Software Centric Innovative Methodology for Ontology Development

Santhosh John[1], Nazaraf Shah[2], Craig Stewart[2] and Leon Samlov[2]
[1]*Training Unit, Middle East College, Knowledge Oasis Muscat, Sultanate of Oman*
[2]*School of Computing, Engineering and Mathematics, Coventry University, U.K.*

Keywords: Hybrid Methodology, Ontology Development, Stages, Workflows, Activities, Ontology Development Life Cycle.

Abstract: Ontologies are mainly used to establish ontological agreements explicitly which serves as the basis for communication between either humans or software agents. In the aspect of knowledge representation, knowledge base starts where ontology ends. Ontology Engineering, a branch of knowledge engineering derived exclusively for the methods, methodologies, techniques and technologies used for the design, development and maintenance of ontologies. Though ontology engineering and software engineering are two complementary engineering branches, there exists a significant gap between them in terms of maturity level and popularity. Absence of effective methodologies eligible to claim the tag 'standardized' aimed at supporting the development of large scale ontologies is one of the reasons behind the gap. This paper attempts to bridge this gap by proposing a software centric innovative methodology (SCIM) for ontology development by extending the process models of software engineering with a defined ontology development life cycle (ODLC).The proposed methodology defines the stages, workflows, activities and techniques for the development of an ontology regardless of domain in a systematic manner for the practitioners to follow.

1 INTRODUCTION

Linked data supports the evolution of web where both data and documents are linked using the structural model of Semantic Web (Linkeddata.org, 2017). Ontologies, in particular, fulfil the requirements for knowledge representation for the Semantic web. Semantic web technologies based on ontology have emerged as an appropriate engineering solution to the problems of developing systems that ensure the integration of data from different sources with high level of interoperability to provide seamless services to web users.

Ontology is a broad term used for various purposes such as natural language processing, information extraction, intelligent search engines, digital libraries and business process modeling etc. It is mainly used to establish ontological agreements explicitly to serve as the basis for communication between both human and software entities. Therefore, it is mandatory to reduce the language ambiguity and differences in knowledge between parties involved to avoid confusions, errors and inefficiency (Blanco et al., 2011).Ontology engineering, the branch of knowledge engineering mainly deals with the formal principles to build and maintain ontology. This includes the processes such as development, management, analysis and reuse of ontologies. The mentioned processes covers the aspects of methods, methodologies and the diverse set of tools used for designing, visualizing, developing, editing and maintaining ontologies.

Development of an ontology exhibits both structural and logical complexity comparable to the development of software systems. However, ontology development is more complex compared to any kind of software development due to various factors such as the necessity of diverse tools support, the requirement of higher learning curve, compatibility among heterogeneous platforms, dynamic changes in business needs, lack of performance engineering, and low failure tolerance etc. Unlike software engineering, absence of effective and standardized ontology development methodologies restricts the development of large scale high quality ontologies regardless of the domain concerned. Though a few methodologies provide an engineering approach with adequate details, most of the available methodologies lack sufficient details of techniques and activities employed in them (Iqbal et al., 2013)

John S., Shah N., Stewart C. and Samlov L.
Software Centric Innovative Methodology for Ontology Development.
DOI: 10.5220/0006482901390146
In *Proceedings of the 9th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (KEOD 2017)*, pages 139-146
ISBN: 978-989-758-272-1

The main contribution of this paper is to explicate the derivation of a Software Centric Innovative Methodology (SCIM) for ontology development. Both philosophical and engineering aspects of the proposed methodology have been defined inline with the existing approaches. The proposed methodology is hybrid in nature in terms of its underpinning philosophy. The components of hybrid model have been extended from well proven software engineering process models/methodologies. In Section 2, we describe necessity of a novel methodology for ontology development. The engineering aspects, background and overview of the proposed methodology are discussed in Section 3. Section 4 discusses the defined ontology development life cycle with the stages, workflows, activities and techniques. The final section concludes the paper and highlights some future research directions.

2 NEED OF AN INNOVATIVE ONTOLOGY DEVELOPMENT METHODOLOGY

A methodology in ontology engineering is composed of methods, techniques, processes and activities and may follow several approaches for the development of ontology. It has been observed from the literatures that, most of the methodologies proposed in ontology engineering in the past, lacking the details of techniques and activities employed in them with appropriate mapping of underpinning philosophy and approaches (Fernández-López and Gómez-Pérez, 2002) (N.Foy, 2001) Few of the available methodologies are influenced by software engineering (SE) methodologies but failed to map fully with SE. Since 1995, methodologies such as KEM, TOVE etc. were proposed by practitioners for ontology development. However, they are focused more on the problem or the domain concerned. Unlike the predecessors, a methodology called METHONTOLOGY was proposed by Foundation for Intelligent Physical Agents (FIPA) with the inclusion of activities and techniques to build domain ontologies from scratch (Fernandez, 1997). However, this methodology does not propose any techniques to carry out the activities in a formal manner.

A software engineering approach, Unified Process for Ontology (UPON) (De Nicola, Missikoff and Navigli, 2009) has been proposed for ontology development based on a rich set of resemblances between software engineering and ontology engineering. One of the weaknesses of the UPON methodology is that it does not target the development of generic ontologies. It focuses on ontologies that serve its specific consumers and automated systems. Moreover, UPON fails to provide comprehensive details for collaborative ontology construction aspect (Iqbal et al., 2013). eXtreme Programming of Knowledge-based systems (XP.K) (An Agile Development Methodology for Knowledge-Based Systems Including a Java Framework for Knowledge Modeling and Appropriate Tool Support, 2002) has been proposed as a lightweight agile methodology for the development of knowledge base systems extended from Extreme Programming (XP). It follows the values of XP but generalises the value of communication to community. Collaborative ontology development has been highly encouraged by this methodology as its emphasizes humility. Few of the additional practices applied to XP.K are *Joint ontology design and pair modeling*, round trip-engineering, testing and constraint testing XP.K is suitable for the development of ontologies for knowledge base systems as the additional practices supports collaborative ontology development. However, a defined ontology development life cycle is missing in XP.K. The summary of a review among major existing ontology development methodologies based on relevant criterion is presented in Table 1. The first four parameters used for the review are reflecting the higher level aspects of concerned methodology whereas the last four parameters are specific and technical ones.

Based on the analysis, it has been observed that most of the existing methodologies failed to provide adequate details for the techniques employed in them with a defined ontology development life cycle (ODLC). Notion of reusability is limited to few development methodologies. Since ontology development faces higher learning curve as one of the obstacles against its growth, we see the scope of an innovative software centric approach which assists software practitioners for ontology development can make a significant difference in large scale ontology development. The ODLC of proposed methodology has been defined in such a way to reduce the learning curve significantly to software practitioners. Therefore we propose a hybrid methodology extended from well proven software engineering process models with a defined ODLC. The proposed methodology is inclusive of a complete coverage of employed methods and activities.

3 OVERVIEW OF SCIM

The underpinning philosophy of SCIM is the software centric approach. This has been embedded in a hybrid model of linear waterfall and iterative Rational Unified Process (RUP).

Process models have been extended from software engineering. The engineering aspects of the

Table 1: Review of existing methodologies.

Name of Methodo-logy	**Mode of develop-ment**	**Support for collabora-tive ontology develop-ment**	**Support for reusabi-lity**	**Support for inter-operabi-lity**	**Extent of Applica-tion dependency**	**Ontology Life Cycle support**	**Coverage of employed methods and activities**
Ushold and King (KEM)	Stage based	No	No	No	Application dependent	No	Limited coverage available for purpose identification, ontology building and evaluation.
Gruninger and Fox (TOVE)	Stage based	No	Yes	No	Application Semi-independent	No	Limited coverage available for informal specification, formulation of competency question…
METHONTOLOGY	Evolutionary prototype	No	Yes	No	Application independent	Yes	Abstract level coverage for ontology building stages
Ontoligua	Modular development	No	Yes	Yes	Application independent	No	Limited coverage on ontology development and integration.
On-To-Knowledge	Evolutionary prototype	No	No	No	Application dependent	Yes	Limited coverage on ontology design and development
UPON	Evolutionary prototype	No	Yes	No	Application independent	Yes	Limited coverage on ontology design and development
XP.K	Evolutionary prototype	Yes	No	No	Application independent	No	Limited coverage on ontology development

SCIM have been synchronized with existing methodologies and standards with software representations for model elements. The engineering behind the SCIM is the intermediate representation of ontology milestone deliverables in terms of model elements. More of software engineering approach has been applied in the selection of model elements and techniques. For e.g, conceptual model is represented in Unified Modeling Language (UML). Formal model of ontology will be implemented in formal representation language (e.g. RDF, OWL). The hierarchy followed in SCIM methodology is stages, workflows, activities and techniques. Incremental approach has been applied among the stages and iterative approach is applied among the activities within specific workflows. Unlike other software engineering centric ontology development methodology such as UPON, SCIM focuses generic domain ontology development.

SCIM classifies the core ontology development process into four stages such as Planning, Conceptualization, Development followed by Implementation and deployment. SCIM uses UML to represent the conceptual model in intermediate representation. This will be an added benefit for software designers and practitioners who intend to develop ontology. SCIM follows the approaches of RUP (Kruchten, 2003) such as Cycles, Iterations, Phases and Workflows in workflow modeling. However activities and techniques under each workflow have been customized for ontology development.

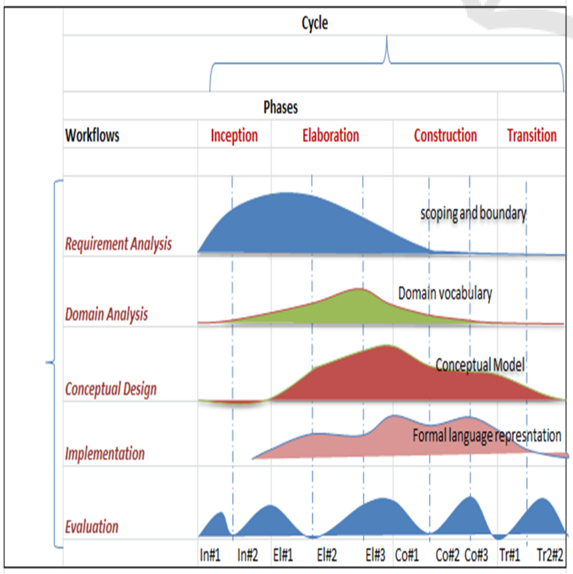

Figure.1: SCIM workflows and deliverables per phase.

The customized five workflows of SCIM have been mapped against that of RUP are requirement analysis, domain analysis, conceptual design, implementation and evaluation. Initial phase is focusing more on the pre development whereas the remaining phases focuses on the development and post development of ontology respectively.Figure1 represents the customized framework of SCIM with an abstract level indication of deliverables per phase.

4 ONTOLOGY DEVELOPMENT LIFE CYCLE

Unlike Software Engineering Methodologies, most of the existing ontology development methodologies failed to define ontology development life cycle (ODLC) with the detailed coverage of employed techniques and methods. The order of execution of the ontology development activities are realized in the ODLC which specifies the set of activities for ontology development.

SCIM treats stages as the base for the sequence of workflows. In SCIM iterative approach has been proposed for the activities within the workflows. Nine software engineering related disciplines of RUP have been mapped to five exclusive ontology development workflows. These workflows were placed within the respective stages and subsequently activities are embedded within the workflows. Guidelines for various techniques to be applied in each activity mentioned as part of the ODLC. Since deliverables against milestones are measurable entities, details of deliverables are proposed against each workflow. Figure 2 illustrates the abstract view of the ODLC of SCIM.

4.1 Planning Stage

The ultimate goal of planning stage is to produce a semi-formal, formal or a mixed mode Ontology Requirement Specification (ORS). The extent of formalism varies as per the techniques applied for the preparation of ORS. In SCIM, a systemized approach with efficient agile techniques has been proposed for ORS preparation. Requirement analysis workflow is embedded in this stage with feasibility analysis as the sole activity. Unlike other development methodologies, SCIM proposes agile techniques to complete the activity embedded.

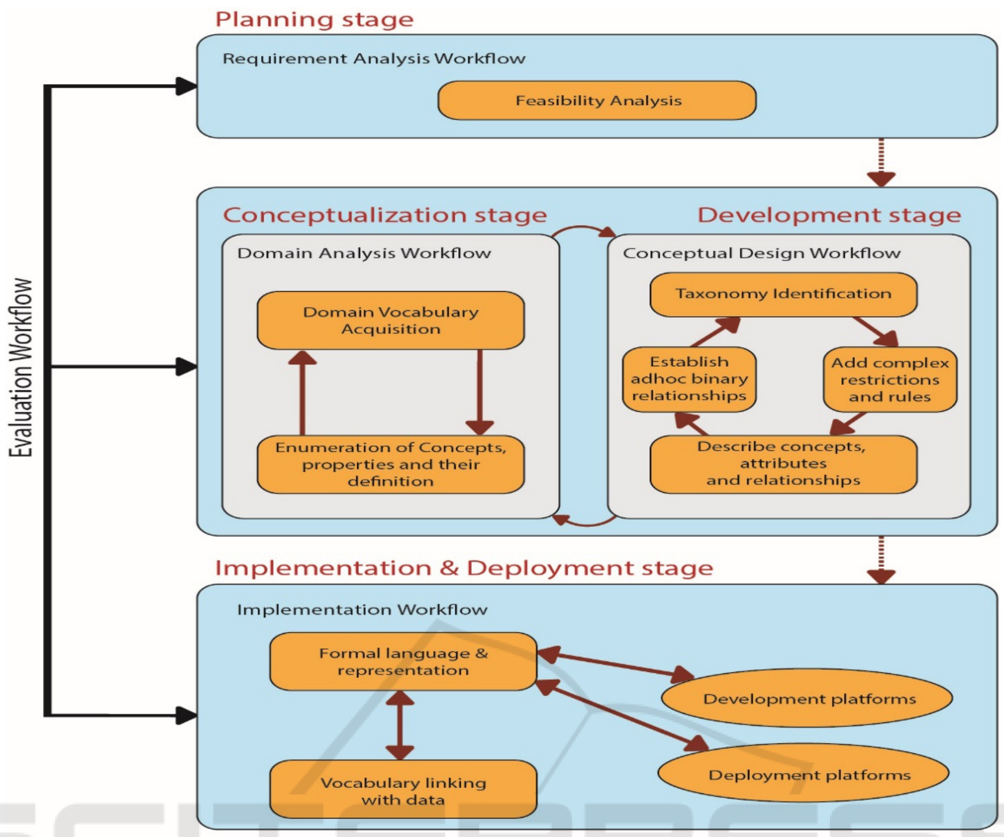
Planning stage
Requirement Analysis Workflow
Feasibility Analysis
Conceptualization stage
Development stage
Domain Analysis Workflow
Domain Vocabulary Acquisition
Enumeration of Concepts, properties and their definition
Conceptual Design Workflow
Taxonomy Identification
Establish adhoc binary relationships
Add complex restrictions and rules
Describe concepts, attributes and relationships
Implementation & Deployment stage
Implementation Workflow
Formal language & representation
Development platforms
Deployment platforms
Vocabulary linking with data
Evaluation Workflow

Figure 2: Abstract view of the ODLC of SCIM.

Requirements Analysis Workflow

Feasibility analysis is the core activity integrated to this workflow. Feasibility analysis activity covers requirement analysis by specifying the semantic requirements of the ontology users. The scope of this activity includes the identification of scope, purpose, boundary, intended users and ontology requirements. The techniques applied in this activity define the degree of formality of the Ontology Requirement Specification (ORS). Variation of agile technique followed by light weight software methodologies such as Joint Ontology Development (JOD) sessions will be conducted among users and Domain Expert (DE) to fulfil the requirements of this activity. Other agile techniques followed to complete the activity are storyboarding, usage of competency questions and modeling of the analysis use cases. Functional and non- functional requirements of intended domain ontology are defined during the JOD sessions. Functional requirements are the domain specific and they are ones referred to knowledge represented in the ontology whereas non-functional requirements are the generic referred to the aspects not specific to the domain concerned, but should have been satisfied by the ontology in its realization.

Scope and boundary definition includes the identification of first level key concepts. Defining business purpose of the ontology finalizes the categories of ontology users. During storyboarding, DE sketches the outline of sequential activities belonging to a scenario. Usage of competency questions ensures the availability of aspects that ontology must answer at its conceptual level. The competency questions can be used for the evaluation of this workflow at evaluation stage.

4.2 Conceptualization Stage

Conceptualization stage focusses on the identification of the domain concepts with their properties and the relationships. The goal of the conceptualization stage is to produce a conceptual model. This can be delivered in the form of a domain vocabulary in its minimal form. Domain analysis workflow has been mapped to this stage. The activities embedded in this workflow are domain vocabulary acquisition and the enumeration of concepts, properties and their definition. The deliverable against the completion of this stage will eventually be a glossary with clear definition of all the concepts with the details of properties belonging to them. Techniques proposed for the activities are *survey, extraction of concepts from documents, concept mapping*.

Domain Analysis Workflow

The development of an ontology starts from the definition of concepts related to the scope of the domain concerned. The activities embedded are *domain vocabulary acquisition* and *enumeration of concepts, properties and their definitions.* This workflow refines the requirements finalized in the requirements analysis workflow. A more generic domain vocabulary which includes a finer list of domain concepts along with the list of properties will be delivered by this workflow against the completion of activities embedded in it. The deliverable mentioned will eventually turns to the final glossary with the clear definition of concepts.

Domain Vocabulary Acquisition

Classes are the key elements of domain ontologies which are derived from the domain concepts after refinement. These classes are derived as a result of this activity. DE plays a vital role in this activity based on his/her best knowledge in the domain concerned. Domain vocabulary is built after detailed identification of domain specific concepts (classes). UML representation of design level classes with attributes is a kind of representation of concept structure. The more refined domain vocabulary eventually extending to a domain glossary is built on top of the concepts extracted from existing domain specific documents such as reports, policies, procedures and standards.

Enumeration of Concepts, Properties & Their Definition

This activity lists the concepts/classes and the properties of the concepts included in the refined glossary developed in the previous activity. Properties are the ones which provide structure to the concepts and this activity enumerates them which can be represented as atomic and complex. In formal representation languages, properties are of two major categories. They are data properties and object properties. Data properties describe the property with its value whereas object properties link an instance of the concept to another instance. This activity leads to the derivation of both data properties and object properties. In a nutshell, the completion of embedded activities will deliver a well refined glossary on the domain concerned for the usage of remaining workflows.

4.3 Development Stage

This stage is related to the conceptualization stage and therefore they are interconnected in SCIM framework. The goal of the development stage is the transformation of conceptual model defined into to a semi-formal model. The conceptual design workflow is mapped to the development stage. The core activities embedded in this workflow are *taxonomy identification, establish adhoc binary relationship*, *add complex restrictions and rules* followed by *describe concepts, attributes and relationships*. The activities stated above are generic in nature and they are applied to the development of domain ontology regardless of the domain concerned. UML modeling has been suggested as a technique for the semi model.

As part of conceptual model development, this stage emphasize on the core elements of ontology such as concepts, relations, attributes, instances, constants, axioms and rules. While completing the activity of taxonomy identification, an ontological structure to be built among the concepts is identified as the activity sets a concept hierarchy. The conceptual model development is not possible to realize in a sequential manner completely. However, it needs an order to work with for model representation.

Conceptual Design Workflow

This workflow is associated with the development of a conceptual model as its deliverable. *Taxonomy Identification*, *add complex restrictions and rules*, *describe concepts, attributes and relationships* and *establish adhoc binary relationships* are the core activities in this workflow.

Taxonomy Identification

Once after the glossary of terms gets a reasonable set of refined concepts, Knowledge Engineer (KE) shall to work on this activity. It defines the concept hierarchies of the ontology by building concept taxonomy. During this activity, concepts are organized into a hierarchical taxonomy based on the taxonomic relationships. Hence an ontological structure is set to refined vocabulary/glossary derived in the previous workflow. Concepts from the vocabulary/glossary are organized into hierarchical structure based on four formal relationships in SCIM. The four relationships used in this activity are

Subclass Of, Disjoint-Decomposition, Exhaustive-Decomposition and Partition.

Add Complex Restrictions and Rules

This activity identifies the formal axioms required in the ontology and describes them explicitly. In this activity, property constraints can be set which limits the set of possible values for a property. These include the value types, its allowed values, and the number of values that can/must have (cardinality).

Domain and range for a property can be specified by explicitly stating the possible value or enumerated values. Besides this, this activity identifies the rules required in the ontology. The instrument rule table has been used to record the credentials of rule applied.

Establish Adhoc Binary Relationships

This activity establishes the appropriate semantic/structural relationships between identified concepts with respect to the domain concerned. During this activity, the binary relationship targets to establish adhoc relationship between the same or different concepts from the conceptual model produced in the previous workflow. The adhoc binary relationships include inverse, transitive, symmetric and reflexive relationships between the concept taxonomy. For every adhoc binary relationship, KE must specify the name of relationship, between the source and target concepts with cardinality.

4.4 Implementation & Deployment Stage

This stage of SCIM suggests the usage of an appropriate ontology development environment (ODE). The selected ODE should support the formal language representation as the deliverable of this stage is the well evaluated formal language coded ontology. Therefore SCIM recommends an ODE with formal language code generation facility against the conceptual model. To ensure the correctness of the aspects applied in ontology development, ODE uses the reasoner. Implementation workflow has been mapped against this stage with necessary activities. *Formal language representation* and *vocabulary linking with data* are the activities integrated with the Implementation stage.
A sequential flow among stages has been proposed for SCIM has been defined in the life cycle. Set of workflows defined briefed in the next session with integrated activities.

Implementation Workflow

The implementation of ontology requires an ontology development environment with the capability of formal language code generation. Protégé has been proposed by SCIM. During this workflow, encoding of the ontology with formal language takes place. Formal language representation and vocabulary linking with data are the two activities associated with the workflow mentioned. The expressive power and

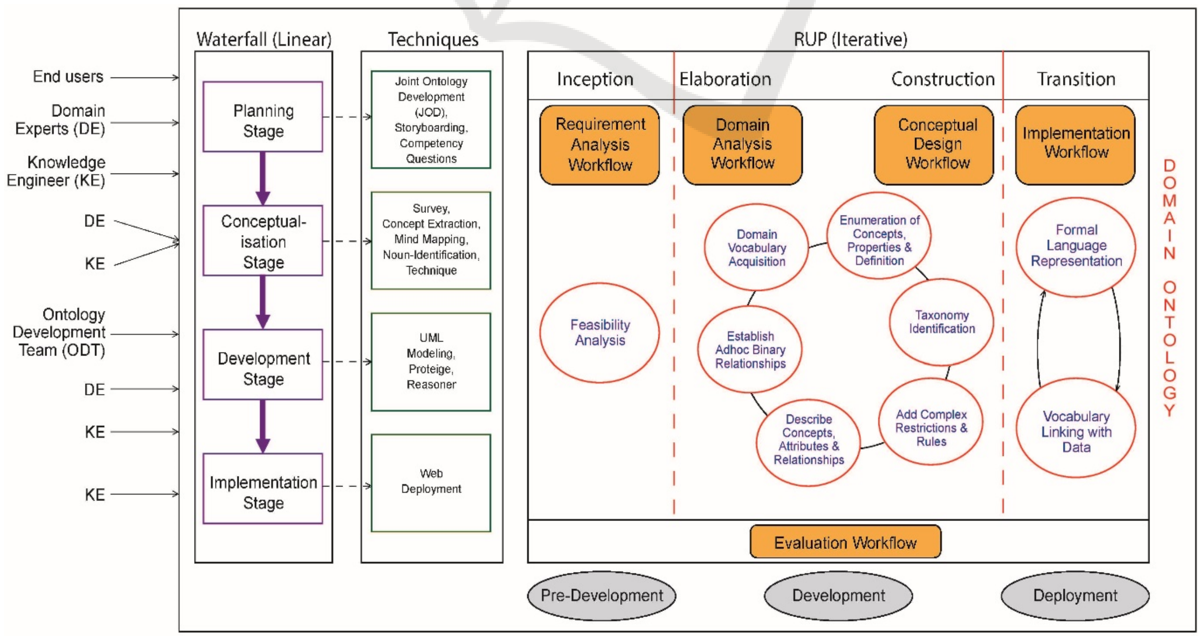
Waterfall (Linear)
Techniques
RUP (Iterative)
End users
Domain Experts (DE)
Knowledge Engineer (KE)
DE
KE
Ontology Development Team (ODT)
DE
KE
KE
Planning Stage
Conceptual-isation Stage
Development Stage
Implementation Stage
Joint Ontology Development (JOD), Storyboarding, Competency Questions
Survey, Concept Extraction, Mind Mapping, Noun-Identification, Technique
UML Modeling, Proteige, Reasoner
Web Deployment
Inception
Elaboration
Construction
Transition
Requirement Analysis Workflow
Domain Analysis Workflow
Conceptual Design Workflow
Implementation Workflow
Feasibility Analysis
Domain Vocabulary Acquisition
Enumeration of Concepts, Properties & Definition
Taxonomy Identification
Establish Adhoc Binary Relationships
Describe Concepts, Attributes & Relationships
Add Complex Restrictions & Rules
Formal Language Representation
Vocabulary Linking with Data
DOMAIN ONTOLOGY
Evaluation Workflow
Pre-Development
Development
Deployment

Figure 3: Final framework of SCIM.

computational complexity of the associated reasoning method and the level of acceptance are the core parameters being considered for choosing the formal language. Web Ontology Language (OWL)(Steffen Staab, 2010) is one of the best available formal languages to encode an ontology even in semantic web context. KEs are the key role players in this workflow as implementation delivers a physical model of the ontology as its output.

The final framework of SCIM illustrates the consolidation of stages, workflows, activities and techniques along with indication of the ontology development team representation. The framework is illustrated in Figure 3.

5 FINAL REMARKS

We propose a new methodology for ontology development with the complete coverage of methods and activities embedded in it. A well-defined ODLC has been proposed with the details of milestone and deliverables. Detailed literature review and analysis of exiting methodologies have been carried out to assess their limitations. This research attempts to address these limitations by proposing a novel methodology. Therefore, the proposed methodology supports state of the art technology to improve large scale ontology development. Further to that a validation will be carried to assess the accuracy and applicability of the methodology.

REFERENCES

Linkeddata.org. (2017). IJSWIS Special Issue on Linked Data: Call for Papers. [online] Available at: http://linkeddata.org/docs/ijswis-special-issue [Accessed 16 Apr 2017].

Blanco, C., Lasheras, J., Fernández-Medina, E., Valencia-García, R. and Toval, A. (2011). Basis for an integrated security ontology according to a systematic review of existing proposals. *Computer Standards & Interfaces, 33(4), pp.372-388.*

Iqbal, R., AzmiMurad, M.A., Mustapha, A. and Mohd Sharef, N. (2013) 'An Analysis of Ontology Engineering Methodologies: *A Literature Review', Research Journal of Applied Sciences, Engineering and Technology, 16(6). pp. 2993-3000.*

Gómez-Pérez, A. and Suárez-Figueroa, M.C., 2008. NeOn methodology: scenarios for building networks of ontologies. Poster and Demo, p.43.

Fernández-López, M. and Gómez-Pérez, A. (2002). Overview and analysis of methodologies for building ontologies. *The Knowledge Engineering Review, 17(02), pp.4-1-4-13.*

Natalya F. Noy and Deborah L. McGuinness (2001). Ontology Development 101: A Guide to Creating Your First Ontology. [online] Available at: http://protege.stanford.edu/publications/ontology_development/ontology101.pdf [Accessed 3 May 2016].

Fernandez, M., Gomez-Perez, A. and juristo, N. (1997) METHONTOLOGY, From *Ontological Art Towards Ontological Engineering.*

De Nicola, A., Missikoff, M. and Navigli, R. (2009).A software engineering approach to ontology building. *Information Systems, 34(2), pp.258-275.*

An Agile Development Methodology for Knowledge-Based Systems Including a Java Framework for Knowledge Modeling and Appropriate Tool Support. (2002). Ph.D. *Universität Ulm, Fakultät für Informatik, Abteilung Programmiermethodik und Compilerbau.*

Kruchten, P. (2003). The Rational Unified Process: An Introduction. 3rd ed. Boston, USA: *Addison-Wesley Longman Publishinh Comp.*

Steffen Staab, R. (2010). *Handbook on Ontologies.1st ed. Springer Berlin Heidelberg.*