World Academy of Science, Engineering and Technology
International Journal of Industrial and Manufacturing Engineering
Vol:8, No:12, 2014

A Collaborative Platform for Multilingual Ontology Development

Ahmed Tawfik, Fausto Giunchiglia, Vincenzo Maltese

***Abstract*—**Ontologies provide a common understanding of a specific domain of interest that can be communicated between people and used as background knowledge for automated reasoning in a wide range of applications. In this paper, we address the design of multilingual ontologies following well-defined knowledge engineering methodologies with the support of novel collaborative development approaches. In particular, we present a collaborative platform which allows ontologies to be developed incrementally in multiple languages. This is made possible via an appropriate mapping between language independent concepts and one lexicalization per language (or a lexical gap in case such lexicalization does not exist). The collaborative platform has been designed to support the development of the Universal Knowledge Core, a multilingual ontology currently in English, Italian, Chinese, Mongolian, Hindi and Bangladeshi. Its design follows a workflow-based development methodology that models resources as a set of collaborative objects and assigns customizable workflows to build and maintain each collaborative object in a community driven manner, with extensive support of modern web 2.0 social and collaborative features.

***Keywords*—**Knowledge Diversity, Knowledge Representation, Ontology Development.

I. Introduction

RECENTLY, there have been great advances in semantic-aware and context-aware applications [12]. Semantic-aware applications are mainly intended to assist with information retrieval. They are designed to return more accurate search results by trying to extract the embedded meaning of the search keywords. On the other hand, context-aware applications are smart applications capable of detecting the user's social and physical surroundings (i.e. physical location or weather forecast) and provide in-site recommendations and short answers to user's queries submitted in natural language. Both semantic-aware and context-aware applications rely on knowledge-based approaches, i.e. approaches which exploit the semantics of the information in order to deliver timely and useful information. Examples of knowledge based approaches include: automatic classifications [14], ontology matching [15]-[17], ontology mapping [11], and common sense reasoning [2], and natural language data and metadata understanding [28].

One of the main requirements of knowledge-based approaches is to consider the diversity in human knowledge as people in different parts of the world have different ways of living and thinking. Diversity appears in natural language as the same word may refer to more than one meaning (homonymy) and the same meaning might be referred to with different words (synonymy). Diversity appears in knowledge has a function of local goals, school of thought, culture. A major challenge appearing here is how to deal with diversity in order to increase the accuracy of semantic-aware and context-aware applications. In fact, this requires huge multilingual resources which must provide adequate coverage for the diversity of the world and means of transforming this big amount of linguistic data into useful domain-specific knowledge that could be shared and reused effectively. This challenge reflects two main research directions that we need to go through: (1) Defining methodologies for capturing and organizing multilingual information in a formal way; and (2) Designing and implementing usable tools for gathering diverse terminologies and cross-culture knowledge.

Our main contribution in this paper is a collaborative platform that facilitates the management of diversity across cultures, in language and knowledge, via the development of localized domain ontologies. The collaborative platform is designed to work on the content of the Universal Knowledge Core (UKC), a multi-language resource we have been developing in collaboration with several partners world-wide[1]. Its data model is in line with the work described in [12]. In fact, the UKC offers a neat separation between natural language and formal language which we believe is a fundamental feature to be able to manage diversity in knowledge.

The platform provides an interactive and user-friendly web environment that allows geographically distributed linguistic and domain experts to contribute in a collaborative manner. Their collaboration takes place following a collaborative development methodology, based on the notions of *collaborative objects* and *collaborative workflows*, which specifies the development processes, user roles, and access rights. The workflow-based development methodology proposed in this paper frames linguistic resources as a set of collaborative objects and assigns customizable workflows to build and maintain each collaborative object in a community driven manner. The platform supports both the development and validation phases which are typically considered as two distinct phases in the ontology development and maintenance life cycle, where the latter strictly follows the former. The platform is also equipped with extensive support of modern web 2.0 social and collaborative features. Such features

Ahmed Tawfik, Fausto Giunchiglia, and Vincenzo Maltese are with the Department of Information Engineering and Computer Science, Faculty of Computer Science, University of Trento, Italy (e-mail: {tawfik, fausto, maltese}@disi.unitn.it).

[1] The UKC project is coordinated by the University of Trento and sees the collaboration of several universities world-wide, each of them responsible of one language.

facilitate communication between the experts and allow them to discuss various domain-related topics, share ideas and reach to common agreements.

The rest of this paper is organized as follows. Section II provides an overview of the UKC methodology. Section III presents usage scenarios and the results of an early experiment which allowed us to collect useful requirements. Section IV presents the collaborative development methodology based on the notions of collaborative objects and collaborative workflows. Section V describes how we applied the methodology for the design of the collaborative platform for the UKC. Section VI describes its architecture and user interface. Section VII summarizes the related work with main focus on web-based ontology development tools. Section VIII concludes the paper and points out to the future work.

II. Universal Knowledge Core

WordNet [5] [19] is among the most widespread used lexical resources. However, it has several limitations. In particular, it is only in British English, and the glosses given for the terms reflect the British society and culture. For instance, the term *"primary school"* is defined as *"a school for young children; usually the first 6 or 8 grades"* which is clearly biased towards the British educational system. Thus, as it is, WordNet is not directly usable in a multilingual and multicultural environment. We address these limitations with the development of the UKC. In fact, the UKC provides a mapping between language-independent concepts connected via semantic relations (called the formal language in [12], stored in the Concept Core component of the UKC) and corresponding synsets grouping words with same meaning in various languages, or a lexical gap in case a lexicalization for the concept does not exist in a certain language (called the natural language in [12], stored in the Natural Language Core component of the UKC). We employ the DERA methodology [9] and its guiding principles [10] to ensure that glosses do not exhibit any cultural, spatial or temporal bias.

A. The Natural Language Core (NLC)

Words are the basic linguistic units of languages. Each word in a natural language may have one or more meanings, known as word senses.

The Natural Language Core (NLC) models a language in a similar way to WordNet as it groups words with same meaning into *synsets* (sets of synonyms). A synset is also characterized by having a natural language gloss and a part of Speech (POS). The POS indicates whether a word is either a noun, adjective, verb, or adverb. Differently from WordNet, and given that we deal with multiple languages, we also account for *lexical gaps*.

Relations between word senses are known as *lexical relations*. The NLC defines several types of lexical relations. For instance, antonym is a symmetric relation connecting two word senses having the same POS and opposite meaning, e.g. early antonym late. Fig. 1 gives an example of the English word "fail" which has two different senses, in turn corresponding to two synsets. The first synset is associated with two senses (fail, and go wrong) which correspond to the meaning of "to be unsuccessful". The second synset is associated with another two word senses (fail, and breakdown) which correspond to the meaning of "stop operating or functioning". In the same figure the first synset corresponds to the Italian synset (fallire, abortire) and the second synset corresponds to a lexical gap in Italian (literally one should use "andare in avaria", which is not a lexical unit).

B. Concept Core (CC)

The Concept Core (CC) codifies information about language-independent *concepts* and *semantic relations* between them. Every synset or lexical gap in the NLC is associated with exactly one language-independent concept in the CC.

Each concept is associated a unique identifier, called the concept ID, and a concept label as a descriptive word obtained from the firstly defined language-dependent synset associated with the concept. Fig. 1 gives an example of associating language-independent concepts to language-dependent synsets, as well as some semantic relations between concepts.

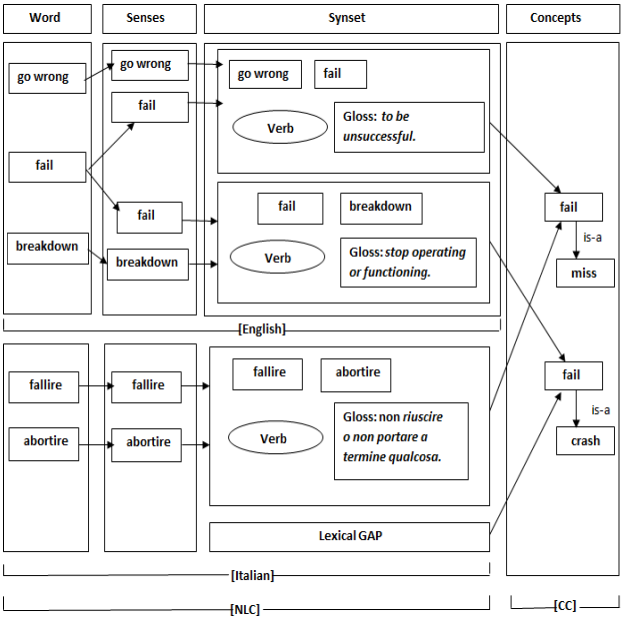

Fig. 1 The relation between the English word "fail", its word senses, synstes and concepts together with the Italian word "fallire", its word senses, synsets and concepts. Concepts are language independent but in this figure instead of providing the concept ID, we show concept labels in English

Concepts are related to other concepts through *semantic relations.* The network obtained forms the actual ontology. We define two main types of semantic relations that may exist between concepts: *hierarchical relations* and *associative relations.* Hierarchical relations are transitive and asymmetric and therefore allow the formation of hierarchies of concepts. For instance, the most common hierarchical relation is the *is-a* relation which is a specialization relation between two concepts that indicates the necessity of specialization, e.g. minivan is-a car. On the other hand, associative relations are

those relations which connect concepts in different hierarchies in the CC. For instance, the *has-member* is a relation between concepts where the source denotes a set and the target is one of its members, e.g. football player member-of football team.

III. Scenario and Early Experiment

A. Description of the Usage Scenario

The main usage scenario that we want to support is the development and maintenance of multi-language ontologies. In particular, here we focus on the UKC. For this task we follow the DERA methodology [9]. DERA aims at the development of domain ontologies as hierarchies of entity classes, relations and attributes. DERA is an evolution of the faceted approach borrowed from library science that is known to guarantee the creation of high quality, extensible and scalable ontologies. Each developed hierarchy is called a facet as it codifies a specific aspect of the domain. For instance, in the geography domain facets of classes include locations (e.g. landforms and bodies of water), facets of relations include containment (e.g. part-of) and direction (e.g. north-of, east-of) relations, attributes include latitude, longitude, depth, and length [3], [10].

The two fundamental steps of the methodology are the *analysis* and *synthesis*. During the analysis each term is analyzed in order to unambiguously determine its meaning and to come up with a suitable gloss for it. The output of the analysis is basically a set of synsets. During the synthesis concepts are generated out of synsets and facets are actually built. We want such process to be collaborative and support both:

(a) **The ontology development** [9], i.e. the process by which an ontology is built starting from a certain development language;

(b) **The ontology localization** [7] of the ontology in other languages, i.e. the process by which each concept in the ontology is associated to either a synset or a lexical gap in the target language. For instance, while the development may start in English, we may decide later on to localize the ontology in Mongolian.

B. Description of the Usage Scenario

We performed an early experiment at the purpose of collecting useful requirements for our collaborative platform. The experiment focused on ontology development.

1. Experimental Setting

With the experiment, conducted in Trento, we aimed at the development of an ontology of *flowing bodies of water* including concepts like river and fiord. Candidates were taken from the GeoWordNet ontology [13] that is an ontology generated by the integration of WordNet with GeoNames[2]. We followed a peer-review approach carried out by one developer and three different reviewers who had to decide about the acceptance or rejection of the submitted candidate terms in a way similar to the paper review process for conferences. We used EasyChair[3] to moderate the assignment and review phases. More in detail, EasyChair was used to support the analysis phase of the DERA development where each synset and corresponding gloss was provided by one developer, accompanied by a detailed explanation of the rationale behind such gloss, and commented by the reviewers who could either accept or reject it. In both cases the reviewers provided feedback and typically suggested modifications to the gloss and/or complained on the rationale.

Reviewers followed guidelines for validation indicated by the DERA methodology. Examples of matters that they needed to check include (a) adequacy of the external resources used, (b) adherence with the guiding principles (e.g. principles of ascertainability, permanence and relevance), (c) correct elimination of redundant concepts and individuals, (d) correct categorization into entity classes, relations and attributes. The synthesis phase was conducted off-line over the synsets agreed during the analysis phase. Reviewers were recommended to provide as much feedback as they could to reduce the probability of rejection after the rebuttal phase.

2. Results of the Experiment

- Figures

The inspection of GeoWordNet led to the selection of 69 candidate terms to be analyzed, given that they looked relevant for flowing bodies of water. The developer spent approximately 190 hours to generate and upload the submissions. For each candidate term, a submission was generated. A submission can correspond to a candidate synset (e.g. *river*), the proposal to ignore the term because irrelevant for the ontology to be developed (e.g. *fountain of youth*) or because it is rather an individual (e.g. *Weser river*).

The first iteration of the review process took approximately 10 hours on average to each reviewer and lead to an initial acceptance rate of 81%. The rebuttal phase took approximately 9 hours of further development time and 2 hours on average to each reviewer. Overall the two iterations lead to a final acceptance rate of 87%.

- Advantages and Limitations of EasyChair

Concerning the advantages, we found out that EasyChair nicely supports the assignment, collection and moderation of the reviews; it partially supports communication between participants via email facilities; it helps converging to commonly agreed decisions. However, EasyChair is not properly designed for ontology development and validation, but rather for paper review. We identified the following weaknesses:

- **Pull vs. Push approach:** it is based on a pull (authors submit) rather that push (developers are assigned a task) approach.
- **Static Workflow:** the workflow is static and cannot be changed. It does not support continuous refinement loops, but only up to one rebuttal phase. In case of rejection from the reviewers, a synset can be resubmitted, but it is

[2] http://www.geonames.org/

[3] http://www.easychair.org/

hard to keep track of how the submission and the reviews evolve (i.e., what has been changed by the developer with the refinement? Did the developer accommodated for the feedback received?).

- **Levels of development/validation:** it does not provide a broad view of the implications of an acceptance, i.e. the position that a certain concept would take in the ontology if accepted (w.r.t. the parent, the siblings and the children). In fact, EasyChair can be used to only support the DERA analysis, and not the synthesis (i.e., we cannot get an overview of how the facet is overall getting shape).
- **Order of development/validation:** given that deeper nodes are defined in terms of higher nodes, the order of review should be top-down, i.e. from the root to the leaves (and not in the order of submission); the tool does not give any suggestion about the order.
- **Cost of the process:** the process is too costly in terms of time. Everything was submitted and resubmitted as document attachments. This turned out to be impractical as it took significant time and it is not even possible to reconstruct the sequence of submissions as new ones override old ones.
- **Reputation:** there is an issue of appropriately engaging developers and validators. EasyChair does not support the possibility to maintain a social network of experts to be allocated on demand on the basis of their skills.

3. Next Steps

The experiments motivated us to develop a more flexible collaborative platform that overcomes the limitations described above.

IV. Collaborative Methodology

Our collaborative methodology incorporates web 2.0 features to the development process through the use of *collaborative objects* and *collaborative workflows*. We start by defining the concepts of collaborative objects and collaborative workflows, highlighting the differences between collaborative and non-collaborative objects and between collaborative workflows and standard process management workflows. Then we proceed by explaining the proposed collaborative methodology in a step by step basis and applying the methodology for UKC development and localization tasks.

A. Collaborative Object

A collaborative object is a web-based item that could be instantiated in a collaborative manner. Examples of well-known collaborative objects include web based online meetings and social events organized using shared online calendar. The main difference between a collaborative and non-collaborative object stands in the fact that the former needs to be defined based on common agreement.

B. Collaborative Workflow

In order to explain clearly the concept of a collaborative workflow, we initially start by defining the *standard process management workflow* as an automation of a work process during which the tasks are passed from one participant to another according to predefined rules and each participant is assigned a specific user role (such as the role of developer or validator). An efficient process design and implementation should result in an improved work process and elimination of any unnecessary steps. The standard workflow process is designed and implemented using workflow management software. On the other hand, we define the *collaborative workflow* as an automated process implemented using workflow management software augmented with social collaboration software (online discussions, interactive polls, or any other collaborative software tool). The collaboration software is introduced in order to facilitate communication via facilities supporting discussions and exchange of ideas among the participants. The collaborative workflow is expected to provide significant efficiency gains to the process by removing the communication barrier between participants and transforming the single-user decision making steps into common decision agreement steps.

The main requirement for supporting a collaborative workflow is to provide social collaboration facilities and a work breakdown structure of an automated process. The work breakdown structure is provided in the form of different types of process nodes and user roles that are meant to constitute the main structure and sequence of workflow process steps.

We define and use six different types of nodes. A node can be *a state, a human task, a condition, a fork, a join, a timer*, and *a notification.* Each node has a unique set of properties; we explain them briefly as follow:

1- **A state:** represents a step in the workflow process that executes immediately and requires no user intervention. Any workflow process starts with an initial state and terminates with a final state (Fig. 2).

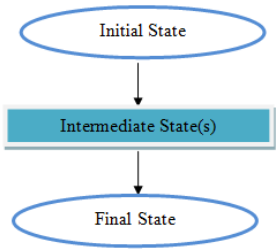

Fig. 2 Workflow States

2- **A task:** represents a step in the workflow process that requires a user input. The task is blocked until user input completes. Tasks are linked with defined user roles or user groups sharing a common role, i.e. a user who can complete a validation task must be holding the reviewer role (Fig. 3).

Fig. 3 Human task which requires user intervention

3- **A condition**: represents a decision making step and based on the condition the workflow takes a specific route (Fig. 4).

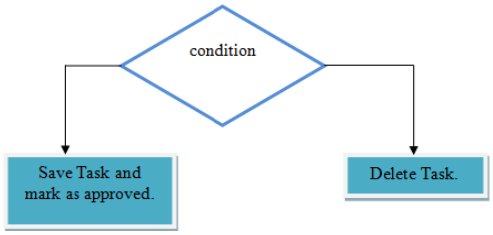

Fig. 4 Workflow Condition

4- **Fork and Join:** are used together to model parallel processing in a workflow process. The fork node splits the flow into two parallel sub-flows in order to perform parallel processing tasks then the join node merges them back and retains the original workflow only in case of successful completion of the parallel sub-flows (Fig. 5).

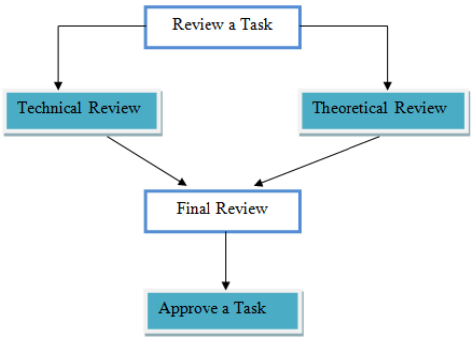

Fig. 5 Workflow Fork and Join

5- **A timer:** assigns a specific duration for tasks in order ensure that important tasks in a workflow aren't forgotten or left undone for a long period of time due to absence of users (Fig. 6).

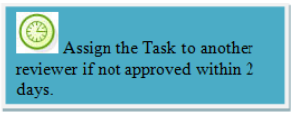

Fig. 6 Workflow Timer

6- **A notification:** is a message sent to one of the participants to communicate a piece of information (Fig. 7).

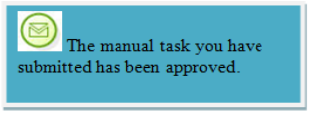

Fig. 7 Workflow Notification

C. Steps in the Methodology

The methodology we propose is not limited to UKC ontology development and localization but it is generic enough to be applied to any collaborative development. In fact, we are considering the UKC as a practical case study in order to explain and verify the methodology. The methodology is in three main phases:

1- **Definition of the Collaborative Objects:** This phase requires modeling the project as a collection of collaborative and non-collaborative objects.

2- **Definition of the Collaborative Workflows:** This phase consists in the definition of the user roles and of the collaborative process needed to manipulating the collaborative objects. We may define one or more collaborative workflows based on the structure and nature of the project under development. The complexity of the defined workflows may vary based on the nature of the project under development and the number of participants required.

3- **Mapping Objects to Collaborative Workflows:** During this phase the designer needs to specify which collaborative workflow needs to be employed for each collaborative object. However, such mapping can be partial as it is not mandatory to map all the collaborative objects to collaborative workflows. In fact, unmapped collaborative objects are directly manipulated without going through any process management procedure.

We argue that the proposed methodology is fine grained and highly customizable which is expected to provide a high level of accuracy and time saving w.r.t. the early experiment we presented in Section III.

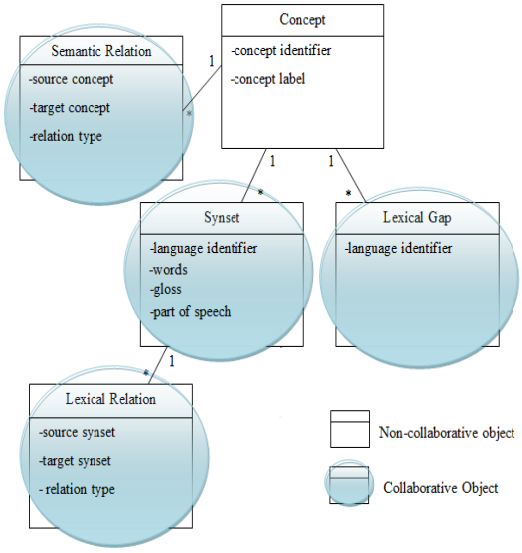

Fig. 8 Collaborative and non-collaborative Objects in UKC

V. The Collaborative Platform for the UKC

In this section, we describe the collaborative platform we developed to support the UKC development. We designed and implemented it following our collaborative methodological approach. The UKC constitutes an excellent practical use case to explain and verify the methodology.

The collaborative process takes place in two main phases: *development phase* and *validation phase*. The conditions,

constraints, transitions between the two phases are defined using collaborative workflows.

A. Definition of the Collaborative Objects

In UKC (Fig. 8), four collaborative objects were defined: *Synset, Lexical Gap, Semantic Relation, and Lexical Relation.* The *Concept* object is the only non-collaborative object as it is auto-generated together with the creation of a synset which does not have yet a correspondence with already defined concepts. In other words, the first language defining a new notion also triggers the generation of the concept.

B. Definition of the Collaborative Workflows

At the purpose of defining the user roles, we performed specific user studies by conducting interviews to knowledge experts. The user studies revealed that the following four different categories of users are necessary:

- **UKC Public Users:** those who are allowed to view the content of the UKC in read only mode.
- **UKC developers:** knowledge experts and domain developers who perform CUD maintenance operations (i.e., Create, Update, or Delete) on the UKC content.
- **UKC validators:** top level experts who can approve or reject CUD maintenance operations performed by developers.
- **UKC Administrators:** those who have full access to system features and can perform administrative tasks such as creating users, creating groups and assigning user roles to users, assigning collaborative workflows to collaborative objects and creation of interactive polls.

We then defined two types of workflows involving UKC developers and validators: (1) The *single approval workflow*; and (2) The *group approval workflow*. The single approval workflow (Fig. 9) process is instantiated when a participant holding the UKC developer role manipulates a collaborative object via CUD operations. The workflow process then assigns a validation task to a participant holding the UKC validator role. The validator decides whether to accept or reject the developer's recent manipulation. In case of rejection, the task is sent back to the same developer who needs to revise and resubmit iteratively until the task is finally accepted. The whole process is augmented with social collaboration features (comments, discussions boards, and other features).

The group approval workflow (Fig. 10) is similar to the single approval workflow except that the approval decision is taken by a group of validators based on a specific condition; for instance, at least two out of three validators (majority vote) should accept the manipulation.

C. Mapping Objects to Collaborative Workflows

In UKC, we decided to follow the group approval workflow for the ontology development tasks and the single approval workflow for localization tasks. Ontology development involves semantic relations and synsets. Localization involves synsets, lexical gaps, and lexical relations. Other objects in UKC are not mapped to any collaborative workflow.

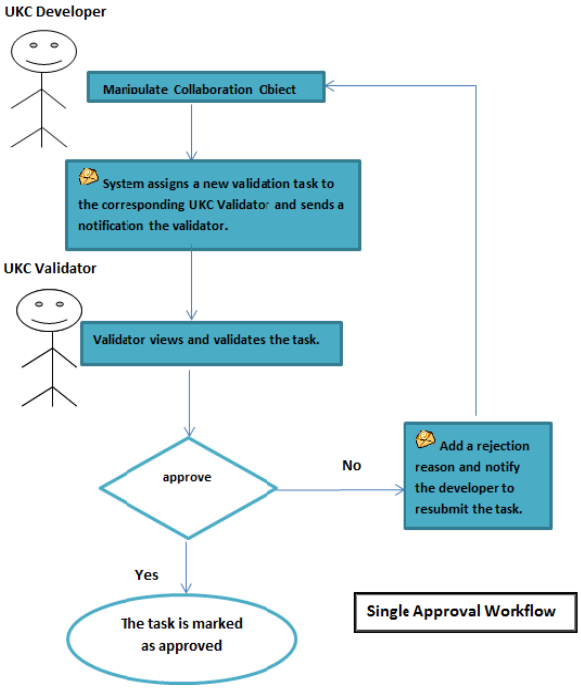

Fig. 9 Single Approval Workflow

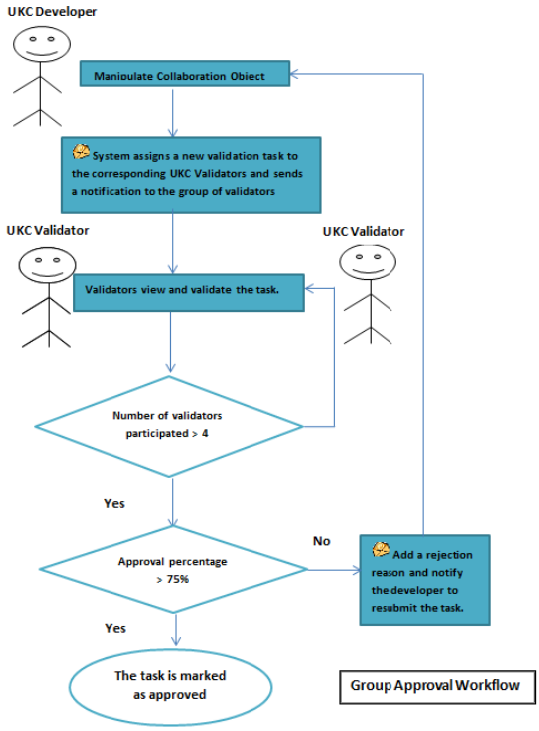

Fig. 10 Group Approval Workflow

VI. Architecture and User Interface

A. Overall Architecture

The UKC collaborative platform is designed and implemented as a *portlet application*. A *portlet* is a self-contained web application, i.e. mini web application that has its own web pages, services and data sources. A *web page* containing a group of portlets integrated together in a

consistent and systematic way is called a *portal page*. Therefore, a *web portal* could be defined as web page that brings multiple web applications or portlets together and allows for effective communication and integration between them.

Portlet applications are managed by a *portlet container*. It provides the environment for portlet management and forms the infrastructure required for running a portlet application. It allows managing portlet instances and handling communication between portlets and with data sources. There are several open-source portlet containers available nowadays. The choice of a portlet container plays an important role in portlet application projects, since it can help reduce the development time by providing built-in portlets and the ability to access container's built-in portlets features from the newly custom portlets. The choice of a suitable portlet container should be driven by the project requirement specifications.

We have conducted a comparative study of the available open-source portlet containers and decided to use *Liferay*[4]. Liferay is an open source portlet container that comes with built-in portlets for web 2.0 social and collaboration features. In addition, it has a built-in workflow engine that allows for running custom defined workflows. Liferay provides a robust platform for building social and collaborative portlets that could be extended and customized according to any project requirement specifications which perfectly fits with our methodology and requirements.

Fig. 11 shows the overall architecture for the UKC collaborative platform. It is composed of two data sources, the UKC database and an information management database (storing administration information, discussions and polls) and a Liferay portlet container ensuring smooth data excahnge between the four portlets:

- **UKC Portlet** is reponsible for the communication with the UKC database, and the management of tasks in their various statuses (assigned, pending approval, accepted, or rejected).
- **Administration Portlet** is reponsible for the administration services such as user management, collaborative objects definition, workflows definition and assigning collaborative objects to collaborative workflows.
- **Discussions Portlet** is reponsible for handling the discussion boards, creating new discussion threads and management of ongoing discussions.
- **Polls Portlet** is reponsible for handling polls, creating polls, displaying polls and counting poll results.

B. User Interface Design

Fig. 12 shows the UKC collaborative platform user interface we developed. The main interface for developers and validators is composed of five tabs: (1) The Home page (2) UKC Portlet, (3) Task Notifications list, (4) Discussion boards; and (5) Interactive Polls. In Fig. 6, the UKC portlet tab is selected.

[4] www.liferay.com

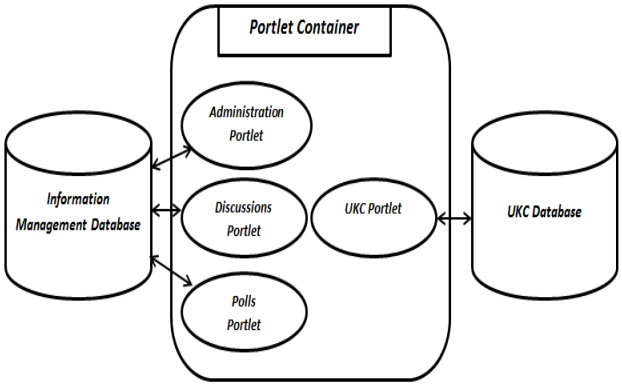

Fig. 11 UKC Collaborative Platform Overall Architecture

At the top region, the user can initiate a new search by typing a word and choosing the desired working and reference languages, respectively. The working language is the default language, when the user performs a search or an update operation; the system applies the changes based on the selected working language. The reference language is mainly for multilingual support in order to view the working language synset in another language or a lexical gap if there is no corresponding synset.

The middle region is divided into two main panels (Synsets and Concepts). Both panels are accompanied with an interactive toolbar to facilitate the manipulation of synsets, lexical and semantic relations by performing CRUD operations (i.e., Create, Read, Update, or Delete). Both panels are also accompanied with a display manager for updating the visual display of the displayed synsets and concepts. For each synset, it's possible to show or hide the: synset gloss, example sentences, the language-independent concept identifier, or the corresponding sysnet in the reference language. In addition, it's possible to filter the displayed synsets by their part of speech or lexical relation type. On the other hand, for each concept, it's possible to show or hide the language independent concept identifier and to filter the displayed concepts by their semantic relation type.

A dynamic synchronization exists between the two middle panels, the synsets panel and concepts panel, which takes place when the user selects any synset from the left region, the system automatically displays the corresponding concept in the right region.

The bottom region contains a set of color legends which are used to differentiate between working language synsets (black font), reference language synsets (blue font), or other language (red font) which is another possible case that could happen when the language independent concept label is obtained from another language different from both working and reference languages. In this case, the concept label will be retrieved from the UKC database and highlighted as a label from another language in a red font.

Using the synchronized synsets and concepts panels accompanied by the interactive toolbars, the display manager which provides full control on the displayed information, and the visual separation of languages using different color coding,

we should end up having an elegant linguistic analysis and manipulation tool which allows linguistic experts to enrich the available linguistic resources with minimal effort while communicating with each other in order to reach a common agreement through the discussion boards and creating interactive polls in case of conflicts.

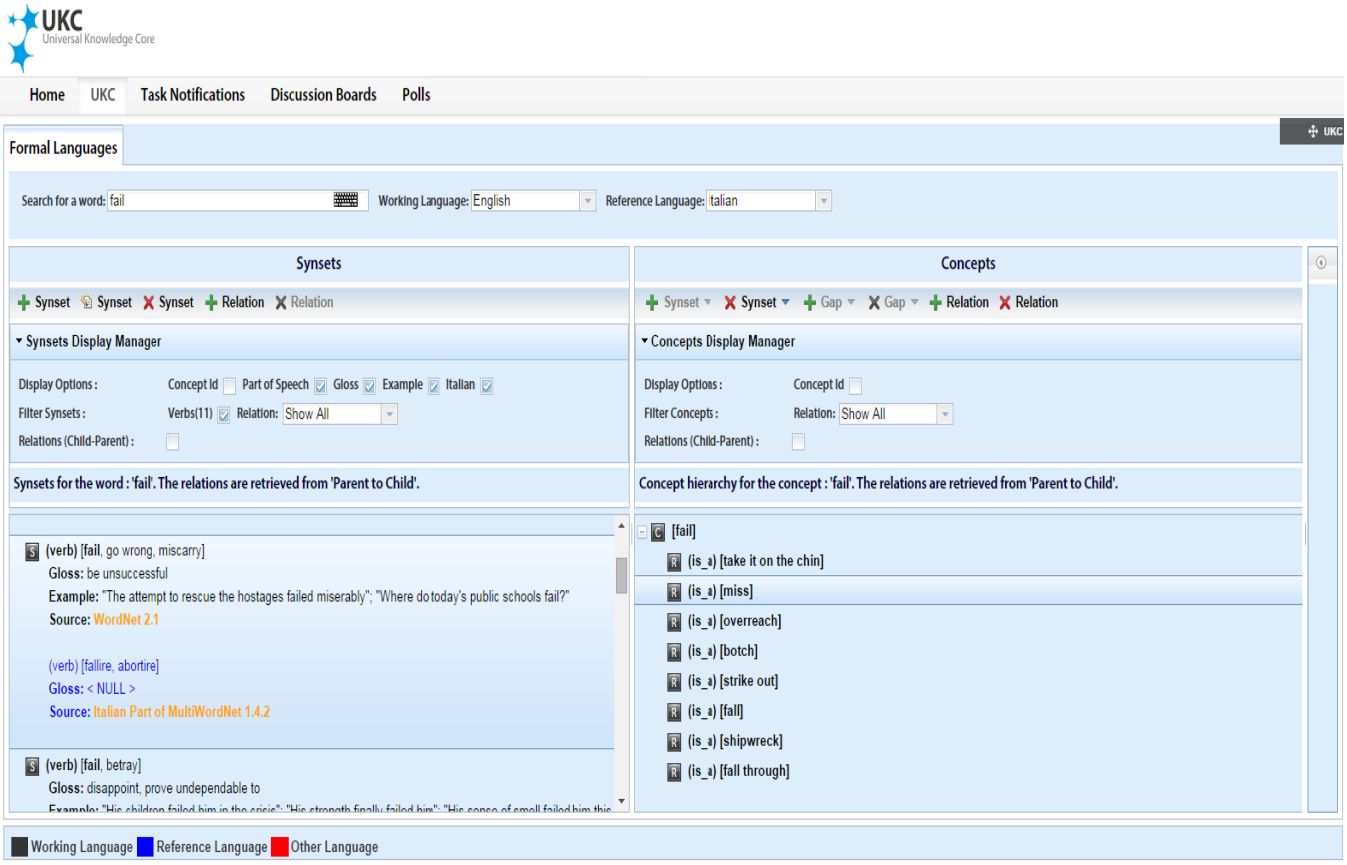

Fig. 12 UKC Collaborative Platform User Interface

VII. Related Work

There are only a few web-based ontology development tools supporting collaborative development. They can be classified into two main categories: (1) Semantic Wiki based tools; and (2) Interactive web based tools.

A. Semantic Wiki Based Tools

Semantic Wiki aims to combine traditional wiki systems with Semantic Web by introducing semantic web technologies and languages, such as RDF and OWL to the traditional Wiki.

Ontology development tools that are based on Semantic Wiki based systems have gained popularity during the past few years with the increase of active contributors to Wikipedia[5], as they could be easily extended and become familiar to many domain experts. Semantic Wiki based tools are similar to the traditional wiki, but they differ in the nature of content and methodology of development. On the following paragraphs we go through the main Semantic Wiki based ontology development tools.

[5] http://www.wikipedia.org/

1- **OntoWiki** [1] provides visual representations of domain ontologies as information maps. Information map entries are represented as web accessible pages and interlinked to related digital resources. OntoWiki also provides contextual views for entities, i.e. map views for locations and calendar views for Instance data. The tool supports collaborative content enrichment by enabling users to rapidly editing or adding contents through an inline editing mode analogous to the WYSIWYG (What You See Is What You Get) editing strategy for text editing, since information can be edited in the same environment as it is presented to users. Social collaboration features are supported by OntoWiki: (i) commenting on contents (ii) tracking all changes performed by contributors such as: contributions to the ontology schema, additions of entities or comments, and information about the contributor; and (iii) entities rating.

2- **CofficientMakna** [23] allows participants to create ontologies from scratch or to import existing ontologies to the wiki. Imported ontologies are mapped to the wiki hypertext model according to a predefined schema. The collaborative development is augmented with the use of an argumentation ontology that formalizes the arguments

exchanged between participants (issues, ideas and discussions) and provides a reasoning mechanism that can alert users about if they agree and disagree on the introduction of the same ontology entity.

3- **MoKi** [8] is a tool for modeling ontologies and enterprise process models in a collaborative MediaWiki[6] based approach. The tool associates a wiki page containing both unstructured and structured information to each entity of the ontology. The unstructured information contains the MediaWiki markup format (text, images, drawings, or any markup format) while the structured information contains description knowledge stored according to the modeling language adopted (RDF or XML) where each entity is described by means of triple having the form (subject, relation, object).

B. Interactive Web Based Tools

Ontology development tools that are designed and implemented as interactive web based tools need to provide facilities for: ontology development; ontology visualization [6] [18]; the design of multi-user interactive interfaces; concurrency control; mechanisms for data storage and alignment. On the following paragraphs we go through the main and the most promising interactive web based development tools.

1- **OntoLingua** [4] is among the first tools developed to provide collaborative ontology development facilities on the web. It supports collaborative ontology construction by providing simultaneous work tasks through group sessions, i.e. a user opens a session and then may assign another group of people ownership to it. This enables any other member of that group to join the session and work simultaneously on the same set of ontologies. One of the main drawbacks of this tool stands in the outdated web standards used; for instance, the server cannot notify users that a change has occurred until they revisit the page again. The tool also has no support for social collaboration.

2- **Protégé** [24], [25] is an open-source tool with a suit of plug-ins that allows domain experts to construct domain models and knowledge-based applications using ontologies. Protégé supports the creation, manipulation and visualization of ontologies in various formats (RDF, OWL, and XML). The Protégé platform supports two main ways of modeling ontologies; The Protégé-frames editor models ontologies as a set of classes organized in a subsumption hierarchy to represent fundamental concepts and a set of slots associated to classes to describe their properties and relationships, The Protégé-OWL editor models ontologies for the semantic web using the Web Ontology Language (OWL). ***WebProtégé*** [26], [27] is an extension project that supports collaborative ontology editing through the web. It allows multiple users to edit the same ontology at the same time and all changes made by one user are seen immediately by other users with the possibility of adding comments and annotations. Collaborative Protégé has an extension for supporting project specific workflows that could be defined using a generic ontology for modeling workflows [21], [22]. A workflow execution engine is required to interact with Protégé to run the modeled workflow for a specific project. Palma, R. [20] also proposed an editorial workflow-based approach for collaborative ontology development but both approaches differ from our approach since their workflow is modeled for a specific project and our approach offers a customized workflow for each collaborative object in the developed project. Table I provides a brief comparison between our platform and the commonly used ontology development tool.

[6] http://www.mediawiki.org/

VIII. Conclusions

In this paper we presented a collaborative methodology and the collaborative platform we developed for the development of Multilanguage ontologies able to address diversity in language and knowledge. The platform, we developed for the UKC, is an effective collaborative ontology development environment that allows for knowledge engineering in multiple languages. We presented the work done in terms of methodology, architecture and user interface. The proposed workflow-based approach is flexible, highly customizable and makes use of recent web 2.0 social and collaborative features. As part of the future work, we are planning to continue evolving the platform to effectively use it for the development of the UKC worldwide.

TABLE I
COMPARISON BETWEEN OUR COLLABORATIVE PLATFORM AND COMMONLY USED ONTOLOGY DEVELOPMENT TOOLS ON THE WEB

Linguistic Tool	Content Analysis Using Interactive Tool	Collaborative Enrichment	Ontology Localization	Web 2.0 Features	Workflow Support	Workflow Per Collaborative Object
Moki	No	Yes	No	Yes	No	No
OntoWiki	No	Yes	No	Yes	No	No
CofficientMakna	No	Yes	No	Yes	No	No
OntoLingua	No	Yes	No	No	No	No
Protégé	Yes	Yes	No	Yes	Yes	No
Our Tool	Yes	Yes	Yes	Yes	Yes	Yes

ACKNOWLEDGMENT

The research leading to these results has received funding from the CUbRIK Collaborative Project, partially funded by the European Commission's 7th Framework ICT Programme for Research and Technological Development under the Grant agreement no. 287704. We would like to thank all the members of the KnowDive group for their effort in developing the UKC. We also would like to thank Amit Kumar Sarangi, Enrico Bignotti and Subhashis Das for their participation in the early experiment conducted using EasyChair. . Finally, special thanks to the EasyChair team for giving us a permission to use their platform in our early experiment.

REFERENCES

[1] Auer, S., Dietzold, S., & Riechert, T. (2006). OntoWiki–a tool for social, semantic collaboration. In The Semantic Web-ISWC 2006 (pp. 736-749). Springer Berlin Heidelberg.

[2] Bouquet, P., & Giunchiglia, F. (1995). Reasoning about theory adequacy. a new solution to the qualification problem. Fundamenta Informaticae, 23(2), 247-262.

[3] Dutta, B., Giunchiglia, F. and Maltese, V. (2012). A facet-based methodology for geo-spatial modelling. *Journal on Data Semantics May 2012, Volume 1, Issue 1, pp 57-73.*

[4] Farquhar, A., Fikes, R., & Rice, J. (1997). The ontolingua server: A tool for collaborative ontology construction. International journal of human-computer studies, 46(6), 707-727.

[5] Fellbaum, C. (1998). WordNet: An Electronic Lexical Database. *Cambridge, MA: MIT Press*.

[6] Fluit, C., Sabou, M., & Van Harmelen, F. (2006). Ontology-based information visualization: toward semantic web applications. In Visualizing the semantic web (pp. 45-58). Springer London.

[7] Ganbold, A., Farazi, F. and Giunchiglia, F. (2014). An Experiment in Managing Language Diversity Across Cultures. The Sixth International Conference on Information, Process, and Knowledge Management.

[8] Ghidini, C., Rospocher, M., & Serafini, L. (2010). Moki: a wiki-based conceptual modeling tool. ISWC 2010 Posters & Demonstrations Track: Collected Abstracts, 658, 77-80.

[9] Giunchiglia, F., Dutta, B., & Maltese, V. (2014). From Knowledge Organization to Knowledge Representation. Knowledge Organization. 41(1), 44-56.

[10] Giunchiglia, F., Dutta, B. and Maltese, V. and Farazi, F. (2012). A facet-based methodology for the construction of large-scale geospatial ontology. Journal on Data Semantics (JoDS), Vol. 1, Issue 1 (2012), pp. 57-73.

[11] Giunchiglia, F., Maltese, V., & Autayeu, A. (2012). Computing minimal mappings between lightweight ontologies. International Journal on Digital Libraries, 12(4), 179-193.

[12] Giunchiglia, F., Maltese, V., B. Dutta (2012). Domains and Context: First steps towards managing diversity in knowledge. Journal of Web Semantics (JWS) Vol 12 - 13 (2012): Special Issue: Reasoning with Context in the Semantic Web.

[13] Giunchiglia, F., Maltese, V., Farazi, F., & Dutta, B. (2010). GeoWordNet: a resource for geo-spatial applications. In The Semantic Web: Research and Applications (pp. 121-136). Springer Berlin Heidelberg.

[14] Giunchiglia, F., Marchese, M., & Zaihrayeu, I. (2007). Encoding classifications into lightweight ontologies. In Journal on data semantics VIII (pp. 57-81). Springer Berlin Heidelberg.

[15] Giunchiglia, F., McNeill, F., Yatskevich, M., Pane, J., Besana, P., & Shvaiko, P. (2008). Approximate structure-preserving semantic matching. In On the Move to Meaningful Internet Systems: OTM 2008 (pp. 1217-1234). Springer Berlin Heidelberg.

[16] Giunchiglia, F., Yatskevich, M., & McNeill, F. (2007). Structure preserving semantic matching.

[17] Giunchiglia, F., Yatskevich, M., & Shvaiko, P. (2007). Semantic matching: Algorithms and implementation. In Journal on Data Semantics IX (pp. 1-38). Springer Berlin Heidelberg.

[18] Katifori, A., Halatsis, C., Lepouras, G., Vassilakis, C. and Giannopolou E. (2007). Ontology visualization methods - a survey. *Journal on ACM Computing Surveys (CSUR) Surveys Homepage archive, Volume 39 Issue 4, Article No. 10*

[19] Miller, George A. (1995). WordNet: A Lexical Database for English. *Communications of the ACM Vol. 38, No. 11: 39-41.*

[20] Palma, R., Haase, P., Corcho, O., Gómez-Pérez, A., & Ji, Q. (2008). An editorial workflow approach for collaborative ontology development. In The Semantic Web (pp. 227-241). Springer Berlin Heidelberg.

[21] Sebastian, A., Noy, N. F., Tudorache, T., & Musen, M. A. (2008). A generic ontology for collaborative ontology-development workflows. In Knowledge Engineering: Practice and Patterns (pp. 318-328). Springer Berlin Heidelberg.

[22] Sebastian, A., Tudorache, T., Noy, N. F., & Musen, M. A. (2008). Customizable workflow support for collaborative ontology development. In 4th International Workshop on Semantic Web Enabled Software Engineering (SWESE) at ISWC (Vol. 2008).

[23] Tempich, C., Simperl, E., Luczak, M., Studer, R., & Pinto, H. S. (2007). Argumentation-based ontology engineering. IEEE Intelligent Systems, 22(6), 52-59.

[24] Tudorache, T., & Noy, N. F. (2007, July). Collaborative Protege. In CKC.

[25] Tudorache, T., Noy, N. F., Tu, S., & Musen, M. A. (2008a). Supporting collaborative ontology development in Protégé (pp. 17-32). Springer Berlin Heidelberg.

[26] Tudorache, T., Nyulas, C., Noy, N. F., & Musen, M. A. (2013). WebProtégé: A collaborative ontology editor and knowledge acquisition tool for the web.Semantic web, 4(1), 89-99.

[27] Tudorache, T., Vendetti, J., & Noy, N. F. (2008b, October). Web-Protege: A Lightweight OWL Ontology Editor for the Web. In OWLED (Vol. 432).

[28] Zaihrayeu, I., Sun L., Giunchiglia,F., Pan, W., Ju,Q., Chi, M. and Huang X (2007). From Web Directories to Ontologies: Natural Language Processing Challenges