Language-Agnostic Knowledge Representation for a Truly Multilingual Semantic Web

Sarika Jain, National Institute of Technology, Kurukshetra, India*
https://orcid.org/0000-0002-7432-8506
Anastasiia Kysliak, Georg-August-Universität, Göttingen, Germany

ABSTRACT

As the internet user communities and the information on the web are increasing exponentially, meaningful search in a multilingual setting is required. It is crucial to develop a simple to use, granular, and adaptable knowledge representation system that could constitute the core of the Semantic Web. It should be well-prepared for multilingual and multicultural settings by virtue of internationalizing the code. In this work, a model for ontology-based language-agnostic Semantic Web application architecture is offered which is independent of today’s heterogeneous encoding of language resources and provides smooth exploitation. The suggested system architecture enables a quick and meaningful search for information independent of the end user’s preferred language and cultural conventions, an easy extension of the application with new information, as well as its easy localization for new language contexts. The model’s usability is proven by a prototype application simulation in the domain of Indian biodiversity.

KEYWORDS

Globalization, Knowledge Representation, Multilingualism, Ontology, Semantic Web

1 INTRODUCTION

Semantic Web is a modern technology aimed to deal with up-to-date challenges of the web caused by data explosion and a constant growth of the number of internet users. It was proposed as an extension of a classical web which can supply data with unambiguous meanings and, thus, make the meaning of the provided information machine-readable. Semantic Web technologies aim to enable a better and easier data exchange between otherwise heterogeneous systems and applications as well as facilitate the interaction between the users and computers.

Nowadays, new challenges for the Semantic Web technologies are posted by the process of globalization and the necessity to deal with multilingualism in the modern world. Although the World Wide Web has opened a lot of possibilities for the internet users all over the world to share and to access data, there is still a lot of space for the improvement of web applications so that they can deal properly with the linguistic and cultural variety of the users’ communities.

Although the major part (54%) of information present on the web is in English the number of non-English speakers (74.80%) using the internet is considerably increasing. Figure 1 shows the
contrast of content languages for websites and internet users by language for ten major languages. This advocates an increasing demand of making the information on the web accessible for non-English-speaking audience. Moreover, the globalization of businesses makes it necessary to facilitate information exchange between linguistic and cultural communities as well. Due to the entire nature of the Semantic Web which is based on a “Bag-Of-Concepts model” instead of a “Bag-Of-Words model” applied in the classical web technology (Sorg and Cimiano 2012), it has a great potential for dealing with such a challenge.

The importance of application’s globalization and adaptation for different linguistic and cultural settings has been recognized by many developers. E.g., Oracle whose Database applications support semantic technologies and provides a globalization support for the users. It includes an automatic adaptation of error messages, date, time, monetary and other conventions to the user’s native language and locale (Jain et al 2011, Jain and Jain 2013, 2014).

However, the Semantic Web technology still lacks a satisfactory solution for the question of representation, storage, interlinking and processing of data provided in different languages (Patel and Jain 2018, Mishra et al 2015). Although different approaches have been suggested in the recent years to incorporate multilingualism in the Semantic Web, there is still no universal solution and a lot of space for improvement remains.

The goal of this paper is to participate in the development of the Semantic Web technologies and to propose a model of a knowledge representation system adjusted to the needs of globalized Semantic Web applications. In the proposed model, the globalization of a target application implies its internationalization and further localization for any new cultural and language context.

The remainder of this work is structured as follows: Section 2 gives an overview of scientific approaches suggested in literature to handle multilingualism in Semantic Web. Section 3 establishes the scope of this work; it specifies the goals and objectives, highlights benefits of the proposed framework, and introduces tools and methods used in course of this work for a prototype implementation. Section
4 gives an in-depth description of the suggested knowledge representation system together with an overall final application architecture; It represents a manual for the usage of the suggested system architecture, it includes information on development and further extension of a corresponding final application. Section 5 describes the prototype application implemented in the domain of biodiversity in course of this work. Section 6 poses conclusions and formulates suggestions for further development and improvement of the proposed model.

2 A REVIEW OF EXISTING APPROACHES

There exists a gap between the user’s information needs and the semantic information available on the web. Researchers around the globe are continuously working for exploiting the potential of semantic web for making it multilingual. (Gracia, et al. 2012) discuss the necessity and challenges of multilingualism in Semantic Web technologies, and propose an initial architecture and a roadmap for implementing their vision. The challenges include, generic representation of lexical information on the Web, Ontology localization, cross-lingual question answering, cross-lingual ontology and data matching, increasing diversity of datasets and their interoperability, issues related to minority and historical languages, whether the two entities are cross-linguistic equivalents contextually or generally etc. To achieve the vision of truly multilingual semantic web, majorly three approaches can be found in literature. One is to use the existing ontologies, localize them in required languages, and then perform necessary mappings and reconciliation; the second approach utilizes the language tagging facility of RDF and the labels of ontologies. The third alternative is linking to dictionaries and lexica. It involves the internationalization of the ontology and the service from scratch and then localizing the effort in as many languages as needed. The fourth approach involves storing a set of language-independent abstract trees transferrable in Controlled Natural Language (CNL) in the ontology. These approaches are discussed here one by one.

2.1 Language Specific Ontologies with Multilingual Mappings

The first approach to multilingualism is maybe the area that attracts the most researchers’ attention nowadays as there are a lot of existing ontologies in various languages and the need to make them interoperable permanently grows. It involves two steps (i) localizing the ontologies in one natural language into another; (ii) formally representing the translation links if the two ontologies do not use the same URI. (Cimiano et al. 2010) have identified many important issues and dimensions related to cross-lingual ontology. The matching helps to identify the overlap between ontologies in different languages. (Shvaiko and Euzenat 2011) presents an overview and evaluation of existing ontology matching systems and points out challenges for the matching field. They deal mainly with monolingual settings. While (Euzenat et al. 2011) reveals that mapping technology has been successfully utilized to automatically align ontologies in the same language, (Fu, Brennan and O’Sullivan 2010) shows one of first attempts of an ontology alignment in a multilingual setting. Several approaches have been proposed to extract multilingual mappings. In the manual approach, human expert extracts mappings as in (Liang and Sini 2006) with a limitation to process large and complex ontologies. In the automated approach, various techniques including machine learning have been utilized. (Trojahn, et al. 2014) provides an outline of mapping methods which have been suggested particularly for multilingual and cross-lingual ontology matching. Among them are:

- Manual processing (as presented by (Liang and Sini 2006)),
- Corpus-based approach (Ngai et al 2002), (Eger and Sejane 2010),
- Linguistic enrichment (Pazienza and Stellato 2005),
- Instance-based approach (Wang, et al. 2009),
- Translation-based approach (Trojahn et al 2008), (Aguirre, et al. 2012), (Fu et al 2012),
- Machine learning-based approach (Spohr et al 2011),
• Indirect alignment composition (Jung et al 2009),
• Image similarity-based approach (Mihic and Ivetic 2012).

Gracia et al. state that there are 3 levels at which relations between ontologies in different languages can be established: 1) conceptual level, 2) instance level, 3) linguistic level (Gracia, et al. 2012). (Embley, et al. 2014) have pointed out three types of matching: structural mappings, data-instance mappings, and the commentary mappings.

Mapping enables independent conceptualizations in each language and that is why it may better capture cultural specificities. For this reason, it is well suited for culturally-influenced domains. This approach is very challenging because it is not trivial to establish cross-lingual mapping or alignments among conceptualizations. Furthermore, linguistic information in this case is again limited to labels and definitions associated with ontology classes (RDF(S) properties).

Although ontology matching is a powerful technique with a lot of advantages, it has its demerits as well. On one hand, it does not imply that exactly the same information is stored in different languages. On the other hand, it may be not that easy to add a new language or extend information stored in the ontology. In order to add a new language, one needs to find or create a new domain-specific ontology in the target language that could be matched with already existing ones. Adding and updating information can be effort-consuming as it would require changes in all involved ontologies. Creating a reasonable mapping could be another challenging task. It would be rather beneficial for space efficiency as well as ease of further learning (ontology content extension, adding new languages) to avoid storing the same ontology structure for every new language.

2.2 Language Specific Annotation

This approach is the most widely spread one because it is well supported by the most popular ontology development languages: RDF(S) and OWL. This mainly relies on two RDF(S) annotation properties, rdfs:label and rdfs:comment that permit to associate word forms and descriptions to ontology elements. The language of labels and definitions can be specified using the “language tagging” facility of RDF literals (e.g., xml:lang="es") (Montiel P. 2011). These properties can be complemented by Dublin Core metadata with elements like title, creator, subject, source or description.

This approach has its advantages. Labels can be created in as many languages as needed providing localization at the terminological level. Adding extra information is possible as well, e.g. with the help of Dublin Core attributes. It is most suitable for internationalized/standardized domains as there is no categorization mismatch in this case and therefore no need for its documentation in the ontology. No modifications of the conceptualization layer are required. Wikidata is a new multilingual ‘Wikipedia for data’ to manage the factual information of the popular online encyclopedia. Wikidata is an example of a Semantic Web application which makes use of the labels approach as it provides labels and descriptions for many terms in different languages.

The multilingual labels approach is not suitable for the real world projects as they demand for more lexical information like syntax, semantics, grammatical features, part-of-speech, phonology, morphology, discourse etc. than just a simple label.

2.3 Associating Ontology with an External Linguistic Model

Formalizing the notion of information began in late 1940s. Utilizing this idea in the semantic web, we will be able to abstractly represent information in a language-agnostic manner. This well-defined structure enables us to work with variables and formulas instead of language-specific string texts. Examples of such approaches include the EU project MOLTO (Multilingual Online Translation). MOLTO is a multilingual semantic wiki based on ACE-GF. The content languages of the wikis are defined by multilingual GF grammars. Each wiki article is stored in a language-neutral format but is viewable and editable via any of the languages supported by the grammar (Kaljurand, et al. 2013).
In this Wiki, every user can edit a document in his own language, and the changes are immediately propagated to all other languages.

The third approach to multilingual web of data deals with standardizing the ontology development process by formalizing the content. It provides a central repository for localization of vocabularies and data, i.e., a Google or Facebook of multilingual data. It allows a separation of conceptual and terminological layers and aims at defining models representing the linguistic description of terminological resources on the web (thesaurus, ontologies, etc.). This approach is the most flexible one: it can account for both kinds of domains (standardized and culturally-influenced); and can serve both functions, documentary and independent, as specified by (Montiel P. 2011).

The linguistic/lexical resources comprises of the multilingual dictionaries, multilingual lexicons, annotated corpora, terminologies and thesauri, wordnet. Wikidata provides labels and descriptions for many terms in different languages; hence can be used as dictionaries to present information to international audiences (Samuel, 2018). Various models exist for associating this lexical information with ontological elements, including the Linguistic Information Repository (LIR) (Montiel P. et al. 2011); LexOnto; LingInfo, the Lexicon Model for Ontologies; SKOS (Simple Knowledge Organization System); LabelTranslator, an ontology localization tool that can be used as an alternative to a manual translation.

Globalization of Semantic Web applications, with multilingualism being an integral part of it, is an important task which has been in scope of scientific research for about a decade already (Buitelaar and Cimiano, 2014), (Buitelaar et al. 2013), (Espinoza et al 2008)). (Auer, et al. 2010) describe the role and potential of URIs and IRIs, review different RDF serializations (like JSON, N-Triples, Turtle, Notation 3, RDFa) as well as Semantic Web APIs (OWL API) and tools (OntoWiki, Protégé, Jena, Virtuoso, Sesame) with regard to their support for internationalization of knowledge bases. Internationalization, often abbreviated as “i18n” (18 represents the number of letters between the first and last), refers to the adaptation of a product or service based in a certain language or locale for another language or locale.

Localization is understood as an adjustment of an application for specific cultural and/or linguistic conditions, internationalization can be described as a prior preparation of the target application for being localized, i.e. designing it in a way that enables further localizations to be rather uncomplicated. The localization includes both translation and cultural adaptation of the content.

Globalization = Internationalization + n* Localization
Localization = Translation + Culturalization

This approach carries the benefits of allowing the ontologies in existing semantic formats such as OWL and RDFS to be linked to rich lexical descriptions. Also, the multilingual repository of data will be outsourced to the language people freeing the technical specialists from dealing with the linguistic details. This approach is rather complicated for those cases when no or not much linguistic information is required.

### 2.4 Using a Controlled Natural Language

A controlled natural language (CNL) is a rich subset (restricted lexicon and grammar and a domain-specific vocabulary) of standard natural language designed to serve as knowledge representation language. In addition to being unambiguous, CNLs preserve the coherence and expressiveness of the base language (Gao, 2019). CNLs fill the conceptual space between natural languages and formal languages such as propositional logic.

The approach as is described in (Kaljurand et al 2015) is to store in an ontology not a language-specific annotation, but a set of abstract trees which present information in a language-independent form and can be transferred into a Controlled Natural Language like e.g. Attempto Controlled English (ACE). The scheme proposed in (Kaljurand et al 2015) has ACE as a core. Abstract trees stored in the ontology can be mapped one-to-one to sentence strings in ACE or (not necessarily one-to-one) with other controlled languages supported by the application. This is a promising approach which
allows to store data in the ontology in a truly language-independent way and to localize the extracted information for different languages. No ontology code duplication is required.

Although this approach is a very promising method, its main disadvantage is a restriction due to a limited amount of CNLs developed for different languages. It makes enriching of a Semantic Web application with new languages strongly dependent on the existence of corresponding CNL and could require creation of a new CNL for the target language which is by no means a trivial task.

3 A RESEARCH MODEL FOR A MULTILINGUAL SEMANTIC WEB

This work aims to participate in the upbringing of the Semantic Web by bringing information closer to an average user independent from his/her language background. The goal is to develop a language-agnostic knowledge representation system for a multilingual Semantic Web which once internationalized would allow people without special programming knowledge to enrich and enlarge the stored data as well as let geographically and linguistically distributed users easily find required information in a convenient for them language. An approach taken in this paper enables an easy adaptation of the final application for the global audience that vary in languages, regions, and cultures.

3.1 Establishing the Scope

It is really good to have such models as discussed in section 2 to represent all linguistic details of lexica, but we also need to think about their usage. Rich models such as Lemon are complex to implement. Indeed, details as parts of speech, morphology, etc. need linguistic experts to determine them and formalize them. This task is very hard, especially for large and complex ontologies like SNOMED-CT. Consequently, there is a need of specific tools to support these models use in order to convince stakeholders in the web of data to adopt them.

Even though a great work has been done in the area of ontology internationalization, there is no universal solution for the multilingualism challenge and more work has to be done. Each of the methods proposed has its pros and contras; or could be improved with new suggestions. None of them fully suits to the purposes of the application discussed in this paper. In summary, not a wide proliferation of models exists for representing lexica. The ones which exist do not represent all the features, an application aim to translate. There is lack of granularity of the knowledge available in existing models. Language resources (dictionaries, terminologies, corpora, etc.) are often encoded in heterogeneous formats and developed in isolation with one another making their discovery and use a daunting task. When attempting to model data from lesser-known languages spoken by minorities, we find that the list of language codes by ISO 639 are simply not available.

3.2 Conceptual Framework

The model proposed in this work implies a clear separation between the ontology structure and the linguistic information. Separate resource bundles are used to describe the linguistic characteristics and cross-lingual specifications. Without such separation, an ontology code can become bulky, hardly readable, and grow enormously with a big amount of new languages added. This method is good for every standardized domain as the one chosen for this work. The ontology itself includes concepts relevant for the selected knowledge domain (these concepts are represented as classes, individuals, properties in the ontology code) and defines relations between them. Any information related to the mentioned concepts which has to be represented to the end user in a language-specific way is stored separately and can be extracted from the corresponding language-specific files with a help of keys – the names of ontology entries which require language-specific representations. The main objectives of this work are:
1. To propose a multilingual ontology-based knowledge representation appropriate for a globalized Semantic Web application.
2. To design an overall architecture for a multilingual Semantic Web application.
3. To implement a prototype web application in the domain of Indian Biodiversity.

The overall architecture of the prototype system is depicted in Figure 2. It comprises of three fundamental components which can be implemented in parallel: 1) the knowledge representation component (KRC), 2) the application logic component (ALC), and 3) the graphical user interface (GUI) component.

**The knowledge representation component** (KRC) consists of the ontology and a Language Specific Data Representation (LSDR) module. Language Specific Data Representation (LSDR) – is a representation of textual data in a language specific manner, i.e. information verbalization in a particular language (e.g. L₁, L₂, etc.). L₁ denotes “base language”, L₂ etc. denote further languages used in the application. File₁, File₂, etc. stand for extra documents where LSDRs in languages LB, L₁, etc. respectively are stored. Figure 3 depicts the correlation between the ontology entries (both URIs and literals) whose names are used as keys in the LSDR files and the language-specific values assigned to these keys in the LSDR files.

**The application logic component** (ALC) provides functionality required to fulfill the various scenarios and modules of any application, e.g. querying the knowledge store, extracting information, structuring the query results for the further processing. It is connected to the KRC and incorporates the relevant functions. The querying is performed on the basis of ontology entries. The information extracted is represented in the form of the ontology entries as well. It is language-independent as there are no language-specific data in the ontology. Thus, the **application logic component** has no connection to the LSDR module related to the ontology and does not have to be changed when a new language is added.
The GUI component manages the communication with the end user. Locale switch, scenario or module choice, input and output for different use cases as well as some additional information (e.g. about the application, user manuals, etc.) are provided at this level. This component is responsible for displaying all the data including input options and the final results of the search in the language- and culture-specific manner, e.g. in the locale specified by the end user. This component is connected to the LSDR module of the underlying ontology but not to the ontology code itself. Communication with the ontology takes place through the application logic component. For each module where an input is required, the GUI gets a list of all ontology entries which could be a relevant input in this case. LSDRs for each of those entries are gathered from the LSDR module of the KRC, from the file responsible for the currently chosen language. These language-specific data are then displayed to the end user as a list of input options. When the user choose an input option from the list, responsible function(s) in the application logic component is called and the ontology entries related to the chosen option (not their language-specific representations) are forwarded to the functions as input if such is required. The application logic component returns results represented in form of ontology entries. The GUI collects once again corresponding language-specific data from the LSDR module of the KRC and displays the results in a user-friendly manner, i.e. with respect to the user’s preferred language, properly formatted and structured.

The GUI component itself also requires a LSDR module, where all language-specific texts related to the GUI and not to the ontology have to be stored. In this work, the LSDR collections of files for the ontology and for the GUI are stored and approached separately. Moreover, only the set of LSDR files for the ontology will be further referred to as an LSDR module. It is due to its importance as an integral part of the knowledge representation system discussed in this paper.

The three components are not completely independent from each other. The implementation of the use cases is strongly dependent upon the ontology structure (however, not on the related LSDR module). For any use case, both the corresponding GUI and the functions in the application logic component have to be developed; the communication through specific interfaces between the involved components has to be established. This means that a thorough planning of the application structure in advance is required. Whereas the LSDR modules of the KRC and the GUI components have to be localized for every new language, the application logic component remains the same after being implemented and does not have to be modified during localization for a new language. Changes in this
component are required only when the ontology structure has been reorganized, more functionality
needs to be added or the performance of the component has to be improved; and not with the demand
of new languages.

3.3 Features and Benefits
The proposed model of multilingual semantic web conforms to the following features as inherent in
the knowledge representation component (KRC):

1. Separation of the ontology code and the linguistic information. This enables a strict separation
   of roles:
   a. A programmer needs to deal with the ontology structure.
   b. A language specialist (e.g. a translator) will localize the data for a new language.
2. Single ontology code + n*data represented in n different languages.

   The above mentioned features of the KRC under consideration provide the following benefits:

3.3.1 Reduction of the Programming Efforts: Because of the strict separation of roles, once the
   internationalized system is implemented for the base language L_B, no major programming efforts
   are required for its further localization in other languages. It will be a language specialist’s task
to prepare data for a new language.

3.3.2 Ease of Learning: The model is well-prepared for extension of ontology, functionality and languages.

3.3.3 Extension of ontology: Involves incorporating ontology learning algorithms and adding new
   information in the existing LSDR files. It does not affect the application logic component and the
   GUI component. In contrary to a cross-lingual mapping, there is no need to provide changes in
   many language-specific ontologies as there is only one ontology used, a language-independent one.

3.3.4 Adding new functionality causes changes in the application logic component and the GUI component
   but does not influence the KRC.

3.3.5 Simplicity of Localization: Localization for a new language is facilitated through the
   internationalization of the system and depends solely on translation option availability. In contrast
to a cross-lingual mapping strategy, no other ontology in the target language is required. Such
   ontology could be unavailable or insufficient. Translation of an already participating ontology
   followed by mapping with the translated version could be an alternative solution in this case but
would require more effort than a simple translation of the language-specific data.

3.3.6 Space efficiency of the KRS: There is no need to duplicate the ontology code (which would be
   unavoidable, e.g., in case of an ontology translation for a cross-lingual mapping).
3.3.7 Editing Efforts minimized: The language-specific data can be easily edited without any influence on the ontology code as well as query performance of the final system.

3.3.8 No redundancy: As every new LSDR is an exact translation of an already existing one, no discrepancies are expected in the data represented in different languages. Meaning that identical information is stored for all participating languages.

3.3.9 Cultural and Linguistic conventions respected: This is achieved through localization of the application for diverse cultural and linguistic settings. In particular, language-specific data as well as Meta data related to the GUI design issues can be stored in separate files and consulted during the application run. Such data can include:

- Component orientation: e.g. Arabic texts have to be displayed from right to left whereas texts in English, German or Russian require a left-to-right displaying style.
- Color conventions.
- Icons used.
- Date, time, number formats, etc.

End User Collaboration: Users involvement and collaboration is now not a daunting task: The model has a good potential for users involvement in the process of data enrichment and improvement. For this purpose, the corresponding data can be published as an open source with an offer to provide a translation into new languages or suggest an improved translation for the existing ones. Thus, one of important up-to-date challenges of the web technology – users’ collaboration – can be addressed.

Flexibility of the model deployment: The model can be used for different domains of knowledge and for different kinds of applications, e.g. mobile apps (in this case, the separate storage of language-specific data is useful as it allows a selective download of only those files containing language-specific data which are explicitly required by the user).

Reusability: LSDRs representing the ontology entries expressed in different languages can be reused for other purposes like other ontologies and applications.

4. OPERATIONAL ANALYSIS

Development of globalized semantic web application described in section 3 starts with designing a working prototype implemented for the base language lb. This step is followed by localizing the application for languages L1, L2,..., LN. This process is described in detail step by step in the following sub-sections.

4.1. Internationalization

Development of an internationalized running system adjusted for Lb encompasses three major tasks: 1) creation of an internationalized ontology-based KRC, 2) application logic component implementation, and 3) creation and internationalization of the GUI localized for Lb.

4.1.1 Internationalized Ontology-based KRC

The following steps have to be performed in order to create an internationalized ontology-based KRC:
1. Create/Import/Adjust an Ontology in $L_B$

An ontology can be obtained by 1) creating a new ontology from databases, term lists, taxonomies, dictionaries etc.; 2) reusing existing ontologies as they are; 3) reengineering existing ontologies, e.g. localizing and adjusting them.

2. Identify and Mark all the Language-specific Data in the Ontology

All ontology entries representing its language-specific content have to be identified, marked and replaced with variable entries. This includes all the data (no matter if it is domain specific data stored in the ontology or its Meta data) which are expected to be displayed to the end user and require a language- and/or culture-specific way of representation (e.g. text literals, time, numbers, internet links etc.).

3. Extract and Store the Language-specific Data from the Ontology Separately in the LSDR Module (FileB)

In this step, the information originally stored in the ontology and represented in a base language $L_B$ (or any other manner if it is numbers, links, etc.) is extracted from the ontology code. The extraction can be conducted manually, semi- or fully automatically depending upon the structure and size of the ontology in question. The extracted data has to be replaced with a new entry (URI, String literal etc.) which can serve as a unique key connecting the ontology code with the extracted data stored in a separate file. All extracted values have to be aligned to their keys within the first LSDR file, FileB. Through these keys, the corresponding information in a chosen language stored in the FileB of the LSDR module will be accessible. Such ontology entries as the names of classes, properties etc. can get language-specific representations as well. In this case, language-specific representations have to be created for these entities and aligned to their names used as keys within the FileB.

This approach allows to store separately not only labels in a narrow sense but any kind of information, including e.g. comments or internet links which can be adjusted to forward users to web sites or their versions in a specified language. Numbers, currency, time and other conventions can be represented in a language- and culture-specific way either.

4.1.2 Application Logic Component

1. Design the Use Cases

Individual use cases have to be defined with respect to the required functionality of the final system and the underlying ontology structure.

2. Implement the Functionality in a Chosen Programming Language.

The application logic component deals only with the ontology itself without interaction with the LSDR module. For each use case, corresponding functions have to be programmed. The functions conduct the search in the ontology and organize the received results in a required format. The predefined interfaces for communication with the GUI pose the format requirements. All the information extracted and forwarded to the GUI is based on the names of the ontology entries. These names are language-independent. No language-specific information is involved in the processing and fulfillment of the use cases.
3. Interfacing ALC to the KRC and the GUI

In this step, the communication between the application logic component and the ontology as well as the GUI has to be established. The communication with the GUI is regulated by predefined interfaces.

Not all the steps described in this section are performed sequentially, some of them (e.g. the step 2. and the step 3.) may intersect.

4.1.3 Internationalized GUI localized for \( L_B \)

The following steps are required in order to implement an internationalized GUI and localize it for the base language \( L_B \):

1. Design a GUI for the Planned Use Cases with Respect to \( L_B \)

   The design of the GUI with respect to \( L_B \) and its further communication with the application logic component has to be modeled. The resulting GUI has to fulfill the following requirements:
   
   - be user friendly;
   - be internationalized, i.e. prepared for adding new languages and switching between the languages on the flow;
   - correlate with the cultural and linguistic conventions related to \( L_B \) (e.g. text alignment depending upon the direction of the writing in \( L_B \)).

2. Internationalize the GUI Analogue to the Internationalized KRC

   Internationalizing the GUI is similar to the internationalization of the KRC described earlier. The language-specific (and/or culture-specific) data needed for the GUI component are stored separately in a corresponding LSDR file (FileB in the LSDR module of the GUI component) and are connected to the GUI through specific keys. However, the actual format of data storage for GUI may differ from the one used for the ontology. Not only textual data, but also metadata related to the GUI design can be localized. The values for the font size, colors, direction of text alignment, etc. can be stored separately as well. These data can be adjusted to new conditions in the course of future localizations for other languages and cultures.

3. Connect the GUI with the ALC

   Adjust the communication between the GUI and the application logic component through predefined interfaces.

4.2 Localization

After the application has been internationalized, it has to be localized for new linguistic and cultural contexts different from \( L_B \). For each new language \( L_i \), 1 \( \leq i \leq n \), n is the number of locales for which the application has to be localized, perform the localization procedure which encompasses two major tasks: 1) localization of the KRC and 2) localization of the GUI.

4.2.1 Localization of the KRC for \( L_i \)

In order to localize the knowledge representation component for \( L_i \), the following steps have to be undertaken:
1. Translate the LSDR from L_b or any other L_j (j < i) into L_i

The information extracted from the ontology and stored in the LSDR module has to be translated into the new language L_i. For a high-quality translation, it should be a manual task performed by an expert. However, machine translation and other strategies for semi or fully automatic translation can be applied as well. A translation into a new language L_i can be performed from any of the already incorporated languages L_n, L_1, …, L_j; j < i.

2. Store the translated data into File_i

The translated data has to be stored separately from the ontology but still connected to the corresponding ontology entries just in the same manner as it was done for L_n. Thus, another LSDR document, File_i, is added to the LSDR module.

**4.2.2 Localization of the GUI for L_i**

The following steps are required to localize the GUI for a new language L_i:

1. Translate the relevant GUI data into L_i.

The translation of GUI data can be performed in the same manner as it is done for the ontology data. Some changes in the GUI code can be required in order to adjust the GUI-LSDR communication and add a new language option for the end user.

2. If Needed, Redesign the GUI for a Culture-specific Context of L_i

A simple GUI localization strategy implies changing texts prepared for the communication with the end user from one language to another as long as the language preference of the user changes. This strategy can be extended to a more sophisticated one: the graphical design (colors, sizes of components, their locations etc.) may vary depending upon the cultural context and other circumstances related to the target audience.

A crucial benefit of the proposed model is that no changes in the ALC are needed for the application localizations. The ALC works with the ontology whose structure is not going to change as a result of the localization. Its communication with the GUI remains unchanged as well. Only adding new functionality to the application (e.g. new use cases) can require changes (e.g. new functions) in the ALC.

**4.3 A Roadmap for Learning**

The suggested model is designed in a way that allows an easy extension of the information stored in the KRC as well as of the application’s functionality. This enables and motivates a permanent improvement of the application with respect to the comprehensiveness of the stored data. As long as the overall ontology structure remains the same and solely new entities or relations are added to the ontology, these new entries have to be added to the LSDR module and localized for all available languages. The following steps have to be undertaken:

- Extend the ontology with new information; ensure that only language-independent keys are stored in the ontology as new entries.
- Add the language-specific data corresponding to the newly created entities to the corresponding language file in the LSDR.
It is important to mention that the new information does not have to be provided in the base language $L_B$. It can be represented in any of the other available languages.

The complete system can be extended with new use cases as well. Only the ALC and the GUI are going to be affected by such extension. It can be achieved with the following steps:

- Implement the corresponding functionality in the ALC: Add new functions and provide interfaces for communication with the GUI.
- Develop the GUI for the new use cases with the end user, localize it for all applicable locales.
- Establish communication between the application logic component and the GUI through the relevant interfaces.

5. CASE STUDY ON INDIA BIODIVERSITY

A simulation prototype has been developed to establish the efficacy of the proposed approach and to demonstrate its usability. This is a multilingual web portal which provides information on Indian biodiversity. It offers several use cases which allow the user to search for species stored in the underlying ontology and extract information about them in a convenient for the user language. The example application has English as its base language $L_B$ and is localized for German ($L_1$) and Russian ($L_2$). All translations have been performed manually.

5.1 Domain Under Study

Ontologies are widely used in the biological domains e.g., NCBO Bioportal with its rich collection of biomedical ontologies (Jain 2018). Indian biodiversity has been chosen as the domain of knowledge for the target application because of several reasons:

- It is a standardized domain of knowledge which has the same conceptualization level in every linguistic and cultural setting. This guarantees the same ontology structure for every new linguistic or cultural condition and allows one-to-one relations between language-specific representations of the data in different languages.
- The scientific importance of the domain is that biodiversity is an essential knowledge area not only for India but worldwide as it gives understanding of the biological resources of the Earth, supports conservation, helps to organize, manage and promote sustainable use of these resources.

5.2 Tools Overview

This section gives a short explanation of tools and techniques used in this work.

5.2.1 Creation of an example ontology

For this work, test ontology was created from scratch, with a help of Protégé 5.1.0., on the basis of such information sources as Catalogue of Life, India Biodiversity Portal and Wikipedia. The ontology is written in OWL, Turtle notation, and complies with the OWL DL. Taxonomic tree units were stored as classes and the rdfs:subClassOf property was used to specify the hierarchical relations between them. Figure 4 shows a taxonomic branch leading to the genus Omobranchus as it is stored in the ontology.
5.2.2 Developing of a prototype web application

An example web application has been implemented with NetBeans IDE 8.2. It is written in Java 8 and makes use of Apache Jena for managing the communication between the application and its underlying ontology. It uses SPARQL as a querying language to query the ontology in question. Furthermore, it applies Java resource bundles and JSTL for storing language-specific data needed for internationalization and localization of the application. For the GUI, JSP (JavaServer Pages) and CSS (Cascading Style Sheets) have been utilized.

5.3 Use Cases

The following use cases have been implemented:

5.3.1 Search by Scientific Name of the Species

This function allows the end user to find all the stored information about a species in question by selecting its scientific name from a drop-down menu. The menu represents a full list of all scientific names present in the ontology. Each entry in the list can be localized for different language and culture communities. In the domain of biodiversity, there is no need for such localization as the scientific names are international.

The figure 5 shows the results for the use case “Search by scientific name” with the scientific name “Pseudois nayaur” already selected from the drop-down menu and then an algorithm is presented.

Figure 5. Search by Scientific Name
Other use cases have been implemented similarly to the one described with the difference that they require more interaction between the GUI and the application logic component and are based on different SPARQL query strings.
5.3.2 Search by Property of the Species:

This option allows the end user to choose any property out of a list of available properties implemented as a drop-down menu. All possible values (countries) of the selected property (say distribution) are extracted from the ontology and displayed in another drop-down menu. After the user chooses a value (say India) and clicks a button “Search” (English version), all entities which have the selected property with the specified value are extracted from the ontology and displayed as a list (Figure 6a). The user can obtain the detailed information for every species in the result list by clicking on the species name. The expanded results are shown as depicted in Figure 6b. Figure 6c shows the same setting in its Russian version.

Figure 6. Search by Properties (a) Results for species with property “distribution” and value “India”, English version. (b) Detailed information of selected species in English. (c) Detailed information of selected species in Russian.

5.3.3 Search in the Taxonomic Tree

In this GUI, drop-down menus with the options related to the levels in the taxonomic hierarchy let the user define a branch in the taxonomic tree to which the sought-after species must belong. For each user’s selection in any level, only those values are displayed in the drop-down menus of the lower levels (if these exist) which are subordinated to the selected value. Figure 7a shows the taxonomic tree form filled in with a taxonomic branch relevant for the kingdom “Animalia”, Phylum “Chordata”, Class “Actinopterygii”, and Order “Perciformes” in Russian. It shows the Latin terms and their Russian analogues, separated with the sign “|”. A list of search results is displayed to the user (Figure 7b). Detailed information of any species just obtained as part of the result list can be obtained by just...
clicking on the species name. Figures 7c, 7d, and 7e display the expanded result representations in German, English, and Russian respectively.

Figure 7. Search in the Taxonomic Tree. (a) Taxonomic Tree Form filled till Order. (b) A List of Search Results (c) Detailed information of species, German version. (d) Detailed information of species, English version. (e) Detailed information of species, Russian version.

6. CONCLUSION

The paper describes lexical and semantic aspects of ontology design that need to be considered for ontology-based applications that cross cultural and/or linguistic boundaries. The web of data has the potential that enable it to be extended to truly multilingual web. In this paper, we have discussed the various approaches for representing multilingual semantic information in web, proposed an architecture for realizing the selected approach and simulated it for the domain of Indian biodiversity.

The suggested architecture enables access to information in a convenient for end users way. It ensures that input and output of the data can be provided in different languages depending upon the end users’ preferences. The proposed model of a knowledge representation system as well as the system architecture suggested in this work are feasible (as shown by the implementation) and can be used for different standardized knowledge domains (not only bio diversity).
The proposed model complies with the goals postulated in Section 3.1. The querying and reasoning are supported as long as the ontology in the core of the proposed KRS is a valid OWL DL ontology. Multilingualism is provided by the corresponding LSDR files collection. The changes caused by extending the ontology with new entries can be easily handled “locally” so that adding a new language doesn’t cause big programming efforts (mainly, it implies a translation and adding a new language-specific file to the LSDR module). Therefore, the proposed architecture is well-suited for a globalized Semantic Web application.

Future Work:
Although a lot of work has been done manually while creating the prototype application, different actions can be executed in an automated or a semi-automated manner. Among them are: retrieval of all labels, comments and other entries from an ontology, creation of the default LSDR file (for \( L_B \)) based on the extracted data and an automatically established keys list, translation of the extracted data into other languages.

At the moment, the system allows only predefined input in specified languages (user can choose out of proposed offers but cannot type in his/her own words). More work is needed to make processing of a multilingual input more flexible. Usage of rdf triples for language-specific data storage and querying through the stored literals for the given input can be a solution.

Another challenging task is the development of an analogous system for culturally-influenced domains where there can be n:m correspondences between the ontology entries in different languages.

ACKNOWLEDGMENT

This work has been supported by the DAAD Rise Worldwide program of the German Federal Ministry of Education and Research. The second author has received summer research internship from this program under the mentorship of the first author.
REFERENCES


---

Sarika Jain graduated from Jawaharlal Nehru University (India) in 2001. Her doctorate, awarded in 2011, is in the field of knowledge representation in Artificial Intelligence. She has served in the field of education for over 19 years and is currently in service at the National Institute of Technology Kurukshetra (Institute of National Importance), India. Dr. Jain has authored or co-authored over 150 publications including books. Her current research interests are knowledge management and analytics, the semantic web, ontological engineering, and intelligent systems. She has been PI of sponsored research projects and is a senior member of the IEEE. For more and most updated information, see https://sites.google.com/view/nitkkrsarikajain/.