Measuring Long-Term Ontology Quality: A Case Study from the Automotive Industry

Nestor Rychtyckyj

Global Manufacturing Engineering Systems
Ford Motor Company
Dearborn, MI 48121
nrychtyc@ford.com

Abstract

The use of ontologies based on knowledge representation architectures to support search and other decision-making problems in production environments has become a critical component of information systems. The process of building such an ontology can now take advantage of tools such as Protégé (Gennari et al. 2002) to build an ontology for any given problem domain. There has also been corresponding work done on the development of tools and utilities that measure the "quality" of an ontology and the metrics that can be used to measure different facets of the ontology. In this paper we analyze an existing ontology that has been in use for fifteen years in the domain of process planning for automotive assembly. This system, originally known as the Direct Labor Management System (DLMS), was developed and deployed at Ford Vehicle Operations in the early 1990s (Rychtyckyj 1999). The requirement for maintaining the DLMS knowledge base over the last fifteen plus years has given us a unique perspective into the various maintenance problems and issues that need to be addressed. This paper will discuss those issues and try to frame the ontology quality issue in terms of our experience at Ford Motor Company.

Introduction

The term "ontology" has passed from the computer science domain into the business world with the usual accompanying shifts in meaning. In the traditional AI world, an ontology is a shared model or representation of a domain that describes the concepts and relationships that exist within that particular domain. An ontology may be based on a taxonomy-like model, but that is not required. The goal of building an ontology is to enable knowledge sharing and reuse among the users of the ontology; these may be human and knowledge-based systems. The work required to build ontologies that represent complex domain models is very substantial, and the return on investment for all of this effort must be justified. For the last fifteen years we have been building and maintaining ontologies that model vehicle manufacturing at Ford Motor Company.

The use of KL-ONE (Brachman & Schmolze 1985) and associated knowledge representation systems for building large complex knowledge bases to support real-world problems has been demonstrated in various application areas (Brachman et al. 1991). One such system is Ford's Direct Labor Management System (DLMS) that has been used since 1990 in the very dynamic domain of process planning for vehicle assembly (Rychtyckyj 1996, Rychtyckyj 1999). The DLMS system has since been modified and integrated into the Global Study Process Allocation System (GSPAS). The long-term maintenance of the DLMS knowledge base has demonstrated both the flexibility and reliability of semantic network-based knowledge bases in a rapidly changing industrial setting. The most critical issue in utilizing knowledge-based systems over a long period of time is the maintainability of the system. Having maintained the GSPAS ontology over these intervening years, we have had to make changes to both the internal knowledge representation and to the system architecture. All of these changes have introduced modifications to the system that can cause unforeseen problems with the system output. Our experience over this time period has given us a unique perspective into the various problems and issues that need to be addressed. These issues focus on the processes and tools that are needed to validate and verify the ontology since it is updated frequently. This allows us to keep up with rapidly changing market conditions. The modifications made to a semantic-network based knowledge base will also impact the structure and design of the network and may degrade the system performance over time if adjustments are not made.

One of our goals in writing this paper is to improve the communication between the KR research community and the business world where knowledge-based systems are deployed and maintained. Our paper will focus on the maintainability of our ontology and discuss several approaches to improve maintenance. One such approach is a method to automatically generate test cases to validate the correctness of the knowledge base as part of the

maintenance process. Another approach is to use evolutionary computation to analyze the knowledge base as a tool to help us maintain and re-engineer as required. We will also give specific examples of knowledge base design decisions that had either a positive or negative effect on future maintenance. A discussion on using ontology metrics will also be included. With this paper, we hope to demonstrate what issues are important in maintaining an ontology base in a dynamic business environment over a long-term period.

In this paper we will discuss the design of the KL-ONE-based DLMS/GSPAS manufacturing ontology and its use in the domain of automobile assembly planning. A brief description of DLMS is contained in the next section. The following section will focus on the evaluation and validation of the DLMS knowledge base and discuss the various techniques that were utilized for this task. The subject of ontology metrics and their usefulness in evaluation ontology quality are then examined. The paper concludes with a discussion of "lessons learned" from long-term experience with maintaining an ontology in a dynamic problem domain, such as automobile assembly.

The Direct Labor Management System

The Direct Labor Management System (DLMS) is utilized by Ford Motor Company's Vehicle Operations division to manage the use of labor on the assembly lines throughout Ford's vehicle assembly plants. DLMS was designed to improve the assembly process planning activity at Ford by achieving standardization within the vehicle process build description and to provide a tool for accurately estimating the labor time required to perform the actual vehicle assembly. In addition, DLMS provides a framework for allocating the required work among various operators at the plant. It also builds a foundation for the automated machine translation of the process descriptions into foreign languages, a necessity in the current global business market.

The standard process-planning document, known as a process sheet, is the primary vehicle for conveying the assembly information from the initial process planning activity to the assembly plant. A process sheet contains the detailed instructions needed to build a portion of a vehicle. A single vehicle may require thousands of process sheets to describe its assembly. The process sheet is written by an engineer utilizing a restricted subset of English known as Standard Language. Standard Language allows an engineer to write clear and concise assembly instructions that are machine-readable. The process sheet is then sent to the DLMS system to be "validated" before it can be released to the assembly plants. Validation includes the following: checking the process sheet for errors, generating the sequence of steps that a worker at the assembly plant must perform in order to accomplish this task and calculating the length of time that this task will require. The DLMS system interprets these instructions and generates a list of detailed actions that are required to implement these instructions at the assembly plant level. These work instructions, known as “allocatable elements,” are associated with MODAPTS (MODular Arrangement of Predetermined Time Standards) codes that are used to calculate the time required to perform these actions. MODAPTS codes are widely utilized within Industrial Engineering as a means of measuring the body movements that are required to perform a physical action and have been accepted as a valid work measurement system (Carey 2001).

The allocatable elements generated by DLMS are used by engineering personnel at the assembly plant to allocate the required work among the available personnel. DLMS is a powerful tool because it provides timely information about the amount of direct labor that is required to assemble each vehicle, as well as pointing out inefficiencies in the assembly process.

All of the associated knowledge about Standard Language, tools, parts, and everything else associated with the automobile assembly process, is contained in the DLMS knowledge base or ontology. This knowledge base structure is derived from the KL-ONE family of semantic network structures and is an integral component in the success of DLMS.

The organization of the ontology is based on the KL-ONE model. The root of the semantic network is a concept known as THING that encompasses everything within the DLMS world. The children of the root concept describe various major classes of knowledge and include such concepts as TOOLS, PARTS and OPERATIONS. Each concept contains attributes or slots that describe that object. The values of these attributes are inherited from the concept's parents. Ranges of valid values can be given for any particular attribute. Any attempt to put an invalid value in that attribute will trigger an error. All of the information pertaining to the organization and structure of the ontology is also contained in the ontology itself. There are four types of links that describe the relationship between any two concepts: subsumes, specializes, immediately-subsumes and immediately-specializes. The subsumption relation describes a link between a parent concept and all of its children, including descendants of its children. The "immediately-subsumes" relation describes only the concepts that are direct descendants of the parent concept. The "specializes" and "immediately specializes" relations are inverses of the subsumption relation. A concept "immediately specializes" its direct parent concepts and "specializes" all of the concepts that are ancestors of its parents. These relationships are stored as attributes of any given concept. They can be utilized to trace any concept through the entire ontology.

The DLMS system utilizes a classification algorithm to create concepts and place them into their appropriate position in the ontology. The classifier utilizes various

attributes of the concept in order to place it into its correct position. These "classifiable" attributes are slot values that play a major role in determining where a concept belongs. For example, the attribute "size" is very important in classification, while the "output format" slot has little value in classification. Classification is performed by finding the appropriate subsumers, linking the concept and then locating all the concepts that should be subsumed by the new concept. The system narrows this search procedure considerably by selecting the appropriate node in the concept to begin the classification process. The concept that is to be classified is placed at the starting node; the system then tries to push the new concept node as far down the tree as possible. The classifiable attributes are used as objective measures to determine if the concept is in its proper place. Within DLMS, this classification algorithm is applied to all of the instances of the input string that describe the process element. In a simple element this may include a verb, an object and an associated tool. When the classifier is complete, each of the above instances will inherit necessary values from the knowledge base in order to build the appropriate operation to describe the required actions. Figure 1 displays the components of DLMS. These components include the following: parser, analyzer, simulator, error subsystem, knowledge base manager and the ontology. DLMS interfaces with the GSPAS database and is integrated within the GSPAS process flow.

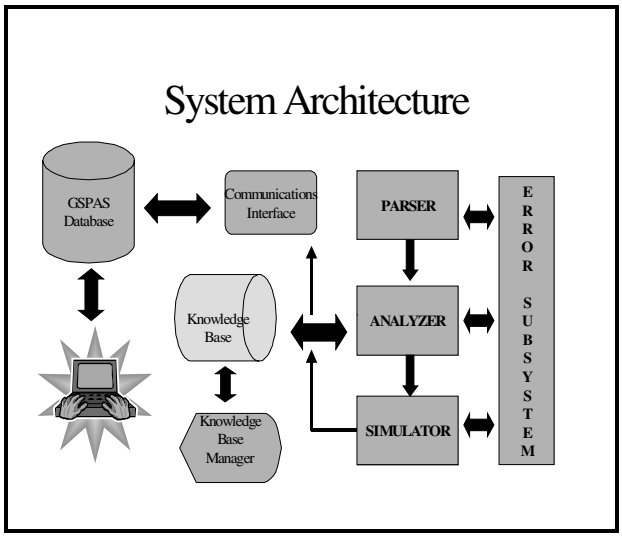

Figure 1: DLMS System Architecture

Currently the DLMS ontology contains over 10,000 concepts and each concept may contain up to 69 different attributes or properties. These concepts are divided into more than 800 classes and the number of links within the system exceeds 110,000. The knowledge encoded into our knowledge base can be divided into Standard Language lexical terms (1/3 of the knowledge base) with the remainder being the tools, parts and operations that describe our manufacturing assembly process. Each concept is described in terms of its properties and its links to parent and child nodes. Figure 2 displays a portion of the DLMS manufacturing ontology. This portion of the ontology contains information about ergonomics within manufacturing and is used to ensure that the work descriptions described conform to ergonomic standards. Each outlined term is a concept and contains a set of properties and values; the ontology is updated through this graphical interface. Edits to the ontology are checked to ensure that they do not violate any constraints.

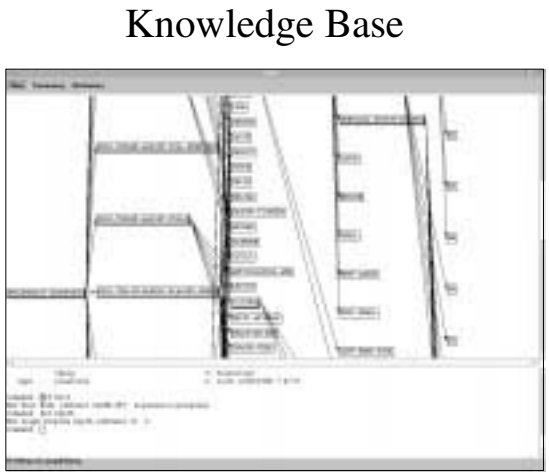

Figure 2: The DLMS manufacturing ontology

Evaluation of the Manufacturing Ontology

As mentioned previously, the DLMS ontology or knowledge base contains all of the relevant information that describes the vehicle assembly process at Ford Motor Company. This includes all of the lexical classes included in Standard Language such as verbs, nouns, prepositions, conjunctions and other parts of speech, various tools and parts utilized at the assembly plants, and descriptions of operations that are performed to build the vehicle.

The DLMS Knowledge Base is maintained through the use of the Knowledge Base Manager (KBM). The Knowledge Base Manager is a graphical tool that is used by the system developers to make important changes to the knowledge base that will affect the actual output generated by the system. Since this output will have a major impact on the assembly process, any such change must be approved by a committee representing all of the interested parties. All changes made to the ontology are logged by the system in order to keep a record of the system's modification history.

The first order of business in knowledge base validation and verification is to develop a baseline of test results that

have been manually inspected and judged to be correct by the user community. Any changes to the ontology are migrated into production only after the regression tests have been completed and accepted. A utility is used to run the series of regression tests to test the system output for a variety of inputs, and the results are then compared to the previous baseline. Any changes that have been introduced are manually examined, and the change is either accepted or rejected. If the change is accepted, a new baseline will be created for this test. A rejected test forces the developers to correct the ontology until the regression test is deemed to be acceptable.

The suite of regression tests must be constantly updated to add and to remove tests as the situation warrants. This updating of the regression tests is done in two different ways: manually and automatically. Manual updates are performed when a developer recognizes that a certain script must be added in order to check a particular scenario. These manual scripts are usually complex, and they include a series of tests that have caused problems for the users. We have also developed a utility that generates test cases automatically against all of the operations in the ontology. An operation can be described as a unique combination of a verb, object and any associated modifiers that will create a distinct work instruction for the assembly plant. This process is accomplished by reading through all of the operations in the ontology, generating a script that will test all of the properties of a particular node by analyzing the attributes of that node, and creating a test that will check those attributes. This test ensures that all of the operations in the ontology execute successfully and return the correct results. Each of these test cases is then executed; the results are then compared against the previous baseline. As with the manually created text cases, the baseline is updated to reflect any changes that have been made to the knowledge base.

However, the use of automatic generation tools has not replaced all of the manually created test scripts due to the following reasons. First, the test creation utility cannot generate complex test scripts that may require a sequence of instructions to fully represent a particular scenario. Second, since the knowledge base is frequently changed, it is often necessary to modify the test utility to keep it current and complete. Third, it is not possible to use the test utility to generate any scripts that model missing or invalid knowledge since that knowledge is not already contained in the ontology. Nevertheless, we have found the use of automated test utilities to be a very useful and productive method to assist with ontology maintenance.

Another approach that was utilized was based on evolutionary computational techniques to analyze and re-engineer the structure of the ontology (Rychtyckyj & Reynolds 2005). The purpose of this effort was to increase the efficiency of the ontology queries by restructuring the knowledge base using a form of evolutionary computation, known as Cultural Algorithms (Reynolds 1994). This approach was successful in terms of re-organizing the network to decrease the cost of subsumption in a typical query by a factor of five. However, it must be noted that the evolutionary approach sometimes develops ontology classes that do not accurately reflect the real-world model and are difficult to understand and maintain.

Ontology Metrics

The recent interest in ontology development has also led to methods for measuring ontology quality. There have been tools developed that attempt to measure the correctness and completeness of an ontology. There are also specific metrics that can be used to measure various facets of an existing ontology (Cross & Pal 2005). The measures that can be used to evaluate the quality of an ontology include the conceptualization complexity, the expressiveness of the ontology language and ontology metrics. The usefulness and completeness of an ontology is directly related to the context in which this ontology is used. In our application, the expressiveness of ontology language is based on the relation of the Standard Language input to the ontology. Since Standard Language is a controlled language, it is not difficult to ensure that all valid Standard Language sentences are correctly represented in the DLMS ontology. Problems with ambiguity do appear on occasion, but can usually be addressed by creating a more specialized concept that can be correctly classified. The main issues with the ontology occur when integrating information that is not in Standard Language. In those cases, we need to be able to identify and match concepts that may be written using different terminology but represent the same entity. Therefore, we use algorithms that utilize various techniques such as matching terms, synonym and abbreviation lookup, spellchecking and heuristics to determine if we are dealing with the same concept. Obviously, these kinds of approaches are less accurate and lead to more errors. This type of "ontology integration" for product life cycle management in the automobile industry has been described using both database structures and business logic (Maier 2003.) In our case, we have also found that terminology plays a critical role in trying to integrate information from different sources. The same concept can be easily described using dissimilar terms and acronyms, abbreviations, misspellings and slang terminology will often cause problems. We have spent considerable time and effort to develop synonym lookups, parsing algorithms and other text-processing techniques to help identify the same concept in different knowledge sources.

Another approach to measuring ontology quality has been to rank ontologies based on the analysis of concept structures (Alani & Brewster 2005). This ontology ranking system, known as AKTiveRank, applies a number of analytic methods to rate each ontology on how well it represents a problem domain through the use of the given

search terms. In our experience, ontology completeness does have significant dependence on finding the specific concepts that match up to a search term. However, the deeper knowledge that is contained within an ontology requires much more information that is not easily identified through search terms or queries. Another significant issue is to get agreement about the specific meaning or use of term that may mean different things to subject matter experts in Europe and the USA.

Since our ontology architecture predates most of the ontology metrics described here, we have had to develop our own internal metrics to evaluate our ontology quality. As we move to integrate our ontology with other knowledge sources we plan to upgrade our architecture and utilize some of the ontology metrics that have been described here. In this paper we will focus on the ontology metrics that were developed for our application. One important facet of ontology quality related to constraint checking. We have developed a utility to scan our ontology for the following types of problems: domain violations, range violations, attribute inverse properties, node value restrictions and forced attribute restrictions. This report allows to identify and correct potential problems in the ontology and to track the ontology performance over time. In terms of ontology metrics, we also utilize both size and structural metrics. Size metrics are mostly used to determine the growth in size of the ontology and to measure various components of the ontology in terms of the number of attributes and properties that are being utilized. Other facets of the ontology, such as number of links, breadth and depth of classes in the ontology, and structural components are also measured on a regular basis. In practice, we have found that concepts that are added into the ontology will stay in the ontology even though they may not be needed anymore. Tools, such as Ontoclean (Guarino and Welty 2002), have been developed to evaluate the ontological decisions during the process of building an ontology. There has even been research to determine if the effort associated with cleaning up an ontology is worth the benefit of improved performance (Welty et al. 2004). Of course, the real-world problem domain that was modeled when the ontology was first developed may have drastically changed in the intervening time period. In our case, the automotive industry has gone through dramatic changes over the last fifteen years and our ontology has been changed to reflect this dynamic processes. The numbers of changes to our ontology for the past fifteen years are shown in Figure 3. This difference in the number of ontology changes over the years can be directly correlated to changes in the underlying manufacturing processes. For example, the large numbers of updates in 2005 were the result of major changes in the Standard Language work descriptions. Other changes in the early 1990s were required to introduce knowledge about manufacturing processes in Europe as they were integrated into our system.

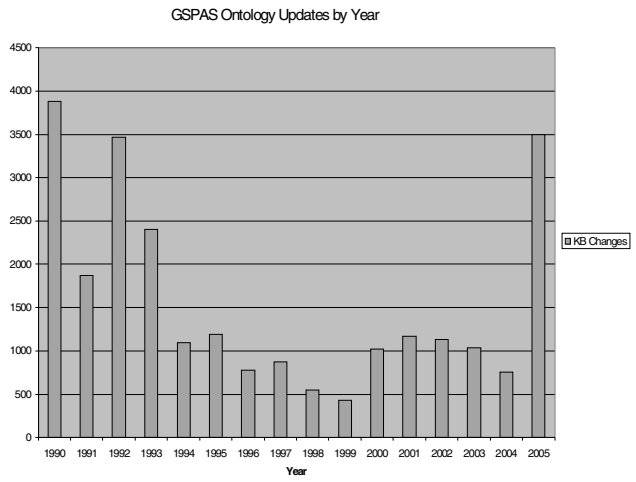

Figure 3: Number of KB updates per year 1990-2005.

Conclusions

In this paper, we discussed some the issues relevant to long-term utilization of knowledge representation systems based on our experience at Ford with DLMS. The recent growth in ontology development and work has given us an opportunity to look back at our experiences with maintaining a manufacturing ontology in the context of some of the recent work on ontology evaluation and metrics. Ontologies are highly dependent on the problem domain that they are modeling, and their completeness must be evaluated in terms of that context. Therefore, our experiences with the automotive manufacturing knowledge may not apply to other problem domains. One important fact that we learned was that it's very difficult to take knowledge out of an ontology; once something gets included – it usually stays in. This is due to several reasons: this information may be needed again in the future, removing information may lead to errors in the system and "ontology cleaning" is never a high priority. This type of inertia can make the ontology grow substantially over time. It's also usually not a good idea to allow user updates to an ontology without very strict controls. We tried this for several years, but discovered that the effort required to verify and fix these user updates was significantly higher than updating the ontology ourselves. An exhaustive validation and verification process is essential; there is nothing that the user community dislikes more than getting unexpected and unexplained results when the input query did not change. Ontology metrics, including performance, are crucial to help detect issues before they affect the user community. However, the most important factor in maintaining any

type of ontology or knowledge base over a prolonged time period is to have a flexible framework that allows for all sorts of unexpected but required changes.

Our experience has shown that knowledge representation systems based on semantic networks provide an excellent framework for developing systems that can exist in a dynamic environment. Long-term maintenance of such systems requires both the development of processes to support the system, as well as the corresponding software tools needed to implement these processes. Flexibility is the key requirement of knowledge representation systems, as the business environment will certainly change in ways that could not be anticipated by the developers. It is also important to utilize new approaches as they become available in order to assist the developers in maintaining and re-engineering the knowledge base over its life cycle. All of these factors contribute to the successful use of knowledge representation systems in very dynamic problem domains as evidenced here.

Acknowledgements

The author thanks the AAAI reviewers for their insightful comments; in addition I would like to thank Mike Rosen, Alan Turski, Rick Keller and Tom Vitale for their work on DLMS. I would also like to thank Erica Klampfl for her assistance in the preparation of this paper.

References

Alani, H., Brewster, C., (2005), "Ontology Ranking based on the Analysis of Concept Structures", *Proceedings of the Third International Conference on Knowledge Capture (K-CAP'05)*, Banff, Alberta, Canada, ACM Press, pp. 51-58.

Brachman, R., Schmolze, J., (1985), "An Overview of the KL-ONE Knowledge Representation System," *Cognitive Science* 9(2), pp. 171-216.

Brachman, R., McGuiness, D., Patel-Schneider, P., Resnick, L., Borgida, A., (1991) "Living With Classic: When and How to Use a KL-ONE-Like Language" in *Principles of Semantic Networks*, ed. J. Sowa, pp. 401-456, Morgan Kaufmann Publishers.

Carey, Farrell, Hui, and Sullivan (2001), *Heyde's Modapts: A Language of Work*. Heyde Dynamics PTY LTD.

Cross, V., Pal, A., (2005), "Metrics for Ontologies", *Proceedings of the North American Fuzzy Information Processing Society (NAFIPS-2005*), Ann Arbor, MI, June 22-25, 2005, pp. 448-453.

Gennari, J., Musen, M., Fergerson, R., Grosso, W., Crubezy, M., Eriksson, H., Noy, Tu, S., (2002), "The Evolution of Protégé: An Environment for Knowledge-Based System Development", *Stanford Medical Informatics Technical Report 2002-0943.*

Guarino, N., Welty, C., (2002), "Evaluating Ontological Decisions with Ontoclean*", Communications of the ACM*, vol. 45, no. 2, pp. 61-65.

Maier, A., Schnurr, H., Sure, Y.,(2003), "Ontology-based Information Integration in the Automotive Industry", *Proceedings of the 2nd International Semantic Web Conference* (*ISWC-2003*), *Lecture Notes on Computer Science*, vol. 2870, Springer-Verlag, pp. 897-912.

Reynolds, Robert G., (1994), "An Introduction to Cultural Algorithms", *Proceedings of the 3rd annual Conference on Evolution Programming"*, Sebalk, A.V. Fogel L.J., River Edge, NJ. World Scientific Publishing, 1994, pp 131-136.

Rychtyckyj, N. (1996), "DLMS: An Evaluation of KL-ONE in the Automobile Industry". in *Proceedings of the Fifth International Conference on the Principles of Knowledge Representation and Reasoning*, pp. 588-596. Morgan Kaufmann Publishers.

Rychtyckyj, N., Reynolds, R.G., (2005), "Using Cultural Algorithms to Re-Engineer Large-Scale Semantic Networks" in the *International Journal of Software Engineering and Knowledge Engineering*, vol. 15, no. 4, pp. 665-693.

Rychtyckyj, N., (1999), "DLMS: Ten Years of AI for Vehicle Assembly Process Planning", *AAAI-99/IAAI-99 Proceedings*, Orlando, FL, July 18-22, 1999, pp. 821-828, AAAI Press.

Welty C., Mahindru R., Chu-Carroll J., (2004), "Evaluating Ontology Cleanup", *Proceedings of the Nineteenth Conference on Artificial Intelligence*, San Jose, CA, pp. 311-316.