Pragmatic Design of Methodology in Process-Based Knowledge Integration

Yew Kwang Hooi, Mohd Fadzil B Hassan, Anouar Abrik, and Azmi M. Shariff

Universiti Teknologi Petronas, Malaysia,

{yewkwanghooi; anouarabrik}@gmail.com,{mfadzil_hassan;azmish}@petronas.com.my

ABSTRACT

This paper proposes a methodology to design a pragmatic ontology. Pragmatism can maximize the interaction between rules and ontology. A formal model provides the template for pattern identification of symbols and rules for manipulation by a logic machine. The ongoing research proposes pragmatics in knowledge engineering. However, current methodology does little to emphasize a truly pragmatic design in an ontology pattern. A critical analysis of the approach against existing methodologies has shown lack of research in this important area of ontology design.

Keywords: Knowledge-based system, ontology engineering, model-driven system.

I INTRODUCTION

Knowledge is power but only if it can be seen. In enterprises, knowledge is hidden in its trove of information such as relational database, XML data, unstructured and semi-structured documents. To be aware of this "obscured" knowledge is a difficult task and many semantic and big data researches have attempted to address this issue.

Knowledge integration is a challenging issue in an enterprise knowledge management. Integration of knowledge is a cognitive activity based on logic and meta-knowledge. Integration requires retrieval and linking. A conventional retrieval is difficult because the search itself must furnish information on the relationship between searched resources. A search engine is unaware of the implicit structure and relationship between linked concepts within the domain of the search. A knowledge model helps with the search by providing a platform to guide the discovery.

A knowledge model is simply logical knowledge and like any other logical knowledge, it does not map well to computer architecture, unlike procedural knowledge or factual knowledge. Thus, logical knowledge is encoded in a formal knowledge model such as ontology as an alternative to source code. The capacity of ontology to support dynamic changes makes it favorable for organizations to address change in management.

II ONTOLOGY AS A FORMAL MODEL

Ontology is a formal, explicit, shared conceptualization for knowledge sharing (Gruber, 1993; Neches et al., 1991; Uschold & Gruninger, 1996; Van Heijst, 1997). Ontology is a sample of domain knowledge and therefore only represents a subset, as shown in Figure 1. The ontology may also overlap with problem-solving knowledge, which is usually in the form of code. Other knowledge that is involved in problem solving is knowledge of data mining and knowledge of the data sets.

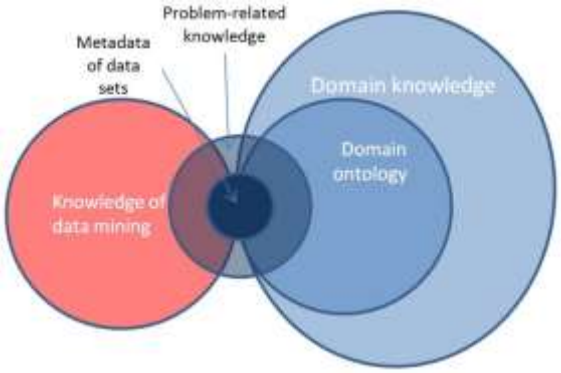

Figure 1. Types of knowledge models (Kuo, Lonie, Sonenberg, & Paizis, 2007)

The structure of ontology is given by O = (S,A) where S is signature that describes the vocabulary and A, the axiom interpretation of vocabulary in the domain of discourse (Kalfoglou & Schorlemmer, 2003).

Ontology is used to provide the semantics of knowledge integration (Basil Ell, 2011) (Gonçalo Antunes, 2014) (Benaroch, 2002; Wache et al., 2001). The semantics provide the details that allow inference of meaning through heuristics rules using relational objects as a reference. The mechanism of integration is through using ontology as a knowledge framework for anchoring and linking instances of information specified by the properties of objects.

Ontology has the potential to be used by intelligent applications for its more powerful reasoning (Noy & McGuinness, 2001) and interoperability (Calvanese et al., 2007) capabilities.

III PROBLEM AND OBJECTIVE

Developing an oversized ontology may be counterproductive, i.e. large but not necessarily well utilized. Knowledge models have received criticism

for the difficulty to determine their scope. Various works state the bottleneck of the work is the lack of expressiveness of the knowledge model to deliver pragmatic results in real applications. To keep only useful concepts, a methodology should consider focusing on the pragmatic concepts by referring to the "verb" elements from business processes and policies.

There is a lack of methodologies that emphasize pragmatic ontology design. This article critically analyzes the advantage of a pragmatic structure in knowledge models and proposes a methodology for such.

There is also limited definition of pragmatic in the context of knowledge management at organizational and technical views.

This paper attempts to define pragmatics from a system development point of view. The definition is used to identify constraints and criteria for the methodology that we claim having potential to bring out a pragmatic design.

The main constituents of a methodology are people, product and process. The focus of this paper is on the process and the corresponding tools of the methodology (Dehghani & Ramsin, 2015). The people and product constituent will be explained incidentally.

IV PRAGMATIC AS THE WAY FORWARD

A. Definition of Pragmatic

The notion of pragmatics is commonly understood as a model that provides shared understanding on the intention and actual use of exchanged content in a given context (Camlon H. Asuncion). Pragmatic refers to actionable knowledge and is basically just the ontology necessary for process work (Thompson, 2005) (Anthony Debons, 2000). A simpler definition is “just that ontology necessary for process work” (Thompson 2005).

Interest in pragmatic:

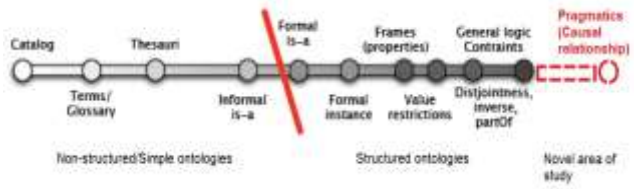

Figure 2. Evolution of Formal Ontology.

There have been proposals for the inclusion of pragmatic in a knowledge model for a more complete and meaningful information exchange, see Figure 2 (Jan Mendling 2007). A formal pragmatic can provide better rational and scientific foundation for an integrated ontology theory as well as practice that extends semantics (Akkermans, 2008). The major aim of pragmatic is to support putting ontologies in context, thus future research should be heading this direction (Giboin, Denis, #233, & Snoeck, 2011).

B. Role of Pragmatic

Less attention has been given to the pragmatic aspect of ontology. The role of pragmatic in relation to ontology can best be illustrated as Figure 3. Most works have been done in the area of semantics but few are actually in pragmatic.

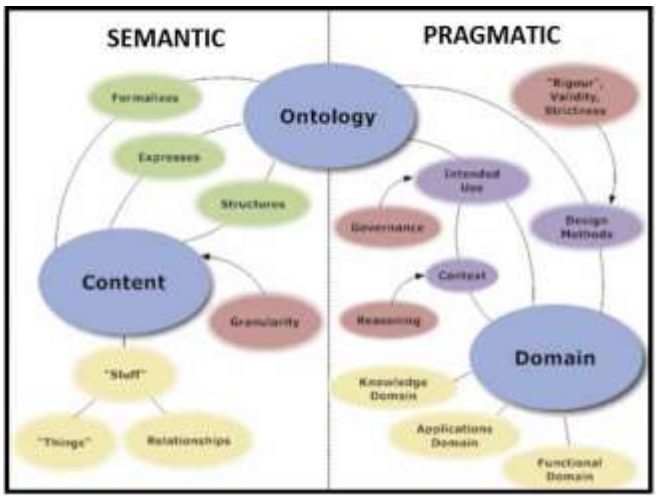

Figure 3. Ontology Framework (Gruninger, Bodenreider, Olken, Obrst, & Yim, 2008)

The figure below illustrates the conceptual relationship of pragmatic with other body of knowledge and applications. The pragmatic perspective is the specific structure of knowledge model that deals with actionable knowledge. It can be viewed as a subset of the domain knowledge model.

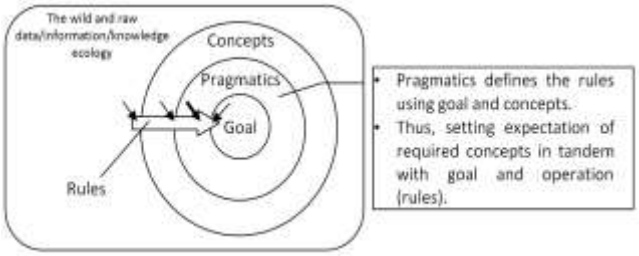

Figure 4. Working definition of pragmatic.

V THE PRAGMATIC ARCHITECTURE

We propose to lay emphasis on the pragmatic perspective to drive the design of the ontology and application system. Pragmatic can optimize the right size of ontology for usage. Figure 5 below illustrates the conceptual relationship of pragmatic with another body of knowledge and applications. The pragmatic perspective is the specific structure of knowledge model that deals with actionable knowledge. It can be viewed as a subset of the domain knowledge model.

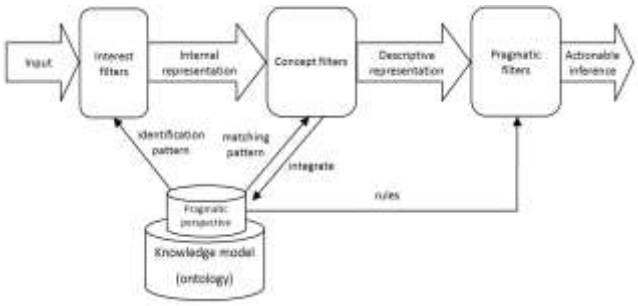

Figure 5. Proposed Pragmatic Architecture

Pragmatic provides patterns and rules to the application system. Interest filters help the system to focus on relevant data/information/knowledge. Concept filter assigns system's meaning to the information. The system uses its own experience, in the form of vocabulary provided by its internal formal knowledge, to describe the representation. This process integrates the internal representation of the input with the model. The pragmatic perspective provides a set of rules associated with the descriptive representation to the inference engine. Naturally, pragmatic should work well with rule-based engine due to the nature of the representation semantics. The outcome of the engine will be actionable inference with respect to the domain knowledge problem solving requirements.

Pragmatics basically provides the problem solving requirements which can be generic as well as domain-specific. To handle generic functions, pragmatic in ontology can be used to check for the truth of an antecedent concept and thus trigger the truth of the consequential concept. From the context of integration, this can be used to check if a relevant piece of information is available to the user, or if the information should be updated. If the information is unavailable, then the search or the capture should be conducted. It is also essential in some application of information that certain information (upon availability or upon unavailability) should become an event that triggers awareness of decision-makers. Examples of domain-specific functions include findings such as missing of information, a possible factor for an outcome, or an outcome of a set of interrelated incidents. All these will trigger use actions.

The firing of pragmatic concepts may be coordinated by a mechanism. Concepts that are similar, or having subsumption relationship, may fire together. Concepts that have a direct causal relationship, may fire consecutively. If the concepts are stimulated repeatedly, the connection between the two becomes stronger, through update of the strength property of a connection.

Conventionally, when the ontology is not expressive enough, the rule layer will be added on top of the ontology layer. A suitable language for rule such as SWRL will be used to express rules in terms of OWL concepts of classes, properties and individuals.

A possible application of pragmatic design is to improve knowledge integration. This is done by developing pragmatic ontologies which are proposed to be more compatible with rule-based systems. Pragmatic refer to a structure that captures and encodes business rules that serve as the driver for integration. Business rules as triples provide causal relations between the antecedent object and the consequent object. Antecedent includes event whereas consequent includes an expected outcome. An event triggers transition to consequent based on properties of the antecedent. The properties of the antecedent are taken from the ontology concepts. Causal relation can maximize rule-based manipulation of ontology and generation of rules that are coupled with facts in the ontology to help with the integration.

The basis of claim that pragmatic can improve integration is derived from overlaps of several postulations:-

- Belief-Desire-Intention (BDI) model for intelligent agent programming. BDI model is used as a programming paradigm to set the world model that software accepts as true (Belief) and form the basis of processing; desire is the state that is achievable, and intention will be a selected function or a set of selected functions to achieve the desire. The BDI model is pragmatic in nature which can be implemented as a set of guided rules.
- Function-Behaviour-Structure (FBS) ontology model for design process. FBS model can be used to extend the Intention component of BDI. The function is achievable by setting the expected behaviour of the software and this can be implemented by using one or combining a few components (Structure).
- Information Integration Theory (IIT) is an awareness phenomenon model that attempts to simplify awareness as activities of merging information that results in an information amount that is bigger than the sum of the merged information [Tononi, 2015 #699].

VI PRAGMATIC METHODOLOGY

Various methodologies have been proposed with different goals, functions and levels of abstraction in mind (Dehghani & Ramsin, 2015; Hooi, Hassan, Abidin, Arshad, & Shariff, 2015). Generically, developing ontology begins with a specification of ontology requirement which specifies the scope through use cases, expert inputs, competency

questions, brainstorming or by referring to existing ontology structure or vocabulary (Chen, 2004).

The next step is conceptualization whereby terms are identified and described using middle-out, top-down or bottom-up strategies (Gawich, Badr, Hegazy, & Ismail; Gómez-Pérez, Fernández-López, & Corcho, 2004; Uschold & Gruninger, 1996). Middle-out strategy focuses on most relevant concept first and is claimed to be more stable and easier (Uschold & Gruninger, 1996).

Few methodologies specifically measure the amount of knowledge for each task (Dehghani & Ramsin, 2015; Sarnikar & Deokar, 2010). The criteria for determining knowledge intensity has been proposed by Eppler et al in (Eppler, Seifried, & Ropnack, 1999).

Pragmatics methodology is proposed to adopt middle-out approach, but more specifically to capture actionable concepts. The last step is to implement the concept by formalizing them into a suitable format such as OWL for facts and SWRL for rules. In this way, applications can use SPARQL query to retrieve information and achieve knowledge discovery.

A. Process and Tools Support Pragmatic

Figure 6 below shows the processes and tools that can be used in the analysis and design of software in the conventional software development sense. In the proposed methodology, these tools can be used to acquire the concepts and rules needed by the ontology. To bridge the objects with the process, we propose the use of a modified Class-Responsibility-Collaboration (CRC) tool. CRC is a diagrammatic tool in object modelling. With some modification, the findings of tools used in process modelling can be included in object-modelling.

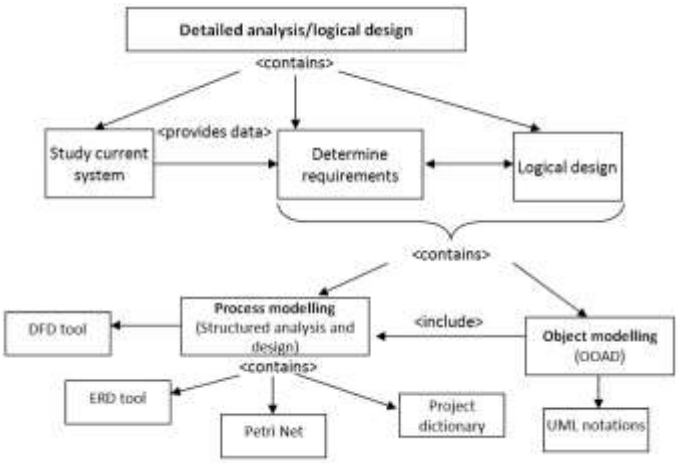

Figure 6. Analysis and design tools.

B. Responsibility-Class-Collaborator (RCC) to Class Responsibility-Collaborator (CRC) Technique

Conceptualization can be done using Class-Responsibility-Collaboration (CRC) frame, as shown in Figure 7. This is modeled after CRC card, a brainstorming tool for quick conceptualization. CRC constrains unnecessary responsibilities. This is an important property of CRC because pragmatic is about focusing on the necessary task and knowledge only.

The constraint can be further improved by developing Responsibility-Class-Collaborator (RCC) frame first. Similar to CRC, RCC proposed here begins with identifying activities, its processes and its tasks. Then, for each task, identify the properties. To systemize the design, the following frame can be used:

Responsibility (R)		
Class1,C1	Collaborator1, Co1	Trigger, temporal, location, personal, collaborator state change
Class2,C2	Collaborator2, Co2	Trigger, temporal, location, personal, collaborator state change
Class3,C3	Collaborator2, Co2	Trigger, temporal, location, personal, collaborator state change
Class2,C2	Collaborator3, Co3	Trigger, temporal, location, personal, collaborator state change

Figure 7. RCC Frame

The following steps are recommended to find the pragmatic concepts and relations:

1. Start by writing a scenario, and then identify the major actions and corresponding actors.
2. Develop RCC frames to describe the actions.
   a. Actions become responsibilities (R).
   b. R is a causal relationship and is defined as the methods that will alter one or both actors' values or states. Collaborators (Co) refer to other actors with whom this actor (aka Class (c)) has causal relationship with. Co is the other class that becomes the object of the relationship; whereas C is the subject and R is the predicate.
   c. Identify properties of R that becomes relevant to the C-Co pair. There should be information on triggering event, parent of responsibility, temporal properties, location properties and person-in-charge.
   d. Identify parent-child relationship between R and store this in a R-

Hierarchy list or table. Identify R between the same C-Co pair or based on expert or guideline information.

3. RCC frames can be collectively presented as a table. The table will have rows of R_1 to R_n. Then, the columns will be populated with Classes, and the cell will be populated with corresponding collaborators. This becomes the RCC table, which is a collection of RCC frames.
4. Convert RCC table into CRC frames:
   a. Read the column followed by row and finally cell. Iterate this for the table. This will generate a collection of CRC frames, as shown in Figure 8.
   b. There should be some information about R, such as if it is a static method or a dynamic rule. This part requires some analysis to see if it is a standard task or context specific and if it has any inferencing property.
   c. Other properties of R are inherited from RCC.
   d. Each CRC frame is inspected for parent-child relationship. Store this information in a separate Class Hierarchy list or table.
      i. Each frame has a relevant class/object in the design.
      ii. Arrange the frame, placing it in closer proximity if the classes/objects are highly related. Class-type relationships are identified.

Class (C)	
Responsibility1, R1	Collaborator1, Co1
Responsibility2, R2	Collaborator2, Co2 Collaborator3, Co3
Responsibility3, R3 Responsibility4, R4	Collaborator4, Co4

Figure 8. CRC Frame

5. Evolve CRC into a class diagram and ontology.
6. Evolve RCC into corresponding rules. Fixed rules such as commonly used techniques or algorithms are translated into methods to be incorporated into class diagram. On the other hand, business rules that may evolve are translated into rule bases.

C. Pragmatic Ontology Design

Inputs from RCC provide the components, properties and relations as building blocks of the ontology. The ontology will subsequently have a significant number of object properties with corresponding domain and range entities:-

a. The object property can then be compared with other object properties of the ontology using hierarchical relation, properties (functional, inverseFunctional, symmetric or transitive) and refined with restriction.
b. Determine rules and axioms from the object properties.
c. Develop ontology from the entities in the relations by establishing relations:- subsumption, equivalent or disjoint.
d. Generate knowledge rules SWRL from the ontology.

The expected output based on the proposed approach using RCC-to-CRC technique is a slimmer pragmatic ontology which can be mapped and is traceable between classes' method, classes' attributes and system rules.

VII DISCUSSION

The methodology is best evaluated through empiric evaluation. The proposed pragmatic methodology is best used with process-based problems which require rich information and knowledge support. Various real-world problems such as clinical diagnosis, loan processing and process safety management in chemical plants may benefit from this.

In particular, pragmatics can provide a richer integration. Ontology has been mainly exploited for its semantic support but many problems in the real world are process-based. Vocabulary based on process is necessary and the capability to make use of existing rule-based systems will further boost the potential of ontology to be used for dynamic and continuous information as well as knowledge integration.

Most knowledge system methodologies do not have empiric backing but are evaluated based on the comparison with existing methodologies or by theoretical arguments. This is based on the findings by (Dehghani & Ramsin, 2015) which has also proposed criteria-based evaluation. This work has been evaluated using the development-related factors of Table V in (Dehghani, 2015 #722). The result is satisfactory achieving 80% of the criteria. However, the criteria metric is very general, i.e. giving only positive or negative as the yard stick. The result of the evaluation is given in Appendix I.

VIII CONCLUSION

This paper proposed a working definition of the pragmatic aspect of ontology. It has argued that most methodologies for knowledge engineering do not provide sufficient detail when capturing and designing a pragmatic model. This paper has shown the use of RCC-to-CRC, an adaptation of CRC card used in requirements elicitation, as a viable approach to capture and build a pragmatic ontology. A pragmatic ontology has the benefit of maximizing

representation of rules in ontology. This will facilitate coupling with rule-based systems and access to rule generation facility. The difference brought by this approach is that the rules are semantically richer by coupling with facts ontology.

ACKNOWLEDGMENT

The authors would like to thank the management of Universiti Teknologi PETRONAS (UTP) for rendering support to this research and Process Safety group of UTP and PETRONAS Penapisan Melaka for knowledge sharing.

REFERENCES

Akkermans, H. (2008). *The Business of Ontology calls for a Formal Pragmatics.* Paper presented at the Proceedings of the 2008 conference on Formal Ontologies Meet Industry,

Anthony Debons, N.-G. D. (2000, November 29-30, 2000). *Need of Education and Training for Information Science.* Paper presented at the Proceedings of XIth Annual Symposium on Reliability and Quality Assurance, Bucharest.

Basil Ell, E. S., Stephan Wölger, Benedikt Kämpgen, Simon Hangl, Denny Vrandečić, Katharina Siorpaes. (2011). Enterprise Knowledge Structures *Context and Semantics for Knowledge Management* (pp. 29-59).

Benaroch, M. (2002). Specifying Local Ontologies in Support of Semantic Interoperability of Distributed Inter-organizational Applications. *Next Generation Information Technologies and Systems - Lecture Notes in Computer Science, 2382*(2002), 90-106.

Calvanese, D., Giacomo, G. D., Lembo, D., Lenzerini, M., Poggi, A., & Rosati, R. (2007). *Ontology-based database access.* Paper presented at the Proceedings of the Fifteenth Italian Symposium on Database Systems.

Camlon H. Asuncion, M. J. S. (2010). Pragmatic Interoperability: A Systematic Review of Published Definitions. *Enterprise Architecture, Integration and Interoperability*(326), 164-175.

Chen, H. (2004). eBiquity Group Meeting. Retrieved from

Dehghani, R., & Ramsin, R. (2015). Methodologies for Developing Knowledge Management Systems: An Evaluation Framework. *Journal of Knowledge Management, 19*(4), 682-710.

Eppler, M. J., Seifried, P. M., & Ropnack, A. (1999). *Improving knowledge intensive processes through an enterprise knowledge medium*. Paper presented at the Proceedings of the 1999 ACM SIGCPR conference on Computer personnel research, New Orleans, Louisiana, USA.

Gawich, M., Badr, A., Hegazy, A., & Ismail, H. (2012). A Methodology for Ontology Building. *International Journal of Computer Applications, 56*(2), 39-45.

Giboin, A., Denis, B., #233, & Snoeck, l. (2011). *Taking "pragmatic dimensions" of ontologies into account: frameworks*. Paper presented at the Proceedings of the 7th International Conference on Semantic Systems, Graz, Austria.

Gómez-Pérez, A., Fernández-López, M., & Corcho, O. (2004). *Ontological Engineering* X. Wu & L. Jain (Eds.),

Gonçalo Antunes, M. B., Rudolf Mayer, José Borbinha, Artur Caetano. (2014). Using Ontologies for Enterprise Architecture Integration and Analysis. *Complex Systems Informatics and Modeling Quarterly (CSIMQ)*(1), 1-23.

Gruber, T. R. (1993). Toward Principles for the Design of Ontologies Used for Knowledge Sharing. *International Journal Human-Computer Studies, 43*(4-5), 907-928.

Gruninger, M., Bodenreider, O., Olken, F., Obrst, L., & Yim, P. (2008). Ontology, taxonomy, folksonomy: Understanding the distinctions, Applied Ontology. *Ontology Summit 2007, 3*(3), 191-200.

Hooi, Y. K., Hassan, M. F., Abidin, A. I. B. Z., Arshad, N. I. B., & Shariff, A. M. (2015, 24th August 2015). *A Generic Ontology Methodology and Factors to Address Design Consistency.* Paper presented at the 5th International Conference on IT Convergence and Security 2015 (ICITCS 2015) Kuala Lumpur.

Jan Mendling , J. R. (2007). Extending the Discussion of Model Quality: Why Clarity and Completeness may not always be enough *19th International Conference on Advanced Information Systems Engineering (CAiSE 2007)* (Vol. 365, pp. 109-121). Trondheim,, Norway: Tapir Academic Press.

Kalfoglou, Y., & Schorlemmer, M. (2003). Ontology Mapping: The State of the Art. *Knowl. Eng. Rev., 18*(1), 1-31.

Kuo, Y.-T., Lonie, A., Sonenberg, L., & Paizis, K. (2007). *Domain ontology driven data mining: a medical case study*. Paper presented at the Proceedings of the 2007 international workshop on Domain driven data mining, San Jose, California.

Neches, R., Finin, R. F. T., Gruber, T., Patil, R., Senator, T., & Swartout, W. R. (1991). Enabling Technology for Knowledge Sharing. *AI Magazine, 12*(3), 35-36.

Noy, N. F., & McGuinness, D. L. (2001). Ontology Development 101: A Guide to Creating Your First Ontology.

Sarnikar, S., & Deokar, A. (2010, 5-8 Jan. 2010). *Knowledge Management Systems for Knowledge-Intensive Processes: Design Approach and an Illustrative Example.* Paper presented at the System Sciences (HICSS), 2010 43rd Hawaii International Conference on.

Thompson, I. (2005). Pragmatic Ontology I: Identifying Propensity as Substance. *Generative Science*. Retrieved from

Uschold, M., & Gruninger, M. (1996). Ontologies: Principles, methods and applications. *Knowledge Engineering Review, 11*, 93-136.

Van Heijst, G., Schreiber, G., Wielinga, B. (1997). Using Explicit Ontologies in KBS Development. *Int. J. Human-Computer Studies, 46*, 183-292.

Wache, H., Vögele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., & Hübner, S. (2001). Ontology-Based Integration of Information - A Survey of Existing Approaches. 108-117.

APPENDIX A: Evaluation of pragmatic methodology using success/failure factors of Knowledge Management System development process (Dehghani & Ramsin, 2015).

Criteria for evaluation of special features of KMS development methodologies-based on success/failure factors			
Name		**Type**	**Result**
Determination of appropriate tools and technologies		Multilevel	☐ Enforced ▨ Not enforced, but using specific tools and technologies is advised. ☐ Not addressed.
Support for KM process.	Embed knowledge-source detection features like knowledge-map	Binary	Positive
	Periodical evaluation of knowledge content	Binary	Positive
	Embedding knowledge storage features	Binary	Negative
	Monitoring the KM process	Binary	Negative
Prototyping		Binary	Positive
Embedding diverse channels for knowledge transition		Binary	Positive
Embedding required features to access the knowledge at any Time and Place		Binary	Positive
Specification of the requirements at Different Levels of Users		Binary	Negative.
Specification of the appropriate time to obtain the Knowledge		Binary	Negative.
Documenting the Problem and System Domain Definition Concepts		Binary	Positive
Specification of appropriate Architecture		Binary	Positive
Specification of Organizational Knowledge Taxonomy		Binary	Positive
Identification and Encoding of Expert Knowledge		Binary	Positive
Prioritization		Binary	Positive
Providing documents for Development and Maintenance Phases		Binary	Positive
Embedding features to receive request knowledge by users		Binary	Negative
Embedding features for monitoring justice-based efficiency of KMS		Binary	Negative
Support for management of human resources		Binary	Negative
Gathering knowledge based on Knowledge Requirements		Binary	Positive
Compatibility check of selected technologies		Binary	Negative
Formation of Maintenance Team(s)		Multilevel	☐ Addressed the formation of maintenance team(s) and has provided criteria for selecting team members. ☐ Only addressed the formation of team(s). ▨ Not addressed.
Checking compatibility with other organizational systems.		Binary	Positive
Embedding the features for monitoring knowledge flows.		Binary	Positive
Provision of methods to extract hidden knowledge of experts		Binary	Negative
User-friendly UI design		Binary	Negative
Basis in practical experiences		Binary	Positive
Periodical validation		Binary	Positive
Embedding knowledge-sources search features		Binary	Negative
Prevention of invalid knowledge encoding		Binary	Positive
Determination of characteristics of organizational knowledge	Determining security levels of organizational knowledge.	Binary	Negative
	Determining obstacles to achieve organizational knowledge.	Binary	Negative
Attention to distinctive characteristics of tacit and explicit knowledge.		Binary	Negative
Planning for tacit KM		Multilevel	☐ Specifically enforced. ▨ Generally enforced. ☐ Not enforced.
Detection of organizational knowledge flows			☐ Activities are specified for knowledge flow discovery. ▨ Only guidelines are provided for this task. ☐ Not addressed.