Product Knowledge Configurator for Requirements Gap Analysis and Customizations

Preethu Rose Anish
Tata Research Development and Design Centre (TRDDC)
Tata Consultancy Services
Pune, India
preethu.rose@tcs.com

Smita Ghaisas
Tata Research Development and Design Centre (TRDDC)
Tata Consultancy Services
Pune, India
smita.ghaisas@tcs.com

***Abstract*—Product knowledge plays an important role in identifying the requirements of the desired variant and configuring the existing product to the present needs of a customer. The success of a product-based business depends to a great extent on how efficiently and accurately the existing product knowledge is utilized for customization needs. Oftentimes however, product knowledge resides with few key individuals in an organization. In the absence of their involvement, project teams may redevelop product features unnecessarily, resulting in an effort overhead. Such overdependence poses a risk to projects. To identify the requirements for the variants accurately and efficiently, we need to have a thorough knowledge of the existing product features. In this paper, we discuss our work on representing product knowledge and reusing it in a Requirements Engineering (RE) exercise for a large project involving product customization. We present our experience from using the configurator for requirements gap analysis and customizations.**

***Index Terms*—Requirements configurator, Ontology, Product Line, Financial product suite, Configuring products, Knowledge assisted configuration.**

I. Introduction

The success of product-based business depends to a great extent on the clarity of customer requirements [1, 2, and 3]. Product configuration is the process of generating a variant from an existing product suite and additional product specifications for a given variant [22]. For an efficient product configuration; customer requirements have to be captured, analyzed, and compared accurately with the existing product variants in the product suite. Product knowledge plays an important role in this respect [1, 4, and 5]. Gap analysis is defined as a method by which customer requirements are compared with the existing product feature specifications in the product suite to arrive at a match or mismatch.

Requirements gap analysis is a core activity in a product configuration and customization exercise. A thorough knowledge of the existing product features in the product suite is essential to accurately identify the requirements for a new product. We can use the knowledge of the existing product features to compare it with the requirements of the new product, perform a gap analysis and use the analysis in customization of the product. In practice, gap analysis is often a manual activity; involving workshop sessions with stakeholders (for instance JAD sessions [14]). Also, since the knowledge resides with a few key individuals in an organization; teams depend heavily on these people. This presents a big risk associated with loss of knowledge if these people leave the organization. In the absence of their involvement, product teams may redevelop already existing features. This not only results in effort-overhead but also makes the whole exercise of requirement gap analysis subjective.

Our interactions with the product experts in our company revealed that manual analysis can unearth only the easily visible and accessible high level details in the product variants. At the high level, all products look similar and hence, during gap analysis, the products can be misconceived to be 100 % similar. The gap analysis done in the absence of adequate product knowledge entails a high probability of leaving out on the unobvious lower level product details. In the product hierarchy, it is often at the lower levels that the actual gaps exist. The unavailability of knowledge-assistance to uncover the deeper level details can make the whole exercise of gap analysis subjective and susceptible to errors and omissions. These concerns are representative of today's business scenario wherein there is a progressive shift from the traditional one-size-fits-all products to more customer-specific products [8-11].

In an earlier publication, we discussed our work on a product knowledge configurator to represent product knowledge so that it can be reused in RE exercises for large projects in the same domain [7]. The approach involves knowledge-assisted product requirements evolution from a knowledge base (KB). The KB consists of product knowledge elements such as features (risk-cover pairs), attribute groups, attributes, business rules and the associated business processes, sub processes and stakeholders. In collaboration with the customers and domain experts, the requirement analysts can use the configurator to perform requirement gap analysis.

We have used the configurator to conduct RE, specifically for gap analysis and configuration of product specifications in a large distributed Insurance domain project in Europe.

The rest of the paper is organized as follows: section II details the solution approach. Section III outlines the case study while section IV presents discussion and conclusion

II. Product Knowledge Configurator

In this section, we elaborate the ontology-based product knowledge configurator in detail. The conceptual model of the configurator has been introduced in our previous publication [7].

The ontologies are organized along three contexts: (1) Product Knowledge (2) Environment (3) Business domain. We use Product Knowledge Ontology to represent the Insurance product and its variants. It captures the product primitives in terms of (1) concepts such as features (risk-cover pairs), attribute groups, attributes, business rules and (2) associations between the concepts. Additionally, we capture associated business processes, sub processes and stakeholders as well. Table I provides some definition along with examples, in the parlance of the product under discussion.

TABLE I. Product knowledge ontology concepts and examples

Concept	Definition	Example(s)
Risk	Risk is the potential of losing something of value	*Permanent disability*
Cover	Cover describes the situations you are insured against.	*Cover against permanent disability*
Attribute Group	Attribute group is a set of logically coherent attributes	*Insured details*
Attribute	An attribute is a quality or feature regarded as a characteristic or inherent part of someone or something [20]	*For insured details, the attributes are: insured name, age, gender, occupation etc.*
Business Rule	Business rules are external constraints that are not under the control of the organization.	*Laws of land, domain specific policies*
Business Process	A business process is a collection of related, structured activities or tasks that produce a specific service or product for a particular customer or customers [21]	*Claims process, Litigation process, Adjudication process*
Sub Process	A sub process is a collection of related, structured activities or tasks within a process	*For the process Claim; Claims intimation, Claim booking are sub processes*
Stakeholder	Stakeholders are entities that are directly or indirectly impacted by the product under consideration	*Insured, Insurer*

The product knowledge is expressed and organized as an instance of the model. The Product Knowledge Ontology interacts with two other ontologies:

- ***Environmental Context Ontology*** which is used to capture the environmental parameters such as Domain (e.g. *Insurance*), Business (e.g. *Life*), Line of Business (e.g. *Group Life*), Product Line (e.g. *Personal*), Geography (e.g. *Europe*) and Company (e.g. *ABC Inc.*). These abstractions allow for representation of knowledge of the environment in which the product is to be deployed. This is important as customers may need to conform to environmental specific laws and may have different organizational policies. Thus, selection of the environmental parameters ensures that only the relevant product specification is made available to the requirement analyst and the analyst's view is not cluttered with any irrelevant information.
- ***Business Domain Ontology*** which is used to capture the intricacies of a given domain such as Insurance and therefore contains business concepts, their relationships and constraints. For example, consider the following scenario from Insurance domain – *For group life claims, in the event of the occurrence of a physical disability; the policyholder may submit a claim request*. The abstractions such as **BusinessEvent (***Occurrence of physical disability***)**, **Party (***Policyholder***)**, and **BusinessAction (***Submit a claim request***)** are used to capture this information.

Figure 1 show examples of the three ontologies depicted using the UML class diagram notation. It should be noted that we have shown only example concepts from the Business Domain and the Environmental Context Ontology to explain their mappings with the Product Knowledge Ontology. A few examples of mappings between the three ontologies are as follows:

- The **BusinessEvents** (e.g., *Policy Purchase*), **BusinessActions** (e.g. *Scrutinize Details*), and **BusinessDecisions** (e.g., *Policy Issuance*) in the Business Domain Ontology are together represented as **BusinessProcess** (e.g., *New Business*) in the Product Knowledge Ontology.
- **BusinessConstraint** (e.g. *Legislation#123*) in the Business Domain Ontology maps to **BusinessRule** (e.g., *Rule 1*) in Product Knowledge Ontology.
- **BusinessValue** (e.g. *Profit Margin*) in the Business Domain Ontology maps to **BusinessGoal** (e.g. *Optimize premium payout*) in Product Knowledge Ontology.
- **BusinessDocument (***e.g. Vehicle Policy***)** in the Business Domain Ontology maps to **Policy (***e.g. ABC Vehicle Policy***)** in Product Knowledge Ontology.

The semantic assistance is achieved by employing the "bridge classes" and inference rules written in the Semantic Web Rule Language (SWRL) [15]. The bridge classes specify

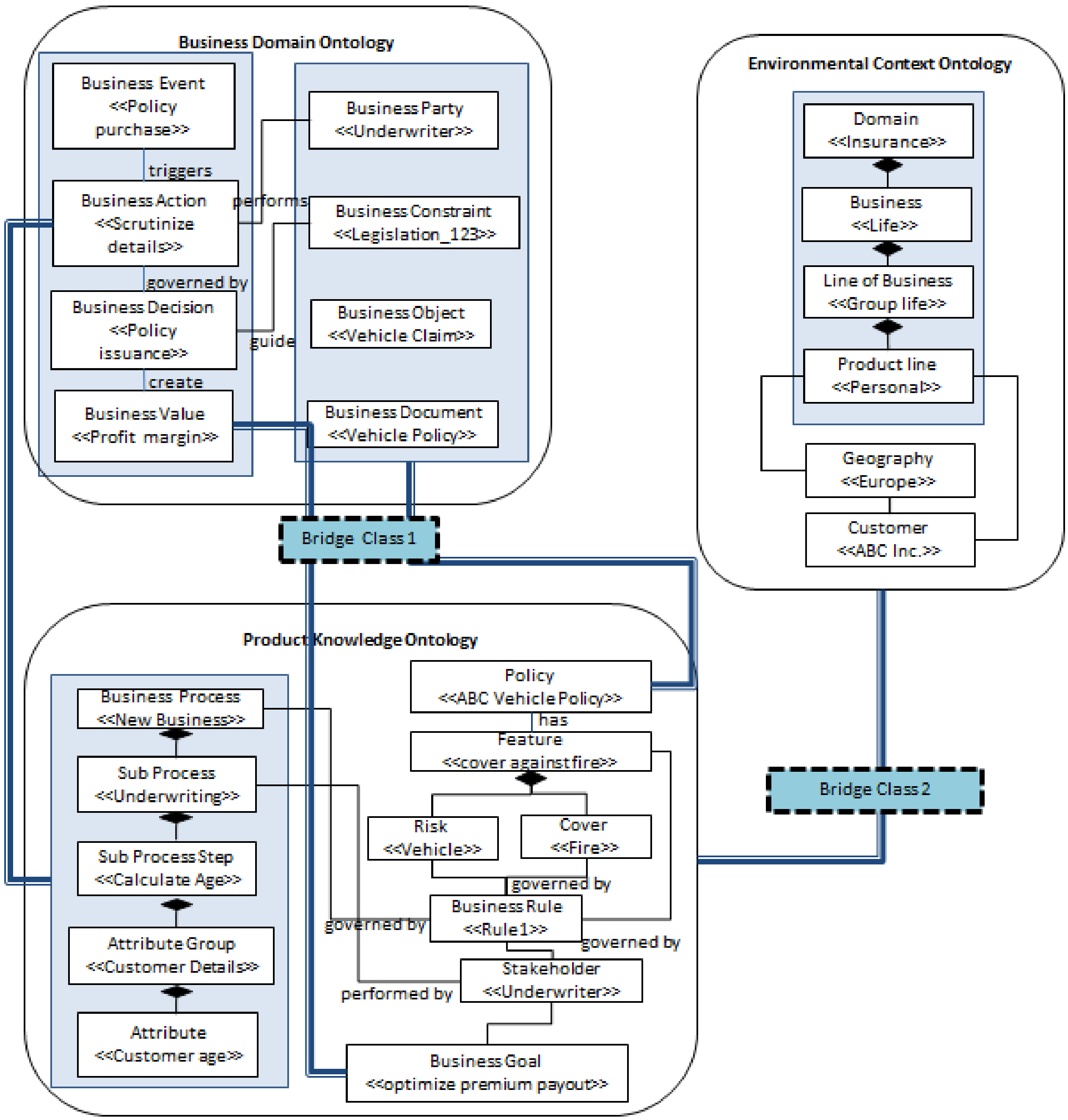

Fig. 1. Example instances of ontologies and bridge classes that enable context-specific recommendations

semantic mappings between the three ontologies. We define rules that refer the ontology instances and provide recommendations based on the integrated inference. The recommendations may be specific to a singular ontology or may span across the three ontologies when necessary, in response to the actions of the business analyst. For example, if the requirement analyst selects a set of environmental parameters specific to the customer such as ***Domain*** (e.g. *Insurance*), ***Business*** (e.g. *Life*), ***Line of Business*** (e.g. *Group Life*), ***Product Line*** (e.g. *Personal*), ***Geography*** (e.g. *Europe*) and ***Company*** (e.g. *ABC Inc.*), the product knowledge configurator would present her a suitable product variant (based on the selected parameters) from previously executed projects, if already available in the repository. Else, a generic specification is presented. The customer representative and the analyst would review this together to determine the point of departure (whether from a product variant or from the generic specification) for product configuration. The customer would then suggest additions /modifications as per their needs. The analyst would begin the process of configuring the generic or the selected variant of the product specification from the product suite to incorporate the customer's inputs. At this stage, the configurator would create a copy of the generic or the selected variant of the product specification and make it available in the workspace designated for the requirement analyst. As the analyst incorporates customer's inputs to create a variant of the product, she receives context-specific recommendations from the product knowledge configurator. The recommendations are facilitated by the underlying ontologies and the semantic rules that operate on it. For instance, when the representative adds a new ***Cover*** (*Permanent Disability)* upon suggestion by the customer, the configurator alerts her that the cover that she has added already exists in some other configured product (e.g. *Product A*) in the product suite. She is recommended to look up the associated artifacts with this Cover (*Permanent Disability)* in *Product A* such as the associated ***Rules*** (*A claim against permanent disability should be intimated within 1 week of its occurrence),* ***Processes*** *(Claim Process),* ***Sub processes*** *(Claim Registration),* ***Attributes*** *(Policy Id.)* etc. When she selects the suggested rule *(A claim against permanent disability should be intimated within 1 week of its occurrence),* she receives an alert to verify if this rule contradicts with some other rule *(All claims must be intimated within 30 days of its occurrence)* that already exists in the selected product specification**.** Similarly, if she modifies a ***Sub Process (e.g.*** *claim Registration)****,*** she is alerted about the impacted artifacts such as ***Sub process step*** *(e.g. Capture claim details),* ***Attribute Group*** *(e.g. Details for Claim intimation),* ***Attribute*** *(e.g. Cause of disability).* Upon completing the configuration as detailed above, the analyst can produce a gap analysis report from the configurator. The report highlights the 'evolution' of the product specification from its generic version to its configured variant. This information is critical to evaluate the amount of further customization that may be needed (if any).

In section III, we evaluate the product knowledge configurator as opposed to the traditional manual technique to identify the gaps that exist between the baseline version and the configured version of the product specification.

III. Evaluation in Industrial Setting

In this section, we present results of using the product knowledge configurator to conduct RE, specifically for gap analysis and configuration of product specifications in a large distributed Insurance domain project in Europe. The project under study involves providing group income protection services to customers.

We created a Group Life Insurance KB using the product knowledge configurator from existing Group Life Insurance product requirement specifications. The creation of KB involves (1) identification of structural details of the product requirement specification, (2) mapping of the document structure to the product knowledge configurator model, and (3) extraction of domain knowledge from documents and its representation in product knowledge configurator. In this paper we do not discuss the details of the process of creating the KB. For details, the reader is referred to our previous work [12, 13]. Table II lists the different tasks and effort involved in creating the KB.

The details pertinent to the size and content of the KB are given in Table III. The KB has the product specifications of 25 Group Life Insurance products. Seven domain experts were involved in analysis, standardization and organization of product specifications.

TABLE II. Tasks and Effort Involved in Creating Knowledge Base

Task	Task Type	Effort
Identification of structural details of the product requirement specification documents	Manual	4 person days
Mapping of the document structure to the product knowledge configurator model	Semi-automated	6 person days
Extracting and organizing knowledge in product knowledge configurator	Semi-automated	9 person day
Review of extracted knowledge elements	Manual	15 person days

The KB was validated for its correctness and completeness by the product manager. This was done by comparing the product specifications written in Microsoft word with the one populated in KB.

TABLE III. Product Knowledge Elements

Knowledge Element	Count
Risk	2
Cover	30
Attribute Group	150
Attribute	2750
Business Rule	961
Business Process	125
Sub Process	227

To evaluate the effectiveness of product knowledge configurator, we conducted an experiment to compare the manual gap analysis process routinely practiced by the project team and the one carried out using the product knowledge configurator. Six product specifications corresponding to customer-specific variants were included in the gap analysis experiment. It should be noted that one of the author (PRA) was a part of the team responsible for KB creation and manual gap analysis.

A. Traditional Gap Analysis Process

In the traditional gap analysis process, the vendor's business analyst team manually compares the configured specification (created by conducting a series of RE sessions with the customer) with the baseline product specification document to detect gaps and necessary customizations. The author followed the manual approach to perform the gap analysis.

During the experiment, the author recorded the following:

- *Total number of gaps identified by domain expert (G_{Total}):* These are the total number of gaps between the configured product specification and the baseline product specification.

- *Total number of gaps identified by the author manually (G_{Manual})*: These are actual number of gaps identified by the author manually by comparing the configured product specification and the baseline product specification.

B. Gap Analysis Using Product Knowledge Configurator

The product specification of the six products received from the customer was configured (using the approach as illustrated in section II above). The configurator generated a gap analysis report based on each of the six configured product specification and the KB specification.

During the experiment, the author recorded the following:

- *Total number of gaps identified by product knowledge configurator ($G_{PdtConf}$):* These are the total number of gaps identified by the product knowledge configurator.
- *Total number of relevant gaps identified by product knowledge configurator ($G_{PdtConfRel}$):* These are the gaps that were identified by the product knowledge configurator and considered relevant by the domain expert.

C. Effectiveness Parameters

In order to measure the effectiveness of gap analysis process using the product knowledge configurator and compare it with the manual approach, we computed precision and recall values based on the data obtained from the experiment.

Precision: We define precision as "percentage ratio of relevant gaps identified to the total number of gaps identified between the configured specification and the KB specification"

Precision (P_M) is the percentage ratio of gaps identified by the author that were considered relevant by the domain expert (to the total number gaps identified by the author)

Precision ($P_{PdtConf}$) is the percentage ratio of gaps identified by the author using the product knowledge configurator that were considered relevant by the domain expert (to the total gaps identified by product knowledge configurator). Thus,

$$P_{PdtConf} = (G_{PdtConfRel} / P_{PdtConf}) \times 100$$

We have assumed that precision (P_M) is 100 % because all the gaps that the author identify are correct. However, the author may or may not identify all the gaps. We have verified this logical assumption with domain experts.

Recall: We define recall as "percentage ratio of gaps identified to the total actual gaps that exist between the configured specification and the KB specification"

Recall (R_M) is the percentage ratio of the gaps identified by the author to the total actual gaps that exist.

Recall ($R_{PdtConf}$) is the percentage ratio of the gaps identified by the product knowledge configurator to the total actual gaps that exist. Thus,

$$R_M = (G_{Manual} / G_{Total}) \times 100$$

$$R_{PdtConf} = (G_{PdtConfRel} / G_{Total}) \times 100$$

D. Results

In this section, we present the results of the evaluation of the effective of the configurator in terms of precision and recall percentage. Table IV lists the parameters discussed in section III C above.

Precision

We computed the average precision of the gap analysis performed using the product knowledge configurator using the following formula:

$$P_{PdtConf} = (\sum G_{PdtConfRel} / \sum G_{PdtConf}) \times 100$$

$$P_{PdtConf} = 100\ \%$$

Thus, the average precision of gap analysis performed using the product knowledge configurator is 100 %

Recall

We computed the average recall of the gap analysis performed using the product knowledge configurator using the following formula:

$$R_{PdtConf} = (\sum G_{PdtConfRel} / \sum G_{Total}) \times 100$$

$$R_{PdtConf} = 92.2\ \%$$

Thus, the average recall of gap analysis performed using the product knowledge configurator is 92.2 %

We further computed the average recall of the gap analysis performed manually using the following formula:

$$R_M = (\sum G_{Manual} / \sum G_{Total}) \times 100$$

$$R_M = 48.05\ \%$$

Thus, the average recall of manual gap analysis is 48.05 %.

TABLE IV. GAP ANALYSIS EXPERIMENT RESULTS

Product	G_{total}	$G_{PdtConf}$	$G_{PdtConfRel}$	G_{Manual}
P1	7	7	7	4
P2	14	12	12	8
P3	11	8	11	3
P4	21	18	18	11
P5	16	15	15	7
P6	8	6	8	4
Total	**77**	**66**	**71**	**37**

The plot in Figure 2 depicts the precision and recall value computed for gaps identified in each of the product specification manually and using the product knowledge configurator.

E. Analysis

Average precision of the gaps identified between the configured product specification and the baseline specification when using product knowledge configurator was found to be 100 %. Average recall using the product knowledge configurator was computed at 92.2 %. Recall using manual approach was found to be only 48.05 %. As discussed earlier, we have involved the RE lead and Domain Expert in charge of the project to review the results of manual gap analysis and analysis done by the product knowledge configurator. The results indicate that there is a significant improvement in the recall percentage of gaps identified using the product knowledge configurator as opposed to the manual approach. This is because the product knowledge configurator makes the unobvious lower level product details easily visible and accessible which have a high probability of getting missed when the gaps analysis is performed manually.

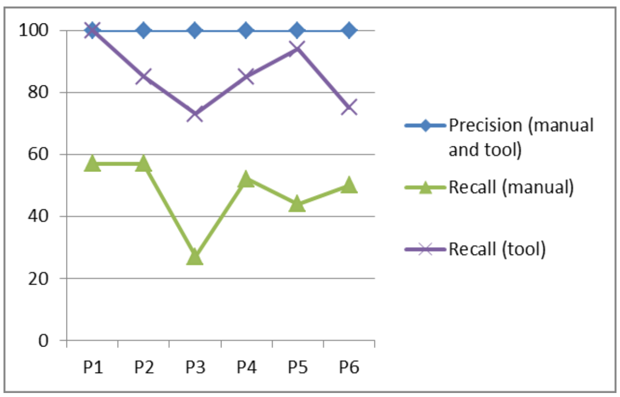

Fig. 2. Precision and Recall of gap analysis: manual and product knowledge configurator based

As an example, in Figure 3 we depict one such lower level product detail. A gap that existed in the business rules associated with Risk -> Cover -> Process -> Sub Process was omitted during the manual gap analysis due to the difficulties involved in identifying such fine grained details while manually comparing the two documents. However, when the gap analysis was done using the configurator, this gap was readily detected.

F. Threats To Validity

The following are the threats to validity:

- The recall percentage is highly dependent on the structure of a product specification desired by a customer. Each project follows its own documentation template and therefore the configurator may not be able to completely map the document structure to the configurator model. This may lead to some information loss while populating the data from the documents to the configurator. The recall percentage obtained in our experiment also indicates some amount of information loss. This dictates the need to further refine our model by testing it against different product specification structures reflected in the templates.
- The author conducted the experiments. As the author has been involved right from conceptualization and development of the product knowledge configurator, she was aware of the functionality and capability of the tool. It remains to be seen whether the results vary when the experiment is conducted by business analysts (who are the real users of the configurator).

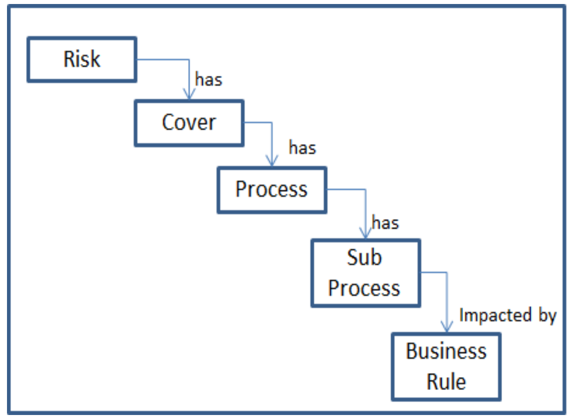

Fig. 3. An example of lower level product details

IV. Discussion And Conclusion

Work in the area of domain specific ontology-based frameworks for capturing and analyzing customer requirements has been reported in literature [2, 16-19]. All these works focus on either the *Manufacturing domain* or the *Production domain*. The product primitives of the Insurance domain however differ from those of the Manufacturing or the Production domain. This motivates the need for a knowledge-assisted framework for product configuration that can cater to the intricacies of Insurance domain. To the best of our knowledge, no work has been reported on ontology-based knowledge-assisted product requirements configuration in the Insurance domain.

In this paper, we present our work on a product knowledge configurator to represent product knowledge and reuse it in Requirements Engineering (RE) exercises for large projects. We capture the product primitives in the form of an ontological model. We associate various concepts in the Business Domain and Environmental Context ontologies with those in the Product Knowledge Ontology and build inference mechanisms based on this foundation. The inference mechanisms are used to provide knowledge assistance while configuring a product requirement set. Upon completing the configuration, the analyst can produce a gap analysis report from the configurator. The report highlights the ‘evolution’ of the product specification from its generic version to its configured variant. This information is critical to evaluate the amount of further customization that may be necessary.

We have demonstrated creation of KB and its reuse in a large Insurance project for product configuration and gap analysis. The evaluation results suggest that using the configurator, requirement analysts can perform a comprehensive gap analysis process. In contrast, analyzing gaps manually is not only inefficient and cumbersome but also error-prone. The configurator assists business analysts in correctly identifying the gaps that exist between the configured specification and the generic product specification. This reduces the chances of redeveloping already existing features. Also**,** by making the tacit knowledge explicit, the configurator reduces the excessive dependence on a small number of experts.

During our interactions with the project team, we realized that our approach has the following limitations:

- The quality of KB that we create is a major determinant of success of the product knowledge configurator.
- Upfront investment in creating a KB can be a hindrance to adopting this approach. The analysts considered creating the KB as a huge overhead.
- Adoption of our approach would require a mindset change that is difficult to achieve in any organization. We observed that the analysts are so comfortable using word documents and the manual approach that they find it difficult to adopt the automated approach in spite of its benefits.
- We anticipate performance issues as the KB grows, especially due to the hierarchies that need to be traversed to detect the knowledge element instances at the leaf-level. (ref fig. 3)
- As the model is subject to refinement in the light of different product specification structures reflected in the different customer templates; the linkages between the concepts and the hierarchies could become complex. This may lead to performance issues as well.

In spite of these limitations, we are optimistic that the evaluation results would be a good motivation to receive buy-in from project teams and customer alike.

REFERENCES

[1] Schierholt K. (2001), Process configuration: Combining the principles of product configuration and process planning. AIEDAM, 15, pp 411-424

[2] Wicaksono, H.; Schubert, V.; Rogalski, S.; Ait Laydi, Y.; Ovtcharova, J., Ontology-driven requirements elicitation in product configuration systems, 4th International Conference on Changeable, Agile, Reconfigurable and Virtual Production (CARV2011), Montreal, Canada 2011

[3] D. Yang, R. Miao, H. Wu, and Y. Zhou, Product Configuration Knowledge Modeling Using Ontology Web Language, Expert Systems with Applications, vol. 36, pp. 4399-4411, 2009

[4] Jiao J.R., Chen C.H., Customer requirement management in product development: A review of research issues, concurrent engineering: research and applications, Vol.14, No. 3, 2006

[5] Mesihovic S, Malmqvist J, Product data management (PDM) system support for the engineering configuration process,14th European Conference on Artificial Intelligence ECAI 2000 Configuration Workshop August 20-25, 2000, Berlin, Germany

[6] Pol P, Patulkar M, Method for Fit-gap Analysis in SAP ERP Projects, White Paper, Infosys. http://www.infosys.com/SAP/thought-leadership/Documents/methods-fit-gap-analysis.pdf

[7] Anish P.R., Sharma S.K., Motwani M, Ghaisas S, Knowledge – assisted product requirements configurator, PLEASE workshop, ICSE 2013, San Francisco, CA

[8] A Falkner,I Feinerer, G Salzer, G Schenner, Solving practical configuration problems using UML, Workshop on Configuration Systems, ECAI 2008

[9] O Djebbi, C Salinesi, Towards an automatic PL requirements configuration through constraints reasoning, Int. Workshop on Variability Modeling of Software Intensive Systems (VaMoS), Essen, Germany, January 2008

[10] R Rabiser, D Dhungana, Integrated support for product configuration and requirements engineering in product derivation, 33rd EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA 2007)

[11] S. Schwarze, Specification mapping - the integration of customer requirements within a product configuration, Institute of Industrial Engineering and Management (BWI), Switzerland, Swiss Federal Institute of Technology (ETHZ), 1993

[12] S. Ghaisas and N. Ajmeri, Knowledge-assisted requirements evolution, Accepted in Managing requirements Knowledge, (MaRK) Ed., Springer

[13] Ajmeri, N., Sejpal R., Ghaisas, S.: A semantic and collaborative platform for agile requirements evolution, In: Third International Workshop on Managing Requirements Knowledge, pp. 32-40. IEEE Press, Sydney (2010)

[14] Wikipedia: http://en.wikipedia.org/wiki/Joint_application_design last accessed on: 6th March 2014

[15] Horrocks I, Patel-Schneider PF, Boley H, Tabet S, Grosof B, Dean M et al (2004) SWRL: a semantic web rule language combining OWL and RuleML. W3C Member Submission 21:79

[16] Yang, D., Dong, M., Miao, R., 2008, Development of a product configuration system with an ontology-based approach Computer-Aided Design, 40:863-878

[17] Kim KY, Manley DG, Yang H. Ontology-based assembly design and information sharing for collaborative product development. Computer-Aided Design 2006; 38(12):1233–50

[18] Zhang JS, Wang QF, Wan L, Zhong YF. Product configuration modeling based on ontology. Computer Integrated Manufacturing Systems 2003; 9(5):344–50

[19] Soininen T, Tiihonen J, Mannisto T, Sulonen R. Towards a general ontology of configuration. Artificial Intelligence for Engineering, Design, Analysis and Manufacturing (AI EDAM) 1998; 12 (4):357–72

[20] Google search: https://www.google.co.in/?gfe_rd=ctrl&ei=pGYdU8atDYuBuATptICADQ&gws_rd=cr#q=attribute+definition Last accesses on 10th March 2014

[21] Wikipedia: http://en.wikipedia.org/wiki/Business_Process last accessed on: 10th March 2014

[22] D. Jannach and M. Zanker, Modeling and solving distributed con-figuration problems: A CSP-based approach, IEEE Transactions on Knowledge and Data Engineering, vol. 25, no. 3, pp. 603–618, 2013