A Method and Framework for Domain Knowledge Assisted Requirements Evolution (K-RE)

Preethu Rose[#1], Manish Kumar, [#2], Nirav Ajmeri[#3], Manas Agrawal[#4],V. Sivakumar[$5] and Smita Ghaisas[#6]

[#]Tata Research Design and Development Center
A Divn of Tata Consultancy Services
54 Hadapsar Industrial Estate,
Pune 411013, India
[1,2,3,4,6] {preethu.rose, manish9.k, nirav.ajmeri, manas.agrawal, smita.ghaisas}@tcs.com
[$]Insurance Industrial Services Unit 1(Insurance ISU1)
Tata Consultancy Services
188 Lloyds Road, Chennai - 600 086, India
[5] vs.sivakumar@tcs.com

***Abstract* - Domain knowledge edge is crucially important in the requirements definition. Requirement analysts are not necessarily domain experts and domain knowledge in an organization is not easily visible and accessible to them. The increasing complexity of software requirements coupled with pressures to reduce time to market have underlined the necessity for a structured mechanism to help the requirement analyst 'jump-start' projects (re)using domain knowledge.**

Most Requirements Engineering (RE) methods treat the requirements engineering exercise as something that begins from nothing and assume a 'clean slate' approach which outlines a series of steps to define, analyze, specify and validate requirements collaboratively with relevant stakeholders. We propose a method and framework to enable Knowledge assisted Requirements Evolution (K-RE). K-RE starts with a seed requirement specification. The seed contains structured domain knowledge as represented by core elements such as business events, actions and decisions (as captured in business processes), constraints, and analysis patterns derived from various resources. Each time a new software application is to be developed; we start with this seed specification and 'evolve' it by way of altering and/or adding to the core to get to the final requirement specification. This is done in a semantically assisted way. The semantic assistance comes from ontologies that can be created, maintained and evolved collaboratively and a context-sensitive alert mechanism that provides online alerts as a requirement analyst evolves her specification from the seed. Each new exercise of requirements definition thus, becomes an evolution of a pre-existing structured domain knowledge base tailored to suit specific projects.

***Keywords*—Domain knowledge, collaboration, ontologies, semantic web, web 2.0.**

I. Introduction

Domain knowledge plays an important role in defining requirements of high quality. Harnessing and reusing available domain knowledge becomes especially important in the software development context when multiple, geographically dispersed teams work together to build large-scale software applications. To facilitate this, we need to structure domain knowledge elements in such a way that their presence is readily visible to the requirement analysts and details are easily accessible. The structure should let a user e.g. a requirement analyst work effortlessly and contribute to perceptible productivity gains and quality of artefacts to be produced. Moreover, the structure should be amenable to evolve and improvise in terms of content and currency of knowledge.

To address this need, we have developed a framework that starts with a seed requirement specification in place that can be evolved into one that suits specific project needs. We term this approach as Knowledge assisted Requirements Evolution (K-RE).

A requirement analyst using K-RE is presented with core business processes and sub-processes relevant to realize the selected feature. In consultation with the customers and/or their Subject Matter Experts (SME), she can alter/delete the existing process steps, add new steps or change the sequence of steps to meet his project needs. As she carries out these tasks, she is presented with an assisting mechanism that helps her complete his specification by presenting other complementary knowledge elements. For instance, business rules and partial domain models that are relevant to the feature are made visible for her to select from. Missing elements in her specification are highlighted. Additionally, the text she enters while defining new/ altering existing process steps is parsed for detection of new business terms. The detected terms are matched with in-built ontologies and assistance is provided on aspects such as (1) synonymy of terms, most commonly accepted terminology in a domain additional terms that complement detected terms in a given context (2) Complementary and conflicting nature of requirements elements that he selects from the domain seed and (3) Inconsistencies, ambiguities and completeness in a requirement specification. She can generate structured requirement documents and view domain models from the specifications and refine them iteratively. Each new exercise of requirements definition thus, becomes an evolution of a generic structured domain knowledge base tailored to suit specific projects instead of a 'clean slate' approach. Hence the term Knowledge Assisted Requirements Evolution (K-RE) as opposed to the clean slate Requirements Engineering (RE). The domain contributor and curator roles are provided so that that selection and refinement of knowledge are enabled and currency of knowledge is maintained based on expert group discussions and voting.

In this paper, we present the details of our work on (1) Structuring of domain knowledge (2) Assistance to use the knowledge and a (3) A preliminary evaluation of the approach.

II. K-RE Framework

This section introduces the K-RE approach and presents details of its model and dynamics and a usage illustration.

A.An Overview

In the K-RE framework, the seed requirements specifications are represented in the form of instances of ontologies which represent concepts in a domain[1]. Semantic assistance is based on three types of ontologies and inference rules.

Based on the environmental parameters selected by the requirement analyst, a suitable seed is made available to him. For example, an analyst working on Life Insurance domain in Asia-pacific region for the customer ABC Insurance gets a seed different from an analyst working on same domain but in Europe region for another customer XYZ Insurance. Here, we address the fact that in globally distributed software development, different stakeholders in different geographies may have different terminologies, preferences or different laws of land.

K-RE provides a semantic assistance to the analyst while evolving over the seed specification. The generic requirements definition assistance is based on the rules defined in the method published elsewhere [2 - and references therein]. The domain specific assistance is provided on aspects such as synonymy of terms, most commonly accepted terminology in a domain, additional terms that complement detected terms in a given context, meaning of a term in a given context, relevant business rules in the selected geography, customer-specific business policies, interactions of the selected domain with other domains etc. Domain ontologies embody this information. Lexical decomposition techniques [3] are used to resolve requirements descriptions (input by users) into constituent terms. Each detected term acts as pointer to concepts in the domain ontology. For example, our domain ontology (detailed in section 2.2) contains the 'synonym' relationship among certain terms. If the analyst uses a synonymous term, she is prompted to replace it with the most commonly used terms. Thus, if the analyst writes 'Verify *customer's details and send notification to the insured',* the K-RE framework will prompt her that '*Customer*' and '*Insured'* are synonymous terms. Similarly if a term such as 'Adjudicator' or 'Middle person' is used in the specification, there will be a prompt saying 'Arbitrator' is the most agreed upon term. The details of different ontologies and inference rules are described in section 2. Figure 1 shows the steps in our method.

1. The analyst selects the environmental parameters
   1.1 Select Domain
   1.2 Select Corresponding Line of Business
   1.3 Select Geography
   1.4 Select Customer
   1.5 Select Type of Project
2. Based on above selection he is presented with features.
3. Select sub-processes corresponding to the features. Edit steps, add new steps or delete some if necessary. Similarly work with corresponding business rules, policies,
3. Review steps in the sub-process
4. Interview stakeholders
5. Edit steps if necessary (add, update or delete)
6. Look for missing requirement elements identified by K-RE alerts
7. Confirm semantic correctness of requirements by resolving inconsistencies identified by K-RE.
8. Do automated and manual verification and validations
9. If necessary, Go to step 4 and repeat the subsequent steps to refine requirement specification
10. Generate final requirements specification and models

Fig. 1. Requirements definition using K-RE

The collaborations between various stakeholders are facilitated on a web2.0 platform. The architecture is selected because requirements definition is a highly interactive process and web2.0 provides architecture of participation.

B.The K-RE Model

In this section, we present the structure of knowledge elements and the logical formalisms for semantic assistance. The following ontologies comprise the foundation for the knowledge structure. The 'instances' of these ontologies form the knowledge base from which a domain knowledge seed can be derived and evolved.

- Environmental Context Ontology

This ontology is designed to capture the environment in which software requirements are to be defined. For example, a requirement analyst may want to capture requirements for a Claims module of a Life Insurance application for a customer ABC Inc. In the Asia-Pacific geography. The concepts, 'Actor', 'Action', 'Domain', 'LineofBusiness', 'Customer', 'Geography', are abstractions used to capture the information.

- Requirements Definition Ontology

The domain seed that we present to the requirement analyst is built around abstractions that capture requirement elements such as features, business processes and sub-processes, business rules, use cases and business entities. The Requirements Definition ontology provides for abstractions that let one capture and organize requirements in terms of these elements and their relationships.

- Domain Ontology

The domain ontology provides abstractions to capture the essence of the problem domain. For example in event of death of a policyholder, a beneficiary may submit a claim request. The abstractions such as 'BusinessEvent', 'BusinessType', 'Party', 'BusinessAction' let one capture this information.

C. Dynamics of the K-RE Framework

K-Re framework facilitates evolution of the domain seed into a project-specific requirement specification by providing a semantic assistance based on the knowledge base (developed using RDF-OWL schema). This is achieved by employing the 'Bridge classes' and inference rules written in the Semantic Web Rule Language (SWRL) that we explain next. The 'Bridge classes' specify semantic mappings of conclusions drawn from one ontology to elements of another ontology. For example, the actor (requirement analyst) performs functions like 'Select domain', 'select geography' etc. Based on the selection, the K-RE framework draws logical conclusion on what modules should be presented to the user. If, he has selected 'Insurance', 'Life', 'Asia', 'ABC Insurance' and K-RE presents him with the modules like 'Claim',' Reinsurance' inferred by the Bridge classes. Figure 2 shows representative example instances of the three ontologies and bridge classes that traverse them.

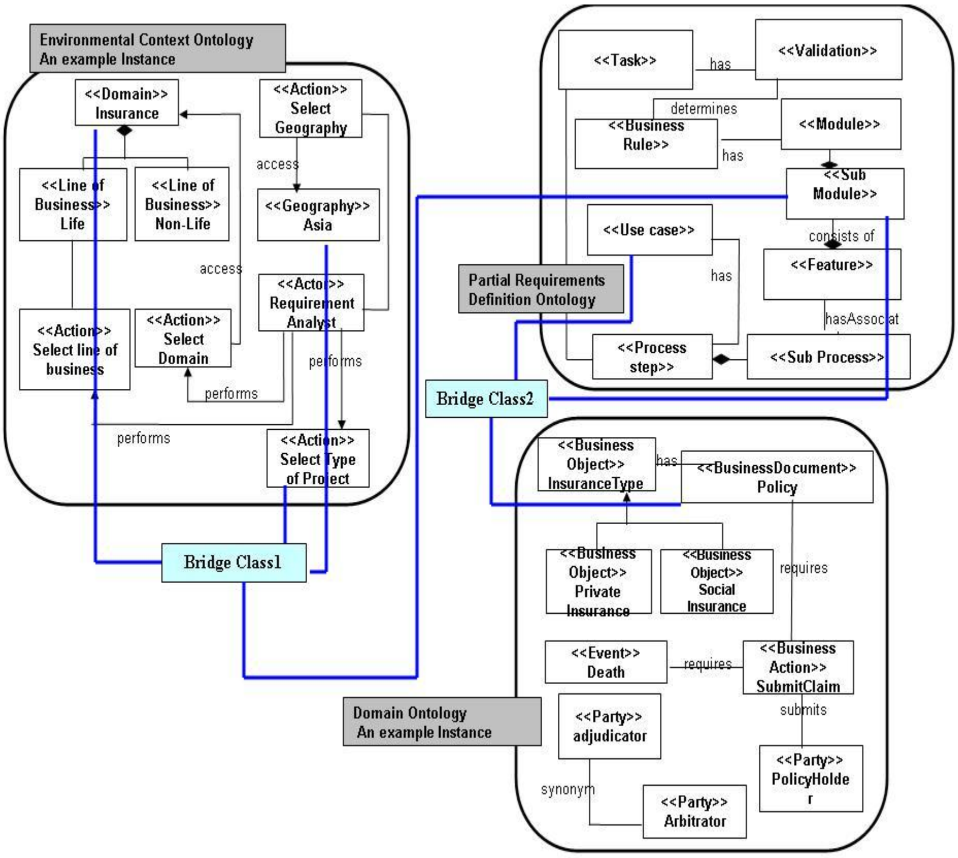

Fig. 2. Example knowledge base instances and bridge classes that refer to them for context-specific guidance.

Also if the requirement analyst selects conflicting features (such as 'Claim intimation for death due to unnatural cause' together with 'Document waiver management'), they traverse the ontologies, sense rules that specify the conflicting nature of the features and provide an alert stating so. Figure 3 shows an example of the semantic guidance using the inference rules in K-RE.

Action

The analyst may select conflicting features to be included in requirements specification. "Claim_*Death_due_to_unnatural_cause_Intimation_and_Booking"* and *"Document_ Waiver_ Management"*.

Conflicting Rule(s)

When a claim is made for death due to unnatural cause , document waiver is not be permissible.

SWRL Human Readable Format

RequirementsAnalyst(?X) ^ Feature("Claim_*Death_due_to_unnatural_cause_Intimation_and_Booking*") ^ Select(?X, "Claim_*Death_due_to_unnatural_cause_Intimation_and_Booking*")
⟶ isSelected("Claim_*Death_due_to_unnatural_cause_Intimation_and_Booking*" , true)

RequirementsAnalyst(?X) ^ Feature("*Document_ Waiver_ Management*") ^ Select(?X, "*Document_ Waiver_ Management*") ^
isSelected("Claim_*Death_due_to_unnatural_cause_Intimation_and_Booking*" , true)
⟶ isSelected("*Document_ Waiver_ Management*" , false)

Result

The analyst is warned about the conflicting nature of the features he has selected "Claim_*Death_due_to_unnatural_cause_Intimation_and_Booking" and "Document_ Waiver_ Management"* together.

Fig. 3. Dynamics of K-RE framework

D. Usage illustration- A case study

This case study illustrates application of K-RE to the Insurance domain. We focus only on the functional requirements definition in this paper. We do not discuss the details of capturing other types of requirements though these are supported in K-RE.

Role: Requirement analyst

Activities

- **Selecting environmental parameters**

Requirements change with changes in environment. For example, the business rules vary from one geography to another, the workflows would be different for one customer from that of another and so on. Therefore, it is necessary to know the environment in which the proposed application is going to work before any seed specification or assistance is presented to the requirement analyst. The requirements analyst selects her parameter set from a list of available parameters. Table I shows an example.

TABLE I: ENVIRONMENTAL PARAMETER SELECTION

Parameter	Value
Domain	*Insurance*
Line of business	*Life*
Module	*Claims*
Sub-module	*Death claim*
Geography	*Asia*
Type of project	*Development*

- **Selecting Features to work on**

Once the selection is made, she is presented with the features based on selected parameter set. For example, some of the features corresponding to the selected parameter set in Table I are:

- Claim Intimation and booking
- Claim Scrutiny
- Waiver management

She can either select from the list of available features or add her own to the existing list. In our example, she selects to work on 'Claim intimation and booking'.

If the requirement analyst selects a feature 'Claim scrutiny' without selecting the feature 'Claim initiation', she will receive an alert stating that 'Claim initiation' is a precursor to 'Claim scrutiny'. If however, she selects the feature "Document waiver management" along with a feature such as 'Claim processing for death due to unnatural cause', she would be alerted about the conflicting nature of the selected features. (since document waiver cannot be considered by the insurer while processing claims related to unnatural causes such as accidents) She can decide about excluding either or at least make an informed decision while retaining them-if the project specifics dictate that both be included in spite of their conflicting nature.

- **Selecting and 'processing' domain knowledge elements from the seed**

She is presented with a seed requirements specification for 'Claim intimation and booking'. The seed contains core elements such as business events, actions and decisions (as captured in business processes). She can view the generic process steps in the selected feature. She can alter/delete the existing steps, add new steps or change the sequence of steps to meet her project needs. Similarly she can look up business rules/policies relevant to the features and include them in her specification in the as-is form or a form modified to suit her project context.

We use this business process description to identify use cases. The design-specific assistance helps in associating use cases explicitly with non-functional requirements, screens, test cases etc.

Example:

Process step: Review validity of claim
Actor(s) who performs the process step: Scrutinizer
Relevant business rule(s): The risk stated in the death claim must be covered under the policy
Associated use case: Check Policy Details

When she selects some of the business rules corresponding to the feature she is working on, she is alerted about the rules that are relevant to that feature and are not selected. She can view a report on the knowledge elements that she selected in the as-is form and those that she chose to add or modify.

- **Ensuring a commonly accepted terminology**

The requirement analyst enters a new process step or a new business rule. K-RE platform detects new business terms/ key phrases within the process step .For example, if the term 'Customer' appears in the process step, and prompts that 'Insured' is a commonly accepted term. She can either decide to replace 'Customer' by 'Insured' or retain it as it is.

- **Making relevant clusters of terms or 'near neighbors' visible upon detecting a given business term**

She enters the step 'Check if claim is duplicated'. K-RE detects the term 'Claim' and presents the related terms such as 'Policy', 'Vehicle' and their associations with 'Claim'. Depending on the project context, she can add some/ all/ none of these terms and their associations to the glossary, domain model.

- **Identifying contextual meaning of a term**

The analyst uses the term 'Ceding company' in some business rule or process step. She is alerted that 'Ceding company is same as Insurer' since the insurer has applied for reinsurance. (This would be inferred from the relationship between the concepts Insurer and Reinsurance in the underlying insurance domain ontology). She can decide to replace the term 'Ceding company' with the term 'Insurer'.

- **Parsing text for detecting new business terms and model elements**

She enters a new business rule or a process step to her specification. If these do not exist in the domain ontology as concepts or synonyms thereof, they are identified as new terms and displayed. She can add these terms to the glossary and as business entities to the domain model that she wants to develop. The newly added terms appear only in her specific instance of requirements specification and do not automatically reside in the knowledge base.

- **Identifying interactions with external domain(s)**

Based on the selected features, the requirement analyst can view possible interactions with other domains and identify entry points into relevant processes. For the 'Death claim intimation', examples of such domains and (entry points thereof) include 'Healthcare (e.g. for reviewing post mortem reports after a claim is booked) Legal (e.g. for litigation management if the payment calculation is challenged in court of law), Banking (e.g. for payment processing after a claim is approved).

- **Semantically enabled collaboration**

During each of above actions, she can start discussions in the form of informal chats on the selected knowledge elements with her colleagues, experts and seek their opinion on her selections from and refinements on the seed specification. She can post topics for discussions on semantically enabled forums and subscribe to alerts when others post their opinions on topic of her current interest.

For example if she selects the following rule to be included in her specification:

'If no. of years of premium paid from the date of commencement is equal to 4 years, then policy acquires paid up value'

But she is not sure if this is valid in India, she can start a forum to discuss this with experts. Upon initiating a forum she will be presented with a set of relevant posts available on the topic. For example, she can view posts related to validity of rules for Life Insurance, Rules for ABC Inc, Rules for India, posts by other experts who contributed Life Insurance rules, rules regarding related terms such as 'date of commencement' premium and select the most suitable thread of discussions in terms of topic, author geography and so on. She can start an entirely new forum as well, if none of the presented ones match her need.

- **Generating and refining artifacts iteratively**

The requirements analyst can generate structured requirements specification documents intermittently. She can view process maps and partial domain models using third party modeling tools. Figure 4 shows a partial process map generated from textual specification for 'Claim intimation and booking'. The analyst can either work on the 'text' or 'diagram' and import/ export to /from either format. This helps in refining artifacts incrementally.

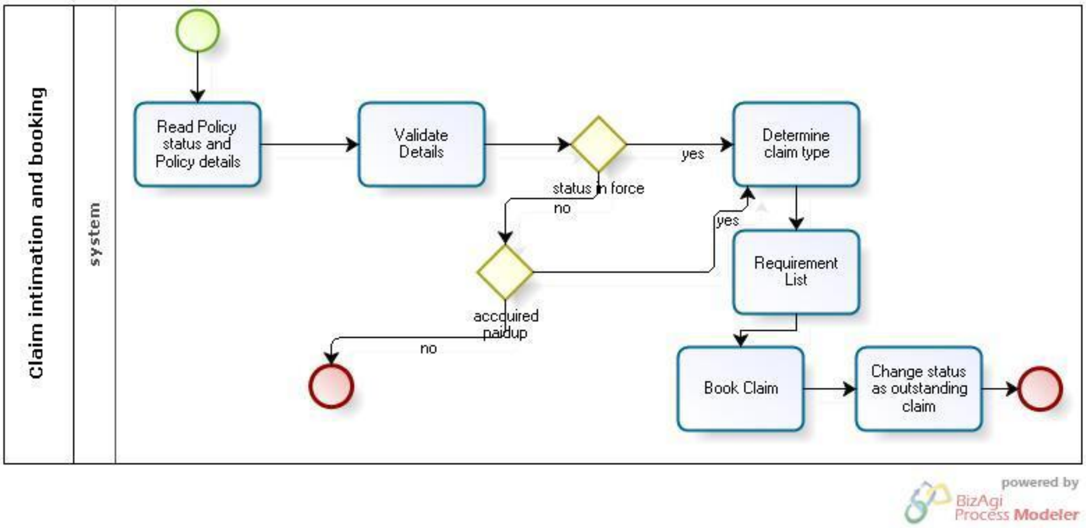

Fig. 4. Example of a partial process map generated from text specification: 'claim intimation and booking'

Starting with a generic seed specification for 'Death claim process', we can thus evolve a specification that suits a specific project. The seed contains a business-process centric view of structured (Insurance) domain knowledge. The evolution is an assisted exercise that helps in adding to or modifying the seed by providing context-sensitive help to a requirement analyst.

The outline of the additional two roles is provided below. We do not discuss the details pertinent to these roles in this paper.

Role: Domain contributor

- She can view existing knowledge elements- for example business rules pertinent to a 'Claims handling' process.
- She can add new elements such as new processes, steps in an existing process, business terms in a glossary, rules and policies
- She can invite discussions and opinions from other experts
- She can submit her contributions and look up status of her submission

Role: Domain curator

- Selecting and refining knowledge
- She can view submissions from domain experts
- She can modify/refine the elements if necessary
- She can select to accept or reject submissions
- She can invite discussions and vote on submissions if necessary

She can finalize the elements that should reside in the knowledge base.

III. EXPERIMENTAL EVALUATION

We have chosen three of the IEEE 830 standard [4] as the parameters to measure the effectiveness of K-RE. We take Requirements Definition Ontology and Domain Ontology as basis for measuring these properties.

A.Parameters

- Completeness

We have chosen two requirement elements viz., Business processes and Use cases for this illustration.

Let us define:

R_p = {r|r is the process related requirement elements identified (excluding the seed)}

R_{pm}= {r'|r' is the process related missing requirement elements}

R'_{pm}= {r''|r'' is the process related missing requirement elements actually removed}

R_u= {or| ur is the use case related requirement elements identified (excluding the seed)}

R_{um}= {ur'|ur' is the use case related missing requirement elements}

R'_{um}= {ur"|ur" is the use case related missing requirement elements actually removed}

C_b is the completeness measured before corrections.

C_a is the completeness measured after corrections.

E_{com} is the overall effectiveness. Therefore,

$$C_b = \frac{\{|R_u| + |R_p|\} - \{|R_{pm}| + |R_{um}|\}}{|R_p| + |R_u|} \times 100$$

$$C_a = \frac{\{|R_u| + |R_p|\} - \{(|R_{pm}| + |R_{um}|) - (|R_{pm}| + |R_{um}|)\}}{|R_p| + |R_u|} \times 100$$

$$E_{com} = C_a - C_b$$

- Consistency

There are two different aspects that we have measured in Consistency. First one is related to concepts like feature selection, rule selection from the domain SEED in K-RE. Second one is related to consistency during changes in requirements. For example, let us consider that the analyst is presented with following business rule in the seed: *'When policy certificate is lost, if claim payout is less than $ 10,000 then an indemnity letter is required'.* The corresponding validation for the rule is: *'Check if claim payout amount is less than $10,000'.* By interviewing the stakeholders, he finds out that the amount mentioned in rule should be in fact *'$15,000 instead of $10000'.* He makes the change in the rule accordingly. The K-RE framework will at this point detect this change and alert him that the corresponding validation should also be changed.

Let us define:

R = {r |r is the requirement elements identified}

I = {i | i is the inconsistencies identified}

I_r = {i_r | i_r is the inconsistencies removed}

I' = {i' | i' is the inconsistencies identified in changed requirements}

I'_r = {i'_r | i'_r is the inconsistencies removed in changed requirements}

Con_b = Consistency measured before corrections

Con_a = Consistency measured after corrections

E_{con} is the overall effectiveness. Therefore,

$$Con_b = \frac{|R| - \{|I| + |I'|\}}{|R|} \times 100$$

$$Con_a = \frac{|R| - \{ (|I| + |I'|) - (|I_r| + |I'_r|) \}}{|R|} \times 100$$

$$E_{con} = Con_a - Con_b$$

- Unambiguity

We measure unambiguity based on the number of times K-RE identifies ambiguity in requirements description. For example, if the description contains a term (Adjugator) which is not commonly used in a domain. K-RE prompts the ambiguity and suggests the suitable term- Arbitrator.

Let us define:

R = {r |r is the requirement elements modified, added or deleted from 'seed'}

A = {a | a is the ambiguity identified}

A_r = {a_r | a_r is the inconsistencies removed}

$Uamb_b$ = Unambiguity measured before corrections

$Uamb_a$ = Unambiguity measured after corrections

E_{uam} is the overall effectiveness. Therefore,

$$Uamb_b = \frac{|R| - |A|\}}{|R|} \times 100$$

$$Uamb_a = \frac{|R| - (|A| - |A_r|)\}}{|R|} \times 100$$

$$E_{uam} = Uamb_a - Uamb_b$$

Additionally, we also define *SEED correctness***.** This parameter is used to measure the correctness of 'seed' presented to the analyst. It is based on how many requirements are unchanged by the analyst. It means those requirements are useful in a format presented to him.

Let us define:

R = {r |r is the seed requirements presented}

R_m = {r_m | r_m is the seed requirements modified}

R_d = { r_d | r_d is the seed requirements deleted}

R_a = {r_d | r_d is the new requirements added to seed}

R_u = {r_u |r_u is the unchanged seed requirements}

Cor = Correctness of seed

$$|R_u| = |R| - \{|R_m| + |R_d|\}$$

$$Cor = \frac{|R_u|}{|R| + |Ra|} \times 100$$

B.Experiment Details

In this experiment, we evaluated our approach with 5 student interns working in our organization. We asked the students to use K-RE in the following way. They were provided with a set of documents we received as "complete" requirements specifications from 2 teams of requirements analysts in our organization. The students were to use the steps illustrated in Section D (Usage illustration- A Case Study) and use contents from the documents. They were to record the gaps detected by K-RE and close them in consultation with the requirement analysts afterwards. The requirement analysts were able to provide inputs to close these gaps either by themselves or upon consulting their clients. It was interesting to note that these were not documented earlier however. We present results from 2 representative experiments here. We have considered two different lines of business in insurance domain namely Claims processes in Life and P&C. The findings obtained from the experimental results are as detailed in tables I, II, III, IV.

TABLE II: COMPLETENESS

	R_p	R_{pm}	$R`_{pm}$	R_u	R_{um}	$R`_{um}$	C_b	C_a	E_{com}
Exp1	20	8	6	12	10	8	43.7	87.5	43.8
Exp2	27	13	9	20	12	10	46.8	87.2	41.6

TABLE IIII: CONSISTENCY

	R	I	I_r	I`	$I`_r$	Con_b	Con_a	E_{con}
Exp1	184	28	19	30	23	68.5	91.3	22.8
Exp2	130	19	15	18	12	71.5	92.3	20.8

TABLE IVII: UNAMBIGUITY

	R	A	A_r	$Uamb_b$	$Uamb_a$	E_{uam}
Exp1	184	27	23	85.3	97.8	12.5
Exp2	130	28	25	78.5	97.7	19.2

TABLE IV: SEED CORRECTNESS

	R	A R_m	R_d	R_a	R_u	Cor
Exp1	152	112	3	32	142	75.1
Exp2	120	S20	5	10	95	73.1

It is interesting to note that even after using the requirement specifications "completed" by the requirement analysts, the gaps in completeness were in the range of 40-45%, incompleteness was in the range of 20-25%, ambiguity ranged between 12- 20%. About 70 to 75% of the seed specification was used without any changes. These results were shared with the requirement analysts and their concurrence on importance of detected gaps was obtained.

IV. RELATED WORK

The related work falls into two categories; one is methods/techniques to structure knowledge and the other is using it for requirements elicitation. Research in the first category leans heavily on development of ontologies and thesauruses [5-8] as a means of representing knowledge. Examples are the DAML ontology library and Word-Net respectively. Extraction of domain knowledge from existing requirements documents is demonstrated in [9]. We reuse these concepts in our approach in seamlessly integrated method for requirements definition. In the second category, examples of approaches use of electronic dictionary for domain knowledge [10] and RML representation of domain model [11] to assist requirements elicitation. Requirements Apprentice (RA) employs reusable templates [12] while PAORE [13] discusses a method for refining requirements using domain thesaurus. Similarly feature diagram is FODA [14] also resembles a domain thesaurus approach. These methods however do not have provisions for guiding requirements elicitation methodically or any assistance for improving the quality of artifacts. We enable the reuse of the generic structured domain knowledge by providing intelligent, context-sensitive assistance and evolving it into project-specific artifacts. In work done by Kaiya [15], requirements concepts are extracted using NLP techniques and the analyst is required to map them to the domain ontologies. The completeness or consistency of requirements specification is verified based on the extent of the mapping. In K-RE, apart from the Domain ontologies, we use generic Requirement Definition ontology as well as Environmental Context ontology as our reference. Moreover, the mapping is done in an automated way, the requirement analyst simply writes the requirements as he would write a Word document, which is the most natural way of working for him. Additionally, we use generic as well as domain specific rules that work on the ontologies and draw logical inferences that enable a context-sensitive guidance to the analyst. A web 2.0 based collaborative platform seamlessly incorporates the ontologies, the inference engine and the guidance engine. We have shown how these three aspects are used in a single web 2.0 based collaborative framework to increase the effectiveness of the specification.

V. Conclusions And Outlook

In this paper, we have presented a requirements definition method and framework that uses domain knowledge seed instead of a clean slate as a starting point. We have seamlessly incorporated various disparate techniques to build a framework that allows users to collaboratively define requirements and provides context-sensitive domain assistance. Additionally, it is possible to derive requirement models in the form of business process maps, partial domain models and use case models and structured requirement specification documents from K-RE. We find that this approach has the potential to improve several desirable properties in a requirement specification. This is brought out by our initial experiments wherein we used a specification that was claimed to be complete; and yet had several gaps and inconsistencies (or undocumented known facts) We realize that this approach will be largely dependent on the quality of domain seed that we are able to provide and that this would require a mindset change for a larger adoption in any organization.

Considering domain knowledge as “seeds” of requirements can potentially be a new Viewpoint in Requirements Engineering.

K-Re also bridges the democratic aspects of web 2.0 based platform and the semantic web concepts. While collaborative identification, discussion and definition of requirements facilitated by web 2.0 are valuable, we cannot entirely do away with moderation in such a highly specialized exercise. The assistance mechanism based on semantic web concepts serves as a moderating mechanism as well. The decision to modify requirements by acting as per the alerts or ignoring them is an informed one. The three ontologies provide pre-defined taxonomies to facilitate classification of requirement elements. Apart from the pre-defined taxonomies (which are built hierarchically top-down way), a user is at liberty to identify new elements. This constitutes the folksonomy which evolves bottom-up. If their usage in the community of practice (in this case stakeholders in requirements definition exercise) is substantial, the elements can be absorbed into the taxonomy.

References

[1] Nigel Shadbolt, Tim Berners-Lee and Wendy Hall, ”The Semantic Web Revisited”;http://eprints.ecs.soton. ac.uk/12614/1/ Semantic_Web_Revisted.pdf.

[2] Smita Ghaisas, Lecture Notes in Informatics (LNI) – proceedings of SENSE 09 Workshopband, Series of the Gesselshaft fur Informatik (GI), Bonn, 2009 , Vol 150, pages 297-308, edited by Munch J. and Liggesmeyer P

[3] M. Saeki, H. Horai, and H. Enomoto.”Software Development Process from Natural Language Specification.” In Proc. of 11th ICSE, pages 64–73, 1989.

[4] IEEE recommended practice for Software Requirements Specifications, 1998, IEEE Std.830-1998.

[5] DAML Ontology Library. http://www.daml.org/ontologies

[6] KAON Tool Suite. http://kaon.semanticweb.org/.

[7] Proc. of 2nd International Workshop on Evaluation of Ontology-basedTools. http://sunsite.informatik.rwthaachen.de/Publications/CEUR-WS//Vol-87/.

[8] WordNet: A Lexical Database for the English Language. http://wordnet.princeton.edu/.

[9] L. Kof. Natural Language Processing for Requirements Engineering: Applicability to Large Requirements Documents. In Proc. of the Workshops, 19th International Conference on Automated Software Engineering, 2004.

[10] K. Breitman and J.C.S.P. Leite. Ontology as a Requirements Engineering Product. In Proc. of RE03, pages 309–319, 2003.

[11] Sol J. Greenspan, John Mylopoulos, and Alex Borgida. On formal requirements modeling languages: RML revisited. In 16th ICSE, pages 135 –147, 1994.

[12] H.B. Reubenstein and R.C. Waters. The Requirements Apprentice: Automated Assistance for Requirements Acquisition. IEEE Trans. on Software Eng.,17(3):226–240, Mar. 1991.

[13] J. Kato, M. Saeki, A. Ohnishi, N. Nagata, H. Kaiya,S. Komiya, S. Yamamoto, H. Horai, and K. Watahiki.PAORE: Package Oriented Requirements Elicitation.In Proc. of APSEC2003, pages 17 – 26, 2003

[14] K. Kang, J. Lee, and P. Donohoe. Feature-Oriented Product Line Engineering. IEEE Software, 19(4):58 –65, 2002. See also Feature-Oriented Domain Analysis (FODA) Feasibility Study by K. Kang, Sholom Cohen, James Hess, William Novak, AS Peterson (1990)

[15] Haruhiko Kaiya, Motoshi Saeki.”Using Domain Ontology as Domain Knowledge for Requirements Elicitation”.