a NAL concept? How many of them are in the example above?
  + Is there one MARE concept?
  + Are there two MARE\_1 and MARE\_2 concepts?
  + Are there three MARE, MARE\_1 and MARE\_2?
  + If so is MARE of the same nature as MARE\_1?

Table 2 Example of transient and non transient PROPOERTIES ACROSS versions

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rec Id | Authority  Code | Label | Creation Date | Pred | Succ | Loc Id |
| COB093 | PUBL | Publications department | 10/06/2009 |  | COB092 | LOC21 |
| COB092 | PUBL | Publications Office | 10/06/2009 | COB093 | COB091 | LOC452 |
| COB091 | PUBL | Publications Office | 01/07/2009 | COB092 |  | LOC452 |

Table 2 presents three simplified records for three versions of PUBL authority code. When a new version is recorded it is usually because the label has changed. Sometimes it is not the preferred label that changes but the long label (for example COB092 and COB091) while the preferred label stays the same. As it is stated before, NALs contain more information beyond the scope of SKOS model. From one version to the other we can see that the non-SKOS properties also change (e.g. LOC21 and LOC452).

The only property that we could guarantee that stays the same is really the authority code. This is the property that grounds in a way that the "concept" represents an unambiguous reference to a resource (within a given context), and very important in the case of authority lists, the reference must persist across time and versions of resource description/definition.

Above example provides us with a dichotomy between transient and intransient properties. To this distinction be will return in a further section, but for now we shall remember that the Authority code is the only property which can be guaranteed that will not change from a concept version to the other.

Arguably, the creation date of the authority code (and similar metadata, examples needed, is any) is also an intransigent property. However If we look at the Creation dates of COB092 and COB091 we can note different creation dates, even if the authority code is the same. So perhaps it is a mistake, of the creation date designate when the record was created and not the creation of the authority code.

Table example of a concept Split into two successive ones

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rec Id | Authority  Code | Label | Parent Concept | Pred | Succ | Loc Id |
| COB452 | DEL\_AU\_NZ | Delegation to Australia and New Zealand | RELEX |  | COB453| COB454 | LOC547 |
| COB453 | DEL\_AUS | Delegation to Australia | EEAS | COB452 |  | LOC547 |
| COB454 | DEL\_NZL | Delegation to New Zealand | EEAS | COB452 |  | LOC566 |

Table 3 presents three simplified records for how the DEL\_AU\_NZ in one version is split into DEL\_AU and DE\_NZ authority code in the next version. This example is set to differentiate three types of relations that hold among records (concept versions) (a) semantic relations, (b) temporal relations, (c) change actions (processes).

Before going into those three types of relations note a specificity of relations between records and authority codes. Parent property (that is present in Corporate Body) is used here to demonstrate relationship between records and authority codes. Even if it appears that the relation of parenthood may be between one authority code and another one the reference is not set fixed and may change from one version to the other. One of the reasons for this is imprecise definition of parenthood relation that changes from one table to the other. But this issue is out of the scope of the current paper and shall be treated elsewhere. It is however clear that there may be intransient relations between records and authority codes.

Let's start by looking at what is intended as change action. To do that, we must distinguish between processes and states. State means *"a particular condition that something is in at a particular time"* while process means *"a series of actions taken for achieving a particular end"*. We say that something changes as a result of a process enactment and goes from an initial state *Si* into a final state *Sf*. Someone could say then that the relationship that holds between the initial and final state is defined exactly by the process that transitioned between them.

Another way the states are related are by the properties they carry, they both comprise of the same nature of properties and this each can be compared in order to describe the direction of change (something got bigger, smaller, hotter, brighter, etc.). In this sense, we can say that a particular aspect of state *Si* is in a comparative relationship to the same aspect of state *Sf*.

This means that regardless of the process the states still stand in a particular relationship to each other in a way describing relations between absolute positions. While the process is explaining how exactly the change has happened as a "delta", the difference between two states, in a way describing the relation in relative terms.

When talking about versions, we are interested in expressing how a particular authority code or concept is defined a particular time, and not what happened to it to have arrived to that particular form. The second type of information is also very useful but it addresses a different set of questions and is out of the current scope. This issue is much better addressed in the context of provenance descriptions.

This underlines the fact that relations that take the verb forms such as continues, replaces, merged from, split into do not fit the kind of descriptions we are aiming for in the current work. We are well aware that for example MARC21 relations defined by the library of congress use only such terms, but they may have had a different paradigm or set of goals in mind. Whereas we aim for more rigorous definitions that would offer the possibility to compare concept aspects without the common sense constraints.

Instead, we advocate for static relations between aspects of an authority code that may have changed. Specifically, we are firstly interested in relating the meaning of an authority code via semantic relations (defined already in SKOS as broader, narrower and related) and secondly in the temporal manifestation captured precisely by interval relations (defined already in Allen's interval algebra such as before, after, overlaps etc.).

## Derived requirements

* Express evolutionary relations among concepts.
* Concept versions may be in various temporal relations, beyond the intuitive successor relation i.e. concomitant, distanced, overlapped, strict successor relations etc. As mentioned earlier, such relations are defined by Allen Interval algebra.
* Relations differ in scope: some strictly concern concepts (regarding their coordination/compositions/decomposition) and some concern temporal relations for evolution (and not only). This is documented by Baker et al (Baker, et al. 2013)
* The model may adopt (a) snapshot approach where all properties are specified for a particular time point or interval or (b) delta approach where only properties with changed/new values are specified (suitable rather in the context of provenance descriptions)
* Maintain a stable Concept URI while still be able to refer to a specific version of the concept. Do not change the URI for a concept with each (scheme) version. The more stable the concept URIs are, the more consistently they will be applied.[[3]](#footnote-3)
* The solution should be SKOS, SKOS-XL, SKOS-AP backward compatible, i.e. maintain the semantics of the legacy models.
* Standard versioning mechanism should be reused where appropriate

# Current proposal: pseudo-versioning via reified properties (SKOS-AP)

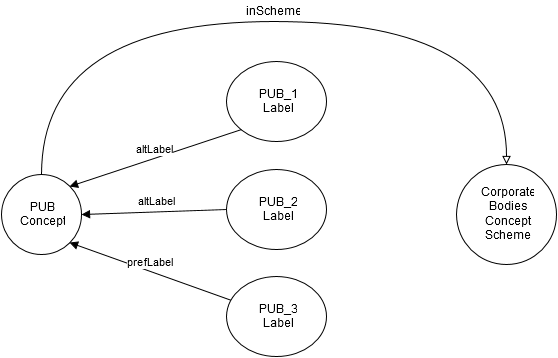


Figure Representation of concept Pseudo-Evolution via reified label property

Currently the evolution of skos Concepts can be captured only as changes in label, note or notation. These properties have been transformed into classes skosxl:Label, euvoc:XlNote and euvoc:XlNotation and allow euvoc:startDate and euvoc:endDate for temporal validity constraints.

The advantage of reifying label relations is that skos-xl:Label instances can carry additional information such as label version, start/end dates for the label validity, label status (whether it is in in use or deprecated), label source etc.

Additionally with a minor modifications of SKOS-AP-EU specification (v29 from Sep 2015) two more properties can be used for this purpose: dct:replaces and dct:isReplacedBy. At the moment their domain is only skos:Concept, but the domain constraints can be relaxed to include every class acting as a reification.

If the structure of the MDR data would be strictly concerned with terminological relations, such a model would perfectly suffice such as in the case of EuroVoc. However the MDR tables vary in their complexity some of which require table specific (data and object) properties. Moreover MDR is expected to create an open number of NALs complexity of which is unknown at the moment.

Given the context of MDR authority tables, current approach has several limitations:

* No abstraction: mix of class and instance levels
  + There is no distinction between concept and the concept snapshot/instance. Such a definition would clearly separate transient (subject to change in time) from intransient properties (time invariant). Where transient properties are only ascribed to concept instances/versions and the intransient ones only to the concepts.
* Lack of scalability
  + Authority Lists (NALs) may be simple (i.e. when the SKOS-XL suffices to express the table logics, such as ConeptStatus, Usecontext etc.) and complex (i.e. when the table has extra properties that are not covered by SKOS-XL and required definition of table specific ones, such as CorporateBodies, Countries, etc. ).
  + In the case of complex NALs, some (table specific) properties are transient meaning that to ensure their versioning with current mechanism, they would have to be reified (transformed into classes) leading to model updates every time there is a change in the model.
* Potential source of errors and difficulty to maintain on the long run
  + Moreover these reified properties are not ontologically established classes but rather technical a work around. This inevitably leads to a large and potentially messy model where business logic is mixed with technical wrappers making it costly and difficult to maintain on the long run; not to mention that it is an additional source of errors and confusion.
  + The concept will have all versions of a label (e.g altLabel-short, long, etc.) and other information attached to the concept. Without a proper editor would, this will become difficult to maintain. At the moment, it is clean and clear because the editing is done in Excel where each new row is a new version of the concept. However in VocBench things this might get less clear especially when the number of versions and the number of new information (other labels and extra properties) per concept-version will grow.

# Towards the new proposal: addressing the limitations and the NAL requirements

## Distinguishing Abstract Concept and Concept Instance

The current solution consists of introducing a distinction between the Abstract concept and the Instance or Snapshot of a Concept. This idea is well documented by Tennis & Sutton (Tennis and Sutton 2008) which have implemented it at the US National Science Digital Library (NSDL).

The Proposed solution is depicted in Figure 2. It is a simplified representation for PUB authority code from the Corporate Bodies NAL. On the left side there is a generic PUB concept whose identity is fixed and considered invariant across time. To it, there are three Concept instances, each corresponding to a concept version with specific preferred and alternative labels (including short, long and other labels).

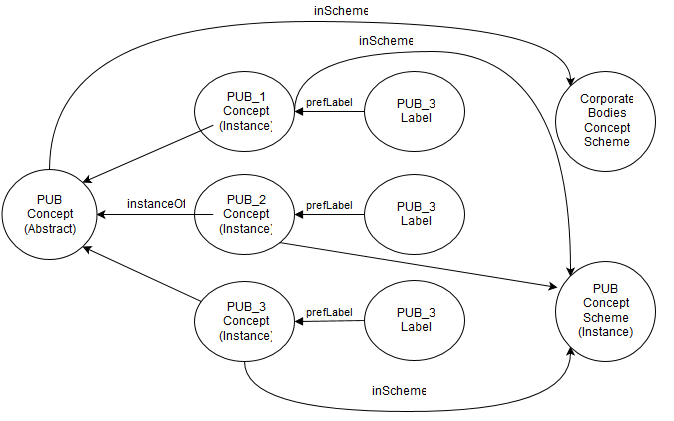


Figure 2 Relationship between abstract concept, concept instance and concept scheme in which the instances are declared

The notion of concept instance can be made explicit in the context of following core assertions.

* **Core Assertion 1:** A **Concept** is an "abstract idea or notion; a unit of though"[[4]](#footnote-4) identified by URI.
* **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 the scheme concepts, relationships and documentation.
* **Core Assertion 6:** There are transient and intransient properties**.** Concept must be ascribed only intransient properties while concept instance should be mainly ascribed transient but it can accommodate both types.

The modeling implications of Abstract-Instance Concept are depicted in Figure 3. The Interval relations are not covered in this figure but they are described in the sections on the minimal set of relations and in the implementation proposal.

D:\Repository\euvoc\euvoc-documentation\SKOS Evolution\yEd_diagrams\skos-evo.emf

Figure 3 Abstract/Instance Concept representation

## Distinguishing Schemes for Concepts and Schemes for instances

Since we separate the types of concepts into two categories based on their level of abstraction, it is instrumental to use the same distinction for the concept schemes as well.

This can come in handy for backward comparability as at the moment only abstract concepts get grouped into concept schemes so it is only useful to distinguish another type of grouping intended for version(s) of concepts.

This distinction however, at the theoretical level, does not come as a need in the concept versioning problem but is merely an intuitive proposal. Thus we consider it as an open question and welcome feedback from other involved parties for supporting or dismissing it.

## Backward compatibility with SKOS-AP-EU

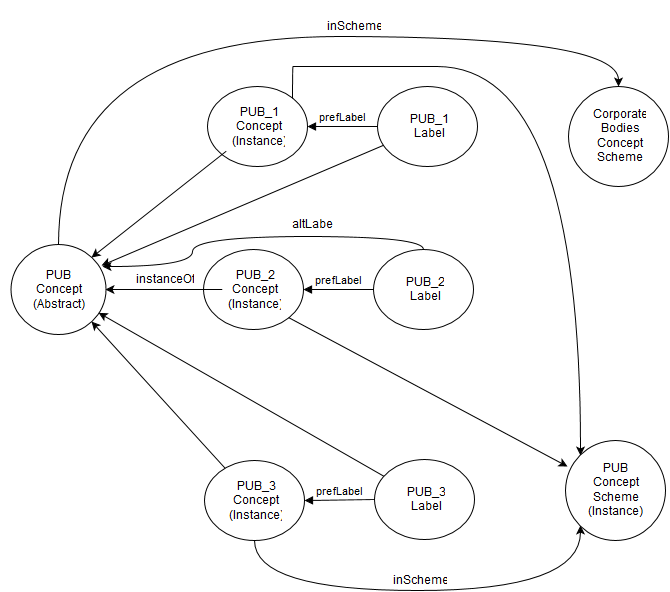


Figure Backwards Compatibility

## Distinguishing the scope of relations: concept meaning, action result and time interval

It is easy to follow intuition and use common words for expressing a relationship suitable for a particular case. However when the context changes to another modeling case, we could find it appropriate to reuse the same relation, with a quite different interpretation.

For example, imagine that in one case a concept A precedes a concept B and in another case a concept P precedes a concept Q but also the beginning of Q is slightly before the end of P. Well we might say that P still precedes Q just like A precedes B with a slight remark. Then the question is, what are the default assumptions for precedes relation or how do we specify such exceptions? Or maybe these are two different precedence relations?

I propose a different approach: to clearly separate the relations scopes. Every time we specify a relationship between concepts we can express explicitly the temporal aspect combined with meaning relation combined with the change result effect. For example, the new concept has a meaning somehow different to the previous one but continues the old one, or is split and has narrower meaning, or is merged and has broader meaning, etc.

The distinguished relationship types are:

* **Meaning comparison relations:** broader, narrower, related, etc (skos relations)
* **Results of a change action:** merge, split, continue, replace etc … (PROV)
* (time) **interval relations:** before, after, overlap, meet, during etc. (Allan relations)

I will take as reference two sets of temporal relationships, the first one is the Allen interval algebra[[5]](#footnote-5) and the second one is MARC 21 Format for Bibliographic Data[[6]](#footnote-6) used by Library of congress (MARC are no needed as they express operations and not relations).

### Allen's interval algebra relations

|  |  |  |
| --- | --- | --- |
| **Relation** | **Illustration** | **Interpretation** |
| X \,\mathrel{\mathbf{<}}\, Y  Y \,\mathrel{\mathbf{>}}\, X | [X takes place before Y](https://en.wikipedia.org/wiki/File:Allen_calculus_before.png) | X takes place before Y |
| X \,\mathrel{\mathbf{m}}\, Y  Y \,\mathrel{\mathbf{mi}}\, X | [X meets Y](https://en.wikipedia.org/wiki/File:Allen_calculus_meet.png) | X meets Y (*i* stands for ***i****nverse*) |
| X \,\mathrel{\mathbf{o}}\, Y  Y \,\mathrel{\mathbf{oi}}\, X | [X overlaps with Y](https://en.wikipedia.org/wiki/File:Allen_calculus_overlap.png) | X overlaps with Y |
| X \,\mathrel{\mathbf{s}}\, Y  Y \,\mathrel{\mathbf{si}}\, X | [X starts with Y](https://en.wikipedia.org/wiki/File:Allen_calculus_start.png) | X starts Y |
| X \,\mathrel{\mathbf{d}}\, Y  Y \,\mathrel{\mathbf{di}}\, X | [X during Y](https://en.wikipedia.org/wiki/File:Allen_calculus_during.png) | X during Y |
| X \,\mathrel{\mathbf{f}}\, Y  Y \,\mathrel{\mathbf{fi}}\, X | [X finishes with Y](https://en.wikipedia.org/wiki/File:Allen_calculus_finish.png) | X finishes Y |
| X \,\mathrel{\mathbf{=}}\, Y | [X is equal to Y](https://en.wikipedia.org/wiki/File:Allen_calculus_equal.png) | X is equal to Y |

### MARC 21 from Library of Congress

It refers to the concepts/codes in the bibliographic data.

• **780** - Preceding entry - Type of relationship

* 0 - Continues
* 1 - Continues in part
* 2 - Supersedes
* 3 - Supersedes in part
* 4 - Formed by the union of ... and ...
* 5 - Absorbed
* 6 - Absorbed in part
* 7 - Separted from

• **785** – Succeeding entry Type of relationship

* 0 - Continued by
* 1 - Continued in part by
* 2 - Superseded by
* 3 - Superseded in part by
* 4 - Absorbed by
* 5 - Absorbed in part by
* 6 - Split into ... and ...
* 7 - Merged with ... to form ...
* 8 - Changed back to

## The minimal set of relations (to cover the examples)

From the analysis of a broader set of examples in Corporate Bodies table and a one class of examples in EU Programme table we came to the conclusion that only semantic and a subset of interval relations suffice.

* Semantic relations
  + Skos:broader
  + Skos:narrower
  + Skos:related
* Interval relations
  + Before
    - Meets
    - Overlaps
  + After (inverse of before)
    - Meets inverse
    - Overlaps inverse
  + During
    - Starts
    - Finishes
  + Includes (inverse of during)
    - Started by
    - Finished by
  + Equals (reflexive)

For most of the NAL cases the semantic relations and the before relation suffice to express all possible variations. However in other cases such as Programmes table, more the concepts may be temporally related in more ways, requiring relations such as "overlaps" or "during". Thus we propose to include all Allen relations as an instrument ensuring expressivity if needed in future NALS.

## Examples representations of NAL concepts

This section demonstrates via examples how current proposal handles the "problematic" cases in NALs.

Each of the examples below contains a RDF graph fragment. The pink circles represent instances of Concept and concept instance Classes and the lines between them represent object properties. The yellow rectangles represent instantiated model classes, in our case the focus is mainly on Concept and ConceptInstance classes as depicted in the Figure 5.

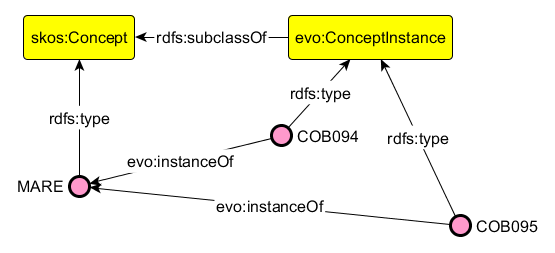


Figure Instantiation of evolutional skos

As a rule of thumb, the nodes named with the record Id following the pattern "COB###" are instances of ConceptInstance class and are usually placed on the right side of the diagram. The nodes named mainly (but not always) with letters are instances of abstract concepts (i.e. of Concept class) and are usually placed on the left side of the diagrams.

The nodes that are quoted text are typed literal values and are connected to instances through data or annotation properties.

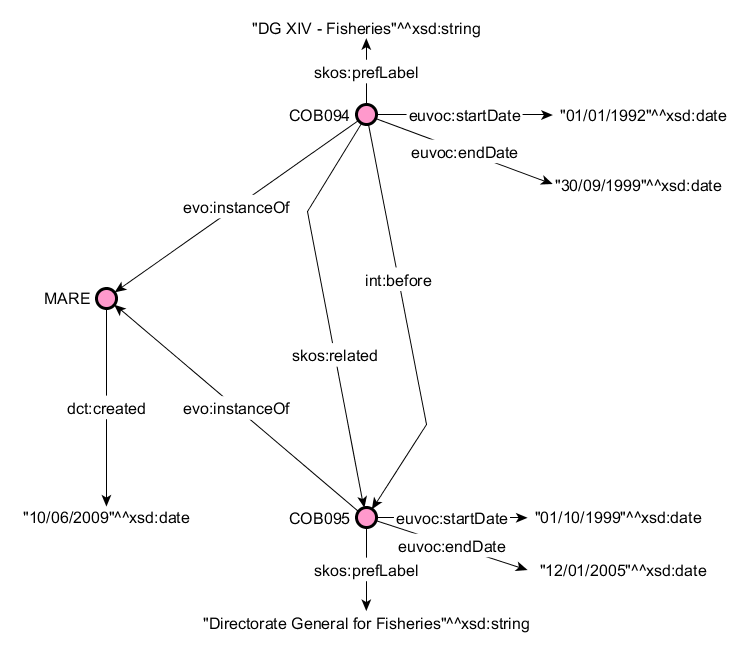


Figure 6 succession of two concept instances of MARE concept

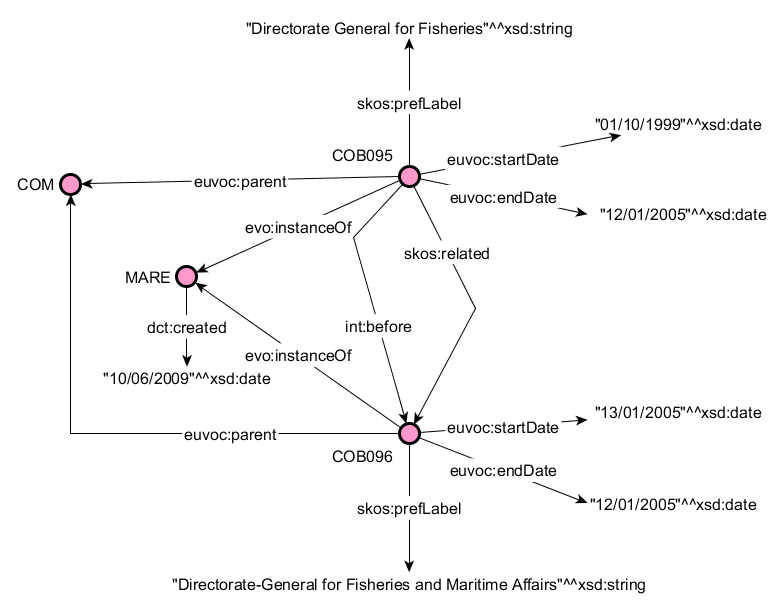


Figure 7 succession of two concept instances of MARE concept with the same parent

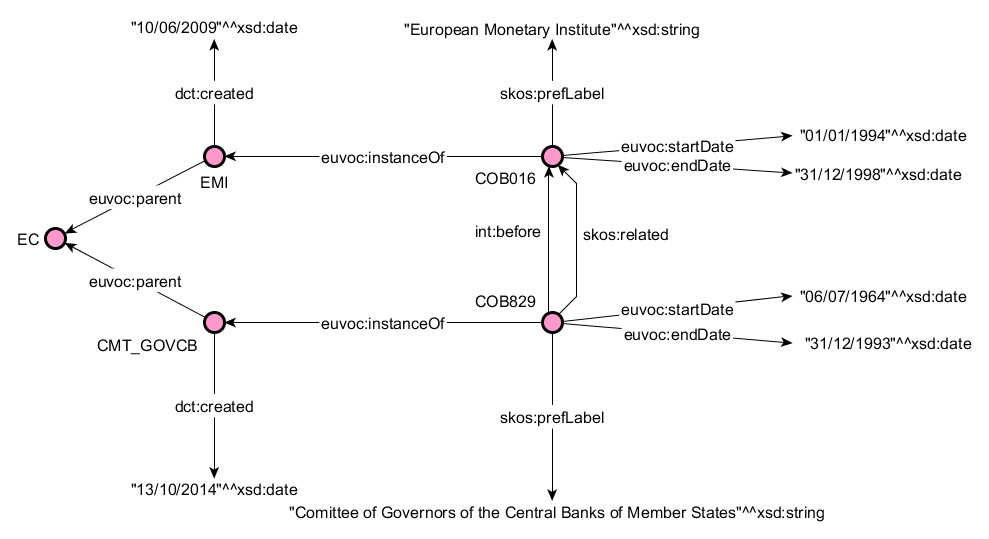


Figure 8 succession of two distinct concepts

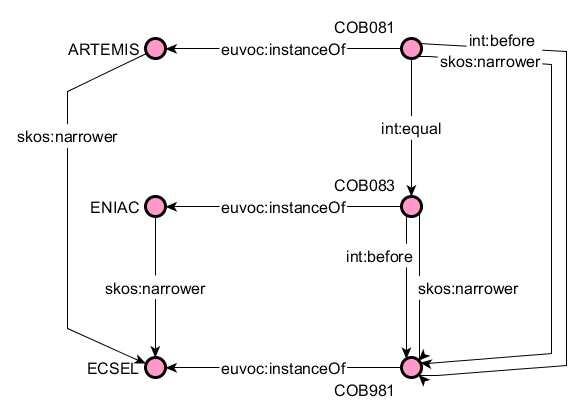


Figure Merge of two distinct concept

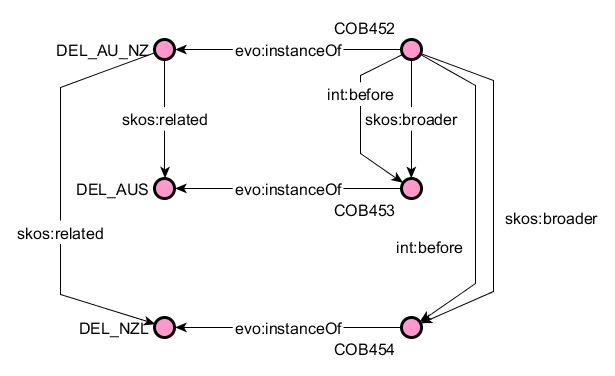


Figure Split of two distinct concepts

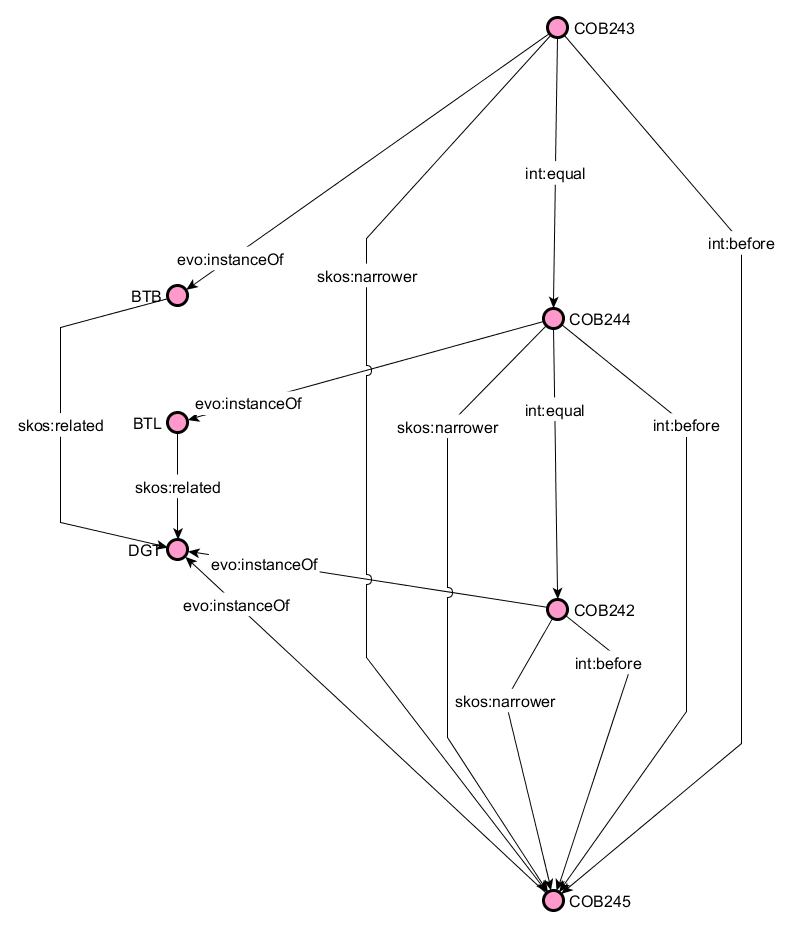
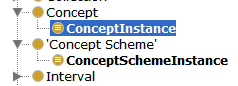
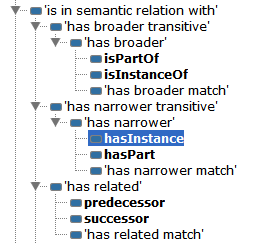
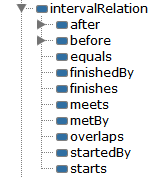


Figure Merge of the same and two distinct concept

## Implementation proposal







# Conclusion

In this paper is described an approach to implement SKOS versioning by adopting the idea of skos concept instance required by MDR named authority lists. A similar need has been implemented at the US National Science Digital Library (NSDL) which is documented by Tennis & Sutton (Tennis and Sutton 2008).

This requires several other choices and theoretical distinctions (underlined and explained in the introductory chapter) to be made in order to accommodate the distinction between abstract and instance concepts. After going through few examples of NAL records, a set of requirements is established that any concept evolution solution needs to fulfill in order to represent NAL tables as semantic web data.

Then is discussed an already proposed model for NAL implementation – SKOS-AP-EU. The discussion highlights the strong points of the model and it's limitations against the set of derived requirements in the introduction.

Then the new proposal sets to address all the raised issues and demonstrate it in action through the same examples from Corporate Bodies table and one reference to EU Programmes table. The solution proposes to introduce the ConceptInstance class, a new relation – evo:isInstanceOf - between Concept and ConceptInstance and a set of interval relations as described in Allen's interval algebra. Three semantic relations that already exist in skos model are reused at the level of concet instance.

# Bibliography

Anderson, J, and J Perez-Carballo. *Information retrival design.* St. Petersburg: FL: Ometeca Institute, 2005.

Baker, Thomas, Sean Bechhofer, Antoine Isaac, Alistar Miles, and Guus Schreber. "Key choices in the design of Simple Knowledge Organization System (SKOS)." *Web Wemantics: Science, services and Agents on the world Wide Web*, 2013.

Congress, Library of. *MARC 21 concise format for classification.* 2000. http://www.loc.gov/marc/ (accessed 12 08, 2015).

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

Tennis, Joseph, and Stuart Sutton. "Extending the simple Knowledge Organization System for Concept Management in vocabulary Development Applications." *Journal of the American society for information science and technology*, 2008: 25-37.

1. https://www.loc.gov/marc/marbi/2005/2005-01.html [↑](#footnote-ref-1)
2. https://www.w3.org/2001/sw/wiki/SKOS/Issues/ConceptEvolution [↑](#footnote-ref-2)
3. http://www.w3.org/wiki/SkosCoreGuideToc/SectionVersioning [↑](#footnote-ref-3)
4. http://www.w3.org/TR/swbp-skos-core-spec [↑](#footnote-ref-4)
5. https://en.wikipedia.org/wiki/Allen%27s\_interval\_algebra [↑](#footnote-ref-5)
6. <http://www.loc.gov/marc/bibliographic/bd76x78x.html> [↑](#footnote-ref-6)