ONTOLOGY DEVELOPMENT FOR PEDAGOGIC CONTENT INFORMATICS

Mark Sunday Ekanem
Department of Electrical/Electronic and Computer Engineering, University of Uyo, Uyo, **NIGERIA**

Simeon Ozuomba
Department of Electrical/Electronic and Computer Engineering, University of Uyo, Uyo, **NIGERIA**

ABSTRACT

This paper reports the development process of ontology on number bases topic in mathematics that will enable personalization in the delivery of pedagogic content when incorporated with an appropriate application to form an e-learning platform. Pedagogic attributes of number bases topic were analyzed, delineated and defined through a requirement engineering process. The concepts arrived at as the outcome of the requirement engineering process were the basis for the design of the ontology which included an algorithm that was used to implement the ontology with an incorporated markup mechanism. Implementation of the ontology was carried out with Protégé software and the resultant ontology was put through different tests including tests using inbuilt reasoners such as HermiT and FaCT++. After the completion of reasoner-based testing D2R server was used as an external testing tool to ensure that all data representations in the ontology were actually accessible. At the end of testing, the outcomes of the tests were evaluated to certify that the ontology had captured all the requirements arrived at during the requirement engineering phase. The positive outcomes of the evaluation processes were the basis for certifying the ontology fit for deployment. The successful outcome of this ontology creation process indicates that availability of pedagogic materials could be increased through the collaboration of education experts and software developers.

Keywords: Ontology, Pedagogy, Semantic, E-learning, Markup.

INTRODUCTION

With the decline in educational quality in certain parts of the world, there has been a need for research into the possible causes of this malaise. One of such studies is presented in Ogbu (2015) where an analysis of the negative implications of lack of instructional materials in the teaching of electrical/electronic engineering education was carried out with the result of the analysis showing that the lack of instructional materials impacted negatively on the students both within the learning environment during the course of their studies and even in the work environment after they must have graduated. This led to a recommendation to stakeholders to increase the availability and accessibility of instructional materials.

Majumdar (2015) reports on analysis of the various ways of increasing availability and accessibility of instructional materials for learners and recommended as one of the ways, the combination of pedagogy and Information and Communication Technology (ICT). Rani, Nayak and Vyas (2015) recommended the creation of educational platforms that offer personalized content which is commendable as learners in the same classroom are rarely on the same level of comprehension of a topic at a given time. This personalization helps the learners to learn at their pace while achieving the learning objective of the full comprehension of such a topic.

Drawing from the recommendation of Majumdar (2015) one of the best ways of increasing availability of instructional materials is through the creation of electronic learning (e-learning) platforms. It could be observed that e-learning platforms could have a wide geographical coverage if so intended as this is achievable through the use of the internet, therefore changes on the structure and form of the internet will surely affect such e-learning platforms. As such, it is wise to consider these characteristics of the internet while designing any e-learning platform. In consideration of recent changes on the internet, it is observed that the computing world is slowly adopting the semantic web to be the new normal and any e-learning platform to be designed now should be semantic web compatible.

In designing a semantic web compatible e-learning platform, ontology shall be required. According to Ristoski and Paulheim (2016), ontology is the standard form of knowledge representation on the semantic web. It could also be noted that knowledge represented in ontology can be marked-up in such a way to allow for personalization in the delivery of query results from ontology (Abdullah and Ibrahim, 2012). The aim of this paper is to report on the development process of ontology that can be used to develop a semantic web compatible e-learning platform.

LITERATURE REVIEW

In Buraga and Cioca (2006), the authors investigate the possibility of utilizing standardized semantic web-based languages, such as RDF and OWL, in order to model a complex e-learning system and to improve the interconnectivity between heterogeneous components of that system. The booklet proposes an XML-based model for attaching metadata, expressing the relations between e-learning system components, and using ontologies to express knowledge within such an application.

The booklet states that most semantic web technologies are based on XML and are structured in three main layers; metadata layer, schema layer and logical layer. RDF is used to associate metadata with the software components and also store the metadata. RDF standard was chosen because it provides interoperability between applications that exchange machine-understandable information on the web. The RDF metadata was also used to describe each user's session information. Where rdf namespace conflicts could occur in the metadata, xml namespace was used. FOAF (Friend of a Friend) statements were used to show connections users make through the tasks they carry out or through the system components they engage in carrying out their tasks. Modelling was also carried out by creating (using OWL), or reusing, ontologies whose semantic nature offers more advantages than those offered by relational and XML database formats. This is because ontologies provide an objective specification of domain information. In order to ensure that interactions between users and software components or between software components meet the standard of interoperability, an XML-based protocol SOAP (Simple Object Access Protocol) was used. The Process ontology described by OWL-S specification describes how these interactions could be modelled.

The model which was the result of this work is indicated to function best with an agent-based e-learning system.

Bouarab-Dahmani, Comparot, Si-Mohammed and Charrel (2015) proposed a general way to operate an ontological modelling of teaching domains for e-learning by doing. This model describes a given domain with concepts, links, pedagogical resource descriptors and rules, valuable for each discipline. This representation will help designers to easily define, for each domain, its specific knowledge base by a semi-automatic instantiation process. The objective is a domain representation appropriate both for information retrieval and for “cognitive” and complex automated teaching activities such as learners’ evaluation by reasoning on detected errors when there are open questions.

This model was first implemented with a self-learning relational database and used by PHP and JavaScript programs for different complex learning activities such as errors diagnosis and learner’s marking. After that, for an easier and more efficient use via the web with semantic web tools, there was an implementation with OWL. This paper shows that it is possible to generalize our approach to any domain based on e-learning by doing mode. The model introduced here shows a domain through two points of views: the specification view given by the ontology and the resources view generally known in e-learning as learning objects. The specification ontology is a granular model where the knowledge of a considered domain is stored, making the use of the ontology for resources retrieval (web semantic use) or for any other learning or teaching activity guided by the domain semantic.

Among the novelties of this model, was the possible propagation of the learner’s evaluation results to each defined components of the expert domain allowing adaptive content generation. The second novelty is related to the integration of pedagogical resource descriptors described using some data elements from the LOM standard metadata scheme. This will give connection between the ontology concepts (considered at the instance level as semantic annotations) to the domain pedagogical resources, which makes the model significantly more powerful.

Results show that the ontological models that have been implemented on specific domain teaching can be generalized to provide a kind of meta-model that becomes reusable for domain knowledge which complies with the structuring process that was defined. The TDO (Teaching Domain Ontology) model was particularly used for pedagogical resources retrieval in the prototype developed using Java servlets in NetBeans6 environment.

METHODOLOGY

In this paper, Inclusive Iterative Incremental Development Methodology (IIIDM) shown in Figure 1 was used to develop the ontology. The methodology which was adapted from Iterative Incremental Development Methodology (IIDM) (MERA, 2017) begins with planning and ends with deployment.

Planning Phase

In the planning phase the scope of the ontology was defined and the tools for implementation of the ontology were also selected. In this paper, the scope was some subsections of Number bases topic in mathematics while the implementation software selected was Protégé.

Requirements Phase

This takes care of the requirement engineering of the ontology development process. The requirement engineering consists of four aspects; requirement elicitation, requirement analysis, requirement specification and requirement validation. The requirement engineering process model adopted in this research work is a User-centric Incremental Requirement Engineering Process Model (UCIREP), as depicted in Figure 2.

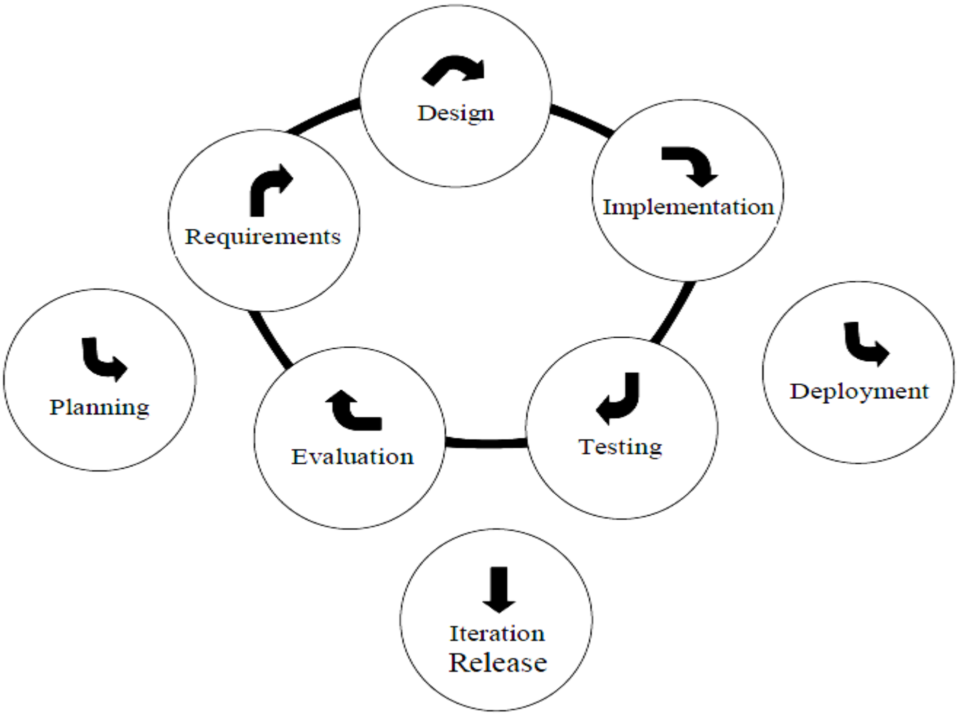

Figure 1: The Inclusive Iterative Incremental Development Methodology (IIIDM)

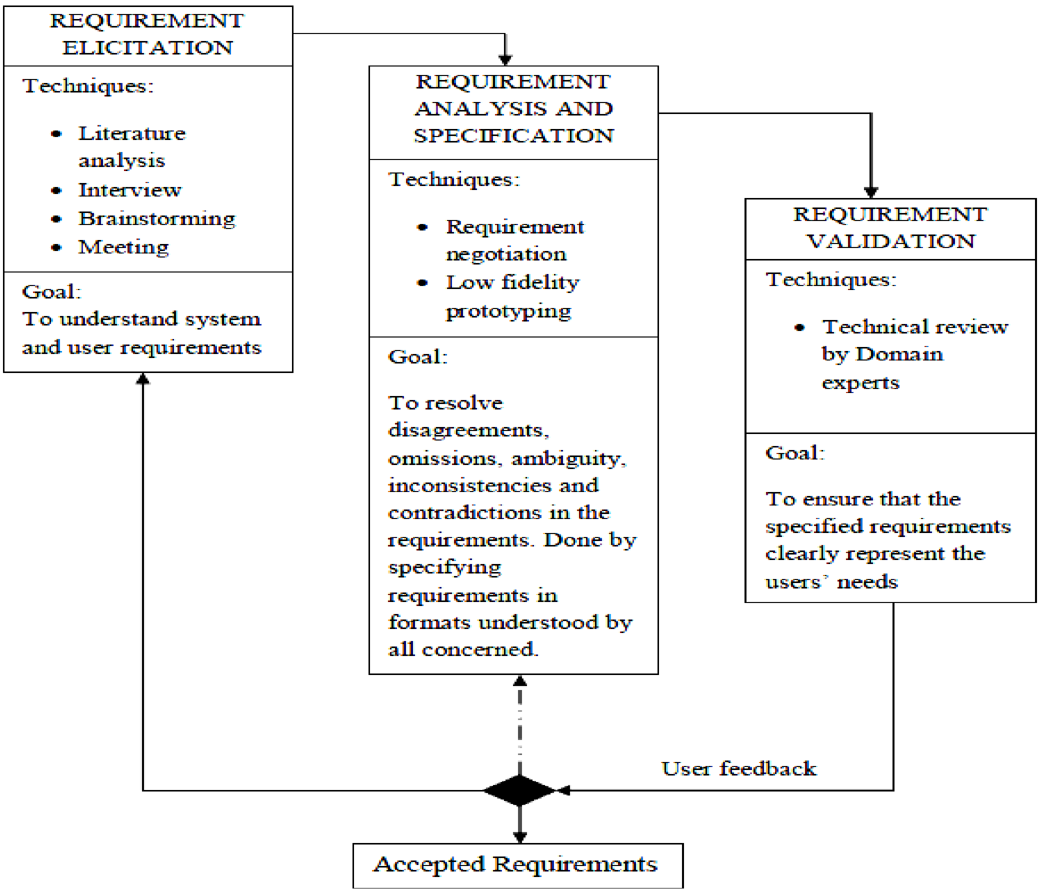

Figure 2: UCIREP model for ontology development

The resultant specification of the requirement engineering gave elements and terms which were used to define classes, class properties, constraints and instances in the ontology. The summary of the number bases pedagogic scope as considered in the requirement phase are as follows:

I. Numbers
   i. Whole numbers
   ii. Integers
   iii. Real numbers
II. Number bases
   i. Binary
   ii. Octal
   iii. Decimal
   iv. Hexadecimal
III. Number base digits
   i. Binary – 0, 1
   ii. Octal – 0, 1, 2, 3, 4, 5, 6, 7
   iii. Decimal – 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
   iv. Hexadecimal - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
IV. Number base conversions
   i. Binary: Binary-to-Octal, Binary-to-Decimal, Binary-to-Hexadecimal
   ii. Octal: Octal-to-Binary, Octal-to-Decimal, Octal-to-Hexadecimal
   iii. Decimal: Decimal-to-Binary, Decimal-to-Octal, Decimal-to-Hexadecimal
   iv. Hexadecimal: Hexadecimal-to-Binary, Hexadecimal-to-Octal, Hexadecimal-to-Decimal
V. Learner level
   i. Beginner – Zero to very insignificant previous knowledge
   ii. Intermediate – Some previous knowledge of topic
   iii. Expert – A lot of knowledge on the topic
VI. Examples and practice questions presentation for learner levels
   i. Beginner – Whole number only
   ii. Intermediate – Integers and some real numbers with small-value integer part
   iii. Expert – Decimals with large value integer part

Design Phase
The design phase takes care of the design of the ontology. With the concepts already defined as stated in the previous section, the concepts are now developed and made ready for actual coding.

The main activity in this design process is the development of an algorithm that will be a guide for the ontology coding process. The algorithm is divided into 2 sections; pre-protégé coding activities and protégé coding activities.

Pre-Protégé Coding Activities
These activities must be carried out to ready the components and elements that will be required in protégé coding activities. These include:

i. Develop/acquire examples
ii. Develop/acquire practice questions
iii. Typeset and convert examples to image format using a suitable photo editor
iv. Save images with appropriate names for ease of reference in ontology coding

Protégé Coding Activities

These are activities that require the Protégé software to be carried out. These include:

i. Launch Protégé
ii. Save new ontology using suitable name
iii. Create classes (super-classes) under the main class "Thing" that is create num-base, num-base-eg, num-base-level and num-base-val
iv. Create instances representing values ranging from digits to pictures of examples
v. Create instances representing the practice questions and answers
vi. Map instances to classes
vii. Add properties to instances and classes that reflect the constraints (the markup mechanism operates through these properties and constraints)
viii. Add annotations where necessary (Annotation is used to add any description that may be attributed to a class, instance or any entity for that matter and may be in a string form or any other form that may be acceptable to the user.)
ix. Compare asserted model to inferred model to ascertain if there are errors
x. If errors exist, these are corrected
xi. Save ontology in required format

Procedure for Creating Ontology with Markup Mechanism using Protégé

The processes involved include developing concepts, defining classes, defining class hierarchy, specifying relationships between classes, specifying class properties, defining constraints which are used to create axioms/rules and in creating instances. This procedure is represented as a structure diagram in Figure 3.

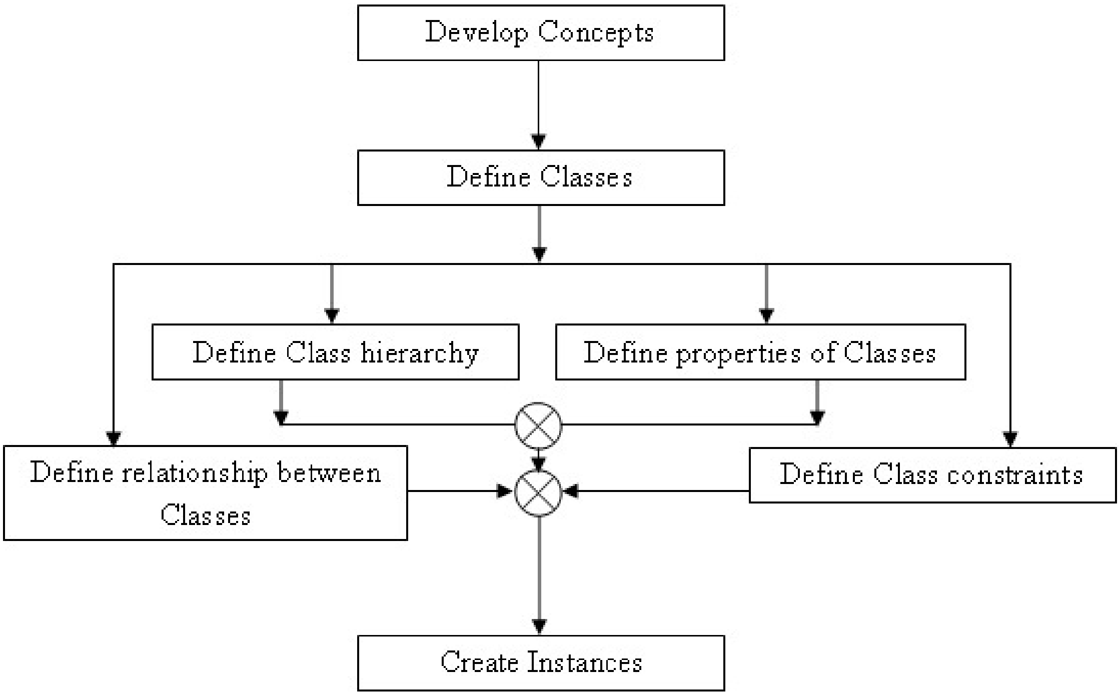

Figure 3: Procedure for creating ontology

Implementation of Ontology and Markup Mechanism

Implementation was carried by using Protégé to code the ontology. The algorithm stated under the Protégé Coding Activities section and the procedures illustrated in Figure 3 were followed and the ontology was created in Protégé software and saved in OWL/RDF format.

Protégé uses a graphical user interface where a user chooses what type of entity is to be created, types in the name and selects the characteristics of such an entity. These characteristics include but are not limited to; functional, inverse functional, transitive, symmetric and reflexive. Also, object properties could be set such as which classes, objects or instances could belong and which classes the objects cannot belong. Data properties are also set for the data types to be inputted such as the digits of the number bases which should be set as integer.

On the blank interface of the software, the user clicks on the Classes tab in order to add the various classes and sub-classes previously outlined in the design stage that will make up the ontology. The resultant specifications included number bases (binary, octal, decimal and hexadecimal) which were mapped under the main class (num-base). Also under the Classes tab interface, class properties such as Equivalent class, Sub-class, Members, Target for key, Disjoint-with and Disjoint-Union-of can be specified which are used to set the constraints and properties of each class. The conversion procedures between the classes were mapped to the class properties. Constraints and properties of the defined classes are instrumental to the creation and assignment of instances (individuals) to the various classes that have been created. These class properties and constraints are also the basis of the markup mechanism's operation in the ontology.

Testing of Ontology and Markup Mechanism

After the creation of the ontology, testing was carried out through the installed inference engines (which are APIs) called reasoner on Protégé. A reasoner on Protégé tests if the logic of the relationships between components within the ontology is correct. The APIs used were FaCT++ and HermiT v 1.3.8. Under the Classes tab, there are two types of class hierarchy; Asserted Class hierarchy and Inferred Class hierarchy. The inferred class hierarchy is a result of the reasoner's inference drawn from the ontology developed (asserted) by the user. The results of this test which show the whole ontology were viewed and printed both in graphical (tree) view (using the APIs OWLViz and OntoGraf) and in expanded linked list view. Also, the D2R server was used to test the workability of the ontology as a database representation by retrieving and checking the standardization of each entry in the ontology. This was an external test carried out without the use of any Protégé-linked tool.

Evaluation of Ontology

The printouts gotten from the OWLViz and OntoGraf APIs during the testing of the ontology were then given to resource persons from education field to verify that the concepts and relationships depicted in the graphical view of the resultant ontology were in agreement with the accepted requirements arrived at on completion of the requirements engineering process. The verifications were carried out and where modifications were suggested, these were effected. These verification outcomes were also compared with the results of testing the ontology with the D2R server before the final acceptance of the finished ontology.

RESULTS

This section shows graphical illustrations and textual descriptions of the processes carried out in the methodology and their outcomes.

Ontology Implementation

With the completion of the implementation process in Protégé, the interface of the software appears as shown in Figure 4. The screenshot in Figure 4 shows the full list of populated classes and a partial list of the populated instances (individuals) that make up the ontology. Each of the individuals shown was then selected and their properties specified as these properties were used to classify these individuals into the classes as per the design.

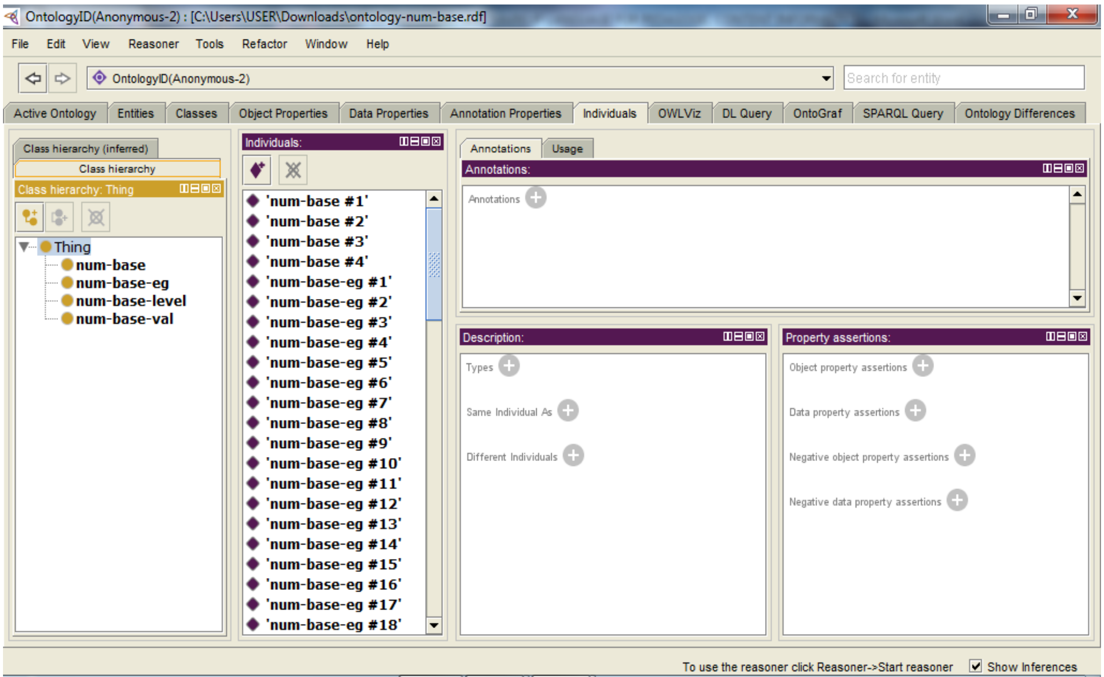

Figure 4: Partial view of a populated Individuals tab

Ontology Testing

Figure 5 shows the Reasoner inferred class hierarchy result which indicates that there were no logical errors during the implementation of the relationships between the entities within the ontology.

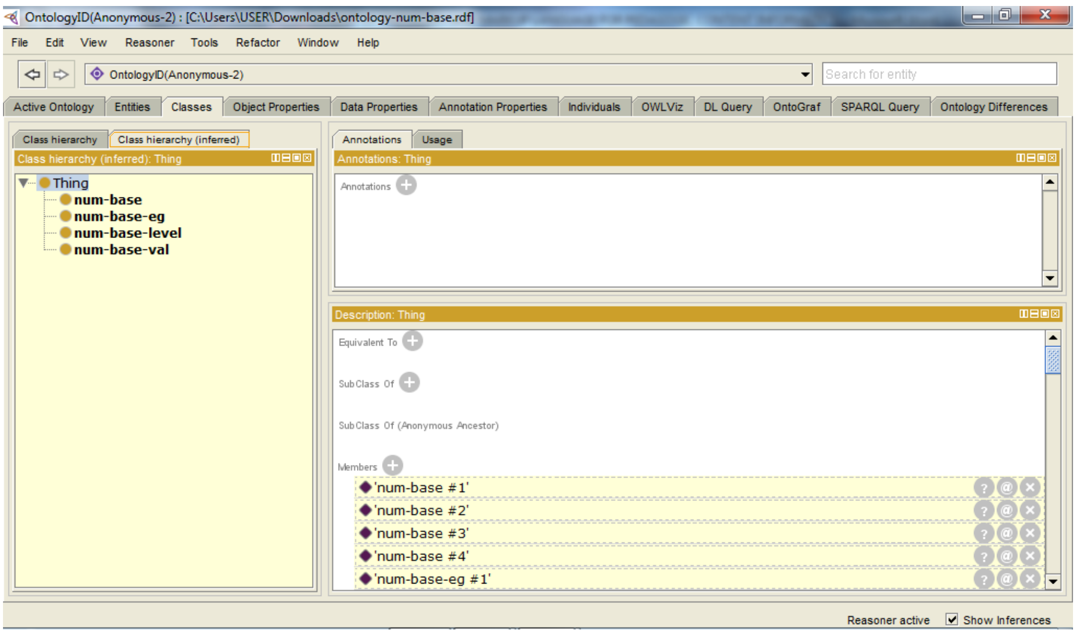

Figure 5: Reasoner inferred class hierarchy

The OntoGraf plugin is used to visualize the ontology both at class and instance level. The OntoGraf view of the classes while specifying instances in the num-base class is presented in Figure 6. Num-base-level instance specification is shown in Figure 7. This test was also carried out for the other classes with positive results confirming that the logical relationships between the entities in the ontology are in order.

The D2R server after launching was loaded through a regular browser (in this case Google Chrome was used). The objects were then browsed one by one to ensure that the respective RDF graphs were properly generated. Screenshots in Figure 8 and Figure 9 show part of the process of querying an object in the ontology.

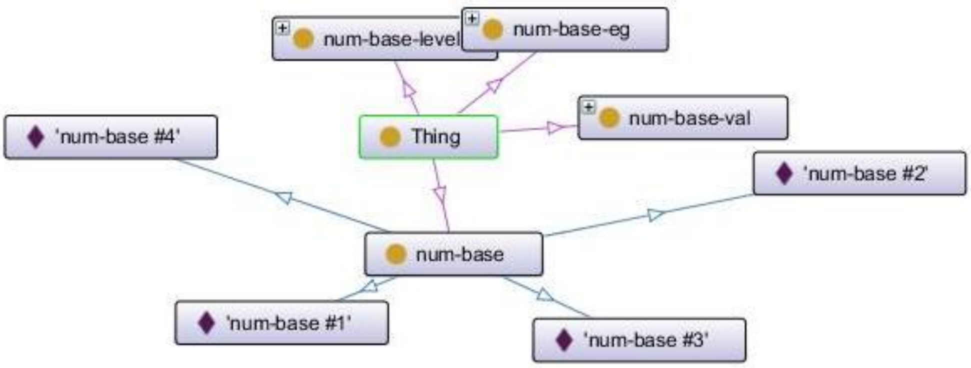

Figure 6: OntoGraf view of classes specifying instances in the num-base class

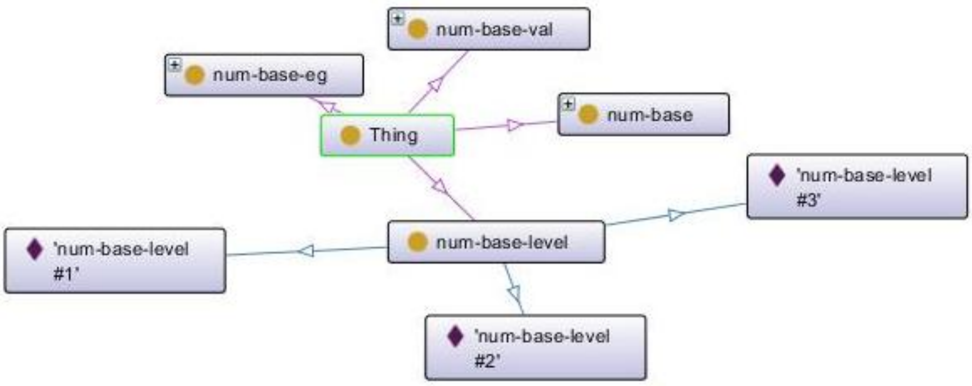

Figure 7: OntoGraf view of classes specifying instances in the num-base-level class

The result shown in Figure 8 shows the hierarchy of objects contained in the num-base class and their respective Uniform Resource Identifiers (URIs). To further deepen the query, num-base #1 is selected and the result of the query is shown in Figure 9.

A further query on one of the items shown in the Figure 9 screenshot is shown in Figure 10. The screenshot in Figure 10 shows metadata on objects linked by the markup mechanism to the data object referenced as <http://localhost:2020/resource/num-base-eg/1> in Figure 9.

Figure 8: Query of num-base (class) link on the D2R server

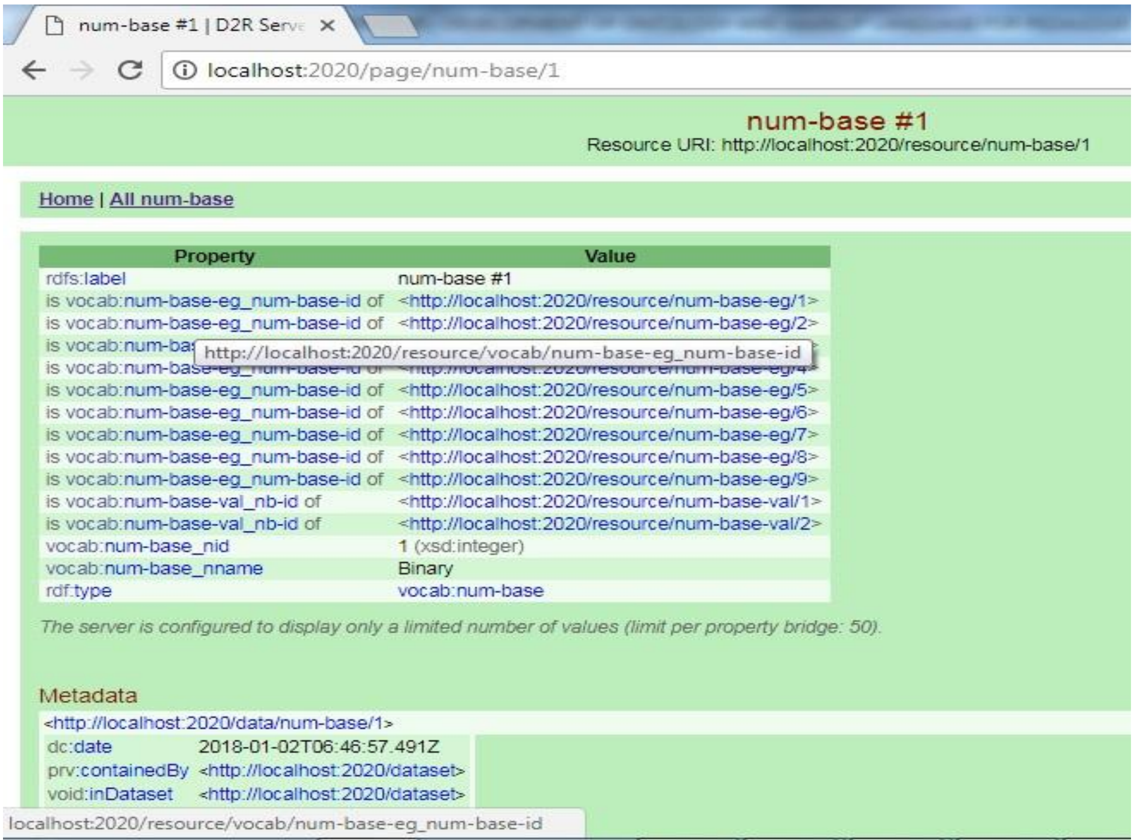

Figure 9: Query of num-base #1 link on the D2R server

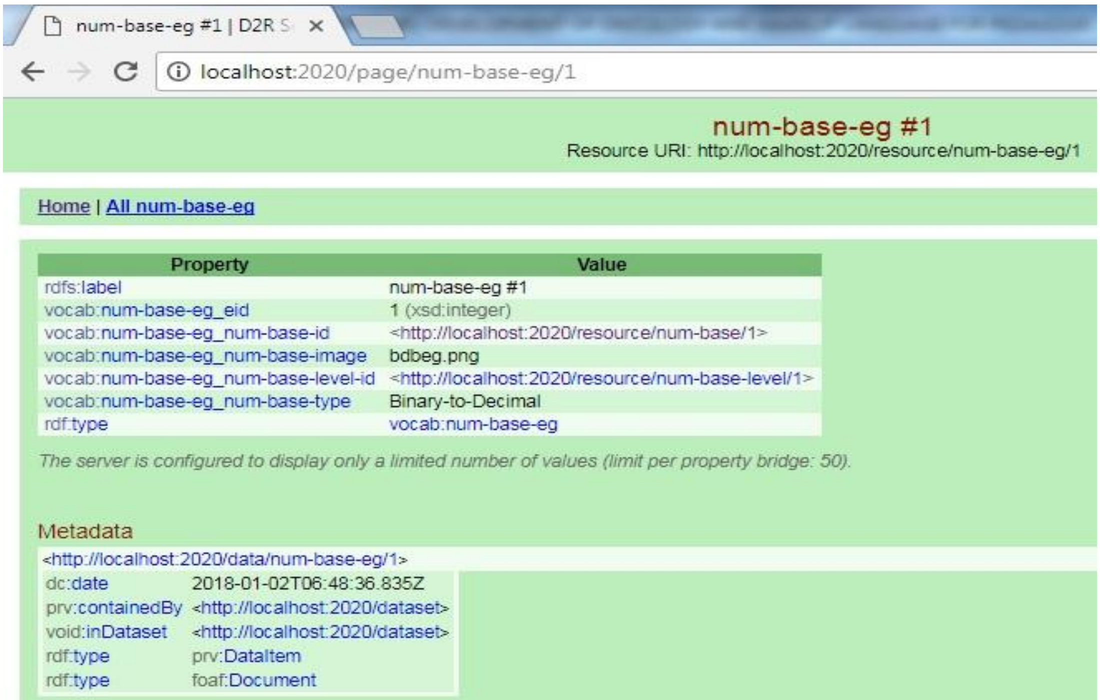

Figure 10: Query of a property value on num-base #1 link on the D2R server

DISCUSSION

The num-base class contains members which are instances pointing to the various number bases referenced in this work. The num-base-eg class contains members which point to the various examples the user will encounter in the course of using the ontology in conjunction with software to learn at the various proficiency levels. The num-base-level class contains members used to specify the various proficiency levels available in the ontology with which the user will be rated. The num-base-val class contains the members pointing to all the values of instances so that when a query for a selected object is received from an application, the ontology picks the value of the selected object and uses that as the basis of its query response.

As shown in Figure 9, the result of the query on num-base #1 shows the properties and values linked to this object in the ontology. On inspection it is noticed that num-base #1 refers to the binary number base with number base id as 1 in the ontology. The query result also contains metadata about num-base #1. A further inspection indicates that examples linked to this class are also listed among the data properties displayed. This shows the markup mechanism in operation as all objects that have been marked to belong to Binary class are shown here. A query on one of the examples is carried out by clicking the first link below the num-base #1 value that is <http://localhost:2020/resource/num-base-eg/1>, the query response is given in Figure 10 where it shows among the various details, that it is an example for binary to decimal conversion explanation.

CONCLUSIONS

The aim of this paper has been to report on the development process of ontology that can be used to develop a semantic web compatible e-learning platform. The methodology used was the Inclusive Iterative Incremental Development Model. Number bases in mathematics was chosen as the case study topic and the pedagogical attributes of this topic were delineated, defined and developed into concepts that can be expressed in a software design. A design was created which had an algorithm that was then implemented through Protégé software. The resulting ontology was tested using various methods to ascertain that pedagogic content attributes previously delineated were well represented and the test results proved that the attributes were properly represented.

The successful development of this ontology which will be used with a web application may be seen to indicate that bridging the learning gap may not be successfully carried out by experts of education field alone. The education experts who should initiate such advancements from researches carried out by them will need experts in systems and software engineering to incorporate such ideas with ICT to produce widespread solutions to educational challenges especially in this digital age where almost nothing works without the involvement of computers.

REFERENCES

Abdullah, N. and Ibrahim, R. (2012) Knowledge retrieval using hybrid semantic web search. *Proceedings of 2012 International Conference on Computer and Information Sciences*. Kuala Lumpur, Malaysia.

Bouarab-Dahmani, F., Comparot, C., Si-Mohammed, M. and Charrel, P. J. (2015) Ontology based teaching domain knowledge management for e-learning by doing systems. *Electronic Journal of Knowledge Management*, 13(2), 155-170.

Buraga, S. and Cioca, M. (2006) *Modeling aspects of semantic web-based e-learning system*. Sibiu: University Press.

Majumdar, S. (2015) *Emerging Trends in ICT for Education and Training* [Retrieved on 8th December 2016] Available from World Wide Web: http://www.stthomascollegebhilai.in/wp-content/uploads/2016/10/emergingtrendsinictforeducationandtraining.pdf

MERA (2017). *Iterative and incremental development | MERA. Software done right* [Accessed 19th April 2017] Available from World Wide Web: https://www.mera.com

Ogbu, J. E. (2015) Influences of inadequate instructional materials and facilities in teaching and learning of electrical/electronic technology education courses. *International Journal of Vocational and Technical Education*, 7(3), 20-27.

Rani, M., Nayak, R. and Vyas, O. P. (2015) An ontology-based adaptive personalized e-learning system, assisted by software agents on cloud storage. *Knowledge-Based Systems*, 90, 33-48.

Ristoski, P. and Paulheim, H. (2016) Semantic web in data mining and knowledge discovery: A comprehensive survey. *Web Semantics: Science, Services and Agents on the World Wide Web*, 36, 1-22.