Noname manuscript No.
(will be inserted by the editor)

Linked Open Ontology Cloud – Managing a System of Interlinked Cross-domain Light-weight Ontologies

Matias Frosterus · Jouni Tuominen · Sini Pessala · Eero Hyvönen

the date of receipt and acceptance should be inserted later

Abstract Traditionally the structure of the controlled vocabularies used for annotation can be utilised for reasoning for information retrieval. However, this can be problematic when applied in the Linked Data context. Linked data typically comes from different organisations and domains with mutually incompatible vocabularies without explicit links between them resulting in data silos. This paper argues that to solve the problem one has to transform the annotation vocabularies into a *Linked Open Ontology* cloud. We present a method for transforming a set of legacy thesauri into a cloud of interlinked ontologies while ensuring the validity of the transitive subclass relations and the means for maintaining the system when component ontologies are updated. Our approach has been used and evaluated in practice building a cloud called KOKO of sixteen ontologies, with a total of 47,000 concepts. KOKO has been published as an ontology service and is in use in various organisations for both data indexing and semantic search.

Keywords Light-weight Ontologies · Linked Open Ontology Cloud · Ontology change propagation · SKOS · Ontologisation · Cross-domain interoperability

1 Introduction

Libraries, archives, museums, and other organisations have been using classifications and thesauri for content indexing (annotation) for a long time. There exist vast amounts of high-quality annotations describing vast amounts of documents and other objects but these metadata descriptions are often divided into silos without machine-traversable links between the values used in the metadata. Semantic interoperability between heterogeneous *cross-domain* data repositories of distributed content providers has become the critical challenge for the Semantic Web and Linked Data [13]. Enabling different thesauri and vocabularies to link to one another in meaningful ways would allow different organisations to benefit from each others' work by enriching a common pool of linked knowledge [15].

1.1 Why a Linked Open Ontology Cloud?

The Linked Data movement[1] has focused its efforts on building cross-domain interoperability by creating and using (typically) `owl:sameAs` mappings between the entities (e.g. places, persons) in the *datasets* of the Linked Open Data (LOD) cloud. However, when linking metadata, not only the data entities but also ontologies used in describing them need to be interlinked for interoperability. This calls for more refined ontology alignment techniques [8] maintaining the integrity of the concept hierarchies.

This paper focuses on aligning light-weight domain ontologies intended for metadata annotations. A light-weight ontology in our terminology is a hierarchy of concepts with subsumption, partitive, and associative relations like in a traditional thesaurus [2], and can be represented using RDFS[2], simple OWL[3] constructs, or SKOS[4].

The research hypothesis of this paper is that the LOD cloud could be complemented by developing one or more light-weight "Linked Open Ontology" (LOO) clouds. The idea of LOO is to provide a shared cross-domain ontology for data annotations based on a set of interlinked domain

Semantic Computing Research Group (SeCo), Aalto University
P.O. Box 15500, 00076 Aalto, Finland
http://www.seco.tkk.fi/, firstname.lastname@aalto.fi

[1] http://linkeddata.org/
[2] http://www.w3.org/TR/rdf-schema/
[3] http://www.w3.org/standards/techs/owl#stds
[4] http://www.w3.org/TR/skos-reference/

ontologies. This idea is also complementary with the idea of "Linked Open Vocabularies"[5] that focus on mapping metadata schemas (e.g. Dublin Core, FOAF, and Bibo) onto each other. In our implementation of the idea, we created the LOO cloud from light-weight ontologies based on existing thesauri. If the resulting LOO cloud is then mapped to, e.g., DBpedia, legacy data annotated using the original thesauri can be linked to the LOD cloud quite easily.

Developing a LOO cloud is in many ways different from linking entities in datasets or elements in metadata schemas. A major difference is that in LOO the linked structure is used for reasoning based on the hierarchical subclass relation, the backbone of ontologies [31]. This fundamental task requires special consideration at the ontology boundaries as otherwise cross-domain reasoning and ontology-based query and document expansion [3, 17] in applications may fail [16].

For example, assume that the concept "Mirror" is present in a given ontology A of daily utensils and has the subclass "Make-up-mirror":

`a:Make-up-mirror rdfs:subClassOf a:Mirror.`

In a related ontology B of furniture, the class Mirror is used as a subclass of the class Furniture:

`b:Mirror rdfs:subClassOf b:Furniture.`

Without context, the concept of mirror looks like the same in both ontologies, i.e.

`a:Mirror owl:sameAs b:Mirror.`

Reasoning and query expansion works fine in A and B separately, but when using A and B linked together, expanding a search query for "furniture" would return falsely handheld make-up mirrors in addition to pieces of furniture. A larger context than the concept alone has to be considered when linking ontologies.

There are also other difficulties specific to developing a LOO cloud. For example, the principle of dividing a shared concept X into subclasses in different ontologies may be different. For example, "clothes" can be divided into subclasses based on the gender or the age of their wearers. Then, from a human perspective, X may have a confusing mixture of subclasses in the linked ontology, hampering its use in user interfaces (e.g. as a search facet). Addressing issues like these is hard to automate, and therefore LOO development in practice requires more coordinated collaboration between the developers of linked ontologies than when linking datasets of instances. By collaboration, better quality links can be created and various critical issues of linked data quality[6] can be addressed. Coordinated collaboration also facilitates larger scale ontology development, which prevents the creation of interoperability problems, and minimises redundant ontology development work in overlapping areas of ontologies.

[5] http://lov.okfn.org/dataset/lov/

[6] Cf. e.g. http://pedantic-web.org/fops.html

Our approach therefore emphasises the systematic development process of a coherent aggregated ontology from a set of component ontologies that are maintained by different domain communities. This *proactive* idea of developing a larger ontology based on different domain ontologies [16] is different from traditional ontology mapping, where one takes a set of existing, independently developed ontologies and tries to map them together *afterwards*. In contrast to simply mapping individual ontologies together we take the mappings as an integral part of the ontologies. The ontologies are considered both as individual entities but also as integral parts of the cloud forming a greater whole. This aspect is taken into account at all levels of the development and publication of the ontologies, thus leading to a different process and underlying philosophy compared with the usual approach of linking independently developed ontologies.

1.2 A National Effort in Building a LOO Cloud

In order to test and evaluate these hypotheses in practice, a LOO cloud called KOKO of cross-domain ontologies (e.g. health, cultural heritage, agriculture, seafaring, government, defence) has been realised in Finland on a national level during the FinnONTO research project (2003–2012). Various libraries, archives, museums, and governmental actors have annotated vast amounts of documents, each with their own thesauri. The aim of the KOKO cloud is to maintain backwards compatibility with the existing annotations while allowing for better interoperability between different datasets. Thus, the system is based on transforming a set of legacy thesauri in use into light-weight ontologies and interlinking them with each other. Currently, KOKO encompasses sixteen ontologised thesauri with more to be integrated into the system in the future. The current version of KOKO is a harmonised global ontology of some 47,000 concepts aligned into a single hierarchy.

In the centre of the KOKO cloud is the General Finnish Ontology YSO, based on the General Finnish Thesaurus YSA[7], which has been developed since 1987 by the National Library of Finland and is used across various organisations in Finland. YSO provides the upper hierarchy, originally inspired by the ideas of "foundational ontologies", such as DOLCE [9], and shared "upper ontologies", such as IEEE SUMO [27]. The idea is to provide the LOO cloud with the common upper concepts needed in many domains. YSO is then complemented by more refined ontologies for specific domains. These component ontologies are developed by the experts responsible for the original thesauri preserving the domain know-how while allowing for asynchronous updating based on the resources of each organisation. From an end user's point of view, KOKO ontology is seen and used

[7] http://vesa.lib.helsinki.fi/ysa/

as a single ontology without boundaries; for the ontology developers, domain boundaries are needed in order to divide the responsibilities of distributed ontology work based on domain expertise needed in different parts of the KOKO cloud.

KOKO ontology was originally published in the ontology library system ONKI[8] [36,37] operating as a living lab research environment. Over the years ONKI has been integrated into, e.g. several museums, libraries, and web portals. In 2013 the National Library of Finland launched a joint project with the Ministry of Finance and the Ministry of Education and Culture to build a permanent, national ontology service Finto[9] based on the ONKI system [35]. The National Library also took on the responsibility of coordinating further development of the KOKO cloud. Finto ontology service provides a centralised publication channel for the ontologies with common interfaces for accessing them in various applications.

1.3 Challenges in Developing a Linked Ontology Cloud

Several issues need to be taken into account when moving from developing individual thesauri into developing and maintaining a cloud of interlinked ontologies. In this paper, the following challenges are discussed.

- **Creation of ontologies.** How is an existing legacy thesaurus transformed into an ontology? How is a new ontology mapped into the linked ontology system? How are the URIs of the concepts formed?
- **Development of ontologies.** How are the overlapping parts of ontologies recognised in order to minimise the duplicate work of the ontology developers? How are the changes in one ontology communicated to other related ontologies? How are the errors and other quality issues recognised in a system of several ontologies?
- **Publication of ontologies.** How should the linked ontology system be presented to the end user in order to make it comprehensible? What kind of services for using the ontologies are needed for different user groups?

1.4 Structure of the Paper

This paper is divided into four main parts. Section 2 presents a model for creating and updating a linked ontology cloud and lists a set of seven principles guiding the process. The following Sections 3–5 describe the development cycle of the cloud in more detail with an emphasis on how a single domain ontology is handled by the process. Throughout, the KOKO cloud provides an illustrative use case on how the process has been applied to practice. Since the traditional, comparative evaluation of the process is difficult, the extensive application to practice has been used as a proof of concept with the process being adjusted based on real-life experiences in accordance with the principles of action research [5]. The main user groups for this have been the ontology developers on the one hand, and the systems that KOKO has been integrated into on the other hand. In the final Section 6, related work is presented and the contributions of the paper are summarised.

2 A Model for Managing a Linked Ontology Cloud

Our ontology development work started by a field study on how thesauri in use are actually developed. The result was that thesauri are typically developed by independent expert groups focusing on concepts in their own domains of interest, with little collaboration between the groups. The situation seems to be more or less similar in other countries, too. Obviously, this model leads to redundant work in developing overlapping areas of thesauri and, at the same time, to interoperability problems between the thesauri, since different parties define their concepts without considering each others' choices. To address these problems we propose a more coordinated collaborative model for developing a linked ontology cloud, which is depicted in Fig. 1. Note that in the following discussion the bolded numbers in parentheses refer to the numbered parts in the figure.

2.1 Ontology Creation Phase

First, existing thesauri (**1**) and ontologies (**2**) are selected for building blocks of the ontology cloud. A thesaurus is converted into RDF format using a shared ontology schema (**3**) and aligned with a general upper ontology (GUO) (**4**). Aligning domain ontologies with a GUO forms the basis for interoperability by providing a complete concept hierarchy and is much easier to maintain than direct, pair-wise mappings between domain ontologies [12]. This idea was suggested, e.g. by the IEEE Standard Upper Ontology (SUO)[10] working group.

The alignment can be done in a semi-automatic fashion by first generating equivalency mappings automatically and then correcting them manually. In addition to equivalency mappings, subsumption and partitive relations might be used in the case of light-weight ontologies. For example, the concept "antique furniture" in a museum domain ontology may be aligned with the concept "furniture" in the upper ontology using a subclass-of relation. In order to create a complete, fully connected linked ontology, all concepts

[8] http://onki.fi/
[9] http://finto.fi/en/
[10] http://suo.ieee.org/

of a domain ontology should be mapped to the GUO either directly or through other concepts in the domain ontology. The transformation is discussed in more detail in [16]. In our case study, a natural basis for a GUO was the General Finnish Thesaurus YSA that was transformed into the General Finnish Ontology YSO. already as a reference in many specialised thesauri.

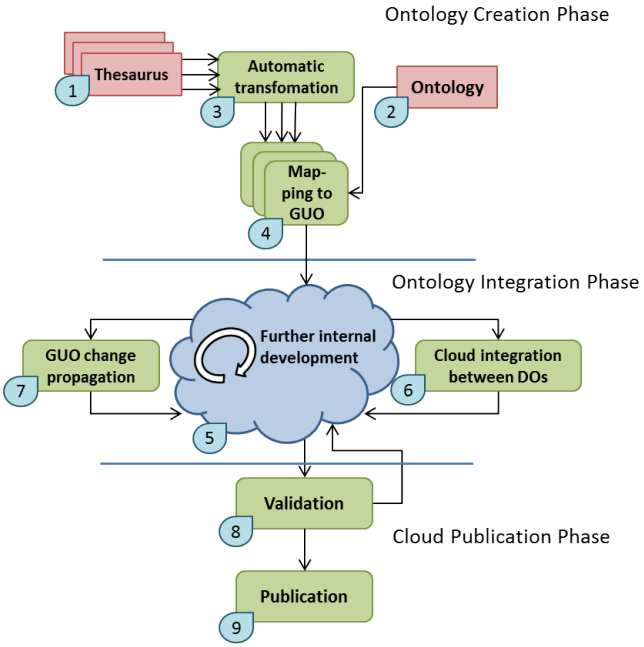

Fig. 1 The model of Linked Ontology Cloud formation and management

2.2 Ontology Integration Phase

The ontology integration phase begins after the domain ontologies have been aligned with the GUO (**5**). There may be mutually overlapping parts between the domain ontologies because the alignments were made between the domain ontologies and the GUO only. To facilitate the integration of domain ontologies (DO) (**6**), processes, and tools for discovering overlapping parts of the ontologies are needed. Based on the analysis, it is possible to eliminate redundant development work by deciding between domain ontology developers which ontologies maintain which overlapping parts.

Changes in the GUO create pressure for the domain ontologies to be updated accordingly to ensure the consistency of the cloud (**7**). For example, in our case study system, some 2,000 changes are made annually in the upper ontology YSO. The changes should be taken into account in the development process of the domain ontologies. Fortunately, not all changes in the GUO are relevant to all domain ontologies, but only those related to them via equivalency, subsumption or partitive relations.

2.3 Cloud Publication Phase

When a development cycle of an ontology cloud has been completed, its logical consistency and other quality aspects should be validated (**8**) making sure that the resulting ontology adheres to the constraints of the properties and classes used. Some of the problems encountered can be fixed automatically [34] (e.g. overlaps in disjoint semantic relations and cycles in the concept hierarchy) but they should nonetheless be communicated to the developers. However, automatic validation has difficulty in finding problems on the semantic level, which usually requires manual checking. Finally, the ontology cloud can be published as services for humans and machines, e.g. via user interfaces, APIs, and downloadable files (**9**).

2.4 Principles for Building a Linked Ontology Cloud

In our case study, the KOKO cloud based on the sixteen ontologised thesauri presented in Table 1 was created. Below, the lessons learned during the work are summarised as a seven-point list of practical building principles. We consider the proposed principles novel, as we are not aware of previous ontology design patterns focused on managing an ontology cloud as a whole.

I The ontology cloud consists of one general upper ontology and several domain ontologies that are linked to the upper ontology with subsumption, equivalency, associative and partitive relations.
Reason: This means that the domain ontologies do not have to be linked to each other pairwise, as the upper ontology acts as semantic glue for joining all the ontologies together. Shared concepts are included in the upper ontology. The idea is to minimise the links between the domain ontologies, which simplifies their development since a given developer needs only concern herself with her own domain ontology and the GUO.

II Every concept in a domain ontology has a subsumption or equivalency relation to a concept in the GUO or a subsumption relation to a concept in the same domain ontology.
Reason: This means that every concept in a domain ontology needs to be able to trace a subsumption relation to a concept in the general upper ontology. This ensures a consistent concept hierarchy for the whole cloud and that domain ontologies can not define new top-level concepts.

Name of the ontology	Domain	Number of concepts
YSO	General upper ontology (GUO)	27,200
AFO	Agriculture and forestry	7,000
JUHO	Government	6,300
KAUNO	Literature	5,000
KITO	Literary research	850
KTO	Linguistics	900
KULO	Cultural research	1,500
LIITO	Economics	3,000
MAO	Museum artefacts	6,800
MERO	Seafaring	1,300
MUSO	Music	1,000
PUHO	Military	2,000
TAO	Design	3,000
TERO	Health	6,500
TSR	Working and employment	5,100
VALO	Photography	2,000

Table 1 The ontologies comprising the LOO cloud KOKO

III If a concept in a domain ontology has an equivalency to a concept in the GUO, it may not have broader concepts in the domain ontology, which lack an equivalency relation to a concept in the GUO.
Reason: This is needed to avoid having dependencies from the GUO towards a domain ontology by forbidding domain ontologies from introducing broader concepts to concepts in the GUO (through inference over equivalency relation). Otherwise domain ontology developers could propagate contradictions or unwanted concept (re)definitions into the GUO, especially in cases where more than one domain ontology is involved.

IV The domain ontologies are focused on a clearly bounded domain and are as self-contained as possible.
Reason: This allows the domain ontology developer to concentrate on the area of her expertise. This also minimises dependencies between domain ontologies and facilitates ontology development work.

V A concept in a domain ontology may not have an equivalency to a concept in another domain ontology. A concept in a domain ontology may have an associative relation or have a broader concept in another domain ontology at the discretion of the developers.
Reason: Dependencies between domain ontologies cannot always be avoided due to inherent relations across the borders of different domains. This means that the developers need to monitor the changes to the other domain ontology but this is allowed since it does not affect the other domain ontology directly. The use of broader and associative relations extends the target domain ontology but does not affect its semantics (strongly), whereas the equivalency relation would possibly introduce new broader concepts to concepts in the target ontology and thus redefine their semantics.

VI The GUO and the domain ontologies use a shared ontology schema and are based on domain-independent standards.
Reason: Thus the ontologies can be easily merged and used together with various data sources outside of those directly linked to the cloud, using standard software, e.g. SKOS or RDFS/OWL tools.

VII The resulting ontology cloud should be logically consistent, e.g. by ensuring the integrity of the concept subsumption hierarchy over ontology boundaries (since transitivity is assumed).
Reason: This allows for reasoning and query expansion over the whole cloud.

3 Creating a Thesaurus-based Ontology for the Linked Ontology Cloud

The creation of a cloud of linked ontologies begins with the formation of the general upper ontology. This ontology provides a completely connected hierarchy of general concepts including the topmost division, e.g. between abstract, endurant and perdurant concepts [9]. This forms the basic structure for the domain ontologies to map into, thus saving them from having to repeat the higher parts of the hierarchy.

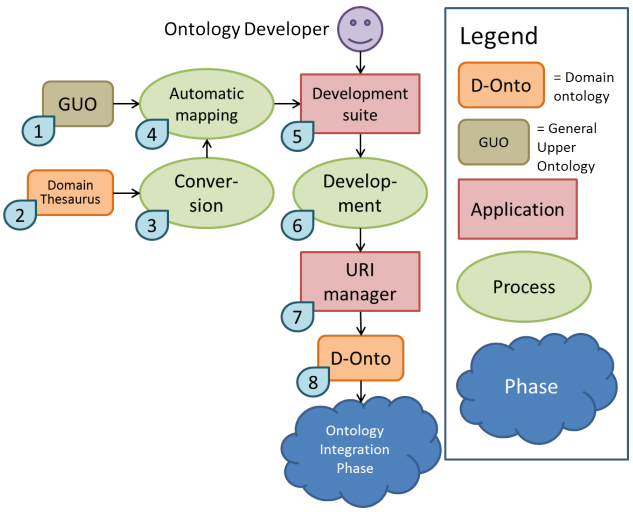

Fig. 2 Ontology Creation Phase

Fig. 2 depicts the process of creating a new domain ontology for the cloud of linked ontologies, which begins with the conversion (**3**) of the domain thesaurus (**2**) from a legacy

format into RDF, typically SKOS. This is often a straightforward operation but in some cases the original thesaurus can include relations that are not easy to convert to SKOS. In these cases it can be a good idea to retain the original relations in the form of a temporary predicate which can then be harmonised in the publication phase of the process.

In the FinnONTO case, we used ad-hoc scripts for the transformation since the domain ontologies were in different formats. We also transformed them originally into a custom OWL-based format since at the beginning of the project SKOS had not yet been established as a standard way of representing light-weight RDF ontologies. We continued this practice in part because of existing tools, such as Protégé[11], but also because some thesauri used relations not present in SKOS. An example of this was the Agriforest[12] thesaurus of agriculture and forestry, which uses different relations to differentiate between names of concepts that were derived from different sources. These the were preserved for the benefit of the ontology developers but were then combined into a common predicate for publication.

The next step of the process is the automatic mapping (**4**) to the GUO (**1**). This can be done roughly through string comparison between labels or, alternatively, by also utilising the structure of the ontologies, depending on the nature of the original thesaurus. Since different thesauri may have different labelling conventions (for example plural vs. singular forms) some sort of stemming or lemmatisation may also be needed for the label comparison. The result is a preliminary mapping which then needs to be checked manually in the next part (**5** and **6**) by a domain expert ontologist since the aim is to produce as good and reliable a result as possible. Aside from checking the mapping, the human development part also entails the ontologisation work proper, which needs to be done when changing vaguely defined terms into more precise ontology concepts [16,21]. In many cases, a term in the original thesaurus becomes several concepts in the ontology depending on whether the term has multiple meanings. When a single term corresponds to several concepts, we have added a qualifier in parentheses to the preferred label for each concept in order to make it easier to differentiate between them. For example, the ambiguous keyword ”child” could be split into concepts ”child (age)” and ”child (family relation)”.

In the FinnONTO project, we did the preliminary mapping between a domain ontology and the GUO by label comparison using lemmatisation and custom scripts, while most of the actual ontologisation work was done by the experts from the organisations that maintained the thesauri. Now there exist specialised ontology matching tools, such as AgreementMakerLight[13], which could be utilised for the initial mapping.

Finally, the URI scheme of the ontology needs to be considered. A good starting point is the URI design principles presented by W3C in *Cool URIs for the Semantic* Web[14]. Human-readable meaning-bearing URIs pose difficulties in that they are language-dependent and, most importantly, in the case where further development ends up changing the label of the concept, the persistent URI can not keep up with the changes, thus gradually leading to the degeneration of the mapping between the labels and URIs. Since we are building on thesauri that have been in active use and development for long, we must also consider the long term effects of our current decisions. Therefore, we decided to use language- and meaning-neutral URIs where the local name consisting of a letter and a string of numbers (e.g. `p3612`).

Since ontologies evolve with new concepts and URIs introduced and old ones possibly deleted, a system (**7**) is needed for tracking the use of the URIs. In FinnONTO, we created our own tool for this called Purify[15]. Purify harmonises all local names in a given namespace to the letter followed by a number format. The tool keeps a log of the mappings between the possible temporary URIs used during the early stages of the development since human-readable URIs were found to be useful at the beginning of the ontologisation. Finally, Purify makes sure that no two resources get the same URI and that if the URI generation needs to be repeated, the result will be the same every time.

With the first version of a domain ontology completed (**8**) and successfully mapped to the GUO, the next step is to integrate it into the ontology cloud proper. Table 1 lists all the ontologies completed for our LOO cloud KOKO at the moment. The name of the ontology is followed by a description of its domain and the number of concepts.

4 Maintaining a Cloud of Interlinked Ontologies

There are two main concerns when integrating and managing domain ontologies in a cloud: how to manage the domain ontologies with respect to one another and how to manage their relation to GUO. This phase is depicted in Fig. 3, continuing from Fig. 2, and is explained in depth below.

4.1 Avoiding Overlap Between Domain Ontologies

The Principles IV and V presented in Section 2 posit that in order to keep relations between domain ontologies to a minimum, the domains covered need to be precisely set. With

[11] http://protege.stanford.edu/
[12] http://www-db.helsinki.fi/agri/agrisanasto/Welcome_eng.html
[13] https://github.com/AgreementMakerLight
[14] http://www.w3.org/TR/cooluris/
[15] http://puri.onki.fi/info/

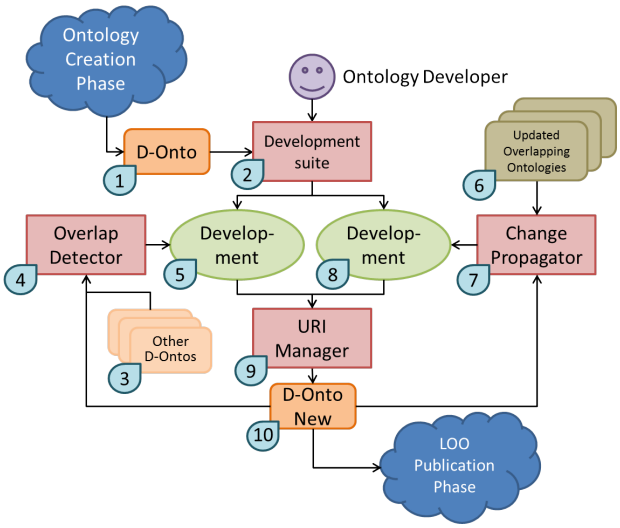

Fig. 3 Ontology Integration Phase

minimal links between domain ontologies, they can be developed independently from one another, but it also means that the curation of the ontological domains is an on-going process. Overlaps can come into being especially in the border areas of two domains where a given set of concepts can not be unequivocally said to belong to either one of two different ontology domains.

In Fig. 3, we can see that the process of discovering these overlaps between a given domain ontology (**1**) and the rest of the domain ontologies in the cloud (**3**) makes use of a tool (**4**) to help the domain ontology developers in finding and even analysing the overlaps. This tool can be based on string matching similar labels and reporting on the potentially overlapping concepts, but it can also make use of the ontological structure and relations to find overlapping concepts [8].

Once an overlap between two or more domain ontologies has been discovered, a dialogue (**5**) should be started between the developers of those ontologies. Its possible end results are as follows:

a) The concepts that overlap are really the same concept and the developers of the domain ontologies and the GUO can agree that the concept is general enough to be included in the GUO. In this case, the concept can be removed from the domain ontologies.
b) The concepts that overlap are really the same concept but the developers can agree that the concept is not needed in both (or all) of the domain ontologies and agree on a single domain ontology that should host the concept in question and handle changes and development further on that concept and its subconcepts. This is most common in situations where the concept is clearly in the domain of one ontology and only included in the other due to historical reasons, for example.
c) The concepts that overlap are really the same concept and the ontology developers wish for it to remain present in both (or all) of the ontologies. A note should be made that if the concept is changed or developed further, the other developers should be informed as needed.
d) The concepts that overlap are actually different concepts but might share a preferred label. In this case, the labels should be differentiated from one another if possible so as to avoid confusion when the ontologies are used together.

Great care should be exercised when choosing option c) since it clashes with Principle V and can easily lead to increasing complexity in development. This should be avoided as much as possible, since one of the main goals of the presented system is to make the asynchronous development of ontologies possible and having the same concept in two domain ontologies means that both sets of developers need to agree on possible further development.

In our case study, we implemented a tool called KOAN, based on simple label matching, for finding ontology overlaps since the light-weight ontologies do not offer much structure to use as a basis for the discovery task. Furthermore, identical labels pose the most difficulty for the annotators using the ontology since they would need to decide between different concepts with the same names based on their place in the hierarchy. Having concepts with different labels that end up being the same is much less of a problem since, when the mistake is discovered, it is relatively easy to combine the annotations made using the duplicated concepts.

When applied to KOKO, KOAN found lots of overlapping concepts even between ontologies of seemingly very distinct domains. Table 2 shows some of the comparison results between ontologies by showing the per cent amount of overlap. For instance, from the first row, second cell, we can see that the agriculture and forestry domain ontology AFO contains 8 % of the concepts of the government domain ontology JUHO. Comparing with Table 1 we can see that even ontologies from seemingly very distinct domains can have a lot of overlap between them. Maintaining the overlaps in multiple places at the same time creates a lot of unnecessary work and can lead to inconsistencies in the transitive relations. Our aim is to implement a systematic process for the elimination of these overlaps.

In addition to overlapping concepts, a domain ontology may have a concept with an associative relation or a broader concept in another domain ontology because cross-domain relations cannot always be avoided. For example, in our use case, the museum domain ontology MAO contains the concept ”catapults”, which is a subclass of the concept “weapons” in GUO. On other hand, the military domain ontology PUHO has the concept “single-shot weapons”, which could be used as the superclass of “catapults” for more refined semantics. However, using relations between domain ontologies may

Ontology	AFO	JUHO	KAUNO	MERO	TERO
AFO		8%	2%	3%	25%
JUHO	7%		16%	5%	40%
KAUNO	2%	12%		1%	28%
MERO	0%	1%	0%		2%
TERO	23%	41%	36%	13%	

Table 2 Number of overlapping concepts in five domain ontologies of KOKO

be a justification of moving such cross-domain concepts ("single-shot weapons" in the example) into GUO in order to minimise dependencies between domain ontologies.

4.2 Keeping the Domain Ontologies up to Date Regarding GUO

The second part of the cloud management process, as depicted on the right hand side of Fig. 3, is the handling of the changes in the upper ontology. As an implication of Principle II, the domain ontologies react to the changes in the upper ontology. In other words, when the GUO developers release a new version (**6**), the domain ontologies need to be potentially updated.

The structure of the cloud aims to allow for the asynchronous updating of the domain ontologies, which can result in long intervals between the updates of a given domain ontology. Additionally, the ontologies are often developed in different organisations, and using separate ontology editors creates a challenge in how to communicate the changes between the developers. The two main approaches to conveying the changes are push and pull synchronisation [6]. The push version propagates the changes in one ontology immediately to the other ontologies, whereas in the pull version the change listing is requested when needed by the ontology developer. The push approach is good for situations where the ontologies are updated frequently, so that the ontology developer can quickly ensure the consistency of the ontology after the changes. However, this approach is challenging if the ontology reacting to changes is update infrequently due to, e.g., lack of resources. Then it would be preferable that the changes could be acquired when needed and not propagated immediately. A long update interval also means that the amount of changes can build up over time and when the update process of the domain ontology is started, the number of changes that need to be checked can be in the thousands.

In order to ease the work of the domain ontology developer, a tool (**7**) needs to be used for propagating the changes. It would also be beneficial to order or categorise the changes of the GUO somehow according to their relevance to the domain ontology in question. A set of criteria for estimating relevance should take into account the differences in ontologies and the relations between them as well as the likely changes that are going to occur in development.

In the FinnONTO project, we created a change propagation tool MUTU and a set of accompanying relevance criteria as described in [29]. The basic idea is that a change in the GUO is likely to be relevant to domain ontology developers if it concerns a concept that has been directly linked to from the domain ontology (a connecting concept). If a concept in the GUO has been marked as equivalent or as a superclass to a concept in the domain ontology, any change to it is likely to be of interest to the domain ontology developers. Furthermore, if the concept hierarchy of an ancestor changes for a connecting concept in GUO, this is likely to be relevant to the domain ontology developer due to the transitive nature of the relation. If a concept is removed, rare though that is, that is deemed as interesting due to the fact that this concept could have been in use by the domain ontology users but has not been duplicated to the ontology itself. Finally, if a concept has been added that has the same label as a concept already existing in the domain ontology, this is always relevant.

MUTU simply finds out the changes between the new version of the GUO and the one that was used for the mapping of the domain ontology, lists these changes according to the type of the change and relevance, and adds helper classes to the development version of the domain ontology for grouping the concepts in the ontology editor suite that need to be checked by the developer (**8**). MUTU also allows the domain ontology developer to configure it according to her needs based on, e.g., a specific priority of languages or on blocking changes from certain properties deemed irrelevant to the domain ontology development.

5 Publishing a Linked Open Ontology Cloud

Once the individual component ontologies have been developed and the links between them have been curated, the ontology cloud needs to be published. The aim is to provide the users with a single, unified whole that can be used essentially as a single ontology like any other. The process of publishing a Linked Open Ontology Cloud is depicted in Fig. 4.

5.1 Validation, Merging and Publication

The publication phase starts when a new version of a domain ontology (**1**) is ready and the developer wants to publish it in the LOO cloud. The first step is to clean up the ontology for publication (**2**), by removing structures needed

only in the development of the ontology. For example, temporary concepts that are used for grouping concepts that are under development and editorial notes for internal use by the ontology developers can be removed. Similarly, temporary ontology-specific predicates should be converted to the common schema.

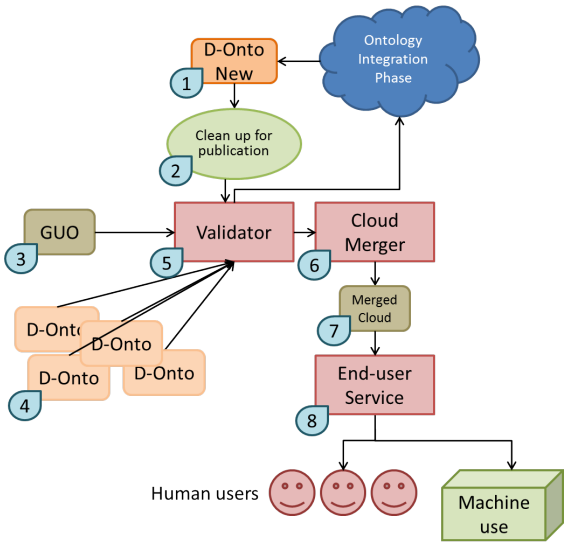

Fig. 4 Cloud Publication Phase

In the FinnONTO case, the domain ontologies were developed in a Protégé project with the general upper ontology included. To help the ontology developer to focus on the concepts of the domain ontology, the topmost concepts of the domain ontology (the ones that do not have subclass-of relation to a concept in the domain ontology) are connected with subclass-of relation to a temporary concept acting as a root concept for the ontology. This root concept is removed in the clean up process as the goal is to present the resulting ontology cloud to the end users as a single complete hierarchy with a single root concept.

Before the new domain ontology version is merged into the cloud it is validated (**5**) for compliance with Principle VII, e.g. by checking the logical consistency and spotting the violations of best practices. The results of the validation are communicated to the ontology developer who can then fix the problems. A validator may also fix some of the problems automatically, e.g. by removing cycles in the concept hierarchy.

For the validation of the ontologies we are using the Skosify tool [34], which converts RDFS/OWL/SKOS vocabularies into proper SKOS format. Overlaps in disjoint semantic relations (`skos:related` and `skos:broaderTransitive`) are checked and inconsistencies are corrected automatically by removing `skos:related` relations in problematic cases. In cases where a concept has more than one `skos:prefLabel` per language one of the labels is arbitrarily selected for use as `skos:prefLabel` while the others are simply converted into `skos:altLabels`. A check is performed in order to detect overlaps in disjoint label properties (`skos:prefLabel`, `skos:altLabel`, and `skos:hiddenLabel`) and the superfluous ones are removed. As a best practice, the cycles in the concept hierarchy are removed, and finally extra whitespaces are removed from concept labels. The violations are also reported to the ontology developer so he may fix the problems in a controlled way instead of relying on the automatic fixing procedure. We also used Skosify to convert the ontologies from the custom OWL format to SKOS for publication. SKOS was chosen due to the amount of tools available and for its suitability for our use case of light-weight ontologies.

After the validation, the new version of the domain ontology is merged (**6**) with the GUO (**3**) and the other domain ontologies (**4**). The idea is to build a single representation of the linked ontology cloud (**7**) by merging equivalent concepts and giving them a single URI. The differing ontology schemas are harmonised so that the end user does not get confused with the ontology-specific structures and naming conventions.

When annotating using the linked ontology cloud, the annotator should be making choices between concepts as opposed to ontologies. To this end, the cloud is merged (**6**) into a single whole (**7**) by taking all the concepts linked with `skos:exactMatch` relations between them (marked as equivalent) and combining them. In case a clearer division between the development version of individual ontologies and the published version of the cohesive cloud needs to be made, new URIs can also be assigned to the concepts of the cloud. Here care needs to be taken in order to ensure that the same concepts always map to the same URIs even in cases where that concept might at first originate from the upper ontology and later have been moved to a domain ontology or vice versa.

In FinnONTO, we gave all the concepts in the KOKO cloud new URIs in a new namespace. The combination of the ontologies listed in Table 1 resulted in a combined cloud of some 47,000 concepts. In order to allow the end user to select only from a single domain, we assigned domain ontology specific concept types as subclasses of `skos:Concept`.

Once the ontology cloud is merged, it is ready for publication (**8**). In accordance with the "open" part of the name LOO, the cloud is published as Linked Data [13], so that individual concept URIs can be referenced from various datasets and URIs are resolvable to information about the concepts in human- and machine-readable forms. The LOO cloud is published for humans via user interfaces for searching, browsing and visualising the ontologies, and as widgets for integrating ontologies into applications. For machine use, additional REST and Web Service APIs are provided to facilitate

even deeper integration of the ontologies. Finally, the ontology cloud is published as a SPARQL endpoint enabling ad hoc queries for more complex needs.

These services were provided by the ONKI Ontology Service [37], part of which was further developed by the National Library of Finland in the form of Finto thesaurus and ontology service. These services act as repositories for vocabularies, thesauri, and ontologies, providing support for publishing, finding and accessing them. The main user groups of ONKI and Finto are ontology developers, content indexers and information searchers. Ontology developers need a way to visualise the ontology they are developing, and especially in the context of the LOO cloud, where the domain ontology developers map their ontologies only to the general upper ontology, there is a need for getting an overview of the whole cloud. This way the developers can see how their domain ontologies are situated in the LOO cloud and discover possible overlaps with other domain ontologies.

Content indexers and information searchers need ways of finding ontologies and concepts suited for their needs. The Linked Open Ontology Cloud approach eliminates the need for finding ontologies and making selections between them, as one ontology system covering all domains of life aims to fulfill the needs of everyone. For finding suitable concepts, ONKI/Finto service provides an ontology browser which visualises the ontologies as a tree hierarchy and shows other relations between concepts. The user may use autocompletion search for finding concepts based on their labels. In addition to dereferenceable URIs, machines are served with REST API, and a SPARQL endpoint[16]. The general ontology service software powering Finto is developed currently by the National Library of Finland as an open source project Skosmos[17] and can be freely used to set up a similar service. The ontology cloud is published under a permissive Creative Commons Attribution license[18].

5.2 Use Cases

During the FinnONTO research project, the adoption of KOKO was hampered by the uncertainty regarding its future. With the ontology service project of the National Library of Finland, the development and publication of KOKO was given sustainable governmental resources thus securing its future. However, due to the length of the process of securing funding, the wide-spread adoption of KOKO is only beginning.

KOKO and ONKI/Finto have been in daily use in many museums, such as the Espoo City Museum, where it has been integrated into the collection management system Kauko. The first large scale system using KOKO was the semantic cultural heritage portal CultureSampo[19] [25] aggregating contents from various data sources. KOKO is a basis of the deployed BookSampo[20] [24] literature portal of the Finnish public libraries with some 60,000 monthly end users on the Web.

At the moment, Finnish archives are integrating KOKO ontologies via Finto into their common search registry service for metadata on both digital as well as conventional documents with the AHAA project [19]. A recent company pilot user of KOKO has been the Swedish language division of the national public service broadcasting company of Finland, Svenska YLE. They have used KOKO in the annotation of their web content and have complemented it with Freebase[21] for instance references, such as people and organisations. The pilot has been a success and they are considering adopting the system in the Finnish part of YLE as well as their archive.

The multi-disciplinary nature of KOKO means that it is especially suited to organisations that deal with cross-domain contents such as media organisations, museums, libraries, and archives. Furthermore, using the concepts from domain ontologies links the content to more in-depth data from the specialist organisations that originally developed the domain ontologies for their data annotation needs.

6 Conclusions and Discussion

We described a process for creating and managing a Linked Open Ontology Cloud, based on existing legacy thesauri. To conclude, we compare our approach to related work, summarise our contributions, and suggest future work.

6.1 Related Work

Our work on linking ontological concepts used in annotations complements the idea of the Linked Open Data cloud, where emphasis is on data entity linking using (typically) `owl:sameAs` properties [11]. We also complement the work on Linked Open Vocabularies (LOV)[22], which focuses on metadata schema linking based on property hierarchies rather than linking domain ontologies. Linking the data through ontologies allows additional interoperability due to the inferenced knowledge gained through the shared ontology semantics [14]. Our approach follows this principle and provides a framework for interlinked ontology cloud develop-

[16] http://api.finto.fi/sparql; The KOKO cloud is available in the named graph http://www.yso.fi/onto/koko/.
[17] http://github.com/NatLibFi/Skosmos/
[18] http://creativecommons.org/licenses/by/3.0/
[19] http://www.kulttuurisampo.fi/
[20] http://www.kirjasampo.fi/
[21] https://www.freebase.com/
[22] http://lov.okfn.org/dataset/lov/

ment. Backwards compatibility with existing annotations is retained by basing the ontologies on legacy thesauri in use.

Another example of highly linked ontologies can be found in BioPortal[23], an ontology repository that has features supporting collaborative (inter-)ontology development. In addition to uploading new ontologies to BioPortal, users can also create and upload mappings between the concepts of different ontologies [28]. The mappings can be used to bridge overlapping ontologies or to extend general ontologies with specialised ones. Users can comment and create discussion threads on ontologies, their parts, and mappings, thus supporting a collaborative and open inter-ontology development process. However, a set of mapped ontologies does not appear as a single whole to the user though one can browse the ontologies by following the mappings between concepts. Moreover, the mappings are not utilised in the ontology development process to propagate the changes of an (upper) ontology to ontologies extending it, as they are in our LOO model.

In order to move from a group of ontologies into a coherent ontology system, the ontologies need to be reconciled. Ontology reconciliation [12] is a broad term, covering ontology merging, alignment, and integration. Most of the reconciliation methods are automatic or semi-automatic, which can lead to lower quality, especially if the ontologies were originally expert-made [7]. Our approach emphasises the importance of manual development work in the reconciliation process.

In ontology modularisation [1], ontologies are divided into smaller interlinked parts to facilitate distributed development and re-use. Our approach merges ontologies based on existing thesauri to form a Linked Open Ontology Cloud, while the development of the individual ontologies continues in a modularised way. Furthermore, we present the resulting interlinked cloud as single whole so that the end users do not have to make selections between ontologies.

There have been several previous efforts on building a general upper ontology [26] that can be used as a foundational basis for domain ontologies. Some of the upper ontologies have been developed from scratch, while, e.g. the Suggested Upper Merged Ontology SUMO [27] was created by merging existing ontologies. We used the General Finnish Ontology YSO, transformed from an existing general thesaurus YSA, as the upper ontology in our Linked Open Ontology Cloud. In contrast to the previous work on upper ontologies, which focuses on the creation of an upper ontology, our work emphasises the model for managing the domain ontologies as part of the ontology cloud and keeping them up to date and synchronised with the changes of the upper ontology.

Related to the principles of forming and managing the ontology cloud presented in this paper, similar guidelines have been presented in the context of the OBO Foundry initiative [30]. However, in OBO Foundry the focus is on coordinating the development of different ontologies under shared principles, but not on merging them into a single ontology cloud and the challenges therein. General, domain-independent ontology design principles have been proposed by several researchers [10, 38]. Our model uses their ideas as a foundation, e.g. by organising orthogonal concept domains into separate ontologies and supporting ontologies of different granularity levels in the form of a GUO and the domain ontologies extending it. The principles presented in this paper extend the general ontology design principles by covering modelling issues related to the management of changes in a linked ontology cloud.

Keeping domain ontologies up to date with the changes of the GUO is closely related to the topic of ontology evolution, which concentrates on addressing the changes in ontologies over time. Different change types have been listed in [33], whereas [20] considers more abstract change patterns constructed from atomic changes. According to [4] users would have liked to see explicitly when a concept created by them had been modified, indicating that not all the changes occurring in an ontology are equally relevant to all developers providing the impetus for our work on building a set of priorities for different changes. The detection of changes in an updated ontology can be done using logs [32] or by comparing two versions of the ontology [22], which was the approach we chose. Additionally, the extra challenges of distributed ontology development have been addressed in [20, 23, 18]. In our approach there is no assumption that all the ontology developers use the same ontology editor, as the support tools are implemented separately from the ontology development suite.

6.2 Contributions and Future Work

We presented the idea of the Linked Open Ontologies (LOO) for fostering data integration. LOO complements dataset entity linking in LOD and metadata schema linking in LOV. The realisation of the LOO cloud was achieved by facilitating an environment of interconnected but easily manageable set of cross-domain ontologies allowing for distributed ontology development. This can be more efficient since the mappings between ontologies can be re-used for different datasets.

Furthermore, we presented a detailed description of managing the overlaps between domain ontologies and the inconsistencies resulting from the asynchronous updating of the GUO compared with the domain ontologies. Finally, we implemented a LOO in practice with sixteen ontologies and a full set of tools for the development cycle from a set of thesauri to a fully realised Linked Open Ontology Cloud. Based on this work, we accompanied the LOO model with

[23] http://bioportal.bioontology.org/

a set of seven principles that guide the process of building and managing an ontology cloud. The principles are general enough to be applied to other ontology clouds in addition to the one we have built.

A central challenge in the linking process is in maintaining the integrity of the transitive relations across different ontologies and we have achieved this through the use of a general upper ontology acting as a central hub for the linking of various domain ontologies. We consider proactive linking as an integral part of the development process itself as opposed to simply mapping independently developed ontologies.

The model was piloted by implementing it into practice in extensive scale in various organisations encompassing many different domains. The model evolved based on lessons learned during the multi-year implementation and development process. Feedback was gathered from ontology developers as well from the systems integrating the linked ontology cloud. Based on the success of the prototype, the model is now being applied on a national scale in archives, libraries, and museums, as well as in ministries and other governmental agencies.

For our future research, we are focusing on building better tools and refining the processes for tracking and communicating the changes in the GUO to the domain ontology developers. To this end, we are also devising a more formal administrative process for the development and overall coordination of the KOKO cloud. Since it is a joint operation by over a dozen different organizations the particulars of the administration need to be both flexible yet well-defined.

References

1. S. B. Abbès, A. Scheuermann, T. Meilender, and M. d'Aquin. Characterizing modular ontologies. In *Proceedings of the 6th International Workshop on Modular Ontologies*. CEUR Workshop Proceedings, `http://CEUR-WS.org`, July 2012.
2. J. Aitchison, A. Gilchrist, and D. Bawden. *Thesaurus Construction and Use: A Practical Manual*. Aslib IMI, 2000.
3. J. Bhogal, A. Macfarlane, and P. Smith. A review of ontology based query expansion. *Information Processing & Management*, 43(4):866–886, 2007.
4. S. Braun, A. Schmidt, A. Walter, and V. Zacharias. The Ontology Maturing Approach to Collaborative and Work Integrated Ontology Development: Evaluation Results and Future Directions. In *Proceedings of the International Workshop on Emergent Semantics and Ontology Evolution at (ISWC 2007)*, pages 5–18, 2007.
5. F. Burstein and S. Gregor. The systems development or engineering approach to research in information systems: An action research perspective. In *Proceedings of the 10th Australasian Conference on Information Systems*, pages 122–134. Victoria University of Wellington, New Zealand, 1999.
6. P. Deolasee, A. Katkar, A. Panchbudhe, K. Ramamritham, and P. Shenoy. Adaptive Push-Pull: Disseminating Dynamic Web Data. In *Proceedings of the 10th international conference on World Wide Web (WWW 2001)*, pages 265–274, 2001.
7. Z. El Jerroudi and J. Ziegler. iMERGE: Interactive Ontology Merging. In *Proceedings of the International Conference on Knowledge Engineering and Knowledge Management (EKAW 2008)*, pages 52–56, 2008.
8. J. Euzenat and P. Shvaiko. *Ontology Matching*. Springer–Verlag, 2007.
9. A. Gangemi, N. Guarino, C. Masolo, A. Oltramari, and L. Schneider. Sweetening ontologies with dolce. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web*, EKAW '02, pages 166–181. Springer-Verlag, 2002.
10. T. Gruber. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, 43(5-6):907–928, 1995.
11. H. Halpin, P. J. Hayes, J. P. McCusker, D. L. McGuinness, and H. S. Thompson. When owl:sameAs isn't the same: An analysis of identity in linked data. In P. F. Patel-Schneider, Y. Pan, P. Hitzler, P. Mika, L. Zhang, J. Z. Pan, I. Horrocks, and B. Glimm, editors, *The Semantic Web – ISWC 2010*, volume 6496 of *Lecture Notes in Computer Science*, pages 305–320. Springer Berlin Heidelberg, 2010.
12. A. Hameed, A. Preece, and D. Sleeman. Ontology reconciliation. In S. Staab and R. Studer, editors, *Handbook on ontologies*, pages 231–250. Springer-Verlag, Germany, 2004.
13. T. Heath and C. Bizer. *Linked Data: Evolving the Web into a Global Data Space (1st edition)*. Morgan & Claypool, Palo Alto, California, 2011.
14. P. Hitzler and F. van Harmelen. A reasonable semantic web. *Semantic Web Journal*, 1(1-2):39–44, 2010.
15. E. Hyvönen. *Publishing and using cultural heritage linked data on the semantic web*. Morgan & Claypool, Palo Alto, California, October 2012.
16. E. Hyvönen, K. Viljanen, J. Tuominen, and K. Seppälä. Building a National Semantic Web Ontology and Ontology Service Infrastructure—The FinnONTO Approach. In *Proceedings of the European Semantic Web Conference (ESWC 2008)*, pages 95–109, 2008.
17. K. Järvelin, J. Kekäläinen, and T. Niemi. ExpansionTool: Concept-based query expansion and construction. *Information Retrieval*, 4(3/4):231–255, 2001.
18. E. Jiménez Ruiz, B. Cuenca Grau, I. Horrocks, and R. Berlanga. Supporting concurrent ontology development: Framework, algorithms and tool. *Data & Knowledge Engineering*, 70(1):146–164, 2011.
19. J. Kilkki, O. Hupaniittu, and P. Henttonen. Towards the new era of archival description – the Finnish approach. In *Proceedings of the International Council on Archives Congress*, August 2012.
20. M. Klein. *Change Management for Distributed Ontologies*. PhD thesis, Vrije Universiteit Amsterdam, 2004.
21. D. Kless, S. Milton, and E. Kazmierczak. Relationships and relata in ontologies and thesauri: Differences and similarities. *Applied Ontology*, 7(4):401–428, 2012.
22. K. Kozaki, E. Sunagawa, Y. Kitamura, and R. Mizoguchi. A framework for cooperative ontology construction based on dependency management of modules. In *Proceedings of the International Workshop on Emergent Semantics and Ontology Evolution (ISWC2007)*, pages 33–44, 2007.
23. A. Maedche, B. Motik, and L. Stojanovic. Managing multiple and distributed ontologies on the Semantic Web. *The International Journal on Very Large Data Bases (The VLDB Journal)*, 12(4):286–302, 2003.
24. E. Mäkelä, K. Hypén, and E. Hyvönen. BookSampo—lessons learned in creating a semantic portal for fiction literature. In *Proceedings of ISWC-2011, Bonn, Germany*. Springer–Verlag, 2011.
25. E. Mäkelä, T. Ruotsalo, and E. Hyvönen. How to deal with massively heterogeneous cultural heritage data—lessons learned in CultureSampo. 3(1), 2012.
26. V. Mascardi, V. Cordì, and P. Rosso. A comparison of upper ontologies. In *Proceedings of the 8th Workshop Dagli Oggetti*

agli Agenti: "Agenti e Industria: Applicazioni tecnologiche degli agenti software" (WOA 2007), pages 55–64, September 2007.
27. I. Niles and A. Pease. Towards a standard upper ontology. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS 2001)*, pages 2–9. ACM, October 2001.
28. Natalya F. Noy, Nicholas Griffith, and Mark A. Musen. Collecting community-based mappings in an ontology repository. In *Proceedings of the 7th International Semantic Web Conference (ISWC 2008)*, pages 371–386. Springer–Verlag, October 2008.
29. S. Pessala, K. Seppälä, O. Suominen, M. Frosterus, J. Tuominen, and E. Hyvönen. MUTU: An analysis tool for maintaining a system of hierarchically linked ontologies. In *Proceedings of the Workshop on Ontologies come of Age Workshop (ISWC 2011)*, 2011.
30. B. Smith, M. Ashburner, C. Rosse, J. Bard, W. Bug, W. Ceusters, L. J. Goldberg, K. Eilbeck, A. Ireland, C. J. Mungall, The OBI Consortium, N. Leontis, P. Rocca-Serra, A. Ruttenberg, S.-A. Sansone, R. H. Scheuermann, N. Shah, P. L. Whetzel, and S. Lewis. The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. *Nature Biotechnology*, 25(11):1251–1255, 2007.
31. S. Staab and R. Studer, editors. *Handbook on Ontologies (2nd Edition)*. Springer–Verlag, 2009.
32. L. Stojanovic, A. Maedche, B. Motik, and N. Stojanovic. User-driven ontology evolution management. In *Proceedings of the International Conference on Knowledge Engineering and Knowledge Management (EKAW 2002)*, pages 133–140, 2002.
33. E. Sunagawa, K. Kozaki, Y. Kitamura, and R. Mizoguchi. An environment for distributed ontology development based on dependency management. In *Proceedings of the 2nd International Semantic Web Conference (ISWC 2003)*, pages 453–468. Springer–Verlag, 2003.
34. O. Suominen and E. Hyvönen. Improving the quality of SKOS vocabularies with Skosify. In *Proceedings of the 18th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2012)*, pages 383–397. Springer-Verlag, October 2012.
35. O. Suominen, S. Pessala, J. Tuominen, M. Lappalainen, S. Nykyri, H. Ylikotila, M. Frosterus, and E. Hyvönen. Deploying national ontology services: From ONKI to Finto. In *Proceedings of the Industry Track at the International Semantic Web Conference 2014*. CEUR Workshop Proceedings, `http://CEUR-WS.org`, October 2014.
36. J. Tuominen, M. Frosterus, K. Viljanen, and E. Hyvönen. ONKI SKOS server for publishing and utilizing SKOS vocabularies and ontologies as services. In *Proceedings of the 6th European Semantic Web Conference (ESWC 2009)*, pages 768–780. Springer–Verlag, 2009.
37. K. Viljanen, J. Tuominen, and E. Hyvönen. Ontology libraries for production use: The Finnish ontology library service ONKI. In *Proceedings of the ESWC 2009, Heraklion, Greece*, pages 781–795. Springer–Verlag, 2009.
38. X. Wang, J. S. Almeida, and A. L. Oliveira. Ontology design principles and normalization techniques in the web. In *Proceedings of the 5th International Workshop on Data Integration in the Life Sciences (DILS 2008)*, pages 28–43. Springer–Verlag, June 2008.