Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

Knowledge and Ontological Engineering Techniques for Use in Developing Knowledge-Based Situation Analysis Support Systems

Jean Roy
Alain Auger
DRDC Valcartier

Defence R&D Canada – Valcartier
Technical Memorandum
DRDC Valcartier TM 2006-757
October 2008

Canada

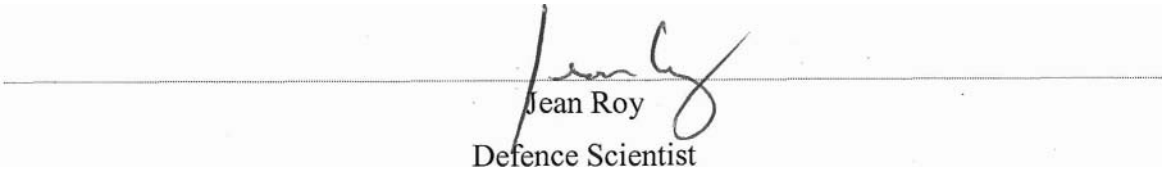
Knowledge and Ontological Engineering Techniques for Use in Developing Knowledge-Based Situation Analysis Support Systems

Jean Roy
Alain Auger
DRDC Valcartier

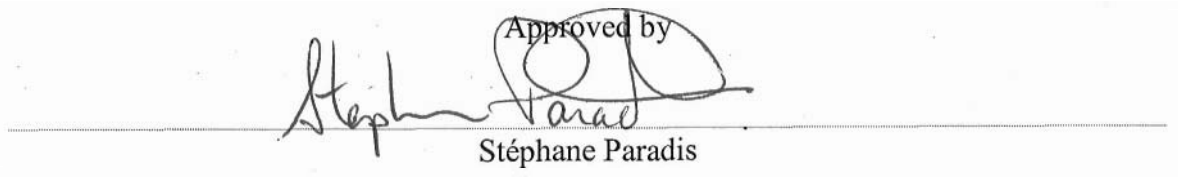
Defence R&D Canada – Valcartier

Technical Memorandum
DRDC Valcartier TM 2006-757
October 2008

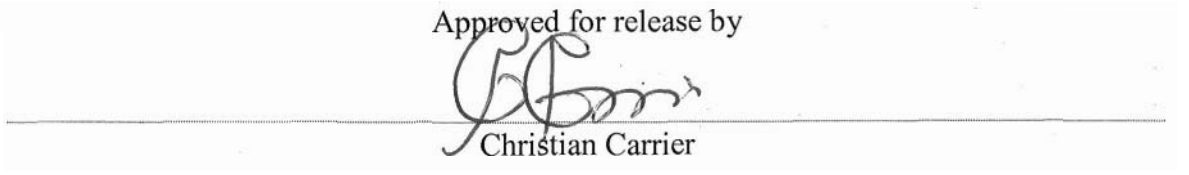
Principal Author



Jean Roy
Defence Scientist



Approved by
Stéphane Paradis
Section Head / Intelligence & Information Section, DRDC Valcartier



Approved for release by
Christian Carrier
Chief Scientist, DRDC Valcartier

This document results from S&T work conducted under DRDC projects: 11bh (Halifax Class Situation Analysis Support Systems), 11hg (Collaborative Knowledge Exploitation for Maritime Domain Awareness), and 120f (Situation Analysis for the Tactical Army Commander)

Abstract

Adopting a knowledge-centric view of situation analysis and information fusion requires that one cares about knowledge engineering (KE) and ontological engineering (OE) techniques. In this regard, this memorandum discusses many aspects of KE and OE, including the basic elements usually incorporated in a knowledge base (KB), the notion of a knowledge domain, and also axioms, definitions and theorems. The properties of good and bad KBs are described, and the distinction between KBs and databases is made. Some of the skills and characteristics that are desirable in knowledge engineers are listed. Discussions on KE versus programming, and also on the synergistic effect of KE, especially regarding knowledge creation and discovery, are provided. The memorandum talks about the main difficulties in knowledge acquisition, e.g., expressing, transferring and structuring the knowledge, and how these difficulties can be somewhat surmounted. Some of the main methods of knowledge acquisition, especially the various forms of interview with subject-matter experts, are reviewed. A methodology that can be used for the development of a KB is presented. The ontology development process, which refers to the management, development and support activities performed when building ontologies, is discussed, along with the evolution of the ontology development methodologies. The memorandum contributes to the development of a foundational R&D framework for two projects at Defence R&D Canada (11hg, Collaborative Knowledge Exploitation for Maritime Domain Awareness, and 12of, Situation Analysis for the Tactical Army Commander).

Résumé

Adopter une vue de l'analyse de la situation et de la fusion d'information centrée sur la connaissance requiert que l'on se soucie des techniques de l'ingénierie de la connaissance (IC) et de l'ingénierie ontologique (IO). À ce propos, ce mémoire discute plusieurs aspects de l'IC et de l'IO, incluant les éléments de base habituellement incorporés dans une base de connaissance (BC), la notion d'un domaine de connaissances, et aussi les axiomes, définitions et théorèmes. Les propriétés des bonnes et mauvaises BCs sont décrites, et la distinction est faite entre BCs et bases de données. Quelques unes des habiletés et caractéristiques qui sont désirables des ingénieurs de connaissances sont listées. Des discussions sont fournies concernant l'IC versus la programmation, et aussi sur l'effet synergique de l'IC, tout spécialement en ce qui concerne la création et la découverte de la connaissance. Le mémoire parle des principales difficultés de l'acquisition de la connaissance, e.g., exprimer, transférer et structurer la connaissance, et de comment ces difficultés peuvent, d'une certaine façon, être surmontées. Certaines des principales méthodes d'acquisition de connaissances, tout spécialement les différentes formes d'entrevues avec des experts du domaine, sont passées en revue. Une méthodologie qui peut être utilisée pour le développement d'une BC est présentée. Le processus de développement d'une ontologie, qui réfère aux activités de gestion, de développement et de soutien qui sont effectuées lors de la construction des ontologies, est discuté, en plus de l'évolution des méthodologies de développement des ontologies. Le mémoire contribue à la mise en place d'un cadre de référence de R&D pour deux projets à R&D pour la défense Canada (11hg, Exploitation collaborative de la connaissance pour l'éveil situationnel du domaine maritime, et 12of, Analyse de la situation pour le commandant d'armée tactique).

This page intentionally left blank.

Executive summary

Knowledge and Ontological Engineering Techniques for Use in Developing Knowledge-Based Situation Analysis Support Systems

Roy, J., Auger, A.; DRDC Valcartier TM 2006-757; Defence R&D Canada – Valcartier; October 2008.

Over the years, situation awareness has emerged as an important concept supporting dynamic human decision-making in both military and public security environments; situation analysis is defined as the process that provides and maintains a state of situation awareness for the decision makers. Information fusion is a key enabler to meeting the demanding requirements of situation analysis in future command and control (C2) and intelligence support systems.

Adopting a knowledge-centric view of situation analysis and information fusion (SAIF) ultimately requires that one cares about knowledge representation (KR) and reasoning, i.e., the area of artificial intelligence (AI) concerned with how knowledge can be represented symbolically (i.e., using formal symbols to represent a collection of propositions believed by some putative agent) and manipulated in an automated way by reasoning programs. This being said, adopting this view also requires that a corresponding well-structured development process be defined and followed to build appropriate knowledge-based situation analysis support systems (SASSs). Combining elements of information fusion and knowledge-based systems, [Roy, 2007-B] has already presented a holistic approach and framework, including some recommended high-level steps, for the overall development and evaluation of any knowledge-based SASS. This memorandum presents knowledge and ontological engineering techniques, which are essential aspects of this approach.

Knowledge engineering is the process of building a knowledge base from human expert knowledge. It involves several tasks such as knowledge acquisition, representation and validation. Ontological engineering refers to the set of activities that concern the ontology development process and life cycle, the methods and methodologies for building ontologies, and, the tool suites and languages that support them.

The memorandum doesn't present “solutions” or findings and results of completed activities. The intent is more to provide a comprehensive description of knowledge and ontological engineering, in order to contribute to the development of a foundational R&D framework for two projects that are just starting under the Applied Research Program (ARP) at Defence R&D Canada: SATAC (Situation Analysis for the Tactical Army Commander), and CKE-4-MDA (Collaborative Knowledge Exploitation for Maritime Domain Awareness). Aspects being discussed include the basic elements usually incorporated in a knowledge base, the notion of a knowledge domain, and also axioms, definitions and theorems. The properties of good and bad knowledge bases are described, and the distinction between knowledge bases and databases is made. Some of the skills and characteristics that are desirable in knowledge engineers are listed. Discussions on knowledge engineering versus programming, and also on the synergistic effect of knowledge engineering, especially regarding knowledge creation and discovery, are provided. The memorandum

describes the main difficulties in knowledge acquisition, e.g., expressing, transferring and structuring the knowledge, and how these difficulties can be somewhat surmounted. Some of the main methods of knowledge acquisition, especially the various forms of interview with subject-matter experts, are reviewed. A methodology that can be used for the development of a knowledge base is presented. The ontology development process, which refers to the management, development and support activities performed when building ontologies, is discussed, along with the evolution of the ontology development methodologies.

Situation analysis and information fusion play a critical role in the C2 and intelligence processes, and have already received significant attention for military applications. Numerous ongoing projects of the Department of National Defence (DND) and also at Defence R&D Canada possess an important SAIF component. However, despite the importance given to SAIF in many DND strategic documents and projects, the current systems and the associated technology often fail to meet the demanding requirements of the operational decision makers; major science and technology advances are required to really achieve the full potential of the related enabling technologies and to best serve the operational communities. The effort reported here constitutes one step in this direction; it contributes to the establishment of a solid basis on which a long-term R&D program should be built.

This memorandum only provides an initial overview of the many issues regarding knowledge and ontological engineering techniques; one needs to dig deeper into these issues to really exploit the techniques and develop appropriate support systems meeting the demanding requirements of the operational communities. Hence, R&D activities have been and are still currently being conducted to further investigate knowledge representation concepts, paradigms and techniques, and reasoning processes, methods and systems for use in knowledge-based SAIF support systems. It is also required that knowledge (i.e., expertise) is eventually acquired from the subject matter experts (SMEs) of the different military and public security application domains. In this regard, the techniques discussed in this memorandum have been and are still being investigated at the moment. Knowledge elicitation and validation sessions with SMEs are about to be conducted.

Sommaire

Knowledge and Ontological Engineering Techniques for Use in Developing Knowledge-Based Situation Analysis Support Systems

Roy, J., Auger, A.; DRDC Valcartier TM 2006-757; R & D pour la défense Canada – Valcartier; octobre 2008.

Au fil des ans, l'éveil situationnel a émergé en tant que concept important en soutien à la prise de décision dynamique par des humains dans les milieux militaires et de sécurité publique; l'analyse de la situation est définie comme étant le processus qui fournit et maintient un état d'éveil situationnel pour les preneurs de décisions. La fusion d'information constitue un habilitant clé pour répondre aux besoins exigeants de l'analyse de la situation dans les systèmes futurs de soutien au commandement et contrôle (C2) et au renseignement.

Adopter une vue de l'analyse de la situation et de la fusion d'information (ASFI) centrée sur la connaissance requiert ultimement que l'on se soucie de la représentation de la connaissance (RC) et du raisonnement, i.e., l'aire de l'intelligence artificielle (IA) concernée par comment la connaissance peut être représentée symboliquement (i.e., en utilisant des symboles formels pour représenter une collection de propositions crues par un agent présumé) et manipulée de façon automatique par des programmes qui raisonnent. Ceci étant dit, adopter cette vue requiert aussi qu'un processus de développement bien structuré correspondant soit défini et suivi pour construire des systèmes appropriés de soutien à l'analyse de la situation (SSAS) basés sur la connaissance. Combinant des éléments de la fusion d'information et des systèmes à base de connaissance, [Roy, 2007-B] a déjà présenté une approche et un cadre holistiques, incluant des étapes recommandées de haut niveau, pour le développement global et l'évaluation de tout SSAS basé sur la connaissance. Ce mémoire présente des techniques d'ingénierie de la connaissance et ontologique, qui sont des aspects essentiels de cette approche.

L'ingénierie de la connaissance est le processus de construction d'une base de connaissances. Elle implique plusieurs tâches telles que l'acquisition, la représentation et la validation de la connaissance. L'ingénierie ontologique réfère à l'ensemble des activités qui concernent le processus de développement et le cycle de vie des ontologies, les méthodes et méthodologies pour construire des ontologies, et, les suites d'outils et les langages qui les supportent.

Le mémoire ne présente pas de “solutions” ou de découvertes et résultats d'activités complétées. L'intention est plutôt de fournir une description d’ensemble de l'ingénierie de la connaissance et ontologique, dans le but de contribuer à la mise en place d'un cadre de base de R&D pour deux projets qui démarrent en ce moment sous le programme de recherches appliquées (PRA) à R&D pour la défense Canada: ASCAT (Analyse de la situation pour le commandant d'armée tactique), et ECCESDM (Exploitation collaborative de la connaissance pour l'éveil situationnel du domaine maritime). Les aspects discutés incluent les éléments de base habituellement incorporés dans une base de connaissance, la notion d'un domaine de connaissance, et aussi les axiomes, définitions et théorèmes. Les propriétés des bonnes et mauvaises bases de connaissances sont décrites, et la distinction est faite entre bases de connaissances et bases de données. Quelques unes des habiletés

et caractéristiques qui sont désirables des ingénieurs de connaissances sont listées. Des discussions sont fournies concernant l'ingénierie de la connaissance versus la programmation, et aussi sur l'effet synergique de l'ingénierie de la connaissance, tout spécialement en ce qui concerne la création et la découverte de la connaissance. Le mémoire décrit les principales difficultés de l'acquisition de la connaissance, e.g., exprimer, transférer et structurer la connaissance, et de comment ces difficultés peuvent, d'une certaine façon, être surmontées. Certaines des principales méthodes d'acquisition de connaissance, tout spécialement les différentes formes d'entrevues avec des experts du domaine, sont passées en revue. Une méthodologie qui peut être utilisée pour le développement d'une base de connaissance est présentée. Le processus de développement d'une ontologie, qui réfère aux activités de gestion, de développement et de soutien qui sont effectuées lors de la construction des ontologies, est discuté, en plus de l'évolution des méthodologies de développement des ontologies.

L'analyse de la situation et la fusion d'information (ASFI) ont un rôle critique à jouer dans les processus de C2 et de renseignement et elles ont déjà reçu une grande attention pour les applications militaires. De nombreux projets courants du Ministère de la Défense Nationale (MDN) et aussi à R&D pour la défense Canada comportent une importante composante d'ASFI. Cependant, en dépit de l'importance donnée à l'ASFI dans plusieurs documents et projets stratégiques du MDN, les systèmes actuels et la technologie associée n'arrivent souvent pas à rencontrer les besoins exigeants des preneurs de décisions opérationnels; des percées majeures en science et technologie sont requises pour vraiment atteindre le plein potentiel des technologies habilitantes associées et pour servir au mieux les communautés opérationnelles. L'effort rapporté ici est une étape en ce sens; il contribue à l'établissement d'une base solide sur laquelle un programme de R&D à long-terme devrait être construit.

Ce mémoire ne donne qu'une vue d'ensemble initiale de la multitude d'aspects concernant les techniques d'ingénierie de la connaissance et ontologique; il s’avère nécessaire d'approfondir ces aspects pour exploiter vraiment ces technologies et développer des systèmes de soutien appropriés et rencontrant les besoins exigeants des communautés opérationnelles. Pour cette raison, des activités de R&D ont été et sont encore menées afin d’étudier plus à fond les concepts, paradigmes et techniques de représentation de la connaissance, et les processus, méthodes et systèmes de raisonnement pour leur utilisation dans des systèmes de soutien à l'ASFI basés sur la connaissance. Il est aussi requis que la connaissance (i.e., l'expertise) soit éventuellement acquise des experts du domaine (ED) des différents domaines d'application militaires et de sécurité publique. À ce propos, les techniques discutées dans ce mémoire ont été et sont encore étudiées en ce moment. Des sessions d'élicitation et de validation de connaissances avec des ED sont sur le point d'être effectuées.

Table of contents

List of figures

List of tables

Important Notice

The work resulting in the present technical memorandum began during the summer of 2005, when the authors were about to undertake new research projects involving many aspects of artificial intelligence (AI), knowledge acquisition and representation, automated reasoning, knowledge and ontological engineering, and other related notions. The level of expertise of the people to be involved in the projects regarding these aspects was somewhat uneven, and the need was felt to produce some sort of "primer" document to help the project participants to get up to speed, and to establish some common background (or foundation) for the projects. Along this line of thoughts, available books were quickly identified and gathered, all from well established authors in these fields (Brachman, Levesque, Russell, Norvig, Stefik, Giarratano, Riley, Ginsberg, Turban, Aronson, Gómez-Pérez, Fernández-López, Corcho). Of course, there are many other authors and references that could have been used, but the set was limited to the ones listed here, given that it was felt sufficient to fulfill our objective. Intense reading and animated discussion sessions with colleagues at DRDC Valcartier then occurred, while very detailed notes were taken, and synthesis diagrams created; some of these notes and diagrams compose the essence of this document.

It is very important to note that many text passages in this memorandum have been taken up integrally from the original documents of the authors mentioned above. This was done, on purpose, to assure integrity of the words and ideas as the author(s) meant them. In part this is because of our own limitations in these subjects, and in part because their words are both adequate and terse. Also, the many literal quotations from the original text have been inserted in this memorandum without using quotation marks ("") to highlight the direct quotations, as is normally required. This was done to avoid making the text too busy and cluttered. However, all of the material used is referenced to the appropriate original author(s) in the body of the text, mostly from paragraph to paragraph, but sometime from sentence to sentence. It is thus easy for the reader (and actually recommended) to go back to the original book for more details, or to put the material back in its original context.

All of the authors mentioned above cover a lot of ground in their respective books. The synthesis process for this memorandum was thus an attempt to extract from each one only the aspects judged relevant to our R&D projects, and then to merge the individual ideas into a unified smooth and fluid text. Unfortunately, the attempt was more or less successful, resulting in some mix of styles, which could be annoying to some readers.

Some may fairly note that most of the reference documents are not "recent" documents. Certainly, the current strong interest in ontologies and other related semantic web technologies for example has produced many recent documents that could be used for our purpose. Things have evolved since the summer of 2005, and they still progress at a very fast pace. This being said, although some of the semantic web technologies may look fresh and new, many of the proposed concepts are actually brought back and resurrected from the huge body of existing work in AI. Some of this recent material can in fact be considered as a "variation on a (somewhat old) theme". It may thus be beneficial to go back to the basics of AI first, in order to better appreciate later the more recent advancements that constitute the state-of-the-art at the moment. Moreover, although some aspects, such as expert systems, may rightfully be considered very mature, they

have not yet been fully exploited in the context of higher-level information fusion in knowledge-based situation analysis support systems (KB SASS).

This technical memorandum was not intended as a contribution to the advancement of the AI domain. Furthermore, no explicit and/or firm recommendation is made on the best approaches to build a KB SASS. Such recommendations are yet to come as a result of the current projects being undertaken in specific applied problem areas of interest to our sponsors (e.g., maritime domain awareness, geo-intelligence, situation analysis for the tactical Army commanders). The discussion is more at the general level, directly based on the established references that were selected.

Writing this document has certainly helped the authors to get up to speed regarding the aspects discussed; it is believed (hoped) that it could similarly help other newcomers.

1. Introduction

Situation awareness (SAW) has emerged as an important concept supporting dynamic human decision-making in military and public security environments. Actually, SAW is a general concept that has been shown to be of interest in a very large number of settings. Given this wide interest for SAW, the author has previously proposed another concept, situation analysis (SA), defined as a process that provides and maintains a state of situation awareness for the decision maker(s) [Roy, 2001].

A key enabler to meeting the demanding requirements of situation analysis in future command and control (C2) and intelligence systems (i.e., in achieving high-quality situation awareness for optimal decision making) is data/information fusion. An initial lexicon, presented in [White, 1987], defined data fusion as a process dealing with the association, correlation, and combination of data and information from single and multiple sources to achieve refined position and identity estimates, and complete and timely assessments of situations and threats as well as their significance. This definition has evolved over the years, and multiple variants have been proposed. Recently, [Lambert, 2001] defined information fusion as the process of utilizing one or more information sources over time to assemble a representation of aspects of interest in an environment. Among the many reasons for interest in this domain, there is that data/information fusion:

- provides extended spatial and temporal coverage, increased confidence, reduced ambiguity, improved entity detection, etc.,
- allows for the management of large volumes of information and the correlation of seemingly unrelated, overlooked, or deceptive information to present a coherent representation of an evolving situation to a decision maker, and,
- enables the commander to cope with the complexity and tempo of operations in modern dynamic operational theatres.

Clearly, situation analysis and information fusion (SAIF) have a critical role to play, and have already received significant attention for military applications. Numerous ongoing projects of the Department of National Defence (DND) have an important SAIF component. Examples, just to name a few, are:

- Project No. 00000276: Land Force Intelligence, Surveillance, Target Acquisition and Reconnaissance (LF ISTAR)
- Project No. 00000806: Marine Security Operations Centres (MSOC)
- Project No. 00000624: Joint Information and Intelligence Fusion Capability (JIIFC)

However, despite the importance surrounding SAIF in many DND strategic documents and projects, the current supporting systems and the associated technology often fail to meet the demanding requirements of the operational decision makers, and major Science and Technology (S&T) advancements are required to really achieve the full potential of the related enabling technologies and to best serve the operational communities. In this line of thought, many R&D

projects ongoing under the Technology Demonstration Program (TDP) at Defence R&D Canada (DRDC) have an important SAIF component as well:

- Joint Command Decision Support for the 21st Century (JCDS 21)
- Innovative Naval Combat Management Decision Support (INCOMMANDS)

SAIF is certainly expected to play a crucial role in the next generation of support systems for aiding decision makers in military and public security operations.

Taking into account the context described above, [Roy, 2007-A] proposed that developing and adopting a knowledge-centric view of situation analysis should provide a more holistic perspective of this process, leading to the development of better, more adequate SAIF support systems for the operational communities. This was mostly based on the fact that awareness ultimately has to do with having knowledge of something. As discussed in [Roy, 2007-A], expert systems constitute a branch of artificial intelligence (AI) that makes extensive use of specialized knowledge to solve problems at the level of a human expert. They are the paradigmatic application of AI techniques to hard problems. Although traditional expert systems have shown limitations, expert system technologies are expected to significantly contribute to the development of the future, state-of-the-art knowledge-based situation analysis support systems (SASSs).

Adopting the knowledge-centric view of situation analysis eventually requires that one cares about knowledge representation (KR) and reasoning. The object of KR is to express knowledge in computer-tractable form, such that it can be used [Russell, Norvig, 1995]. KR research studies the problem of finding a language in which to encode the knowledge so that the machine can use it [Ginsberg, 1993]. According to the dictionary [Merriam-Webster, 2003], to reason is “to use the faculty of reason so as to arrive at conclusions” or “to discover, formulate, or conclude by the use of reason”. Similarly, to infer is “to derive as a conclusion from facts or premises”. The terms reasoning and inference are generally used to cover any process by which conclusions are reached. In general, reasoning in knowledge-based systems is the formal manipulation of symbols representing a collection of believed propositions to produce representations of new ones [Brachman, Levesque, 2004].

KR and reasoning is thus the area of artificial intelligence (AI) concerned with how knowledge can be represented symbolically and manipulated in an automated way by reasoning programs [Brachman, Levesque, 2004]. Figure 1, a very busy diagram, illustrates many aspects of knowledge representation and reasoning in knowledge-based systems, from the perspective of first-order logic (FOL). Knowledge representation should support the tasks of acquiring and retrieving knowledge, as well as subsequent reasoning [Turban, Aronson, 1998]. Actually, as illustrated in Fig. 1, there is an interplay between representation and reasoning; it is not enough to write down what needs to be known in some formal representation language, nor is it enough to develop reasoning procedures that are effective for various tasks [Brachman, Levesque, 2004]. Knowledge representation and reasoning is best understood as the study of how knowledge can at the same time be represented as comprehensively as possible and be reasoned with as effectively as possible. There is a trade off between these two concerns. Needing to reason with knowledge structures has an impact on the form and scope of the languages used to represent a system's knowledge.

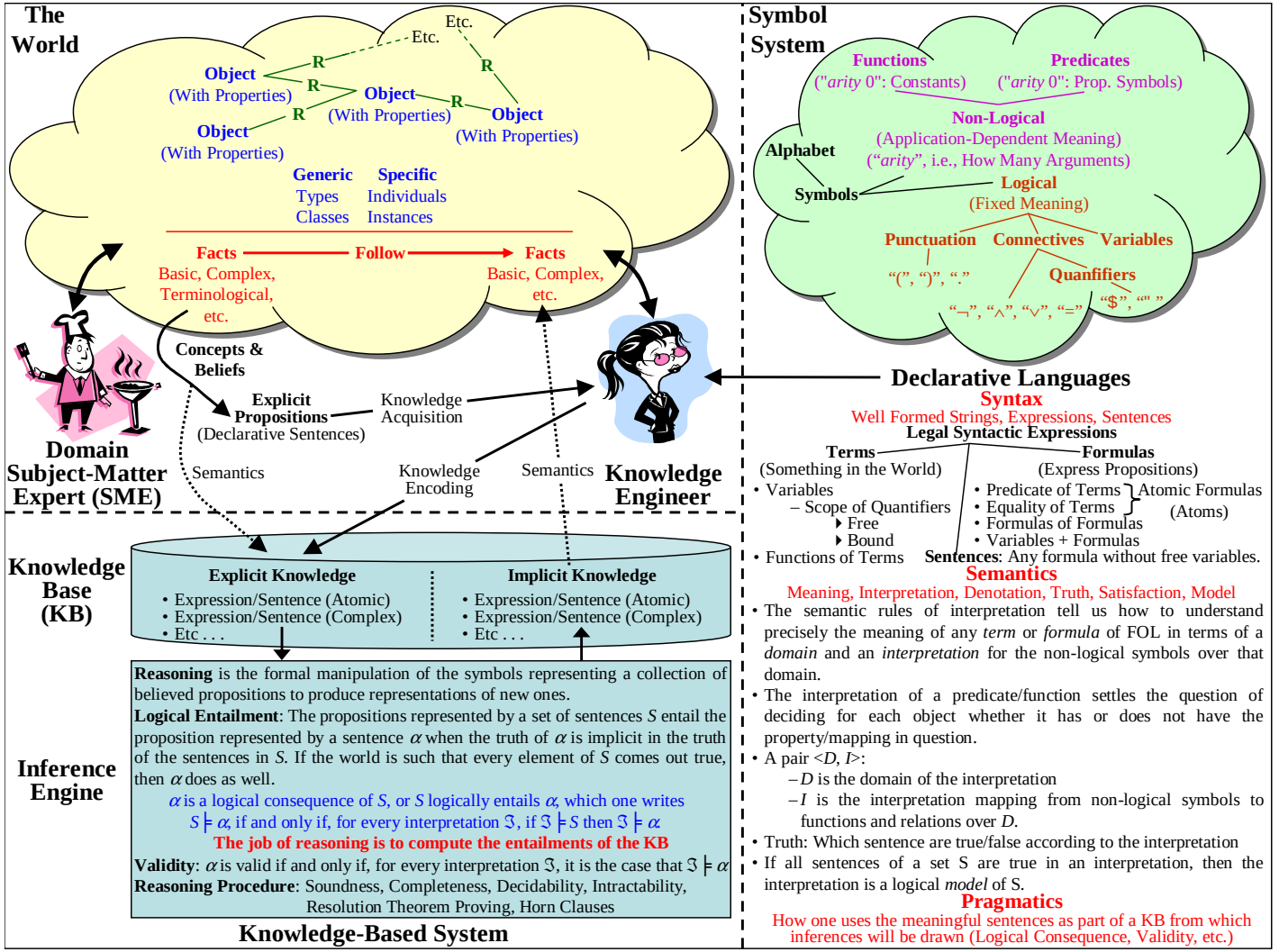


Figure 1: Knowledge representation and reasoning in KBSs (first-order logic perspective)

Although there is this strong link between KR and reasoning, the discussion of these two aspects has previously been split in two documents. [Roy, Auger, 2008-A] talks about knowledge representation concepts, paradigms and techniques for use in knowledge-based SASSs, while [Roy, Auger, 2008-B] discusses reasoning processes, methods and systems for use in such systems.

This being said, adopting the knowledge-centric view of situation analysis also requires that a corresponding well-structured development process be defined and followed to build appropriate knowledge-based SASSs. Combining elements of information fusion and knowledge-based systems, [Roy, 2007-B] presents a holistic approach and framework, including some recommended high-level steps discussed in the report, for the overall development and evaluation of any knowledge-based SASS. Emphasis is given to the knowledge-centric aspects of the approach.

Knowledge and ontological engineering techniques, the object of this memorandum, are essential aspects of the holistic approach presented in [Roy, 2007-B]. On one hand, knowledge engineering is the process of building a knowledge base (i.e., the organized repository for the collection of knowledge related to a domain and used for understanding, formulating, and solving problems in a knowledge-based system). It involves several tasks such as knowledge acquisition, representation and validation. On the other hand, [Roy, Auger, 2008-A] provides a practical definition of the term ontology, and also discusses many aspects related to ontologies, including their utility in SASSs. Ontological engineering refers to the set of activities that concern: the ontology development process, the ontology life cycle, the methods and methodologies for building ontologies, and, the tool suites and languages that support them [Gómez-Pérez et al, 2004]. Ontological engineering is sometime viewed as a subset of knowledge engineering. Actually, according to some wide perspective, the term knowledge engineering describes the entire process of developing and maintaining AI systems.

Note that the memorandum doesn't present “solutions”, or findings and results of completed activities. The intent is more to provide a fair description of knowledge and ontological engineering, in order to contribute to the set up of a foundational R&D framework for two DRDC projects that are just starting under the Applied Research Program (ARP):

- SATAC (Situation Analysis for the Tactical Army Commander), and,
- CKE-4-MDA (Collaborative Knowledge Exploitation for Maritime Domain Awareness), formerly known as AKAMIA (Advanced Knowledge Acquisition for Maritime Information Awareness).

The memorandum is organized as follows. Section 2 begins the discussion of knowledge engineering with a description of the basic elements that are usually included in a knowledge base. The notion of a knowledge domain is briefly presented, along with a discussion on axioms, definitions and theorems. The properties of good and bad knowledge bases are described, and the distinction between knowledge bases and databases is made. Then Section 2 talks about knowledge engineers, the specialists who investigate a particular domain, determine what concepts are important in that domain, and create a formal representation of the objects and relations in the domain. In particular, some of the skills and characteristics that are desirable in knowledge engineers are listed. Section 2 then ends with discussions on knowledge engineering

versus programming, and also on the synergistic effect of knowledge engineering, especially regarding knowledge creation and discovery.

Knowledge is a major resource, and it often lies with only a few experts [Turban, Aronson, 1998]. It is thus important to capture that knowledge so others can use it. Regarding this issue, Section 3 discusses knowledge acquisition, starting with an overview of the main difficulties in knowledge acquisition, e.g., expressing, transferring and structuring the knowledge, and how these difficulties can be somewhat surmounted. Some of the main methods of knowledge acquisition, especially the various forms of interview with subject-matter experts, are then reviewed in Section 3.

The process of representing the knowledge of a domain typically goes through several stages. Section 4 presents a methodology that can be used to help focus the development of a knowledge base and to integrate the knowledge engineer's thinking at the three levels (i.e., domain, representation language, and inference procedure). This methodology goes from understanding the domain and encoding the different types of facts, to posing queries to the inference procedure to get answers, and ultimately to validating and verifying the knowledge.

Then Section 5 presents ontological engineering. The description begins with the ontology development process that refers to which activities are performed when building ontologies. These include the management, development and support activities. The evolution of the ontology development methodologies is then discussed.

Finally, concluding remarks are provided in Section 6.

2. Knowledge Engineering

The process of building a knowledge base is called knowledge engineering [Russell, Norvig, 1995]. Several methods have been published, describing knowledge engineering tasks. One of the most well-known is probably the CommonKads methodology [Schreiber et al., 2002]. Knowledge engineering involves several tasks, and can be viewed from two perspectives: narrow and wide [Turban, Aronson, 1998]. According to the narrow perspective, knowledge engineering deals with:

- knowledge acquisition,
- knowledge representation,
- knowledge validation,
- inference,
- explanation and justification, and
- maintenance.

Alternatively, according to the wide perspective, the term describes the entire process of developing and maintaining AI systems.

2.1 Knowledge Base

A knowledge base is the organized repository for the collection of knowledge related to a domain and used for understanding, formulating, and solving problems in a knowledge-based system [Stefik, 1995]. The basic elements that are usually included in the knowledge base are [Turban, Aronson, 1998]:

- Facts, such as the problem situation and theory of the problem area.
- Special heuristics, rules and hints that direct the use of knowledge to solve specific problems in a particular domain.
- Global strategies, which can be both heuristics and a part of the theory of the problem area.

Informally, a knowledge base is a set of representations of facts about the world [Russell, Norvig, 1995]. The declarative approach to system building expresses knowledge in the form of sentences. This simplifies the construction problem enormously. Knowledge is contained in computer-based agents in the form of sentences in a knowledge representation language that are stored in a knowledge base. There must be a way to add new sentences to the knowledge base, and a way to query what is known.

Once a knowledge base is built, AI techniques are used to give the computer an inference capability based on the facts and relationships contained in the knowledge base [Turban, Aronson, 1998]. That is, the knowledge base contains a data structure that can be manipulated by an inference system that uses search and pattern matching techniques on the knowledge base to answer questions, draw conclusions, or otherwise perform an intelligent function. The actual

separation of declarative knowledge from the reasoning/inference capability is an important aspect. With a knowledge base and the ability to draw inferences from it, the computer can be put to practical use as a problem solver and decision maker. By searching the knowledge base for relevant facts and relationships, the computer can reach one or more alternative solutions to the given problem.

A knowledge base can be organized in several different configurations to facilitate fast inference (or reasoning) from the knowledge [Turban, Aronson, 1998]. Furthermore, the domain knowledge in the knowledge base may be organized differently from the reasoning process knowledge in the inference engine.

2.1.1 Knowledge Domain

Most knowledge bases are limited in that they typically focus on some specific, usually narrow subject area or domain [Turban, Aronson, 1998]. In fact, the narrow domain of knowledge and the fact that an AI system must involve some qualitative aspects of decision making are viewed as critical for AI application success. The knowledge base is updated periodically to reflect changes or extensions to the domain knowledge [Stefik, 1995].

2.1.2 Axioms, Definitions and Theorems

Each logic system relies on formal definitions of its axioms or postulates, which are the fundamental definitions of the system [Giarratano, Riley, 1998]. An axiom is simply a fact or assertion that cannot be proved from within the system.

Axioms are what one writes when one "axiomatises" a domain [Stefik, 1995]. From these axioms, people (and sometimes computer programs) try to determine what can be proved [Giarratano, Riley, 1998]. Mathematicians write axioms to capture the basic facts about a domain, define other concepts in terms of those basic facts, and then use the axioms and definitions to prove theorems [Russell, Norvig, 1995]. In AI, one rarely uses the term *theorem* but the sentences that are in the knowledge base initially are sometimes called axioms, and it is also common to call them definitions. An axiom of the form $\forall x, y\ P(x, y) \equiv \ldots$ is often called a *definition* of P, because it serves to define exactly for what objects P does or does not hold [Russell, Norvig, 1995]. It is possible to have several definitions for the same predicate (a property or a relation). Many predicates will have no complete definition of this kind, because one does not know enough to fully characterize them.

Sometimes we accept certain axioms because they make "sense" by appealing to common sense or observation. Other axioms do not make intuitive sense.

All of this brings up an important question: how does one know when he/she has written down enough axioms to fully specify a domain? One way to approach this is to decide on a set of basic predicates in terms of which all the other predicates can be defined [Russell, Norvig, 1995]. The converse problem also arises: does one have too many sentences? In mathematics, an independent axiom is one that cannot be derived from all other axioms. Mathematicians strive to produce a minimal set that cannot be derived from all the other axioms. In AI, it is common to include

redundant axioms, not because they can have any effect on what can be proved, but because they can make the process of finding a proof more efficient.

2.1.3 Properties of Good and Bad Knowledge Bases

A good knowledge representation language should be expressive, concise, unambiguous, context-insensitive, and effective [Russell, Norvig, 1995]. A knowledge base should, in addition, be clear and correct. The relations that matter should be defined, and the irrelevant details should be suppressed. Of course, there will be trade-offs between properties: one can make simplifications that sacrifice some correctness to gain clarity and brevity.

Every knowledge base has two potential consumers: human readers and inference procedures [Russell, Norvig, 1995]. A common mistake is to choose predicate names that are meaningful to the human reader, and then be lulled into assuming that the names are somehow meaningful to the inference procedure as well. The hard part is for the human reader to resist the temptation to make the inferences that seem to be implied by long predicate names. A knowledge engineer will often notice this kind of mistake when the inference procedure fails to conclude. Although very long names can be made to work for simple examples covering a small, sparse portion of a larger domain, they do not scale up well. Adding another very long name takes just as much work as adding the first one.

In a properly designed knowledge base, facts that were entered for one situation should end up being used in new situations as well [Russell, Norvig, 1995]. As one goes along, one should need fewer new facts, and fewer new predicates. This will only happen if one writes rules at the most general level at which the knowledge is applicable.

Every time one writes down a sentence, one should ask oneself the following [Russell, Norvig, 1995]:

- Why is it true? Could I write down the facts that make it true instead?
- How generally is it applicable? Can I state it for a more general class of objects?
- Do I need a new predicate to denote this class of objects? How does the class relate to other classes? Is it part of a larger class? Does that class have other subclass? What are other properties of the objects in this class?

2.1.3.1 Efficiency

The question of efficiency is a little more difficult to deal with [Russell, Norvig, 1995]. Ideally, the separation between the knowledge base and the inference procedure should be maintained. This allows the creator of the knowledge base to worry only about the content of the knowledge base, and not about how it will be used by the inference procedure.

The same answer should be obtainable by the inference procedure, no matter how the knowledge is encoded [Russell, Norvig, 1995]. As far as possible, ensuring efficient inference is the task of the designer of the inference procedure and should not distort the representation. In practice, some considerations of efficiency are unavoidable. Automatic methods exist that can eliminate the most obvious sources of inefficiency in a given encoding, in much the same way that

optimizing compilers can speed up the execution of a program, but at present these methods are too weak to overcome the determined efforts of a profligate knowledge engineer who has no concern for efficiency.

Even in the best case, then, the knowledge engineer should have some understanding of how inference is done, so that the representation can be designed for maximum efficiency [Russell, Norvig, 1995]. In the worst case, the representation language is used primarily as a way of "programming" the inference procedure.

2.1.4 Knowledge Bases Versus Databases

An often heard question is about whether the information stored in knowledge bases could be stored in databases [Stefik, 1995]. The answer is "Yes, of course.". A knowledge base can be managed like a database [Turban, Aronson, 1998].

This question confounds knowledge with the symbol structures that represent knowledge for an interpreter [Stefik, 1995]. Symbol structures can be stored in databases. General facilities for storing and retrieving symbol structures are useful for knowledge engineering. Databases provide storage and retrieval facilities that can be used for information of all kinds of things, whether the information is tables of facts or list structures or programs.

Database research is concerned with such matters as relational models, transaction processing, robust distributed storage, recovery from failures, and efficient access to mass storage [Stefik, 1995]. These are all important concerns for computer technology, but they are distinct from such theoretical issues as knowledge formulation, problem solving, and use. Thus, database research does not encompass knowledge engineering.

2.1.5 Shared, Community Knowledge Bases

A group of people may agree to share the work of developing a common knowledge base [Stefik, 1995]. Sharing in the building of a knowledge base is motivated by practical concerns about the ability to share knowledge, use common knowledge in closely related applications, and reduce the time or labour of encoding a body of knowledge. The primary motivation for this approach is that the participants believe there is a substantial corpus of knowledge that they want to use but could not develop for themselves. It may be that the common knowledge base requires that they each bring unique knowledge to the process. Alternatively, it may simply be that the scale of a project requires a joint effort. When a knowledge base is shared, changes to it must be coordinated; this requires appropriate mechanisms/policies for collaborative work.

2.2 Knowledge Engineers

A knowledge engineer is someone who investigates a particular domain, determines what concepts are important in that domain, and creates a formal representation of the objects and relations in the domain [Russell, Norvig, 1995]. The knowledge engineer must understand enough about the domain in question to represent the important objects and relationships.

Often, however, the knowledge engineer has strong expertise in AI, in particular regarding knowledge engineering methods (including knowledge representation languages), but he/she is not an expert in the domain at hand [Russell, Norvig, 1995]. The basic model of knowledge engineering thus portrays teamwork in which a knowledge engineer interacts with one or more human experts in building the knowledge base, i.e., he/she mediates between the expert(s) and the knowledge base [Turban, Aronson, 1998]. Knowledge engineering involves the cooperation of human experts in the domain who work with the knowledge engineer to make explicit and codify the knowledge (rules or other procedures) that a human expert uses to solve real problems. It is the expert's job to provide knowledge about how he/she performs the task that the knowledge-based system will perform. The knowledge engineer will usually interview the real experts to become educated about the domain and to elicit the required knowledge in a process called knowledge acquisition [Russell, Norvig, 1995]. Typically the knowledge engineer helps the expert structure the problem area by interpreting and integrating human answers to questions, drawing analogies, posing counterexamples, and bringing to light conceptual difficulties. This occurs prior to, or interleaved with, the process of creating formal representations.

The knowledge engineer first establishes a dialog with the human expert [Giarratano, Riley, 1998]. This stage is analogous in conventional programming to a system designer discussing the system requirements with a client for whom the program will be constructed. The knowledge engineer then elicits knowledge from the expert, refines it with the expert, and represents it (i.e., codes the knowledge explicitly) in the knowledge base [Turban, Aronson, 1998]. Note that when more than one expert is involved, the situation can become difficult if the experts disagree; a consensus must eventually be reached.

The knowledge engineer must understand enough about the representation language to correctly encode the facts, rules, etc. [Russell, Norvig, 1995]. He/she must also understand enough about the implementation of the inference procedure to assure that queries can be answered in a reasonable amount of time. He/she is often also the system builder [Turban, Aronson, 1998].

The use of computers and special methods to overcome the difficulties in knowledge acquisition requires qualified knowledge engineers [Turban, Aronson, 1998]. Unfortunately, one does not become a proficient knowledge engineer just by studying the syntax and semantics of a representation language [Russell, Norvig, 1995]. One cannot do, or understand, knowledge engineering by just talking about it. It takes practice and exposure to lots of examples before one can develop a good style in any language, be it a language for programming, reasoning, or communicating.

The ability and personality of the knowledge engineer directly influence the expert [Turban, Aronson, 1998]. Part of successful knowledge acquisition involves developing a positive relationship with the expert. The knowledge engineer is responsible for creating the right impression, positively communicating information about the project, understanding the expert's style, preparing the sessions, and so on. Some of the skills and characteristics that are desirable in knowledge engineers are:

- computer skills (hardware, programming, software),
- tolerance and ambivalence,
- effective communication abilities (sensitivity, tact, and diplomacy),

- broad educational background,
- advanced, socially sophisticated verbal skills,
- fast-learning capabilities (of different domains),
- understanding of organizations and individuals,
- wide experience in knowledge engineering,
- intelligence,
- empathy and patience,
- persistence,
- logical thinking,
- versatility and inventiveness,
- self-confidence.

These requirements make knowledge engineers in short supply (and costly) [Turban, Aronson, 1998]. The shortage of experienced knowledge engineers is a major bottleneck in expert system construction. Research is being conducted on building machine learning systems that will reduce the need for knowledge engineers.

2.3 Knowledge Formulation

To emphasize the constructive and comparative aspects of building knowledge bases, the term "knowledge formulation" is sometime preferred [Stefik, 1995]. The formulation of knowledge includes identifying it, describing it, and explaining what it means. To emphasize aspects of writing and the choice of representation, one says that knowledge is codified. This suggests that besides elicitation and transfer, the process of building a knowledge base also involves articulating elements of experience and preserving them in writing.

2.4 Knowledge Formalization

Knowledge often goes through stages of refinement in which it becomes increasingly formal and precise [Stefik, 1995]. Part of this process involves identifying the conditions under which the knowledge is applicable, and any exceptional conditions. It also involves organizing the representations so they can be used by a problem solver or other interpreter. To convey these meanings, the term knowledge formalization is used. This term helps one to characterize what happens as one progresses from informal notes to formal computational models.

2.5 Knowledge Engineering Versus Programming

A useful analogy can be made between knowledge engineering (or AI problem solving) and conventional computer programming [Russell, Norvig, 1995] [Ginsberg, 1993]. Both activities can be seen as consisting of five steps.

Table 1: Knowledge engineering (or AI problem solving) and conventional computer programming

Knowledge Engineering (or AI Problem Solving)	**Conventional Computer Programming**
(1) Identify the knowledge needed to solve the problem	(1) Devise an algorithm to solve the problem
(2) Select a language in which that knowledge can be represented	(2) Select a programming language in which that algorithm can be encoded
(3) Write down the knowledge in that language (building a knowledge base)	(3) Capture the algorithm in a program (write a program)
(4) Implement the proof theory	(4) Choose or write a compiler
(5) Use the consequences of the knowledge to solve the problem (infer new facts)	(5) Run the program

In both activities, one writes down a description of a problem or state of affairs, and then uses the definition of the language to derive new consequences [Russell, Norvig, 1995]. In the case of a program, the output is derived from the input and the program; in the case of a knowledge base, answers are derived (making use of the inference engine) from descriptions of problems and the knowledge base.

It is the final step, i.e., determining the consequences of the knowledge, which typically involves the well known search techniques of AI [Ginsberg, 1993]. The first and third steps are harder than the other three. The existing languages used for knowledge representation all have a well-defined syntax, and sentences in these languages have agreed-upon meanings. But there is no general understanding of exactly what knowledge one brings to bear to solve specific problems, and there is little known about how any particular bit of knowledge is to be encoded in any particular language. A logic by itself consists of only the syntax, semantics, and proof theory [Russell, Norvig, 1995]. A logic does not offer any guidance as to what facts should be expressed, nor what vocabulary should be used to express them.

Except maybe for human readability (e.g., for the knowledge engineer and the other system developers), the names of symbols appearing in sentences in a knowledge representation language are irrelevant [Ginsberg, 1993]. What makes a variable in one program different from a variable in another is the collection of all references to the two variables in the two programs; knowledge representation is similar. When one writes a computer program, the variables appearing in the code typically refer to specific objects within the algorithm. The objects appearing in the sentences should have identifiable referents in the real world. This is the declarativist assumption.

Given the similarities with conventional computer programming, what is the point of doing knowledge engineering at all? The main advantage of knowledge engineering is that it requires less commitment, and thus less work [Russell, Norvig, 1995]. A knowledge engineer only has to decide what objects and relations are worth representing, and which relations hold among which objects. A programmer has to do all that, and in addition must decide how to compute the relations between objects, given some initial input. The knowledge engineer specifies what is

true, and the inference procedure figures out how to turn facts into a solution to the problem. Furthermore, because a fact is true regardless of what task one is trying to solve, knowledge bases can, in principle, be reused for a variety of different tasks without modification. Finally, debugging a knowledge base is made easier by the fact that any given sentence is true or false by itself, whereas the correctness of a program statement depends very strongly on its context.

This being said, the challenge in knowledge engineering comes from the difficulty in adequately extracting, making explicit, and representing expert knowledge.

2.6 Synergistic Effect of Knowledge Engineering: Knowledge Creation and Discovery

The customary way of building a knowledge-based system, by having the knowledge engineer repeat the cycle of interviewing the expert, constructing a prototype, testing, interviewing, and so on, is a time-consuming and labour-intensive task [Giarratano, Riley, 1998]. However, knowledge engineering usually has a synergistic effect [Turban, Aronson, 1998]. As knowledge systems are built, it is often the case that participants achieve greater insights about how to carry out the task [Stefik, 1995]. These insights come from the careful examination of particular cases and the search for better generalizations.

This suggests that knowledge that goes into knowledge systems does not simply originate from finished mental representations that already express all the information [Stefik, 1995]. The knowledge possessed by human experts is often unstructured and not explicitly expressed [Turban, Aronson, 1998]. Experts sometimes can give only vague accounts of their thinking [Stefik, 1995]. But they do know how to make their thinking more precise through experiments. In this experimentation process, knowledge is often created or discovered.

The construction of a knowledge base helps the expert to articulate what he or she knows [Turban, Aronson, 1998]. It can also pinpoint variances from one expert to another (if several experts are involved).

3. Knowledge Acquisition

Every knowledge-based system project must determine how and where most of the knowledge will be obtained [Stefik, 1995]. Sometimes the answer is to start from scratch. Sometimes it is to interview people who already do the task of interest, and sometimes the answer is to conduct experiments to gather the right knowledge.

Within AI and knowledge engineering, the term knowledge acquisition refers to any technique by which computer systems can gain the knowledge they need to perform their tasks [Stefik, 1995]. It is the process of collecting, extracting, transferring, accumulating, structuring, transforming and organizing knowledge (e.g., problem-solving expertise) from one or more knowledge sources (human experts, books, documents, sensors, or computer files) for constructing or expanding a knowledge base [Turban, Aronson, 1998]. Knowledge acquisition is actually done throughout the entire development process.

3.1 Difficulties in Knowledge Acquisition

Acquiring knowledge from experts is a complex task [Turban, Aronson, 1998]. This process has been identified by many researchers and practitioners as a (or even as *the*) bottleneck that currently constrains the development and construction of expert systems and other AI systems. The multiplicity of sources and types of knowledge contribute to the complexity of knowledge acquisition.

3.1.1 Transferring Expertise: Knowledge as a "Transportable Substance"

For newcomers and also those who have thought deeply about knowledge, the term knowledge acquisition is problematic [Stefik, 1995]. The term acquisition is odd because it suggests that knowledge is like a substance. In this view, knowledge is delivered using a "transportation" process from someone or somewhere and the crucial step in building a knowledge system is to "get" the knowledge.

The metaphor of knowledge as a "transportable substance" has long been used in informal English conversation [Stefik, 1995]. For example:

- books and teaching are said to "convey knowledge";
- elders "pass along" knowledge to the next generation;
- knowledge and secret information sometimes "leak out";

In general, however, transferring information from one person to another is difficult [Turban, Aronson, 1998]. Several mechanisms can be used to conduct such a transfer, e.g., written words, voice, pictures, music, and not one of them is perfect. There are also problems in transferring any knowledge, even simple messages.

Transferring knowledge in expert systems is even more difficult [Turban, Aronson, 1998]. Knowledge is transferred to a machine, where it must be organized in a particular manner. The machine requires the knowledge to be expressed explicitly, at a lower, more detailed level than human uses. Human knowledge exists in a compiled format. A human being simply does not remember all the intermediate steps used by his/her brain in transferring or processing knowledge. Thus, there is a mismatch between computers and experts.

This problem of transferring human knowledge into a knowledge-based system is so major that it is called the knowledge acquisition bottleneck [Giarratano, Riley, 1998]. Actually, some argue that the transfer-of-expertise model is wrong because the knowledge is not accessible by asking questions to experts [Stefik, 1995]. This is an important line of argument because it says there are fundamental restrictions on the kind of tasks that can be automated by the interviewing of experts. Table 1 lists some issues regarding the transfer-of-expertise model.

Table 2: Some issues regarding the transfer-of-expertise model [Stefik, 1995]

Communication	Some kinds of knowledge cannot be communicated in words or writing. Examples of knowledge that cannot be communicated are easy to find. It is difficult to teach physical tasks, such as swimming, golf, or tennis, by just verbal instructions. The learning experience involves too much in the way of senses and motor skills. Of course, people do learn to swim, play tennis, and play golf. Instruction involves a richer modality of communication. These sports examples are real, but they lie outside the boundaries of information-processing models used in knowledge systems.
Introspection	Some kinds of knowledge are not accessible by introspection. That some elements of competence are not accessible by introspection also relates to phenomena outside knowledge engineering. For example, we are all “experts” at listening to spoken languages. But few of us can be articulate about how we discriminate among different sounds. Some symbol recognition processes take place at subconscious levels. Some activities are not “cognitively penetrable”.
What you say is not what you do!	What experts do is not the same as what they say they do. That domain experts sometimes “rationalize” their methods, saying they solve problems differently from the way they actually do, is a widely recognized methodological issue.
Multiple experts	Some tasks are not carried out by any single expert. In some cases, knowledge about needs and knowledge about the existing system is distributed in the heads of many people. The main challenge would be integrating this knowledge.
Imperfect and unprincipled expert practice	Sometimes, experts solve problems incorrectly and in unprincipled ways. When expert practice is imperfect or imperfect or unprincipled, this can become apparent both to domain experts and to knowledge engineers part way through a project.

Note that [Stefik, 1995] states that knowledge-based systems can be built without using the transfer-of-expertise model.

The metaphor of knowledge as a “transportable substance” reflects a truism [Stefik, 1995]. During education, knowledge becomes known by additional people. Knowledge systems themselves are sometimes thought of as knowledge distribution systems. However, the process of spreading and knowing involves more than transportation. The substance and transportation metaphor implies that knowledge starts out in a domain expert's head and that the object is to get the knowledge into a computer. This sounds like a simple matter that involves asking a few questions of an expert followed by some programming. Unfortunately, on this simplistic line of thought, many knowledge systems projects have floundered in months of misguided attempts at knowledge acquisition that never converge. Some projects fail utterly.

The transportation metaphor ignores the processes of knowledge creation and use [Stefik, 1995]. It does not consider that:

- knowledge must somehow originate;
- knowledge can be implicit in the interactions of an agent with an environment; and
- communication is at least two-way, involving the cooperative activity of agents that need to agree on spoken or written symbols and behaviours.

The process of developing a knowledge base involves more than transportation of facts [Stefik, 1995]. Recognizing other dimensions of what is needed help one to make sense of the process and the methods by which people approach it.

3.1.2 Expressing the Knowledge

To solve a problem, a human expert performs a two-step process [Turban, Aronson, 1998]. First, the expert inputs information about the external world into the brain. This information is transmitted by people, computers, or other media. It may also be collected via sensors or retrieved from memory. Second, the expert uses an inductive, deductive, or other problem-solving approach on the collected information.

This process is internal [Turban, Aronson, 1998]. The knowledge engineer, when collecting knowledge from an expert, must ask the expert to be introspective about his/her (the expert's) decision-making process and about the inner experiences that are involved in it. An expert may find it very difficult to express his/her experiences about this process, especially when the expert's knowledge has never been systematically explored [Giarratano, Riley, 1998], and when the experiences are made up of sensations, thoughts, sense memories, and feelings [Turban, Aronson, 1998]. The expert is often unaware of the detailed process he/she uses to arrive at a conclusion. Therefore, the expert may actually use different rules to solve real-life problems than those stated in a knowledge acquisition interview.

3.1.3 Structuring the Knowledge

In expert systems, it is necessary to elicit not only the knowledge, but also its structure [Turban, Aronson, 1998]. One must represent the knowledge in a structured way (for example, as rules).

3.1.4 Number of Participants

In a simple process of knowledge transfer, there are two participants (a sender and a receiver) [Turban, Aronson, 1998]. For the development and use of an expert system, there could be as many as four participants (plus a computer): the expert, the knowledge engineer, the system designer (builder), and the user. Sometimes, there are even more participants, e.g., multiple experts, programmers, etc. These participants have different backgrounds, use different terminology, and possess different skills and knowledge. The experts, for example, may know very little about computers, and the knowledge engineer may know very little about the problem area.

3.1.5 Other Reasons

Several other reasons add to the complexity of transferring the knowledge [Turban, Aronson, 1998]:

- Experts may lack time or may be unwilling to cooperate.
- Testing and refining knowledge is complicated.
- Methods for knowledge elicitation may be poorly defined.
- System builders tend to collect knowledge from one source, but the relevant knowledge may be scattered across several sources.
- Builders may attempt to collect documented knowledge rather than use experts. The knowledge collected may be incomplete.
- It is difficult to recognize specific knowledge when it is mixed up with irrelevant data.
- Experts may change their behaviour when they are being observed or interviewed.
- There may be problematic interpersonal communication factors between the knowledge engineer and the expert.

3.2 Overcoming the Difficulties in Knowledge Acquisition

Many efforts have been made to overcome some of the problems mentioned in the previous sub-section [Turban, Aronson, 1998]. For example, research on knowledge acquisition tools has begun to focus on ways to decrease the representation mismatch between the human expert and the program under development. One form of this research might be characterized as research on “learning by being told”. The attempt here is to develop programs capable of accepting advice as it would often be given to a human novice.

Several expert system development software packages greatly simplify the syntax of the rules (in a rule-based system) to make them easier for an expert system builder to create and understand without special training [Turban, Aronson, 1998]. Also, a natural language processor can be used to translate knowledge to a specific knowledge representation structure.

The process of knowledge acquisition can be greatly influenced by the role of the three major participants: the knowledge engineer, the expert, and the end user [Turban, Aronson, 1998].

Experts should take a very active role in the creation of the knowledge base. The knowledge engineer should act more like a teacher of knowledge structuring, a tool designer, and a catalyst in the interface between the expert and the end-users. This approach could minimize problems such as inter-human conflicts, knowledge engineering filtering, and end-user acceptance of the system. Also, knowledge maintenance problems can be reduced. It is more appropriate to think of the participants as playing one or more roles, including knowledge sources, agents, and targets for knowledge acquisition processes. Finally, some of the difficulties may be lessened or eliminated with computer aided knowledge acquisition tools and with extensive integration of the acquisition efforts.

3.3 Methods of Knowledge Acquisition

Knowledge elicitation methods are classified in different ways and appear under different names [Turban, Aronson, 1998].

3.3.1 Level of Automation

The elicitation of knowledge from the expert can be done manually or with the aid of computers with some degree of automation [Turban, Aronson, 1998].

3.3.1.1 Manual Methods

Most of the manual elicitation techniques have been borrowed (but often modified) from psychology or from system analysis [Turban, Aronson, 1998]. Manual methods are basically structured around some kind of interview. The knowledge engineer elicits knowledge from the expert or other sources and then codes it in the knowledge base. The process is shown in Fig. 2.

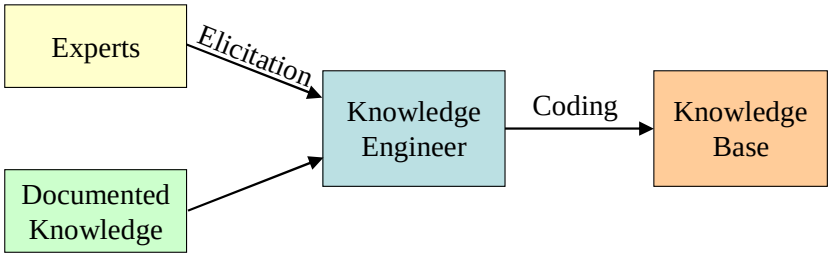


Figure 2: Manual methods of knowledge acquisition [Turban, Aronson, 1998]

The three major manual methods are [Turban, Aronson, 1998]:

- Interviewing (structured, semi-structured, unstructured).
- Tracking the reasoning process.
- Observing.

Manual methods are slow, expensive, and sometimes inaccurate. Therefore, there is a trend to automate the process as much as possible.

3.3.1.2 Semi-Automatic Methods

Semi-automatic methods are divided into two categories [Turban, Aronson, 1998]:

- Those that are intended to support the experts by allowing them to build knowledge bases with little or no help from knowledge engineers. The process is shown in Fig. 3.
- Those that are intended to help knowledge engineers by allowing them to execute the necessary tasks in a more efficient or effective manner (sometimes with only minimal participation by an expert).

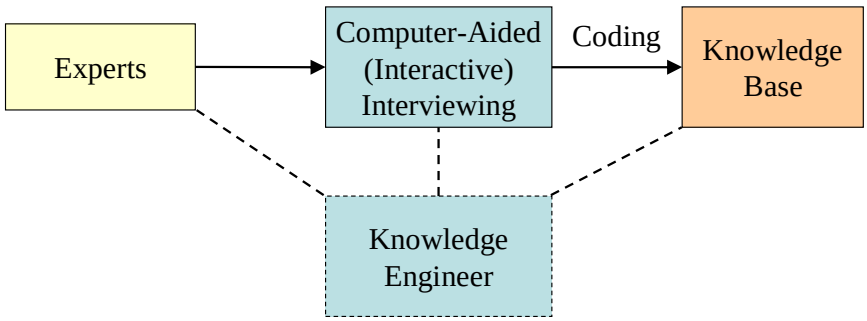


Figure 3: Expert-driven knowledge acquisition [Turban, Aronson, 1998]

3.3.1.3 Automatic Methods

In automatic methods, the role of both the expert and the knowledge engineer is minimized or even eliminated [Turban, Aronson, 1998]. For example, the induction method shown in Fig. 4 can be administered by any builder (such as a system analyst).

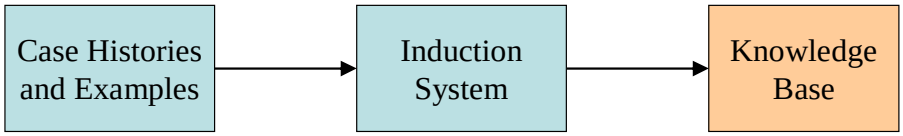


Figure 4: Induction-driven knowledge acquisition [Turban, Aronson, 1998]

The role of the expert is minimal, and there is no need for a knowledge engineer. The term automatic may be misleading. There will always be a human builder, but there may be little or no need for a knowledge engineer and an expert.

3.3.2 Interviews

The most commonly used form of knowledge acquisition is face-to-face interview analysis [Turban, Aronson, 1998]. It is an explicit technique and it appears in several variations. It involves a direct dialog between the expert and the knowledge engineer. Therefore, interpersonal communication and analytical skills are important.

Information is collected with the aid of conventional instruments (such as tape recorders or questionnaires), and it is subsequently transcribed, analyzed, and coded [Turban, Aronson, 1998].

In the interview, the expert is presented with a simulated case or, if possible, with an actual problem of the sort that the expert system will be expected to solve. The expert is asked to “talk” the knowledge engineer through the solution. Sometimes, this method is called the walkthrough method. One variant of the interview approach begins with no information at all being given to the expert. Any facts the expert requires must be asked for explicitly. By doing this, the expert's path through the domain may be made more evident, especially in terms of defining the input an expert system would require.

Because of the specific nature of each interview, it is difficult to provide good guidelines for the entire interview process. However, there are several guidelines, checklists, and instruments that are fairly generic in nature. A recommendation is that a knowledge engineer interviews less knowledgeable or minor experts before interviewing the main experts [Turban, Aronson, 1998]. This helps the knowledge engineer:

- learn about the problem,
- learn its significance,
- learn about the experts,
- learn who the users will be,
- understand the basic terminology, and
- identify archived sources about the problem (for a novice in the area)

The knowledge engineer should next read about the problem domain. Then, the main experts can be interviewed much more effectively.

On one hand, interviewing techniques, though very popular, have many disadvantages [Turban, Aronson, 1998]. They range from inaccuracies in collecting information to biases introduced by the interviewers. The interview process can also be tedious. It places great demands on the domain expert, who must be able not only to demonstrate expertise, but also to express it. On the other hand, it requires little equipment, it is highly flexible and portable, and it can yield a considerable amount of information.

Interviews are important techniques, but they must be planned carefully and the interview results must be subjected to thorough verification and validation methodologies. Interviews are sometimes replaced by tracking methods (cf. subsection 3.3.3). Alternatively, they can be used to supplement tracking or other knowledge acquisition methods.

There are two basic types of interviews: unstructured (informal) interviews, and structured interviews [Turban, Aronson, 1998]; these are discussed next.

3.3.2.1 Unstructured Interviews

Many knowledge acquisition interview sessions are conducted informally, usually as a starting point [Turban, Aronson, 1998]. Starting informally saves time; it helps to get quickly to the basic structure of the domain. Usually it is followed by a formal technique.

Contrary to what many people believe, unstructured interviews are not simple. In fact, they may present the knowledge engineer with some very problematic after-effects. Unstructured interviewing seldom provides complete well-organized descriptions of cognitive processes, for the following reasons [Turban, Aronson, 1998]:

- The domains are generally complex.
- The experts usually find it very difficult to express some of the more important elements of their knowledge.
- Domain experts may interpret the lack of structure as requiring little preparation on their part.
- Data acquired from an unstructured interview are often unrelated, exist at varying levels of complexity, and are difficult for the knowledge engineer to review, interpret, and integrate.
- Because of a lack of training and experience, few knowledge engineers can conduct an efficient unstructured interview.

By interviewing the expert, the knowledge engineer slowly learns what is going on [Turban, Aronson, 1998]. Then the knowledge engineer builds a representation of the knowledge in the expert's terms. The process of knowledge acquisition involves uncovering the attributes of the problem and making explicit the thought process (usually expressed as rules) that the expert uses to interpret these attributes.

The unstructured interview is most common. It appears in several variations. In addition to the talk-through, one can ask the expert to teach-through or read-through. In a teach-through the expert acts as an instructor and the knowledge engineer as a student. The expert not only tells what he/she does, but also explains why and instructs the knowledge engineer in the skills and strategies needed to perform the task. In a read-through approach, the expert is asked to instruct the knowledge engineer how to read and interpret the documents that are used for the task.

3.3.2.2 Structured Interviews

The structured interview is a systematic goal-oriented process [Turban, Aronson, 1998]. It forces an organized communication between the knowledge engineer and the expert. The structure reduces the interpretation problems inherent in unstructured interviews, and it allows the knowledge engineer to prevent the distortion caused by the subjectivity of the domain expert.

Structuring an interview requires attention to a number of procedural issues; the knowledge engineer [Turban, Aronson, 1998]:

- studies available material on the domain to identify major demarcations of the relevant knowledge,
- reviews the planned expert system capabilities and identifies targets for the questions to be asked during the knowledge acquisition session,
- formally schedules and plans (using a form) the structured interviews (planning includes attending to physical arrangements, defining knowledge acquisition session goals and agendas, and identifying or refining major areas of questioning),

- may write sample questions, focussing on question type, level, and questioning techniques,
- ensures that the domain expert understands the purpose and goals of the session and encourages the expert to prepare before the interview,
- follows, during the interview, guidelines for conducting interviews,
- uses directional control, during the interview, to retain the interview's structure.

There are many structured interview methods.

3.3.3 Process Tracking Methods

Process tracking is a set of techniques that attempt to track the reasoning process of an expert [Turban, Aronson, 1998]. It is a popular approach among cognitive psychologists who are interested in discovering the expert's train of thought while he/she reaches a conclusion. The knowledge engineer can use the tracking process to find what information is being used and how it is being used. Tracking methods can be informal or formal. The most common formal method is protocol analysis.

3.3.4 Machine Learning

As previously mentioned, experts often can't explain how they solve problems; the solutions just come to them [Giarratano, Riley, 1998]. If the expert can't explain how a problem is solved, it's not possible to encode the knowledge in an expert system based on explicit knowledge. In this case, the only possibility is programs that learn by themselves to emulate the expert. These are programs based on induction and artificial neural systems (ANS). Some rule-based expert systems allow the system to learn the rules by examples (provided by human experts), through rule induction, in which the system creates rules from tables of data.

However, although rule induction helps, only limited types of knowledge can be put into an expert system this way [Giarratano, Riley, 1998]. Actually, although machine learning is ideal because many cases can be examined quickly, most knowledge-based systems are based on knowledge acquired from human experts [Turban, Aronson, 1998].

3.3.5 Knowledge Discovery

Among the many advancements in knowledge management, knowledge discovery is one aspect that has a significant impact on knowledge acquisition.

3.3.5.1 Databases

Many expert systems have been constructed from knowledge that is extracted in full or in part from databases [Turban, Aronson, 1998]. However, with the increased amount of knowledge stored in databases, the acquisition of such knowledge becomes more difficult.

3.3.5.2 Internet

With the increased use of the Internet, it is possible to reach vast amounts of knowledge [Turban, Aronson, 1998]. The acquisition, accessibility, and management of knowledge via the internet are becoming critical issues for the construction and maintenance of knowledge-based systems, particularly because they allow for the acquisition and dissemination of large quantities of knowledge in a short time across organizational and physical boundaries.

3.4 Incremental Knowledge Acquisition

Knowledge can easily grow incrementally in a rule-based expert system [Giarratano, Riley, 1998]. That is, the knowledge base can grow little by little as rules are added so that the performance and correctness of the system can be continually checked. If the rules are properly designed, the interactions between rules will be minimized or eliminated to protect against unforeseen effects. The incremental growth of knowledge facilitates rapid prototyping so that the knowledge engineer can quickly show the expert a working prototype of the expert system. This is an important feature because it maintains both the expert's and management's interest in the project. Rapid prototyping also quickly exposes any gaps, inconsistencies, or errors in the expert's knowledge or the system so that corrections can be made immediately.

4. Constructing a Knowledge Base

The process of representing knowledge of a domain goes through several stages. To help focus the development of a knowledge base and to integrate the knowledge engineer's thinking at the three levels (i.e., domain, representation language, and inference procedure), the following eight-step methodology can be used [Russell, Norvig, 1995] [Brachman, Levesque, 2004]:

1. understand the domain and decide on what to talk about,
2. decide on a basic vocabulary of predicates, functions, and constants,
3. encode a description of the basic facts,
4. encode a description of the complex facts,
5. capture the terminological facts,
6. define and introduce new abstract individuals,
7. complete the KB with the capture of other sorts of facts,
8. pose queries to the inference procedure and get answers.

These are discussed next.

4.1 Understand the Domain and Decide on What to Talk About

The knowledge engineer has to understand the domain well enough to know which objects and facts need to be talked about, and which can be ignored [Russell, Norvig, 1995]. This is about establishing the scope of the domain that will be represented in the knowledge base. In some cases, this can be the hardest step. Many knowledge engineering projects have failed because the knowledge engineers started to formalize the domain before understanding it.

4.2 Decide on a Basic Vocabulary of Predicates, Functions, and Constants

At one point, one has an informal list of the concepts in the domain, i.e., a conceptualization. The knowledge engineer eventually has to translate the important domain-level concepts into logic-level names [Russell, Norvig, 1995]. In creating a KB, it is a good idea to start with the set of domain-dependent predicates and functions that provides the basis for the statement of facts about the KB's domain [Brachman, Levesque, 2004]. This involves many choices, some arbitrary and some important. Once the choices have been made, the result is a vocabulary that can be related to an ontology of the domain (cf. section 5).

What sort of objects will there be in the world of interest? The most obvious place to start is with the named individuals; in first-order logic (FOL), these would be represented by constant symbols [Brachman, Levesque, 2004]. After capturing the set of individuals that will be central to the "world" to be considered by the system, it is next essential to circumscribe the basic types of objects that those individuals are. This is usually done with one-place predicates in FOL.

Another set of one-place predicates that is crucial for the domain representation is the set of attributes that the objects can have [Brachman, Levesque, 2004]. So one needs a vocabulary of properties that can hold for individuals. The syntax of FOL is limited in that it does not allow for the distinction between such properties and the object-types suggested above. This usually does not present a problem, although if it were important for the system to distinguish between such types, the language could be extended to do so.

The next key predicates to consider are n-ary predicates that express relationships [Brachman, Levesque, 2004]. Also, one cannot forget relationships of higher arity than 2.

Finally, one needs to capture the important functions of the domain [Brachman, Levesque, 2004]. These can take more than one argument, but are most often unary. One thing to note is that all functions are taken to be total in FOL.

[Roy, Auger, 2008-A] discuss the elements used in FOL (predicates, functions, constants, etc.) in more details.

4.3 Encode a Description of the Basic Facts

Now that one has the basic vocabulary in place, it is appropriate to start representing the simple core facts of the world of interest [Brachman, Levesque, 2004]. Such facts are usually represented by atomic sentences and negations of atomic sentences. If the conceptualization is well thought out, this step will be easy [Russell, Norvig, 1995]. It will mostly involve writing simple atomic sentences about instances of concepts that are already part of the conceptualization.

For example, one can use the type predicates, applied to individuals in the domain, to represent some basic truths [Brachman, Levesque, 2004]. Such type predications would define the basic ontology of this world. Once one has set down the types of each of the objects, one can capture some of the properties of the objects. These properties will be the chief currency in talking about the domain, since one most often wants to see what properties (and relationships) are implied by a set of facts or conjectures.

Basic facts like these yield what amounts to a simple database [Brachman, Levesque, 2004]. These facts could indeed be stored in relational tables. For example, each type predicate could be a table, with the table entries being identifiers for all of the known satisfiers of that predicate. Of course, the details of such a storage strategy would be a symbol-level (cf. [Roy, Auger, 2008-A]), not a knowledge-level issue.

Another set of simple facts that are useful in domain representation are those dealing with equality [Brachman, Levesque, 2004]. One use for equalities would be for naming convenience, as when an individual has more than one name.

4.4 Encode a Description of the Complex Facts

Many of the facts one would like to express about a domain are more complex than can be captured using atomic sentences [Brachman, Levesque, 2004]. Thus, one needs to use more complex formulas, with quantifiers and other connectives, to express various beliefs about the domain. A type of fact that needs a complex formula to express it is one that expresses incomplete knowledge about the world. Another useful type of complex statement about the domain is what one might call a closure sentence, used to limit the domain of discourse. For example, one could carve out the full set of individuals in the domain of discourse; this ensures that a reasoner would not postulate a new, hitherto unknown object in the course of its reasoning.

Finally, it is useful to distinguish formally between all known individuals with a set of sentences; this would prevent the accidental postulation that two individuals were the same [Brachman, Levesque, 2004].

4.5 Capture the Terminological Facts

The kind of facts represented so far are sufficient to capture the basic circumstances in a domain, and give grist for the reasoning mill. However, one would typically also think in terms of relationships among the predicate and function symbols one has exploited. In order to support common and useful inferences, one needs to provide a set of facts about the terminology one is using. Terminological facts come in many varieties [Brachman, Levesque, 2004]:

- *Disjointness*: Often two predicates are disjoint, and the assertion of one implies the negation of the other.
- *Subtypes*: There are many predicates that imply a form of specialization, wherein one type is subsumed by another.
- *Exhaustiveness*: This is the converse of the subtype assertion, where two or more subtypes completely account for a super-type.
- *Symmetry*: Some relationships are symmetric.
- *Inverse*: Some relationships are the opposite of others.
- *Type restrictions*: Part of the meaning of some predicates is the fact that their arguments must be of certain types.
- *Full definitions*: In some cases, one wants to create compound predicates that are completely defined by a logical combination of other predicates. One can use a biconditional to capture such definitions.

Terminological facts are typically captured in a logical language as universally quantified conditionals or biconditionals.

4.6 Abstract Individuals

The FOL language gives the basic elements for representing facts in a domain, but in many cases there is a great deal of flexibility that can be exercised in mapping objects in that domain onto

predicates and functions [Brachman, Levesque, 2004]. There is also considerable flexibility in what one considers to be the individuals in the domain. It is sometimes useful to introduce new abstract individuals that might not have been considered in a first analysis.

This idea of making up new individuals is called reification and is typical of systems like description logics and frame languages [Brachman, Levesque, 2004]. Reification might be useful, for example, when the arity of a predicate depends on how much detail one will want to express, which one may not be able to predict in advance. The approach is to take the "predicate" itself to be an abstract individual. Then, to describe this "predicate" at any level of detail that one finds appropriate, one needs only use 1-place predicates and functions. For less detail, one simply leaves out some of the conjuncts; for more, one includes others. The big advantage is that the arity of the predicate and function symbols can be determined in advance.

In a similar way, one can capture in a reasonable fashion complex relationships [Brachman, Levesque, 2004]. In representing commonsense information, one also finds that he/she needs individuals for numbers, dates, times, addresses, and so on. Basically, any "object" about which one can ask a wh-question (i.e., who, what, where, when, etc.) should have an individual standing for it in the KB so it can be returned as the result of a query.

4.7 Other Sorts of Facts

The apparatus described so far should be sufficient to represent the basic facts and individuals of a commonsense domain. However, it is important to point out that there are a number of other types of facts about domains that one may want to capture [Brachman, Levesque, 2004]. Each of these is problematical for a straightforward application of FOL, but they may be represented with extensions of FOL, or with other knowledge representation languages.

The choice of the language to use in a system will ultimately depend on what types of facts and conclusions are most important for the application. Examples of other sorts of facts are [Brachman, Levesque, 2004]:

- Statistical and probabilistic facts: These include those that involve portions of the sets of individuals satisfying a predicate, in some cases exact subsets and in other cases less exactly quantifiable.
- Default and prototypical facts: These cite characteristics that are usually true, or reasonable to assume true unless told otherwise.
- Intentional facts: These express people's mental attitudes and intentions, i.e., they can reflect the reality of people's beliefs, but not necessarily the "real" world itself.

Of course, this is not the end of what one would like to be able to express in a KB. Ultimately, a knowledge-based system should be able to express and reason with anything that can be expressed by a sentence of English, indeed, anything that one can imagine as being either true or false.

4.8 Pose Queries to the Inference Procedure and Get Answers

By writing logical sentences or axioms about the terms in the conceptualization, one accomplishes two goals [Russell, Norvig, 1995]. First, one makes the terms more precise so that humans will agree on their interpretation. Second, one makes it possible to run inference procedures to automatically derive consequences from the knowledge base. This is where the reward is: one can let the inference procedure operate on the explicit axioms and problem-specific facts to derive the implicit facts one is interested in knowing.

Once the axioms are in place, one can say that a knowledge base has been produced [Russell, Norvig, 1995]. Of course, nobody expects a knowledge base to be correct and complete on the first try. There will be a considerable debugging process. The main difference between debugging a knowledge base and debugging a program is that it is easier to look at a single logic sentence and tell if it is correct. Programming language statements tend to depend on a lot of context, whereas logic sentences tend to be more self-contained. In that respect, a sentence in a knowledge base is more like an entire procedure in a program, not like an individual statement.

4.9 Knowledge Engineering with Frames

The description of the construction of a KB as provided in the subsections above is largely from a FOL perspective. However, developing a frame system (cf. [Roy, Auger, 2008-A] and [Roy, Auger, 2008-B]) is a form of knowledge engineering that is quite different from the logical approach [Brachman, Levesque, 2004]. One starts with a sketchy description of some circumstance and embellish it with defaults and implied values. The IF-ADDED procedures can make update easier and help to maintain consistency; the IF-NEEDED procedures allow values to be computed only when they are needed. There is a trade-off here, of course, and which type of procedure to use in an application will depend on the potential value to the user of seeing implied values computed up front versus the value of waiting to do computation only as required.

4.10 Knowledge Validation and Verification

There may be inconsistencies, ambiguities, duplications, or other problems with the expert's knowledge that are not apparent until attempts are made to formally represent the knowledge in a knowledge-based system [Giarratano, Riley, 1998]. The process of developing such a system thus has an indirect benefit since the knowledge of human experts must be put into an explicit form for entering into the computer. Because the knowledge is then explicitly known instead of being implicit in the expert's mind, it can be examined for correctness, consistency, and completeness.

The knowledge must be validated and verified (for example, by using test cases) until its quality is acceptable [Turban, Aronson, 1998]. Test case results are usually shown to the expert to verify the accuracy of an expert system. The knowledge may then have to be adjusted or re-examined, which improves the quality of the knowledge [Giarratano, Riley, 1998].

5. Ontological Engineering

[Roy, Auger, 2008-A] provides a practical definition of the term *ontology*, and also discusses many aspects related to ontologies, including their utility in situation analysis support systems. Ontological engineering refers to the set of activities that concern [Gómez-Pérez et al, 2004]:

- the ontology development process,
- the ontology life cycle,
- the methods and methodologies for building ontologies, and,
- the tool suites and languages that support them.

Ontologies are usually built cooperatively by different groups of people in different locations [Gómez-Pérez et al, 2004]. The 1990s and the first years of this new century have witnessed the growing interest of many practitioners in approaches for building ontologies from scratch, for reusing other ontologies, and for using semi-automatic methods that reduce the knowledge acquisition bottleneck of the ontology development process. Until the mid-1990s, this process was an art rather than an engineering activity. Each development team usually followed their own set of principles, design criteria and phases for manually building the ontology. The absence of common and structured guidelines slowed the development of ontologies within and between teams, the extension of any ontology, the possibilities of ontologies of being reused in others, and the use of ontologies in final applications.

[Gómez-Pérez et al, 2004] presents the main methodologies and methods used to build ontologies:

- from scratch,
- by reusing and re-engineering other ontologies,
- by a process of merging, or,
- by using an ontology learning approach.

5.1 Ontology Development Process

The ontology development process refers to which activities are performed when building ontologies [Gómez-Pérez et al, 2004]. It is crucial to identify these activities if agreement is to be reached on ontologies built by geographically distant cooperative teams. It is advisable to carry out the three categories of activities presented in Fig. 5 and steer clear of anarchic construction.

Note that the ontology development process identifies which activities are to be performed. However, it does not identify the order in which the activities should be performed. The ontology life cycle identifies when the activities should be carried out, i.e., it identifies the set of stages through which the ontology moves during its life time, describes what activities are to be performed in each stage and how the stages are related (relation of precedence, return, etc.).

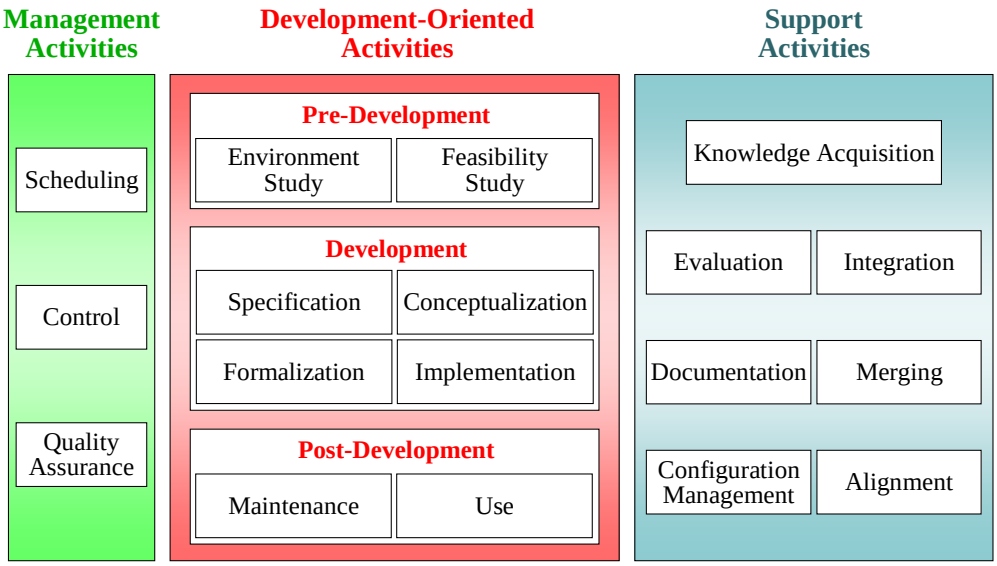


Figure 5: Ontology development process [Gómez-Pérez et al, 2004]

5.1.1 Management Activities

Ontology management activities include scheduling, control, and quality assurance [Gómez-Pérez et al, 2004]. The scheduling activity identifies the tasks to be performed, their arrangement, and the time and resources needed for their completion. This activity is essential for ontologies that use ontologies stored in ontology libraries or for ontologies that require a high level of abstraction and generality. The control activity guarantees that scheduled tasks are completed in the manner intended to be performed. Finally, the quality assurance activity assures that the quality of each and every product output (ontology, software and documentation) is satisfactory.

5.1.2 Development Oriented Activities

Ontology development oriented activities are grouped, as presented in Fig. 5, into pre-development, development and post development activities [Gómez-Pérez et al, 2004]. During the pre-development, an environment study is carried out to know the platforms where the ontology will be used, the applications where the ontology will be integrated, etc. Also during the pre-development, the feasibility study answers questions like: is it possible to build the ontology? Is it suitable to build the ontology?, etc..

Once in the development, the specification activity states why the ontology is being built, what its intended uses are, and who the end-users are (note that some consider specification as a pre-development activity). The result of this activity is an ontology description (usually in natural language) that will be transformed into a conceptual model by the conceptualisation activity. The conceptualization activity structures the domain knowledge as meaningful models at the knowledge level. The formalization activity transforms the conceptual model into a formal or semi-computable model. The implementation activity builds computable models in an ontology language.

During the post-development, the maintenance activity updates and corrects the ontology if needed. Also during the post-development, the ontology is (re)used by other ontologies or applications.

5.1.3 Support Activities

Finally, ontology support activities include a series of activities performed at the same time as the development-oriented activities, without which the ontology could not be built [Gómez-Pérez et al, 2004]. They include knowledge acquisition, evaluation, integration, merging, alignment, documentation and configuration management.

The goal of the knowledge acquisition activity is to acquire knowledge from experts of a given domain, or through some kind of (semi-)automatic process, which is called ontology learning. The evaluation activity makes a technical judgment of the ontologies, of their associated software environments, and of the documentation. This judgment is made with respect to a frame of reference during each stage and between stages of the ontology's life cycle.

The merging and integration activities are required when building a new ontology by reusing other ontologies already available. The objective then consists in obtaining a new ontology starting from several ontologies on the same domain. The resulting ontology is able to unify concepts, terminology, definitions, constraints, etc., from all the source ontologies. The merge of two or more ontologies can be carried out either in run-time or design time.

The alignment activity establishes different kinds of mappings (or links) between the involved ontologies. Hence, this option preserves the original ontologies and does not merge them. The documentation activity details, clearly and exhaustively, each and every one of the completed stages and products generated. The configuration management activity records all the versions of the documentation and of the ontology code to control the changes.

5.2 Ontology Development Methodology Evolution

A series of methods and methodologies for developing ontologies from scratch have been reported [Gómez-Pérez et al, 2004]. But methods and methodologies have not been created only for building ontologies from scratch. When (re)using an existing ontology it might happen that the ontology is implemented in a language with an underlying knowledge representation paradigm different to the knowledge representation conventions used by the ontology that reuses it, or that the ontology to be reused has different ontological commitments, etc. A re-engineering method could be used for solving some of these problems.

Although one of the main purposes of ontologies is to reduce the knowledge acquisition bottleneck, to acquire knowledge for building ontologies still requires a lot of time and resources. As a consequence, ontology learning methods have been thought up to decrease the effort made during the knowledge acquisition process. Such methods are used with several purposes: to create a new ontology from scratch, to enrich an existing one with new terms, and to acquire knowledge for some tasks. [Gómez-Pérez et al, 2004] present ontology learning methods mainly based on natural language analysis from texts.

Ontologies aim to capture consensual knowledge of a given domain in a generic and formal way to be reused and shared across applications and by groups of people. From this definition, one could wrongly infer that there is only one ontology for a domain. In fact, one can find in the literature several ontologies that model, though in different ways, the same kind of knowledge or domain [Gómez-Pérez et al, 2004].

Two approaches can be distinguished for unifying the terminologies of the ontologies [Gómez-Pérez et al, 2004]: ontology alignment and ontology merging. Ontology alignment methods establish different kinds of mappings (or links) between the ontologies, hence this option preserves the original ontologies. However, ontology merging methods propose to generate a unique ontology from the original ontologies. The merging process usually requires to establish mappings between the ontologies to merge. Given the current state of affairs, it is more suitable to establish ontological mappings between existing ontologies on the same topic than to pretend to build "*the*" unified knowledge model for such a topic from scratch.

Some approaches consider collaborative and distributed construction of ontologies. Some methods of such approaches include a protocol for agreeing new pieces of knowledge with the rest of the knowledge architecture, which has been previously agreed upon. Ontologies cannot be reused by other ontologies or used by applications without evaluating first their content from a technical point of view. In this regard, a method has been proposed to analyze and clean the taxonomy of an existing ontology by means of a set of principles based in philosophy [Gómez-Pérez et al, 2004].

6. Conclusion

When adopting a knowledge-centric view of situation analysis and information fusion (SAIF), one eventually has to care about knowledge and ontological engineering techniques. This memorandum presented these important fields, and discussed their main aspects.

The memorandum didn't present “solutions”, or findings and results of completed activities. The intent was more to provide a fair description of the techniques, in order to contribute to setting up a foundational R&D framework for two projects that are just starting under the Applied Research Program (ARP) at Defence R&D Canada: SATAC and CKE-4-MDA. Among other things, the basic elements usually incorporated in a knowledge base, the notion of a knowledge domain, and also axioms, definitions and theorems were discussed. The properties of good and bad knowledge bases were described, and the distinction between knowledge bases and databases was made. Some of the skills and characteristics that are desirable in knowledge engineers were listed. Discussions on knowledge engineering versus programming, and also on the synergistic effect of knowledge engineering, especially regarding knowledge creation and discovery, were provided. The memorandum talked about the main difficulties in knowledge acquisition, e.g., expressing, transferring and structuring the knowledge, and how these difficulties can be somewhat surmounted. Some of the main methods of knowledge acquisition, especially the various forms of interview with subject-matter experts, were reviewed. A methodology that can be used for the development of a knowledge base was presented. The ontology development process, which refers to the management, development and support activities performed when building ontologies, was discussed, along with the evolution of the ontology development methodologies.

SAIF has a critical role to play in the C2 and intelligence processes. For this reason, numerous ongoing DND and DRDC projects have an important SAIF component. However, despite the importance given to SAIF in many DND strategic documents and projects, the current systems and the associated technology often fail to meet the demanding requirements of the operational decision makers; major science and technology advancements are required to really achieve the full potential of the related enabling technologies and to best serve the operational communities. The effort reported here is one step in this direction; it contributes to the establishment of some solid basis on which a long-term R&D program should be built.

Clearly, adopting a knowledge-centric view of SAIF and developing a complete support system along the approach and framework presented in [Roy, 2007-B] is a very challenging, multidisciplinary enterprise. Project teams will certainly have to seriously deepen the aspects described in this memorandum along the way. Along this line of thought, R&D activities have been and are still currently being conducted to further investigate knowledge representation [Roy, Auger, 2008-A], and also reasoning processes, methods and systems [Roy, Auger, 2008-B] for use in knowledge-based SAIF support systems. Adopting the knowledge-centric view of SAIF also requires that knowledge (i.e., expertise) be eventually acquired from the subject matter experts (SMEs) of the different military and public security application domains. In this regard, knowledge and ontological engineering techniques have been and are still being investigated. Knowledge acquisition and validation sessions with SMEs are about to be conducted.

References

[Brachman, Levesque, 2004], Brachman, R. J. and Levesque, H. J., Knowledge Representation and Reasoning, Elsevier, Morgan Kaufmann, San Francisco, CA, 2004.

[Giarratano, Riley, 1998], Giarratano, J. and Riley, G., Expert Systems - Principles and Programming, PWS Publishing Company, Boston, 1998.

[Ginsberg, 1993], Ginsberg, M., Essentials of Artificial Intelligence, Morgan Kaufmann, San Francisco, California, USA, 1993.

[Gómez-Pérez et al, 2004], Gómez-Pérez, A., Fernández-López, M. and Corcho, O., Ontological Engineering, Springer, London, 2004.

[Lambert, 2001], Lambert, D. A., An Exegesis of Data Fusion, accepted for publication in Soft Computing in Measurement and Information Acquisition, Edited L. Reznik and V. Kreinovich. Studies in Fuzziness and Soft Computing, Physica Verlag.

[Merriam-Webster, 2003], Merriam-Webster's 11th Collegiate Dictionary, Software, Version 3.0, 2003.

[Roy, 2001], Roy, J., From Data Fusion to Situation Analysis, Proceedings of the Fourth International Conference on Information Fusion (FUSION 2001), Montreal, Canada, August 7-10, 2001.

[Roy, 2007-A], Roy, J., A Knowledge-Centric View of Situation Analysis Support Systems, DRDC Valcartier Technical Report, TR 2005-419, 2007.

[Roy, 2007-B], Roy, J., Holistic Approach and Framework for the Building of Knowledge-Based Situation Analysis Support Systems, DRDC Valcartier Technical Report, TR 2005-420, 2007.

[Roy, Auger, 2008-A], Roy, J. and Auger, A., Knowledge Representation Concepts, Paradigms and Techniques for Use in Knowledge-Based Situation Analysis Support Systems, DRDC Valcartier Technical Memorandum, TM 2006-755, 2008.

[Roy, Auger, 2008-B], Roy, J. and Auger, A., Reasoning Processes, Methods and Systems for Use in Knowledge-Based Situation Analysis Support Systems, DRDC Valcartier Technical Memorandum, TM 2006-756, 2008.

[Russell, Norvig, 1995], Russell, S. and Norvig, P., Artificial Intelligence - A Modern Approach, Prentice Hall, New Jersey, 1995.

[Schreiber et al., 2002], Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., Van de VElde, W., and Wielingia, B., Knowledge Engineering and Management: The CommonKADS Methodology, MIT Press, London, 2002.

[Stefik, 1995], Stefik, M., Introduction to Knowledge Systems, Morgan Kaufmann Publishers, San Francisco, California, 1995.

[Turban, Aronson, 1998], Turban, E. and Aronson, J.E., Decision Support Systems and Intelligent Systems, Fifth Edition, Prentice Hall, New Jersey, 1998.

[White, 1987], White, F. E. Jr., Data Fusion Lexicon, Joint Directors of Laboratories, Technical Panel for C^3, Data Fusion Sub-Panel, Naval Ocean Systems Center, San Diego, 1987.

This page intentionally left blank.

List of symbols/abbreviations/acronyms/initialisms

AI	Artificial Intelligence
AKAMIA	Advanced Knowledge Acquisition for Maritime Information Awareness
ANS	Artificial Neural System
ARP	Applied Research Program
ASCAT	Analyse de la situation pour le commandant d'armée tactique
ASFI	Analyse de la situation et fusion d'information
BC	Base de connaissance
C2	Command and Control
C2	Commandement et contrôle
C4ISR	Command and Control, Communication, Computer, Intelligence, Surveillance, and Reconnaissance
CKE-4-MDA	Collaborative Knowledge Exploitation for Maritime Domain Awareness
DND	Department of National Defence
DRDC	Defence R&D Canada
ECCESDM	Exploitation collaborative de la connaissance pour l'éveil situationnel du domaine maritime
ED	Expert du domaine
FOL	First-Order Logic
IA	Intelligence artificielle
IC	Ingénierie de la connaissance
IKM	Information and Knowledge Management
IO	Ingénierie ontologique
INCOMMANDS	Innovative Naval Combat Management Decision Support
JCDS 21	Joint Command Decision Support for the 21st Century
JIIFC	Joint Information and Intelligence Fusion Capability
KB	Knowledge Base
KBS	Knowledge-Based System
KE	Knowledge Engineering
KR	Knowledge Representation
LF ISTAR	Land Force Intelligence, Surveillance, Target Acquisition and Reconnaissance
MDN	Ministère de la Défense Nationale
MSOC	Marine Security Operations Centres

OE	Ontological Engineering
PRA	Programme de recherches appliquées
R	Relation
RC	Représentation de la connaissance
R&D	Research & Development
SA	Situation Analysis
SAIF	Situation Analysis and Information Fusion
SASS	Situation Analysis Support Systems
SATAC	Situation Analysis for the Tactical Army Commander
SAW	Situation Awareness
SME	Subject Matter Expert
SSAS	Systèmes de soutien à l'analyse de la situation
S&T	Science and Technology
TDP	Technology Demonstration Program
TM	Technical Memorandum

Distribution list

Document No.: DRDC Valcartier TM 2006-757

LIST PART 1 − Internal Distribution by Centre (All copies to be distributed in PDF format):

1 - Director General
3 - Document Library
1 - Head/ Intelligence & Information Section
1 - Head/C2 Decision Support Systems Section
1 - Head/System of Systems Section
1 - Head/ Spectral and Geospatial Exploitation Section
1 - Head/ Tactical Surveillance and Reconnaissance Section
1 - Mr Claude Roy (SDA)
1 - Mr Jean Roy (author)
1 - Dr Alain Auger (author)
1 - Mrs Régine Lecocq
1 - Mrs Valérie Lavigne
1 - Mr Yaïves Ferland
1 - Dr Michel Gagnon
1 - Mr Denis Gouin
1 - Dr Luc Pigeon
1 - Mr Alain Bergeron
1 - Mr François Lemieux
1 - Mr Marc-André Morin
1 - Mr Alexandre Bergeron-Guyard
1 - Mr Réjean Lebrun
1 - Dr Pierre Valin
1 - Dr Hengame Irandoust
1 - Dr Patrick Maupin
1 - Mr François Réhaume
1 - Dr Abderezak Benaskeur
1 - Dr Adel Guitouni
1 - Mrs Micheline Bélanger
1 - Dr Anne-Claire Boury-Brisset
1 - Mr Abder Sahi
1 - Mrs Catherine Daigle
1 - Mr Martin Blanchette
1 - Dr Richard Breton
1 - LCol Michel Gareau
1 - Mr Jean-Claude St-Jacques
1 - Dr Marielle Mokhtari
1 - Mr François Bernier
1 - Mr Michel Lizotte
1 - Mr Mario Couture

1 - Mr David Ouellet
1 - Mr Gaetan Thibault
1 - Mr Éric Dorion
1 - Mr Dany Dessureault

TOTAL LIST PART 45

LIST PART 2 − External Distribution by DRDKIM (All copies to be distributed in PDF format)

1 - Director - R&D Knowledge and Information Management

Library and Archives Canada
395 Wellington Street
Ottawa, ON K1A 0N4

1 - Library and Archives Canada

National Defence Headquarter (NDHQ)
101 Colonel By Drive
Ottawa ON K1A 0K2

1 - ADM(S&T) / Associate DG RDP
1 - ADM(S&T) / Directorate Science and Technology (Air)
1 - ADM(S&T) / Directorate Science and Technology (Land)
1 - ADM(S&T) / Directorate Science and Technology (Maritime)
1 - ADM(S&T) / Directorate Science and Technology (Policy)
1 - ADM(S&T) / Directorate Science and Technology (Human Performance)
1 - ADM(S&T) / Directorate Science and Technology (C4ISR)
1 - ADM(S&T) / Centre for Security Science (CSS); Attn: Dr Anthony Ashley – DG

DRDC Atlantic
9 Grove Street
Dartmouth NS B2Y 3Z7

1 - Dr Ross Graham – DG
1 - Mr Jim S. Kennedy
1 - Cdr Siegfried Richardson-Prager (SMO)
1 - Dr Mark McIntyre
1 - Mr Anthony Isenor
1 - Mrs Francine Desharnais
1 - Mrs Liesa Lapinski
1 - Mr Bill Campbell

DRDC Ottawa
3701 Carling Avenue
Ottawa ON K1A 0Z4

1 - Mr Dan Brookes
1 - Mr Chris Helleur, DRDC Ottawa
1 - Mr David DiFilippo

DRDC Toronto
1133 Sheppard Ave W.
Toronto, ON, M3M 3B9

1 - Mrs Carol McCann

TOTAL LIST PART 22

TOTAL COPIES REQUIRED (IN PDF FORMAT): 67

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.)

Defence R&D Canada - Valcartier, 2459 Pie-XI Blvd. North, Quebec (Quebec), Canada, G3J 1X5

2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.)

Unclassified

3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C, R or U) in parentheses after the title.)

Knowledge and Ontological Engineering Techniques for Use in Developing Knowledge-Based Situation Analysis Support Systems

4. AUTHORS (last name, followed by initials – ranks, titles, etc. not to be used)

Roy, J., Auger, A.

5. DATE OF PUBLICATION (Month and year of publication of document.)	6a. NO. OF PAGES (Total containing information, including Annexes, Appendices, etc.)	6b. NO. OF REFS (Total cited in document.)
October 2008	40	16

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Technical Memorandum

8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.)

N/A

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.)	9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)
Projects 11bh, 11hg, and 12of.	N/A

10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.)	10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)
DRDC Valcartier TM 2006-757	N/A

11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.)

(X) Unlimited distribution
( ) Defence departments and defence contractors; further distribution only as approved
( ) Defence departments and Canadian defence contractors; further distribution only as approved
( ) Government departments and agencies; further distribution only as approved
( ) Defence departments; further distribution only as approved
( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected.))

Unlimited announcement

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

Adopting a knowledge-centric view of situation analysis and information fusion requires that one cares about knowledge engineering (KE) and ontological engineering (OE) techniques. In this regard, this memorandum discusses many aspects of KE and OE, including the basic elements usually incorporated in a knowledge base (KB), the notion of a knowledge domain, and also axioms, definitions and theorems. The properties of good and bad KBs are described, and the distinction between KBs and databases is made. Some of the skills and characteristics that are desirable in knowledge engineers are listed. Discussions on KE versus programming, and also on the synergistic effect of KE, especially regarding knowledge creation and discovery, are provided. The memorandum talks about the main difficulties in knowledge acquisition, e.g., expressing, transferring and structuring the knowledge, and how these difficulties can be somewhat surmounted. Some of the main methods of knowledge acquisition, especially the various forms of interview with subject-matter experts, are reviewed. A methodology that can be used for the development of a KB is presented. The ontology development process, which refers to the management, development and support activities performed when building ontologies, is discussed, along with the evolution of the ontology development methodologies. The memorandum contributes to the development of a foundational R&D framework for two projects that are just starting at Defence R&D Canada.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

Knowledge-Based System, Knowledge Engineering, Ontological Engineering, Knowledge Base, Knowledge Acquisition

Defence R&D Canada

Canada's Leader in Defence
and National Security
Science and Technology

R & D pour la défense Canada

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale

DEFENCE R&D DÉFENSE

www.drdc-rddc.gc.ca