A Knowledge Modeling Framework for Computational Materials Engineering

Raghavendra Reddy Yeddula, Sushant Vale, Sreedhar Reddy, Chetan P. Malhotra, B. P. Gautham, Pramod Zagade
TRDDC, Tata Consultancy Services Limited
Pune, India

***Abstract*— Integrated computational materials engineering (ICME) is a new approach to the design and development of materials, manufacturing processes and products. The approach proposes using a combination of modeling and simulation, data-driven reasoning and knowledge-guided decision making. Industrialization of this approach requires strong automation support. Modeling and simulation is a highly knowledge-intensive activity and integrated design requires knowledge cutting across several design domains. For the industrialization vision to succeed, it is essential to capture this knowledge and make it available in a usable form for people not so skilled in these areas. With this motivation, we are building a comprehensive computational platform to support this emerging design paradigm. We present an overview of the platform and discuss its knowledge engineering requirements. We present a knowledge modeling framework to address these requirements.**

Keywords-ICME; Knowledge Engineering; Knowledge Representation; Ontology;

I. INTRODUCTION

Integrated computational materials engineering (ICME) is a new method for integrated design of materials, products and manufacturing processes using modelling and simulation, knowledge-guided decision making and data-driven reasoning [1,2]. ICME is widely recognized as a paradigm changer that is expected to significantly reduce the dependence on trial and error based experimentation cycles, leading to a) faster development of new materials, and b) significant improvement in quality and time-to-market of products by integrating material design with product design. However, industrialization of this approach has many roadblocks to overcome [3]. Modelling and simulation is a highly knowledge-intensive activity. In an integrated design, one has worry about a multitude of phenomena occurring at different length scales. Choosing right models for these phenomena, at right scales, with right parameters, and ensuring integration across these models is a non-trivial task. It calls for a great deal of domain knowledge and expertise. Similarly, developing right design workflows also calls for extensive domain knowledge and expertise. Shortage of skilled people in these areas can be a significant roadblock for industrial adoption of ICME. For the industrialization vision to succeed, it is essential to capture this knowledge and make it available in a usable form for people not so skilled in these areas. Hence, a software platform that supports ICME must have a strong knowledge engineering capability. It should capture knowledge about a variety of material systems, products and processes, etc., in the form of building-block models, design templates, workflows, rules, etc. It should use this knowledge to systematically guide a designer towards right solutions. With this motivation, we are building a comprehensive computational platform called PREMΛP [4] that supports concurrent design of materials, manufacturing processes and engineered components.

Fig. 1 shows a view of the architecture of the platform from the knowledge engineering perspective. The architecture supports three user roles, namely, knowledge engineer, design solution builder and designer. The design solution builder is responsible for building design solutions for various engineering problems. A designer uses these solutions to solve his/her design problems. The knowledge engineer is responsible for knowledge management: modeling the knowledge space, capturing and curating knowledge from different sources such as domain experts, literature, etc.

At the heart of the platform we have a knowledge repository that contains engineering design knowledge in the form of models, workflows, design templates, design rules, design cases, etc. A design solution builder consults this knowledge base while building his solutions. The knowledge engineering component plays an active role in this consultation process. It processes the design context to infer knowledge that is suitable for the problem context and offers context-specific guidance. This could be in the form of design guidelines, design templates, workflows, models, past design cases and so on.

For instance, building a solution for designing a gear consists of building a gear component model, developing a design process comprising steps such as geometry design, material selection, manufacturing process design, etc., selecting simulation models, defining services to carry out computations, specifying rules to guide design decisions, and so on. Knowledge-guided assistance plays a key role at each stage in this process. The effectiveness of this guidance depends on the quality of the underlying knowledge engineering framework.

The rest of the paper is organized as follows. In section 2 we identify the requirements of the knowledge engineering framework. In section 3 we present a knowledge modeling framework to address these requirements. In section 4 we discuss how this model supports context-sensitive knowledge retrieval. In section 5 we discuss related work, and in section 6 we summarize our contribution.

DOI reference number: 10.18293/SEKE2016-124

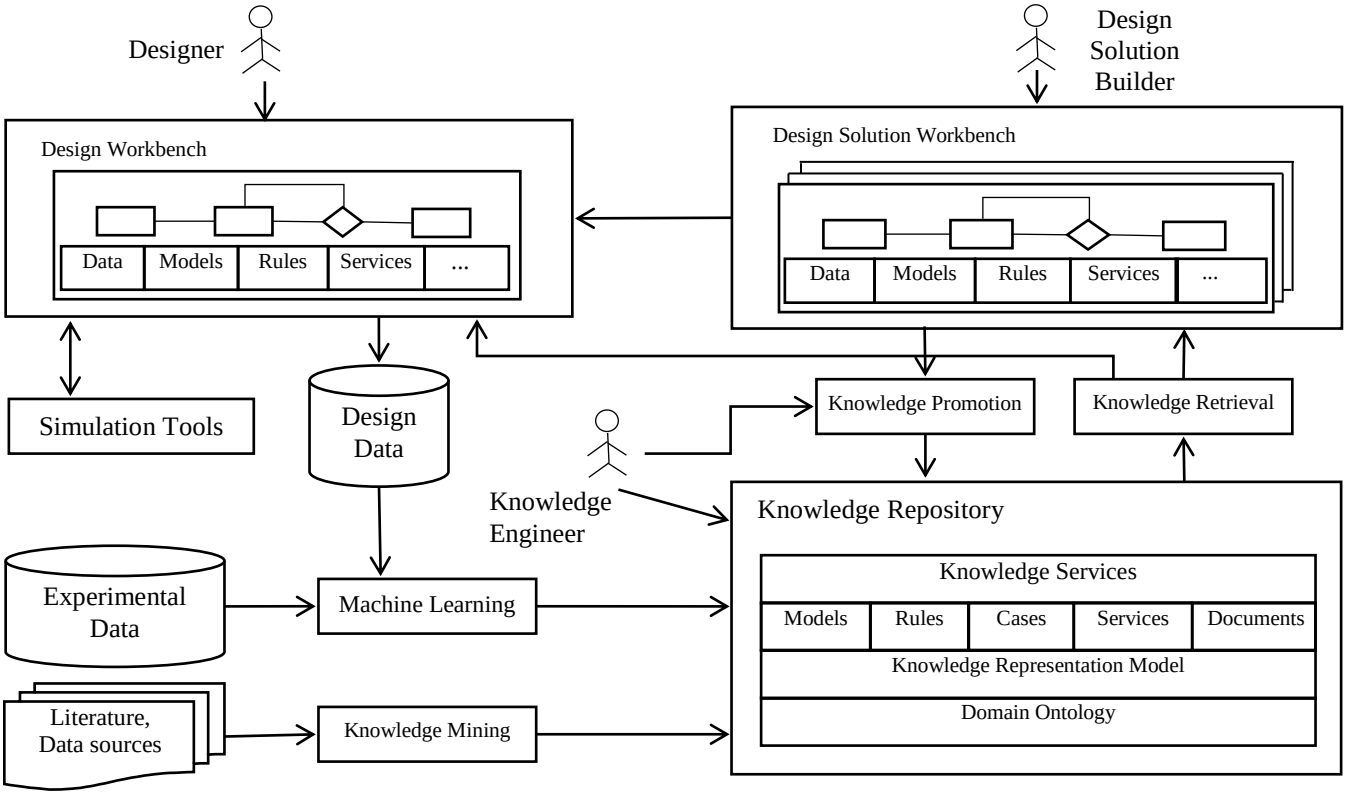

Figure 1. PREMΛP architecture from Knowledge Engineering Perspective

II. PREMΛP KNOWLEDGE ENGINEERING FRAMEWORK

To effectively serve the knowledge needs of a design platform such as PREMΛP, the knowledge engineering framework has to satisfy several requirements:

1. PREMΛP is designed to support integrated design cutting across a number of design domains. These design domains are not always known a priori. Existing design domains also change quite frequently. For instance, there may be a new material that needs to be added or a new manufacturing technique that needs to be supported. The knowledge engineering framework should be flexible to support this.

2. In ICME, knowledge exists in multiple forms such as models, rules, cases, design templates, documents and so on. Some of this knowledge is in executable form with automated reasoning (e.g. rules, models, etc.), while some of it is in the form of documents and media that can only be leveraged by a human. It should be possible to represent both kinds of knowledge.

3. Knowledge is intended to serve a certain purpose. It is essential to model this intent space, so as to be able to capture right kind of knowledge for the right purpose.

4. Knowledge is applicable in a certain context. It is essential to capture this context.

5. Knowledge retrieval should be context-sensitive. I.e. retrieved knowledge should be right for a given problem context.

6. Knowledge reuse should be facilitated. I.e. wherever possible, knowledge from other similar problem contexts should be made available for reuse.

7. Knowledge comes from a variety of sources such as domain experts, literature, online sources, and so on. A lot of knowledge also lies hidden in data. It should be possible to mine knowledge from all these different sources.

We use a model-driven approach to realize the knowledge engineering framework. The starting point is a model of the knowledge space that specifies what kinds of knowledge should be captured and how it should be captured. The knowledge space model again has two parts to it: 1) domain ontology that specifies the conceptual space of the domain, and 2) knowledge representation model.

In this paper, we do not discuss the knowledge mining part (requirement 7).

III. KNOWLEDGE MODELING FRAMEWORK

A. Ontology Model

Domain ontology provides the semantic foundation for capturing knowledge. To maximize reuse and utility of knowledge, the conceptual space of a domain has to be modelled at the right level of abstraction. So, getting the domain ontology right is the first step towards modelling the knowledge right. However, ontology varies from subject to subject, and, being a generic platform, PREMΛP has to cater to a wide range of subjects. For instance, the ontology of steel is

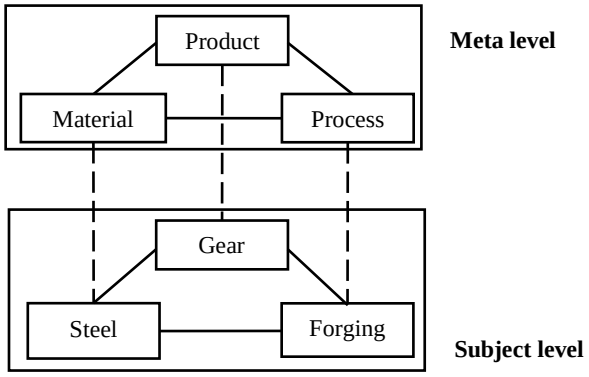

Figure 2. Domain Ontology Layers

different from the ontology of a composite material. This calls for a flexible ontology engineering framework that enables us to create and evolve subject specific ontologies without hard coding them into the platform. To address this, we have conceptualized domain models at two ontological levels - a meta level and a subject level as shown in Fig. 2.

Models relevant for ICME domain can be broadly categorized into three subject areas - materials, products and processes. Corresponding to these subject areas we have three related meta-models -- material meta-model, component meta-model and process meta-model. Subject specific ontologies are created as instances of these meta-models. This layered architecture provides two benefits: 1) Since all subjects of a subject area are described using the same meta-model, the meta-model provides the semantic basis for discovering relations among these subjects, and 2) It lends extensibility to the platform, by enabling new subjects to be created as instances of meta-models. Subject-specific ontologies thus become first class entities in the platform. To achieve this flexibility, we build our ontology modeling framework on a reflexive modeling framework [5] where meta-classes can be used to build multiple modeling levels.

Fig. 3 shows the ontology of Gear as an instance of the component meta-model. A meta-model specifies a set of meta-classes and their relations. A meta-class is a class whose instances are also classes. The figure shows classes in <class>:<meta-class> format and objects in <object>:<class> format. In the figure, the component meta-model has meta-classes *Component*, *GeometricFeature*, *FunctionalFeature* and *Parameter*. The Gear ontology is created as an instance of this meta-model. A gear is a component whose geometry has features such as hub, web, rim and teeth. Its function is to transmit motion, with a change in rotational speed. Accordingly, the *Gear* class is an instance of the meta-class *Component*, *Hub* class is an instance of the meta-class *GeometricFeature*, and so on. The geometric feature '*Hub*' has *diameter* and *width* as parameters. The figure also shows a specific gear (NanoCarGear) with its dimensions, as an instance of the gear ontology.

Reasoning with the Ontology

Classification is one of the key reasoning capabilities in an ontology. Given a description of a concept, the reasoner should automatically discover concepts that generalize the given concept (subsumption relation). In the context of knowledge engineering, we want to be able to generalize knowledge and reuse it in multiple contexts. The ability to automatically discover generalization relationships is therefore a fundamental capability. We use a description logic (DL) [6] reasoner for this purpose. From subject-specific ontology descriptions, we automatically generate the equivalent DL fragments. This is achieved by specifying generation templates at the meta-class level in the meta-model. For example, to generate DL fragments for components, we attach the following template with the meta-class 'Component' in the meta-model shown in Fig. 3.

```
[template DLOfComponent(c: Component)]
[c.name/] =
  Component
  and
  [for (ff:FunctionalFeature | c.functionalFeature)
separator('and')]
    exists functionalFeature.{[ff.name/]}
  [/for] and
  [for (gf:GeometricFeature | c.geometricFeature)
separator('and')]
    exists geometricFeature.{[gf.name/]}
  [/for]
[/template]
```

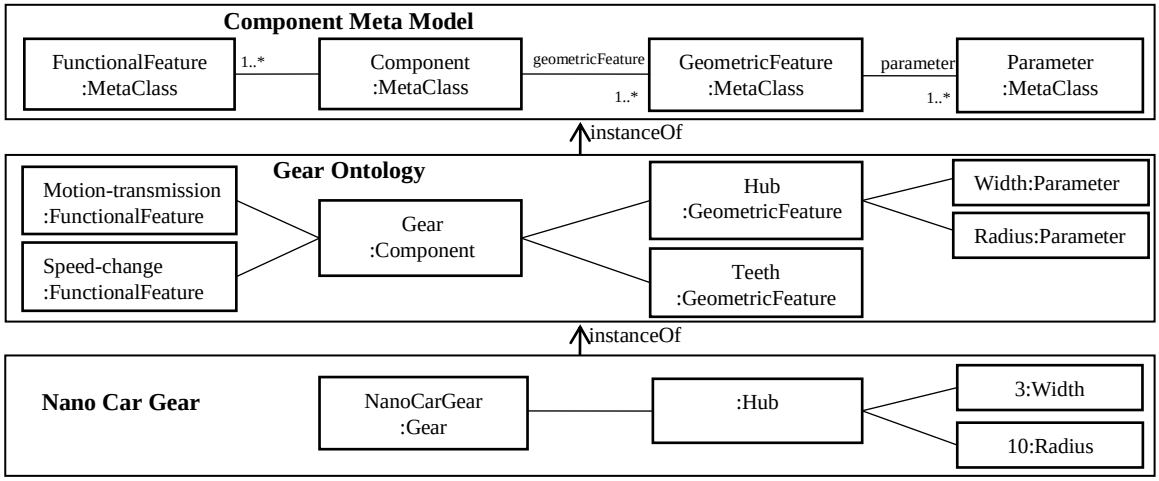

Figure 3. Component Modeling Layers

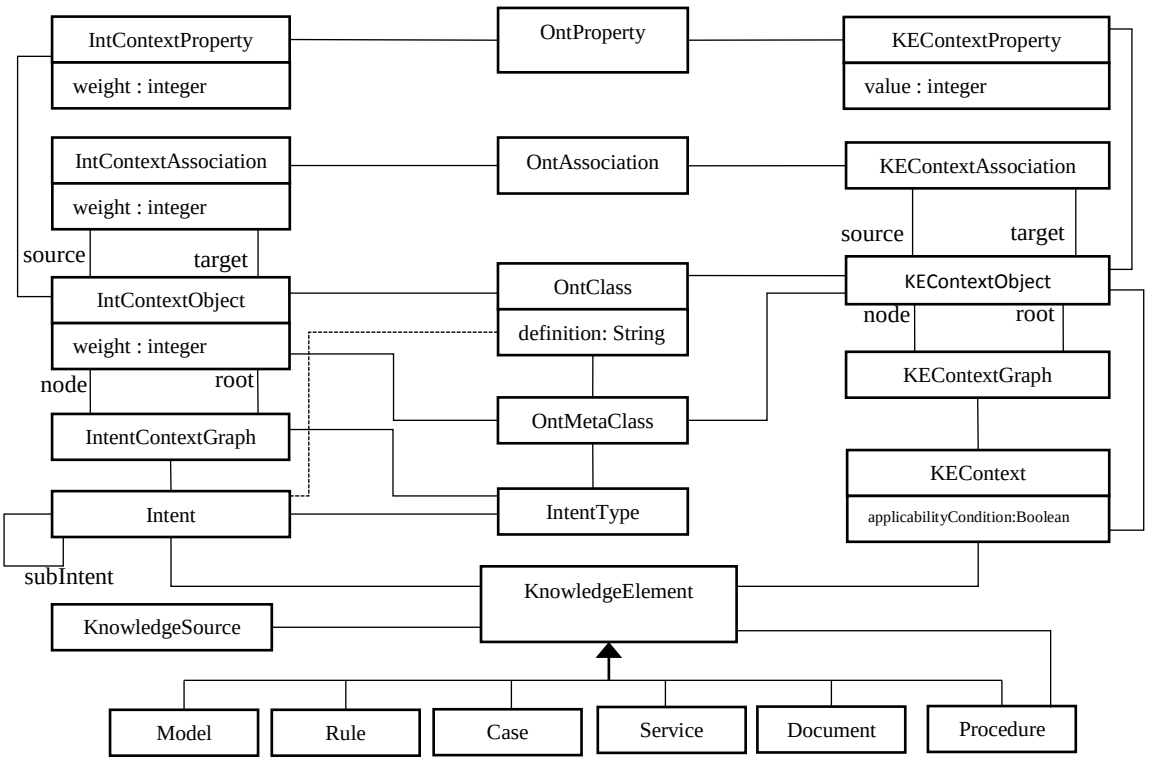

Figure 4. Knowledge Representation Model

The template is specified in Mof2Text specification language [14]. When this template is executed on the gear ontology fragment shown earlier, it generates the DL definition shown below.

Gear =

Component and exists functionalFeature.{Motion-transmission}

and exists functionalFeature.{Speed-change}

and exists geometricFeature.{Hub} and exists geometricFeature.{Teeth}

B. Knowledge Representation Model

Fig. 4 gives an overview of the knowledge representation model. It has four parts: domain ontology, intent model, knowledge element model and knowledge element context model.

The domain ontology model was discussed in the previous section. Ontology provides the domain vocabulary to represent different knowledge elements. Ontology also serves to specify the contexts in which those knowledge elements are applicable. In Fig. 4, *OntMetaClass* refers to an ontology meta-class (e.g. Component), *OntClass* refers to an ontology class (e.g. Gear), *OntAssociation* refers to an ontology association and *OntProperty* refers to an ontology property.

1) Intent Model

Knowledge is required to serve a certain intended purpose in the domain. For example, knowledge to guide requirement specification of a component, knowledge to design a process, knowledge to select a material, and so on. It is essential to model these intents (or topics) in order to ensure that right knowledge is captured for the right purpose. We identify intents at two levels. At the meta-level, the model element *IntentType* specifies intents of a meta-class. At the subject level, the model element *Intent* specifies intents of a subject-specific class. For example, the meta-class Component has intent types 'Geometry design' and 'Material selection'; the meta-class ManufacturingProcess has the intent type 'Process design'. At the subject level, the class Forging has 'Forging process design' as an intent. This is an instance of the intent type 'Process design' of the meta-class ManufacturingProcess. An intent may have sub intents. For instance, forging process design has preform design and die design as sub intents.

An intent may also identify the information context necessary to make decisions about the intent. For example, forging process design decisions depend on the geometry of the component, its performance requirements and the material used. An intent specifies its information context by means of a context graph that identifies a set of connected ontology elements that together provide the information context for the intent. Nodes of the graph identify ontology objects and edges identify the associations between them. Fig. 5 shows the intent context graph for forging process design. An intent's context

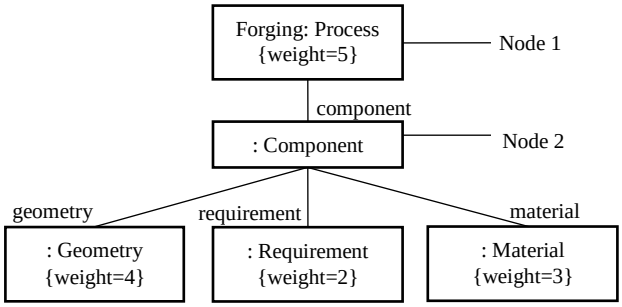

Figure 5. Intent Context Graph for Forging Process Design Intent

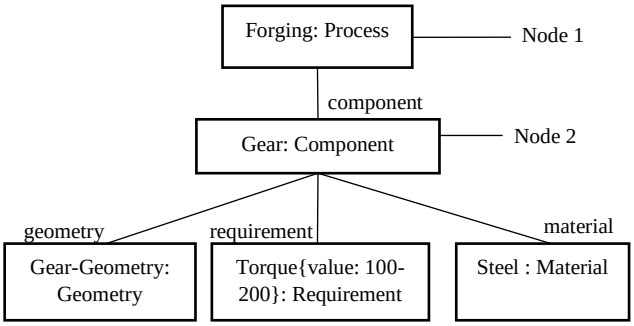

Figure 6. Query graph for fetching forging process design knowledge

graph is inherited by all its sub intents, unless explicitly overridden.

An intent context graph provides the template to construct knowledge queries to fetch knowledge for a given intent. The query is constructed from the information available in the problem context. For example, in a problem context where gear is the component, steel is the material and the required torque range is 100-200, the query graph shown in Fig. 6 is constructed from the context graph shown in Fig. 5. An intent context graph may also specify weights for objects, associations and properties, signifying their relative weightages during query processing. The graph in Fig. 5 specifies that a match on geometry has higher weightage than a match on material.

An object node in an intent context graph may specify a class or a meta-class. When a meta-class is specified, the corresponding node in the query graph can be instantiated with any class of the meta-class. This makes a context graph generic and reusable across multiple problem contexts. For example, in Fig. 5, object node 2 specifies the meta-class 'Component'. When constructing the query graph for forging of a gear component, this node would be instantiated with the class 'Gear'; when constructing the query for a clutch component this would be instantiated with 'Clutch', and so on.

2) Knowledge Element Model

A *KnowledgeElement* is a unit of knowledge that serves an intended purpose in a context. Knowledge elements can be represented in multiple forms such as models, rules, cases, services, documents, etc. Of these, models, rules, cases and services are executable knowledge elements, whereas documents are for human consumption. A special knowledge element called 'Procedure' enables composition of multiple knowledge elements into a step-by-step procedure. The model also tracks the provenance of knowledge elements (e.g. from which domain expert, from which publication, etc.).

3) Knowledge Element Context Model

Knowledge is generated in a certain context. It is essential to capture this context to ensure that right knowledge is applied in the right context. We model the context by means of knowledge element context graphs. A knowledge element context graph is essentially a description of the state of the world in which a knowledge element is applicable, expressed as a set of conditions on ontology classes and meta-classes. The graph specifies a set of ontology objects (nodes), their property values and their associations (edges). Its semantics is that the knowledge element is applicable in the context of the given ontology objects when they are related to each other in the specified manner and when they have the specified property values. A knowledge element context may additionally specify a boolean expression (applicabilityCondition in the model) that these ontology elements must satisfy. In the context graph, an object node may specify a class or a meta-class; when a meta-class is specified, the knowledge element is applicable in the context of 'any class' of the meta-class.

Example 1:

The context graph of knowledge element 'Material Selection Rules for Gear' is shown in Fig. 7. This graph specifies that the knowledge element is applicable for material selection for automobile gears.

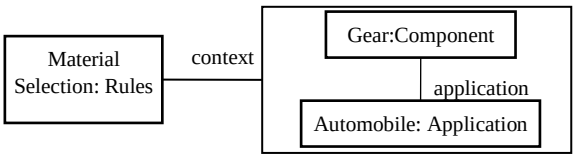

Figure 7. Knowledge element context graph for gear material selection

Example 2:

The context graph for knowledge element 'Billet volume estimation procedure' is shown in Fig. 8. This graph specifies that the billet volume estimation procedure is applicable when forging steel components with axisymmetric geometry.

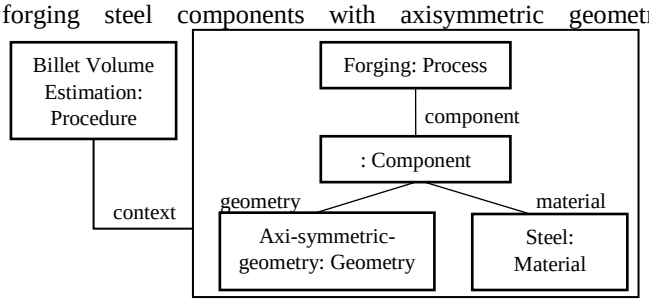

Figure 8. Knowledge element context graph for billet volume estimation procedure

IV. CONTEXT SENSITIVE KNOWLEDGE RETRIEVAL

Knowledge retrieval is done by matching the problem context with contexts stored in the knowledge repository and fetching knowledge elements from matching contexts. This is done by constructing a query graph from the current context as specified by the intent context graph.

To give an example, we consider the gear design case. To start with, we assume that a design solution builder has already developed the gear ontology and a partial design workflow consisting of requirements capture, geometry design, material selection, manufacturing process selection, etc. Let's also assume that the gear is required to handle a torque range of '100 – 200' and the material selected is steel. Let's assume that the solution builder is currently working on designing a forging process for the gear, and in particular on estimating the initial billet volume which is one of the sub steps of the design process.

So the current intent is 'Estimate Initial Billet Volume' which is one of the sub intents of 'Forging process design'. The intent context graph for this intent is shown in Fig. 5. From this

graph and the information available in the problem context, the system constructs the query graph shown in Fig. 6. This query graph is matched against the context graphs of knowledge elements stored in the repository. Matching is based on a distance measure that takes into account class similarity, property similarity and related object similarity (recursively) [7]. Matching also takes into account the relative weights of entities in the query graph. A detailed description of the matching algorithm is beyond the scope of this paper.

The query graph of Fig. 6 matches the context graph of the knowledge element 'Billet Volume Estimation Procedure' shown in Fig. 8. Note that gear geometry matches Axi-symmetric geometry since ontological reasoning infers that the latter subsumes the former.

V. Related Work

CommonKADS [8] is a well-known method that proposes a model-based approach to knowledge engineering. However, CommonKADS models are general purpose in nature and are meant to provide a broad roadmap. In contrast, we propose a knowledge modeling framework that is specifically designed for the needs of an engineering design domain that supports integrated design across a number of subject areas, with an open ended architecture for adding new design domains. To support this, we propose a meta-modeling approach to model ontologies, intents, contexts and knowledge elements.

Knowledge Based Engineering (KBE) [9, 10] is another methodology with its roots in computer-aided design (CAD) and product lifecycle management (PLM) systems. It facilitates the creation of a product design based on best practices, geometry and data related to a product family. However, it does not provide a framework for modeling knowledge spanning multiple problem dimensions such as material, product and manufacturing process. Neither does it provide a means to process knowledge in a context-sensitive manner.

Expert systems have long been used for building knowledge bases. While expert systems support automated inference, not all knowledge is amenable to formal representation and reasoning. We need a system that supports both formal and informal knowledge representations (such as documents), and a means to reason about the knowledge itself so as to be able to deploy right knowledge in right problem contexts.

The role of context in reasoning has been extensively studied in the AI literature [11, 12, 13]. Context also plays a big role in knowledge retrieval. Knowledge is generated in a certain context. At the same time, knowledge to serve a given intent varies from problem context to problem context. It is essential to match these contexts to retrieve right knowledge. We propose an ontological approach to describe contexts, and an ontology-driven context-matching method to retrieve knowledge. Moreover parts of a context can be specified at the meta-level, thereby generalizing the context.

Case-based reasoning (CBR) provides a means to retrieve cases based on problem context matching. There are also approaches that marry CBR with ontologies [7], where both problems and queries are described in terms of ontologies. This is somewhat similar to our approach where we also describe knowledge element contexts and query graphs in terms of ontologies. However, the key difference is the intent model, using which we can automatically generate context-appropriate knowledge queries.

VI. Summary/Conclusions

In this paper, we have given an overview of a platform for integrated computational materials engineering. We have discussed its knowledge engineering requirements and proposed a knowledge modeling framework to address these requirements. We have discussed a meta-modeling framework for flexible ontology modeling. We have discussed a knowledge representation model where intents can be modelled and ontological contexts of knowledge elements can be captured. We have also discussed how these models can be used for context-sensitive knowledge retrieval.

References

[1] NRC Report (2008) Integrated Computational Materials Engineering: A Transformational Discipline for Improved Competitiveness and National Security. The National Academies Press, National Research Council, Washington, D.C.

[2] TMS Study Report on Integrated Computational Materials Engineering (ICME) – Implementing ICME in the Aerospace, Automotive and Maritime Engineering, (2015) TMS, http://www.tms.org/ICMEStudy.

[3] Konter, A.W.A., Farivar, H., Post, J. and Prahl, U. Industrial Needs for ICME. JOM: the journal of the Minerals, Metals & Materials Society, 2015.

[4] Gautham, B.P., Singh, A.K., Ghaisas, S.S., Reddy, S. S. and Mistree, F. (2013a) PREMAP: A Platform for the Realization of Engineered Materials and Products, A. Chakrabarti and R. V. Prakash (eds.), ICoRD'13, Lecture Notes in Mechanical Engineering, Springer India, pp. 1301-1313.

[5] Kulkarni, V., Reddy, S., & Rajbhoj, A. (2010). Scaling up model driven engineering–experience and lessons learnt. In Model Driven Engineering Languages and Systems (pp. 331-345). Springer Berlin Heidelberg.

[6] Baader F., Calvanese D., et al. The description logic handbook: theory, implementation, and applications. Cambridge University Press.

[7] Ralph B., Martin S. Case-Based Reasoning and Ontology-Based Knowledge Management: A Perfect Match?. Journal of Universal Computer Science. vol. 9, no. 7 (2003), 608-626.

[8] Guus T.S., Hans A. Knowledge engineering and management: the CommonKADS methodology. MIT Press Cambridge, MA, USA.

[9] Verhagen, W. J., Bermell-Garcia, P., van Dijk, R. E., & Curran, R. (2012). A critical review of Knowledge-Based Engineering: An identification of research challenges. Advanced Engineering Informatics, 26(1), 5-15.

[10] La Rocca, Gianfranco. "Knowledge based engineering: Between AI and CAD. Review of a language based technology to support engineering design." Advanced engineering informatics 26.2 (2012): 159-179.

[11] Öztürk, Pinar, and Agnar Aamodt. "A context model for knowledge-intensive case-based reasoning." International Journal of Human-Computer Studies 48.3 (1998): 331-355.

[12] Montani, Stefania. "How to use contextual knowledge in medical case-based reasoning systems: A survey on very recent trends." Artificial intelligence in medicine 51.2 (2011): 125-131.

[13] Brézillon, Patrick. "Context in problem solving: a survey." The Knowledge Engineering Review 14.01 (1999): 47-80.

[14] MOF Model to Text Transformation Language. www.omg.org/spec/MOFM2T/1.0