Handbook of Ontologies for Business Interaction

Peter Rittgen
University College of Borås, Sweden

Information Science REFERENCE

INFORMATION SCIENCE REFERENCE

Hershey · New York

Copyrighted material

Acquisitions Editor: Kristin Klinger
Development Editor: Kristin Roth
Senior Managing Editor: Jennifer Neidig
Managing Editor: Sara Reed
Copy Editor: Jeannie Porter
Typesetter: Amanda Appicello
Cover Design: Lisa Tosheff
Printed at: Yurchak Printing Inc.

Published in the United States of America by
Information Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue, Suite 200
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com/reference

and in the United Kingdom by
Information Science Reference (an imprint of IGI Global)
3 Henrietta Street
Covent Garden
London WC2E 8LU
Tel: 44 20 7240 0856
Fax: 44 20 7379 0609
Web site: http://www.eurospanonline.com

Copyright © 2008 by IGI Global. All rights reserved. No part of this publication may be reproduced, stored or distributed in any form or by any means, electronic or mechanical, including photocopying, without written permission from the publisher.

Product or company names used in this set are for identification purposes only. Inclusion of the names of the products or companies does not indicate a claim of ownership by IGI Global of the trademark or registered trademark.

Library of Congress Cataloging-in-Publication Data

Handbook of ontologies for business interaction / Peter Rittgen, editor.
p. cm.
Summary: "This book documents high-quality research addressing ontological issues relevant to the modeling of enterprises and information systems in general, and business processes in particular covering both static and dynamic aspects of structural concepts. It provides reference content to researchers, practitioners, and scholars in the fields of language design, information systems, enterprise modeling, artificial intelligence, and the Semantic Web"--Provided by publisher.
Includes bibliographical references and index.
ISBN-13: 978-1-59904-660-0 (hardcover)
ISBN-13: 978-1-59904-662-4 (ebook)
1. Business enterprises--Computer networks. 2. Ontologies (Information retrieval) 3. Knowledge representation (Information theory) 4. Conceptual structures (Information theory) 5. Semantic Web. I. Rittgen, Peter, 1964-
HD30.37.H365 2008
658.4'038011--dc22
2007023438

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

All work contributed to this book set is original material. The views expressed in this book are those of the authors, but not necessarily of the publisher.

If a library purchased a print copy of this publication, please go to http://www.igi-global.com/reference/assets/IGR-eAccess-agreement.pdf for information on activating the library's complimentary electronic access to this publication.

Copyrighted material

Table of Contents

Copyrighted material

Copyrighted material

Copyrighted material

Detailed Table of Contents

Copyrighted material

Section II
General Domain Ontologies for Business Interaction

This chapter is a good example of the framework approach to a general domain ontology of business. The authors' framework is called content ontology design patterns (CODePs) where the constituents are described by modular, interoperable ontologies, for example, for descriptions and situations and plans. These CODePs can be used to reconstruct existing business modeling languages in terms of a common formal context.

This chapter takes a completely different approach towards a domain ontology for business interaction. Instead of following a line of philosophical reasoning, the authors take their point of departure in experiences from action research projects and generalize them into a theory called business action theory. This theory in turn is grounded in a general, albeit informal, ontology of the social realm, socio-instrumental pragmatism, where the focus is on social (inter)action that is mediated by artifacts.

This chapter starts from the same ontology as the previous chapter but aims at a different goal: formalizing the existing framework of socio-instrumental pragmatism by concretizing and refining the basic constituents, for example, actors, actions, and objects, and by providing an axiomatization in the form of associations between the constituents. The authors thus arrive at a meta-model that they apply to the reconstruction of an existing business modeling language to demonstrate the generality and descriptive power of the meta-model.

In this chapter the authors start from the assumption that self-awareness is an important prerequisite for business action, both human and organizational. But while self-awareness comes as a natural ingredient with human beings, it has to be developed and maintained in the case of organizations. To support this endeavour, the authors suggest an architecture and an ontology as a high-level business modeling framework. This framework combines social, organizational, and psychological theories with enterprise modeling approaches.

Copyrighted material

In this chapter, the authors aim at supporting the communication between business and IT experts at the requirements stage of an information systems development project. Their approach is supposed to facilitate the creation of a specific enterprise model that captures knowledge about the organization and its processes and that can be used to build an agent-oriented requirements specification of the information system to be built and the organizational environment in which it operates. To this end they develop an integrated meta-model or ontology of an enterprise in general that includes concepts from the managerial and information systems domains. These general concepts are instantiated with concrete entities from the particular organization.

Section III
Specialized Domain Ontologies

This chapter introduces a basic ontology of ICT management that comprises the concepts policy, project, assets and evaluation. The authors then go on to refine this core ontology by studying the possible contributions that some of the major organizational theories can make: stakeholder theory, theory of fit, theory of behavioral integration, agency theory, transaction cost theory, and theory of images of organization.

This chapter provides a specialized domain ontology for the memory of an organization. The development of this ontology follows a five-step process, two of which are elaborated in the chapter: analysis and structuring, and evaluation. The former addresses the classification of concepts derived from the literature and how they are mapped to ontological constructs. The results of this step are then validated in the evaluation step by assessing the conceptual coverage of the ontology.

In this chapter the authors start by identifying the interface between organizations and their information systems as the primary source of security risks. In order to address security issues, we therefore have to model the information systems together with their organizational environment. The authors provide a modeling language for this purpose that comprises a number of relevant concepts based on permission, delegation, and trust, and their Datalog semantics.

Copyrighted material

Copyrighted material

Copyrighted material

Section VI
Ontology Management

One aspect of ontology management is that of making ontologies dynamic, that is, providing a context-aware access to them. This chapter takes up that issue. The basic idea is to provide users with information that is meaningful in their current work context. This is achieved by generating views on ontologies which applications can use to query highly specialized knowledge bases.

This chapter views business networks as networks of service agents that describe their services in service descriptions. As each such description, and likewise each service request, is written in the light of the particular agent's ontology, semantic inconsistencies arise that lead to undetected matches or wrongly assumed matches between offers and requests. To solve this issue the authors introduce a compatibility vector system, based on schema-based ontology-merging, to determine and maintain ontology compatibility and to help with the identification of suitable business partners.

This chapter deals with an issue that arises in the creation of large ontologies, which are often built by merging smaller existing ontologies from relevant domains. Much of this work had to be done manually so far. The authors of this final chapter propose an automatic method for this task that can handle inconsistencies, redundancies, and different granularities of information.

Copyrighted material

Foreword

This book can be regarded as philosophical in talking about ontologies for business interaction, but, as I will argue, it is a rather practical book as well, impacting effectiveness in business interaction and information systems design. First I will just say a few words about myself so that you get an idea of this person advising you to spend time with this book.

To be honest, I myself do not talk so much about ontology because the classical concept of ontology refers to an idea that we can know about the basic structure, relations, and functions of the world: the ontology. In connection to that, we assumedly can also use clever strategies to reach this knowledge, epistemologies.

My thesis is that we all have different views of the world around us and that these views are partly manifested in language to describe, reflect, and act in the world. As humans we can also codesign such views and agree upon them as "views in action," making it possible for us to both communicate and act in new ways. Implementing these views in computer applications reinforces the power of human action many times. That is why it is so important to reflect upon the process of finding and using the best possible views or ontologies. And that is what the book is about.

In the book there are discussions that range from high-level ontologies that cover the whole idea of business and business development to specific areas and application domains. Inspired by that, I would like to take a "high-level" example to show the importance of this book.

We can use different ontologies on what constitutes a living human being. The two most well-known ontologies are hearth death and brain death. If we use the brain death ontology, it will open up a whole new business area with new options, dilemmas, and problems for a lot of people.

We can also be sure that the brain death ontology will evolve over time. We want to be absolutely sure that a person who has been declared dead will not become alive again, but we also want to make use of all the possibilities regarding transplantation and research that arise when a person is dead. Therefore, specialized domain ontologies are developed for different types of transplantations.

In all these cases, the ontology serves as the basis for the development of instruments and routines that include computing resources to a high degree. In other words, the ontology is the fundament that allows us to both communicate in the domain of specialized transplantations and to develop computer applications supporting successful transplantations.

This was a top-level general example of life and death. But the same principles do apply in all businesses and organizations. Ontologies are the backbone of new innovations and services, as many of the articles in this book describe.

As some of the chapters indicate, there is an even more important aspect of ontologies that has to be mentioned. In most cases the suggested solutions manifested as computer applications and work routines will not serve the intended outcome if the people involved are not involved in the process. Often we talk about this challenge as user participation or requirements management but, as you can see by reading this book, ontologies play a fundamental role even in this context.

Let us go back to the transplantation case. If people cannot trust the acting ontology, they will not sign agreements for transplantation and the whole idea will fail. It is, therefore, important that the ontology is translated into ordinary language so that people can have a chance to feel safe with the acting ontology.

Copyrighted material

This challenge is very fundamental and causes a lot of problems in development projects in many organizations and businesses. In this book we can find clues to successfully handling this challenge with the help of metaphors. That is the art of using existing languages when discussing new phenomena. This is an advanced task but of crucial importance if new ontologies are going to have positive impact on human life. This book gives some advice in this direction and my estimation is that we will find a lot more research about this in the future. May be it will not just be a question of life and death for business ideas but also for civilizations.

Olov Forsgren
University College of Borås, Sweden

Copyrighted material

Preface

MOTIVATION

Even the well-disposed reader might ask the question: Why should we concern ourselves with ontologies for business interaction? The answers to this question are many-fold. For one, a renewed interest in ontologies has only recently been fueled by the efforts around the Semantic Web and Web 2.0 (Shadbolt, Hall, & Berners-Lee, 2006) where ontologies are a core technology. But the involvement of ontologies in today's business world goes deeper than that. This is witnessed by the vast amount of literature on enterprise engineering (Davenport & Short, 1990; Fox, Gruninger, & Zhan, 1994; Gustas & Gustiene, 2004; Jochem, 2002) and enterprise modeling (Barrios & Nurcan, 2004; Fox, 1994; Fox, Barbuceanu, & Gruninger, 1996; Fox, Barbuceanu, Gruninger, & Lin, 1998; Fox & Gruninger, 1998; Gruninger & Fox, 1996; Jureta & Faulkner, 2005; Liles & Presley, 1996; Shinkawa & Matsumoto, 2001). These disciplines are at the heart of many information systems projects and ontologies play a central role even there (Dietz, 2006; Dietz & Habing, 2004; Fox, Barbuceanu, & Gruninger, 1996; Fox, Barbuceanu, & Gruninger et al., 1998; Guarino, 1998; Jackson, 2004; Kof, 2004; Opdahl & Henderson-Sellers, 2002; Uschold, King, Moralee, & Zorgios, 1998; Wand & Weber, 1989; Weber, 1997).

But business interaction is a wide field and building ontologies for it is not a straightforward endeavor. There is not a unique vocabulary or terminology that we can use as a starting point but rather a multitude of languages that differ from industry to industry, from functional unit to functional unit, from organization to organization, and even from person to person. This makes it impossible to devise "the" business ontology. In order to cope with the intrinsic complexity of this task, ontology levels have been suggested.

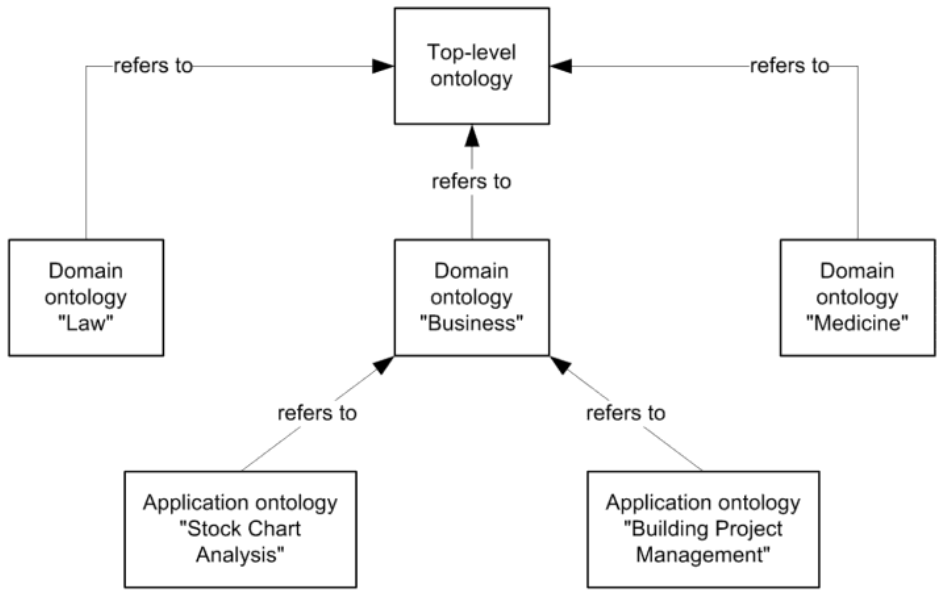
ONTOLOGY LEVELS

Ontologies are typically divided into foundational (or top-level), domain, and application ontologies (Bugaite & Vasilecas, 2005). Foundational ontologies cover the most general categories that can be expected to be common to all domains, such as "individuals" vs. "universals" or "substantials" vs. "moments." They are, therefore, domain-independent. Domain ontologies are tailored for a specific area of human activity, for example, medicine, electrical engineering, biology, or business. Application ontologies further restrict attention to a particular activity in a domain, for example, the diagnosis of lung diseases in medicine or a computer-based order handling system in business. Figure 1 shows the level architecture and names a few examples on each level.

It can be argued, though, whether three levels of ontology are adequate to cover the whole breadth of ontological endeavors. In the business domain, for example, we can identify any number of dimensions that justify further ontological levels. Let us consider a few examples. We distinguish between private-sector and public-sector organizations. Each organization belongs to some industry (banking, car manufacturing, retail, etc.) and it is divided into functional units such as procurement, production, marketing, sales, and so on. Along the hierarchy we have the strategic, tactical, and operational levels. In addition to these we might also consider a level below the application domain level, the personal level that takes

Copyrighted material

Figure 1. Ontology levels



into account, for example, the way in which an individual uses a particular information system for a particular task which is often different from the way others use the same system for the same or a similar task (Carmichael, Kay, & Kummerfeld, 2004; Dieng & Hug, 1998; Haase, Hotho, Schmidt-Thieme, & Sure, 2005; Huhns & Stephens, 1999).

Domain-Level Ontologies

The diversity of phenomena along all these dimensions makes it difficult to find an adequate level of abstraction that fits the whole business domain. In organizational theory, a number of metaphors have been suggested to understand and explain organizational behavior at a high level of abstraction. Metaphors establish a link between a source field and a target field and explain phenomena in the target field in terms of the source field. For organizational theory as a target field the following source fields have been proposed: the machine metaphor (Scott, 1997), living systems (biology) (Kendall & Kendall, 1993), open systems (Flood, 2005), the brain metaphor (Gareth, 1997), learning systems (Senge, 1990), social networks (Davern, 1997), complex adaptive systems (Anderson, 1999), autopoietic social systems (Luhmann, 1990), and so on. Using a metaphor implies a shift of domain. Existing ontologies for the source domain can, therefore, be transferred to the business domain.

But metaphors also imply some severe restrictions. By viewing organizations as, for example, living systems, we fail to capture those parts of organizational behavior that are not found in biology. Established approaches to a business ontology draw therefore on a number of different related theories to develop a richer picture of the domain. Theoretical contributions can come from communication theories, for example, Speech Act Theory (Austin, 1962; Searle, 1969, 1979) and Theory of Communicative Action (Habermas, 1984); social theories, for example, actor network theory (Law, 1992; Walsham, 1997) or structuration theory (Giddens, 1984); economic theories, for example, agency theory (Jensen & Meckling, 1976; Ross, 1973) or transaction cost economics (Coase, 1937; Klein, Crawford, & Alchian, 1978; Williamson, 1975, 1981, 1985); and others.

Copyrighted material

Examples of existing approaches to a general ontology of the business domain are found in Dietz (2006), Fox, Barbuceanu, and Gruninger et al. (1998), Fox and Gruninger (1998), Goldkuhl (2002, 2005), Goldkuhl and Lind (2004b), and Uschold et al. (1998).

Application-Level Ontologies

As such a general ontology of the business domain cannot be used directly in any concrete business application. It is therefore necessary to have at least one more level, the application ontology. Some researchers suggest additional levels, for example, task ontologies (Guarino, 1998). But instead of introducing a multitude of levels, we propose to interpret all these levels as different domain ontologies because most of the interesting problems already occur in the presence of a second level. So we just abstract from complexity levels that do not contribute to our discussion. We do not argue that a reduction to three levels is indeed sufficient. According to this definition, a domain ontology can be task-specific, company-specific, and so forth.

When we take a look at the application-ontology level we discover that the idea of having a separate ontology for every application is fraught with a severe problem, as many individuals and organizations make use of several applications within the context of a single task or business process. Let us consider two of the solutions that have been proposed to solve this problem. The first one, a bottom-up approach, aims at integrating the affected application ontologies, each of which could have been developed independently, to derive a higher-level domain ontology for the task or the specific organization. An example of this is given in Corbett (2003).

The second solution is top-down. It assumes the existence of a library of ontologies that is used to build an application ontology (e.g., on the task level) by re-using existing domain ontologies (e.g., on the business process level). Systems that support this are called ontology library systems. Examples of such systems are WebOnto (Domingue, 1998), Ontolingua (Farquhar, Fikes, & Rice, 1997), and SHOE (Heflin & Hendler, 2000).

THE STRUCTURE OF THE BOOK

The structure of the book roughly follows the ontology levels stipulated above in the first three sections. We have decided, though, to drop the (unqualified) term domain ontology and the problematic term task/application ontology and rather speak of general vs. specialized domain ontologies instead. The remaining sections then deal with the development, use, and management of such ontologies. In a *Handbook of Ontologies for Business Interaction*, there is naturally a strong focus on domain issues as witnessed by the eight chapters in sections two and three. But domain issues also have considerable impact on the design of a foundational ontology. Evidence of this is given in Chapter III, where the authors identify problems in the business domain that call for the introduction of a unique object identifier already on the foundational level. We have therefore introduced a section that is devoted to ontological foundations. In the following, we give an overview of each section's content.

Ontological Foundations

This section provides an introduction to ontologies and addresses foundational issues. The first chapter, *Overview of Semantic Technologies*, is written by Anne Cregan. It shows the importance of Semantic Technologies for the future of computing and the role that ontologies play in that context. It delivers a compact introduction into a wide field and helps the reader in developing a better appreciation of the remaining chapters that highlight particular aspects in greater detail.

The second chapter is authored by Marcus Spies and Christophe Roche and is titled *Aristotelian Ontologies and OWL Modeling*. It shows how Aristotelian ontologies can be realized with the Web ontology language (OWL). The authors argue for the benefits of the Aristotelian approach to ontological modeling and discuss a detailed example of an OWL representa-

Copyrighted material

tion of such an ontology. They also deliver a number of reasons indicating advantages of an epistemological approach over the commonly used object-oriented approach in the area of domain knowledge engineering.

The third chapter by Werner Ceusters and Barry Smith, *Referent Tracking for Corporate Memories*, concludes this section. The authors take a realist stance in approaching business ontologies with the aim of turning them into a more faithful representation of the targeted portion of reality. They suggest realism-based ontologies as the foundation, in particular, basic formal ontology and granular partition theory, to describe the generic aspects of corporate memories. Referent tracking is used to capture the specific aspects, such as keeping track of each individual business entity.

After the foundational issues relevant for business interaction have been discussed thoroughly in the first section, we proceed to the domain level in sections two and three. Section two discusses domain ontologies on a general level, that is, not restricted to a specific task or application within the business domain. The third section then takes up solutions that are more specialized, that is, directed towards a specific business issue such as security.

General Domain Ontologies for Business Interaction

General domain ontologies try to capture the business domain in its breadth. This means that they claim to address all the essential constituents of enterprises and their behavior. As a consequence, these approaches do not cover any particular issue or constituent at a greater level of detail. They can rather be seen as frameworks that outline the contours of the business world. Such a framework can be used as a frame of reference by more specialized ontologies to fill it with content. The first chapter in this section, Chapter IV in the book, is a good example of this approach: *Ontology Design for Interaction in a Reasonable Enterprise* by Aldo Gangemi and Valentina Presutti. Their framework is called content ontology design patterns (CODePs) where the constituents are described by modular, interoperable ontologies, for example, for descriptions and situations and plans. These CODePs can be used to reconstruct existing business modeling languages in terms of a common formal context.

Chapter V, *Grounding Business Interaction Models: Socio-Instrumental Pragmatism as a Theoretical Foundation*, by Göran Goldkuhl and Mikael Lind, takes a completely different approach towards a domain ontology for business interaction. Instead of following a line of philosophical reasoning the authors take their point of departure in experiences from action research projects and generalize them into a theory called business action theory. This theory, in turn, is grounded in a general, albeit informal ontology of the social realm, socio-instrumental pragmatism, where the focus is on social (inter)action that is mediated by artifacts.

Chapter VI, *Towards a Meta-model for Socio-Instrumental Pragmatism*, is authored by Peter Rittgen. It starts from the same ontology as the previous chapter but aims at a different goal: formalizing the existing framework of socio-instrumental pragmatism by concretizing and refining the basic constituents, for example, actors, actions, and objects, and by providing an axiomatization in the form of associations between the constituents. The author thus arrives at a metamodel that he applies to the reconstruction of an existing business modeling language to demonstrate the generality and descriptive power of the meta-model.

Chapter VII, *Towards Organizational Self-Awareness: An Initial Architecture and Ontology*, is written by the team of Marielba Zacarias, Rodrigo Magalhães, Artur Caetano, H. Sofia Pinto, and José Tribolet. They start from the assumption that self-awareness is an important prerequisite for business action, both human and organizational. But while self-awareness comes as a natural ingredient with human beings it has to be developed and maintained in the case of organizations. To support this endeavor the authors suggest an architecture and an ontology as a high-level business modeling framework. This framework combines social, organizational, and psychological theories with enterprise modeling approaches.

Chapter VIII, *An Agent-Oriented Enterprise Model for Early Requirements Engineering*, by Ivan J. Jureta, Stéphane Faulkner, and Manuel Kolp, concludes this section. The authors aim at supporting the communication between business and IT experts at the requirements stage of an information systems development project. Their approach is supposed to facilitate the creation of a specific enterprise model that captures knowledge about the organization and its processes and that can

Copyrighted material

be used to build an agent-oriented requirements specification of the information system to be built and the organizational environment in which it operates. To this end they develop an integrated metamodel or ontology of an enterprise in general that includes concepts from the managerial and information systems domains. These general concepts are instantiated with concrete entities from the particular organization.

The ontologies for business interaction contained in this section target the whole business. The following section addresses specific business activities such as ICT management, or particular aspects of business such as security and organizational memory.

Specialized Domain Ontologies for Business Interaction

The first chapter in this section, Chapter IX in the book, is written by Roy Gelbard and Abraham Carmeli. Its title is *Towards an Ontology of ICT Management: Integration of Organizational Theories and ICT Core Constructs*. It introduces a basic ontology of ICT management that comprises the concepts policy, project, assets and evaluation. The authors then go on to refine this core ontology by studying the possible contributions that some of the major organizational theories can make: stakeholder theory, theory of fit, theory of behavioral integration, agency theory, transaction cost theory, and theory of images of organization.

Chapter X, *KnowledgeEco: An Ontology of Organizational Memory*, is authored by Hadas Weinberger, Dov Te`eni, and Ariel J. Frank. It provides a specialized domain ontology for the memory of an organization. The development of this ontology follows a five-step process, two steps of which are elaborated in the chapter: analysis and structuring, and evaluation. The former addresses the classification of concepts derived from the literature and how they are mapped to ontological constructs. The results of this step are then validated in the evaluation step by assessing the conceptual coverage of the ontology.

Chapter XI, *An Ontology for Secure Socio-Technical Systems*, is written by Fabio Massacci, John Mylopoulos, and Nicola Zannone. The authors start by identifying the interface between organizations and their information systems as the primary source of security risks. In order to address security issues we therefore have to model the information systems together with their organizational environment. The authors provide a modeling language for this purpose that comprises a number of relevant concepts based on permission, delegation, and trust, and their Datalog semantics.

Chapter XI concludes this section and also the first half of the book and addresses the foundational and domain levels. The remaining sections deal with development, application, and management of business interaction ontologies. The next section, Section IV: Building Business Interaction Ontologies, shows how a concrete instance of an ontology can be created and filled with content.

Building Business Interaction Ontologies

This section contains two chapters that deal with the development of particular ontologies. The first one, Chapter XII in the book, is written by Paul Jackson and Ray Webster: Linking *Ontological Conceptions and Mapping Business Life Worlds*. The authors present a method for eliciting knowledge for the design of a corporate intranet within a government agency to solve knowledge management-related issues, for example, work duplication, document location, and accessing tacit expertise. The method combines soft systems methodology, causal cognitive mapping, and brainstorming to create a knowledge ontology using UML class diagrams. It is suitable for understanding nonroutine but rigorous knowledge and making it accessible to the designers of solutions.

Chapter XIII, *Modeling Semantic Business Process Models*, is authored by Agnes Koschmider and Andreas Oberweis. It focuses on the integration of business processes at the interface between partners in a value chain or network. This integration is tedious because partners do not only differ in the way they organize their processes but also in the languages they speak. This chapter attempts to solve the integration of diverging vocabularies by enriching the process modeling language of Petri nets with the Web ontology language (OWL).

Copyrighted material

Applying Ontologies in a Business Context

Section V subsumes five chapters that apply ontologies in a specific business context, for example, in the form of a case study in a particular company or a number of cases studies in an industry. The first chapter in this section, Chapter XIV in the book, is written by Juhnyoung Lee. Its title is *Ontologies for Model-Driven Business Transformation*. This chapter applies ontology to a model-driven approach to business analysis and transformation. It relates business processes and components on the one hand to IT solutions and capabilities on the other hand at different stages of the transformation. This is done by semantic models that show potential causes of problems during transformation and help with the identification of possible solutions. The authors also present a corresponding ontology management system that can be used in model-driven business transformation.

Chapter XV, *Ontology as Information System Support for Supply Chain Management*, is by Charu Chandra. It suggests a framework for information organization that is formalized as a reference model. This framework captures the specifics (e.g., dynamics and uncertainty) and functional requirements (e.g., information standardization and problem-orientation) of a supply chain which is interpreted as a managerial, dynamic, complex, and open system. It comprises an information modeling language that captures different aspects of the information system support for supply chains: a system taxonomy, a problem taxonomy, ontology, and ontology-driven information system.

Chapter XVI, *Matching Dynamic Demands of Mobile Users with Dynamic Service Offers* by Bernhard Holtkamp, Rüdiger Gartmann, Norbert Weißenberg, and Manfred Wojciechowski, describes the use of ontologies for personalized and situation-aware information and service supply of mobile users in different application domains. This is supported by a modular application ontology that is composed of upper-level ontologies for location and time and of domain-specific ontologies. This application ontology is used as a semantic reference model for a matching description of demands and offers in a service-oriented architecture.

Chapter XVII, *Knowledge Management Support for Enterprise Distributed Systems*, is written by Yun-Heh Chen-Burger, and Yannis Kalfoglou. It addresses issues associated with the overflow of information and the demand for semantic processing on the Web. The authors propose a semantic-based formal framework (ADP) that makes use of existing technologies to create and retrieve knowledge. Effectiveness is achieved by reusing and extending existing knowledge. The authors claim that the approach can also be used for organizational memories and knowledge management.

Chapter XVIII is jointly written by Carol Kort and Jaap Gordijn. It is titled *Modeling Strategic Partnerships Using the e3value Ontology: A Field Study in the Banking Industry*. The authors study a case from the banking industry where they evaluate strategic partnerships with the help of the so-called *e3value* ontology. The principle idea behind this approach is to model partnerships as networks for the mutual exchange of business values. It has been extended to cover investment arrangements and outsourcing which are relevant for strategic partnerships.

Chapter XIX, the final chapter of this section, is authored by Peter Weiß, *Towards Adaptive Business Networks: Business Partner Management with Ontologies*. The chapter investigates the support that ontologies can provide to manage business partner relations in large business communities. In such communities the task of building and maintaining a large number of relations becomes too complex to be handled by individual organizations or a central network manager. The chapter suggests an appropriate ICT infrastructure as a solution where ontologies offer support for communication processes and complex interactions of business entities in collaborative spaces.

Ontology Management

The first five sections of this book discussed how ontologies can be designed and deployed. The final section, Section VI, explores how they can be managed. One aspect of management is that of making ontologies dynamic, that is, providing a context-aware access to them. Chapter XX, *POVOO: Process Oriented Views On Ontologies Supporting Business Interaction*, by Eva Gahleitner and Wolfram Wöß, takes up this issue. The basic idea is to provide users with information that is

Copyrighted material

meaningful in their current work context. This is achieved by generating views on ontologies which applications can use to query highly specialized knowledge bases.

Chapter XXI, *Ontology-Based Partner Selection in Business Interaction*, by Jingshan Huang, Jiangbo Dang, and Michael N. Huhns, views business networks as networks of service agents that describe their services in service descriptions. As each such description, and likewise each service request, is written in the light of the particular agent's ontology, semantic inconsistencies arise that lead to undetected matches or wrongly assumed matches between offers and requests. To solve this issue the authors introduce a compatibility vector system, based on schema-based ontology-merging, to determine and maintain ontology compatibility and to help with the identification of suitable business partners.

Chapter XXII, *A Language and Algorithm for Automatic Merging of Ontologies*, by Alma-Delia Cuevas-Rasgado and Adolfo Guzman-Arenas, deals with an issue that arises in the creation of large ontologies, which are often built by merging smaller existing ontologies from relevant domains. Much of this work had to be done manually so far. The authors of this final chapter propose an automatic method for this task that can handle inconsistencies, redundancies, and different granularities of information.

Peter Rittgen
University College of Borås, Sweden

REFERENCES

Anderson, P. (1999). Complexity theory and organization science. *Organization Science, 10*(3), 216-232.

Austin, J.L. (1962). *How to do things with words*. Oxford: Oxford University Press.

Barrios, J., & Nurcan, S. (2004, June 7-11). Model driven architectures for enterprise information systems. In A. Persson & J. Stirna (Eds.), *Advanced Information Systems Engineering, 16th International Conference, CAiSE 2004, Riga, Latvia, Proceedings* (pp. 3-19). Berlin, Germany: Springer.

Bugaite, D., & Vasilecas, O. (2005). *Framework on application domain ontology transformation into set of business rules*. Paper presented at the International Conference on Computer Systems and Technologies - CompSysTech' 2005.

Carmichael, D.J., Kay, J., & Kummerfeld, B. (2004). Personal ontologies for feature selection in intelligent environment visualizations. In J. Baus, C. Kray, & R. Porzel (Eds.), *Artificial intelligence in mobile systems* (pp. 44-51). Saarbrücken, Germany: Universität des Saarlandes.

Coase, R.H. (1937). The nature of the firm. *Economica, 4*, 386-405.

Corbett, D. (2003, October 28-31). Comparing and merging ontologies: A concept type hierarchy approach. In N. Zhong, Z.W. Ras, S. Tsumoto, & E. Suzuki (Eds.), *Foundations of Intelligent Systems, 14th International Symposium, ISMIS 2003,* Maebashi City, Japan (pp. 75-82). Berlin: Springer.

Davenport, T.H., & Short, J.E. (1990). The new industrial engineering: Information technology and business process redesign. *Sloan Management Review, 32*(5), 554-571.

Davern, M. (1997). Social networks and economic sociology. A proposed research agenda for a more complete social science. *American Journal of Economics and Sociology, 56*(3), 287-301.

Dieng, R., & Hug, S. (1998). Comparison of <<personal ontologies>> represented through conceptual graphs. In H. Prade (Ed.), *ECAI 98. 13th European Conference on Artificial Intelligence* (pp. 341-345). New York: John Wiley & Sons.

Dietz, J.L.G. (2006). *Enterprise ontology: Theory and methodology*. Heidelberg, Germany: Springer.

Copyrighted material

Dietz, J.L.G., & Habing, N. (2004, October 25-29). A meta ontology for organizations. In R. Meersman, Z. Tari, A. Corsaro, P. Herrero, M.S. Pérez, M. Radenkovic, et al. (Eds.), *On the move to meaningful Internet systems 2004: OTM 2004 Workshops. OTM Confederated International Workshops and Posters, GADA, JTRES, MIOS, WORM, WOSE, PhDS, and INTEROP 2004,* Agia Napa, Cyprus (Vol. 3292, pp. 533-543). Berlin, Germany: Springer.

Domingue, J. (1998, April 18-23). Tadzebao and Webonto: Discussing, browsing, and editing on the Web. In B. Gaines & M. Musen (Eds.), *Proceedings of the 11th Knowledge Acquisition for Knowledge-Based Systems Workshop,* Banff, Canada.

Farquhar, A., Fikes, R., & Rice, J. (1997). The ontolingua server: Tools for collaborative ontology construction. *International Journal of Human Computer Studies, 46*, 707-728.

Flood, R.L. (2005). Unleashing the "open system" metaphor. *Systemic Practice and Action Research, 1*(3), 313-318.

Fox, M.S. (1994). Issues in enterprise modeling. In S.Y. Nof (Ed.), *Information and collaboration models of integration*. Dordrecht: Kluwer.

Fox, M.S., Barbuceanu, M., & Gruninger, M. (1996). An organization ontology for enterprise modeling: Preliminary concepts for linking structure and behavior. *Computers in Industry, 29*, 123-134.

Fox, M.S., Barbuceanu, M., Gruninger, M., & Lin, J. (1998). An organization ontology for enterprise modeling. In M. Prietula, K. Carley & L. Gasser (Eds.), *Simulating organizations: Computational models of institutions and groups* (pp. 131-152). Menlo Park, CA: AAAI/MIT Press.

Fox, M.S., & Gruninger, M. (1998). Enterprise modeling. *AI Magazine, 19*(3), 109-121.

Fox, M.S., Gruninger, M., & Zhan, Y. (1994). Enterprise engineering: An information systems perspective. In L. Burke & J. Jackman (Eds.), *3rd Industrial Engineering Research Conference Proceedings* (pp. 461-466). Norcross, GA: Institute of Industrial Engineers.

Gareth, M. (1997). *Images of organization*. London: Sage.

Giddens, A. (1984). *The constitution of society. Outline of the theory of structuration*. Cambridge: Polity Press.

Goldkuhl, G. (2002, April 29-30). *Anchoring scientific abstractions – Ontological and linguistic determination following socio-instrumental pragmatism.* Paper presented at the European Conference on Research Methods in Business and Management (ECRM 2002), Reading.

Goldkuhl, G. (2005). *Socio-instrumental pragmatism: A theoretical synthesis for pragmatic conceptualization in information systems.* Paper presented at the 3rd International Conference on Action in Language, Organizations and Information Systems (ALOIS), University of Limerick.

Goldkuhl, G., & Lind, M. (2004a, June 14-16). *Developing e-interactions – A framework for business capabilities and exchanges.* Paper presented at the 12th European Conference on Information Systems, Turku, Finland.

Goldkuhl, G., & Lind, M. (2004b). *The generics of business interaction: Emphasizing dynamic features through the BAT model.* Paper presented at the 9th International Working Conference on the Language-Action Perspective on Communication Modeling, Rutgers University.

Gruninger, M., & Fox, M. S. (1996). The logic of enterprise modeling. In P. Bernus & L. Nemes (Eds.), *Modeling and methodologies for enterprise integration*. London: Chapman & Hall.

Guarino, N. (1998). Formal ontology and information systems. In *Proceedings of the First International Conference on Formal Ontology in Information Systems (FOIS'98)* (pp. 3-15). Amsterdam: IOS Press.

Copyrighted material

Gustas, R., & Gustiene, P. (2004). Towards the enterprise engineering approach for information system modeling across organizational and technical boundaries. In *Enterprise information systems V* (pp. 204-215). Amsterdam: Kluwer Academic.

Haase, P., Hotho, A., Schmidt-Thieme, L., & Sure, Y. (2005). Collaborative and usage-driven evolution of personal ontologies. In A. Gómez-Pérez & J. Euzenat (Eds.), *Proceedings of the 2nd European Semantic Web Conference,* Heraklion, Greece (Vol. 3532, pp. 486-499). Heidelberg, Germany: Springer.

Habermas, J. (1984). *The theory of communicative action 1 - Reason and the rationalization of society*. Boston: Beacon Press.

Heflin, J., & Hendler, J. (2000). Dynamic ontologies on the Web. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)* (pp. 443-449). Menlo Park, CA: AAAI/MIT Press.

Huhns, M.N., & Stephens, L.M. (1999). Personal ontologies. *IEEE Internet Computing, 3*(5), 85-87.

Jackson, P. (2004). *Ontology and business: Creating structure for storing and accessing organisational knowledge on intranets*. Paper presented at the 13th European Conference on Information Systems, the European IS Profession in the Global Networking Environment, ECIS 2004, Turku, Finland.

Jensen, M.C., & Meckling, W.H. (1976). Theory of the firm: Managerial behavior, agency costs and ownership structure. *Journal of Financial Economics, 3*, 305-360.

Jochem, R. (2002). Enterprise engineering - The basis for successful planning of e-business. In L.M. Camarinha-Matos (Ed.), *Collaborative business ecosystems and virtual enterprises* (pp. 19-26). Amsterdam: Kluwer Academic.

Jureta, I., & Faulkner, S. (2005, October 24-28). An agent-oriented meta-model for enterprise modeling. In J. Akoka, S.W. Liddle, I.-Y. Song, M. Bertolotto, I. Comyn-Wattiau, S. Si-Said Cherfi, et al. (Eds.), *Perspectives in Conceptual Modeling, ER 2005 Workshops AOIS, BP-UML, CoMoGIS, eCOMO, and QoIS* (pp. 151-161). Berlin: Springer.

Kendall, J.E., & Kendall, K.E. (1993). Metaphors and methodologies: Living beyond the systems machine. *MIS Quarterly, 17*(2), 149-171.

Klein, B., Crawford, R., & Alchian, A. (1978). Vertical integration, appropriable rents, and the competitive contracting process. *Journal of Law and Economics 21*, 297-326.

Kof, L. (2004). Using application domain ontology to construct an initial system model. In M. Hamza (Ed.), *IASTED International Conference on Software Engineering* (pp. 18-23). Calgary, Canada: ACTA Press.

Law, J. (1992). Notes on the theory of the actor-network: Ordering, strategy and heterogeneity. *Systemic Practice and Action Research, 5*(4), 379-393.

Liles, D.H., & Presley, A.R. (1996). Enterprise modeling within an enterprise engineering framework. In J.M. Charnes, D.J. Morrice, D.T. Brunner, & J.J. Swain (Eds.), *Proceedings of the 28th Winter Simulation Conference* (pp. 993-999). New York: ACM.

Luhmann, N. (1990). The autopoiesis of social systems. In N. Luhmann (Ed.), *Essays on self-reference* (pp. 1-21). New York: Columbia University Press.

Opdahl, A.L., & Henderson-Sellers, B. (2002). Ontological evaluation of the UML using the Bunge-Wand-Weber Model. *Software and Systems Modeling, 1*(1), 43-67.

Ross, S. (1973). The economic theory of agency: The principal's problem. *American Economic Review, 63*(2), 134-139.

Scott, A. (1997). Modernity's machine metaphor. *The British Journal of Sociology, 48*(4), 561-575.

Copyrighted material

Searle, J.R. (1969). *Speech acts - An essay in the philosophy of language*. London: Cambridge University Press.

Searle, J.R. (1979). *Expression and meaning. Studies in the theory of speech acts*. London: Cambridge University Press.

Senge, P.M. (1990). *The fifth discipline. The art and practice of the learning organization*. New York: Doubleday.

Shadbolt, N., Hall, W., & Berners-Lee, T. (2006). The Semantic Web revisited. *IEEE Intelligent Systems, 21*(3), 96-101.

Shinkawa, Y., & Matsumoto, M. J. (2001). Identifying the structure of business processes for comprehensive enterprise modeling. *IEICE Transactions on Information and Systems, 84-D*(2), 239-248.

Uschold, M., King, M., Moralee, S., & Zorgios, Y. (1998). The enterprise ontology. *Knowledge Engineering Review, 13*(1), 31-89.

Verharen, E. (1997). *A language-action perspective on the design of cooperative information agents.* Tilburg: Katholieke Universiteit Brabant.

Walsham, G. (1997). Actor-network theory: Current status and future prospects. In A.S. Lee, J. Liebenau, & J.I. Degross (Eds.), *Information systems and qualitative research*. London: Chapman & Hall.

Wand, Y., & Weber, R. (1989). An ontological evaluation of systems analysis and design methods. In E.D. Falkenberg & P. Lindgreen (Eds.), *Information systems concepts: An in-depth analysis* (pp. 79-107). Amsterdam: North-Holland.

Weber, R. (1997). *Ontological foundations of information systems*. Melbourne, Australia: Coopers & Lybrand and the Accounting Association of Australia and New Zealand.

Williamson, O.E. (1975). *Markets and hierarchies*. New York: Free Press.

Williamson, O.E. (1981). The modern corporation: Origins, evolution, attributes. *Journal of Economic Literature, 19*, 1537-1568.

Williamson, O.E. (1985). *The economic institutions of capitalism*. New York: Free Press.

Copyrighted material

Acknowledgment

The editors would like to acknowledge the help of all involved in the collation and review process of the book, without whose support the project could not have been completed. A further special note of thanks goes to the staff at IGI Global, whose contributions throughout the whole process from inception of the initial idea to final publication have been invaluable. Deep appreciation and gratitude is due the University College of Borås for providing a unique research and teaching environment that stimulated and supported this year-long project.

Most of the authors of chapters included in this book also served as referees for articles written by other authors. Thanks go to all those who provided constructive and comprehensive reviews. However, some of the reviewers must be mentioned as their reviews set the benchmark. Reviewers who provided the most comprehensive, critical, and constructive comments include: Alma-Delia Cuevas-Rasgado of the Instituto Politécnico Nacional in Mexico City, Hadas Weinberger of the Holon Institute of Technology in Israel, Marielba Zacarias of the University of Algarve in Portugal, and Maurizio Ferraris of the University of Torino in Italy.

Special thanks go to the publishing team at IGI Global. In particular to Kristin Roth, who continuously prodded via e-mail to keep the project on schedule and to Mehdi Khosrow-Pour, whose enthusiasm motivated me to initially accept his invitation for taking on this project.

Special thanks also goes to my colleagues at University College of Borås who gave me many new insights and provided inspiring thoughts while enduring my lack of understanding. And last but not least, I am grateful to my colleague, Mikael Lind, for his unfailing support and encouragement during the months it took to give birth to this book.

In closing, I wish to thank all of the authors for their insights and excellent contributions to this book. I also want to thank all of the people who assisted me in the reviewing process. Finally, I want to thank my partner for her love and support throughout this project.

Peter Rittgen, PhD
Borås, Sweden
June 2007

Copyrighted material

Section I
Ontological Foundations

Copyrighted material

Copyrighted material

Chapter I
Overview of Semantic Technologies

Anne M. Cregan
National ICT, Australia
University of New South Wales, Australia

ABSTRACT

Semantic technologies are a new wave of computing, using explicit representation of meaning to enable data interoperability and more powerful and flexible information services and transactions. At the core of semantic technologies are ontologies, which capture meaning explicitly and may be used to manipulate and reason over information via its semantics. Unlike traditional data schemas or models, ontologies are capable of representing far more complex relations, may be linked directly to the data they describe, and have a formal logical semantics, facilitating automated deductive reasoning. This chapter introduces the vision of semantic technologies, and provides an overview of the approach and the techniques developed to date. It provides both an executive summary and an orienting framework for reading more technical material.

INTRODUCING THE VISION

I have a dream for the Web [in which computers] become capable of analysing all the data on the Web—the content, links, and transactions between people and computers. A "Semantic Web," which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy and our daily lives will be handled by machines talking to machines. The "intelligent agents" people have touted for ages will finally materialize. (Berners-Lee & Fischetti, 1999, p. 169)

Technology visionaries like Sir Tim-Berners Lee, the inventor of the World Wide Web, have long dreamed of such a seamless information technology platform (Berners-Lee & Fischetti, 1999) to support distributed business and government and personal interactions, as well as other information-based activities like research, learning, and entertainment. The benefits

Copyright © 2008, IGI Global, distributing in print or electronic forms without written permission of IGI Global is prohibited.

Copyrighted material

of sharing and using knowledge seamlessly, globally, and on demand hold great promise for the future of economics, government, health, the environment, and all areas of human life. Semantic technologies, which are designed to process information at the level of its meaning, hold the key for delivering this vision.

The amount of worldwide digital data generated annually is now measured in exabytes (10^{18} bytes), (Lyman, & Varian, 2003) providing access to unprecedented amounts of information. While methods and technologies to store data and retrieve it reliably and securely over distributed environments are well-developed and generally highly effective, the ready availability of vast amounts of data is, in itself, not enough. Each data store is designed within its own organization or business unit for a specific purpose, and the resulting vocabularies, data formats, data structures, data value relationships, and application processing vary considerably from one system to another. Faced with information overload and a spectrum of incompatibility, most organizations are experiencing a constant struggle to find, assemble, and reconcile even a portion of the potentially relevant and useful data, even within the enterprise itself, and the potential benefits of leveraging the knowledge implicit in this data are largely untapped.

Semantic Technologies are a new wave of computing (Niemann, Morris, Riofrio, & Carnes, 2005) that enable one system to make use of the information resident in another system, without making fundamental changes to the systems themselves or to the way the organization operates. In the same way that a universal power adaptor enables an Australian appliance to be plugged into a PowerPoint in Europe, the U.S., or Asia without the need to change the local power grid, semantic technologies enable semantic interoperability for IT systems with different data structures, formats, and vocabularies, without changing the core systems themselves. By providing more effective ways to connect systems, applications, and data, greater capabilities like intelligent search, automated reasoning, intelligent agents, and adaptive computing become possible, and the potential to leverage existing information for far greater benefits becomes realizable.

HARNESSING SEMANTICS

Typically, each IT system reflects the unique missions, work flows, and vocabularies of its own organization. Differences in syntax, structure, and the concepts used for representation prevent the interoperability of information across systems and organizations. Whilst middleware and data exchange standards like XML (Bray, Paoli, Sperberg-McQueen, Maler, & Yergeau, 2006) address some of the problems, they provide only a partial solution. The main obstacle in achieving efficient and seamless system integration is the lack of effective methods for capturing, resolving, and using meaning, a field referred to as "semantics."

To date, information processing has been primarily at the syntactic or symbol-processing level, whilst the semantic level—the level of the *meaning* of the information—has been relatively inaccessible to machine processes. The knowledge of exactly what the data means resides in the mind of the database architect, system designer, or business analyst, or, if made explicit, in a document or diagram produced by these people. Such documentation is not in an executable form and without a direct function in the live system it quickly becomes out of date. On the other side of the coin, the understanding of the needs and wants of the information consumer resides in their mind, and traditionally there has been no way for them to represent this directly or to match their needs with the system.

Semantic technologies provide the capability to handle information on the basis of its meaning, or semantics. The core idea of semantic technologies is to use logical languages to make the structure and meaning of data explicit, and to attach this information directly to the data, so that at run-time, automated procedures can determine whether and how to align information across systems. By enabling this "semantic interoperability" across systems, a linked virtual data structure is created, where the relevant data can be searched, queried, and reasoned over across multiple native data stores based on its common meaning.

Copyrighted material

KEY STRATEGIES OF SEMANTIC TECHNOLOGIES

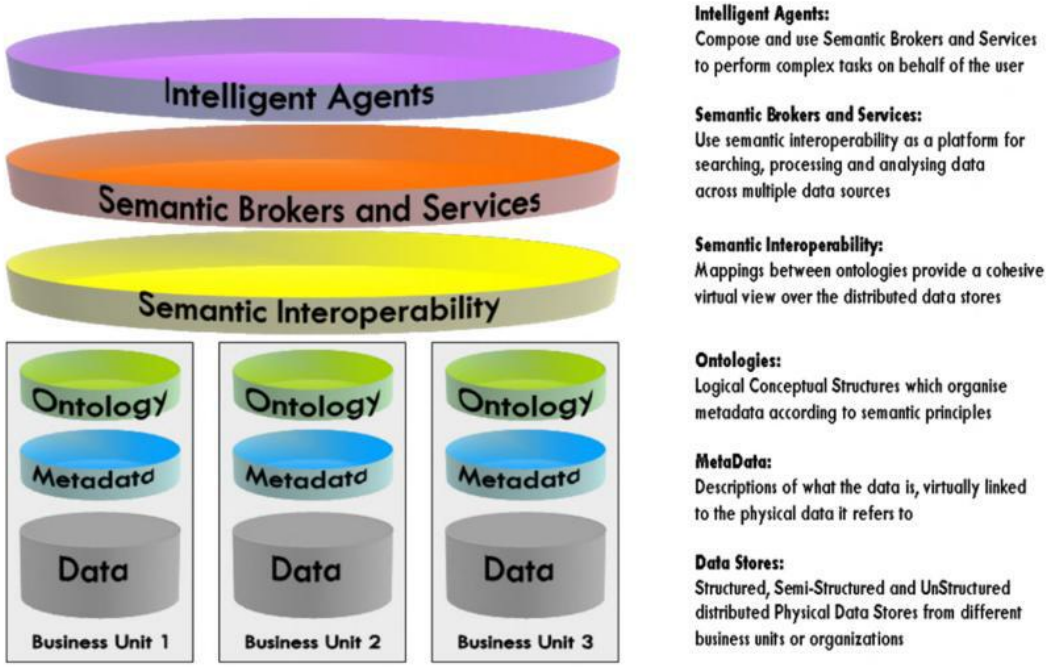
The goals of semantic technologies are twofold: firstly, to make distributed, disparate data sources semantically interoperable so that data can be retrieved and aligned automatically and dynamically on demand, and secondly, to provide techniques and tools to enable machines to intelligently search, query, reason, and act over that data.

Semantic technologies capitalize on the availability of data in sharable, processable electronic form. Some of these forms (e.g., databases and XML documents) contain structured data, and some contain less structured or unstructured data (e.g., text documents and Web pages). Semantic technologies can work with data in any form, providing it can be directly electronically linked into an ontology through some form of unique identifier. Ontologies are explicit, machine-readable specifications of the structure and meaning of data concepts, enabling automated processes to map and reconcile the data into a conceptually cohesive whole for searching and intelligent processing over the virtual data store created.

The key strategies used by semantic technologies are:

- Tagging physical data with metadata describing the data. Metadata is unlimited, in the sense that it can describe anything about the data. Additionally, because it links directly to the data it is about, the tag provides a handle for data identification and retrieval.
- Metadata tags are organized into logical structures called ontologies, which capture the logical and conceptual relationships between the tags, and provide a semantic map overarching the data.
- Aligning and mapping ontologies produces a semantic map over all the data sources, creating semantic interoperability, and providing the possibility for coordinated and seamless searching, querying, and processing over the virtual data structure.

Figure 1. Semantic technologies overview



Copyrighted material

- As ontologies are underpinned by formal logics, they support automated reasoning over the amassed data. Semantic interoperability thus provides a basis for semantic brokers and semantic services. Intelligent agents then may compose these services to perform more complex tasks on behalf of the user.

As shown in Figure 1, each level builds successively upon the previous one, while emphasizing the decoupling of data from applications for greater reuse and modularity.

Data

Rather than replacing existing database technology, semantic technologies allow data to continue to physically reside in its native environment, while providing improved access to the data via a conceptual virtual layer. By making the meaning of the data explicit, it may be harvested more easily for new uses. The data simply needs to be linked to a metadata tag via some form of unique identifier. For World Wide Web resources, unique resource identifiers (URIs) perform this function.

Metadata

Metadata is data about data. XML, for instance, is a common standard used to attach metadata tags to raw data. Metadata can be used to capture anything at all about data: its format, syntax, structure, semantics, pragmatics, or any other relevant aspect. Metadata about format and syntax can be used to guide processes which physically link and retrieve the data, while metadata about the data's structure and meaning can guide semantic alignment of the data. Pragmatic metadata can be used to capture information about how the data can be used in action. Semantic Technologies represent metadata in a form suited to logical manipulation, and are thus a tool which may be applied in any or all of these scenarios. As ontologies support the co-existence of multiple kinds of meta-data over the same data, there is no limit to the kind or amount of metadata that may be used to describe and organize the same information. Complex relationships within and between the various metadata can be harnessed and used for knowledge processing.

Ontologies

Ontologies are the key component of semantic technologies, whether for the Semantic Web or other applications. The word was borrowed from philosophy, but as applied to Semantic Technologies, it is commonly defined as "the specification of a conceptualization" (Gruber, 1993), and may be thought of as an explicit conceptual model representing some domain of interest.

Ontologies organize metadata tags, capturing the logical and conceptual relationships between them, and electronically linking each tag directly to the data or resource it represents. Typically ontologies describe the individuals, concepts, and relationships that are relevant for conceptualizing some real-world domain. The kind of knowledge they capture include:

- The concepts of the domain, and relations between the concepts such as broader, narrower, and disjointed. These set up the basic terminology of the domain.
- Properties that relate concepts to each other and to data fields, specifying the nature of the relationship, constraints on the relationship, and ranges for data values.
- Assertions or facts about individuals in the domain; for example, that a particular individual is an instance of a particular concept.

Ontologies are closely related to existing data modeling methodologies, but enable more explicit, richer descriptions, with more emphasis on the multiplicity of relationships and on precise formulation of logical constraints. One of the key principles of semantic technologies is to decouple information from applications, so that it can be redistributed and

Copyrighted material

re-used by other applications, both inside and outside the enterprise. Whilst current methodologies implicitly reference logical relationships, ontologies capture these explicitly, decouple them from the application, and make them available for machine processing.

For instance, a coded application procedure may make use of the programmer's knowledge about the way years, months, weeks, days, and hours are related in order to process temporal data, without actually making this knowledge explicit in a way that can be reused by other applications, or redeployed for unforseen purposes. In contrast, an ontology captures such knowledge explicitly, removing the need for it to be coded in at the application level, making the knowledge available for automated reasoning, and supporting reuse by other applications. As ontologies are the key enabler for semantic technologies, they are examined in depth in the section titled "Exploring Ontologies."

Semantic Interoperability

Mapping and aligning ontologies provides a cohesive semantic view of multiple data sources, enabling searching, querying, and reasoning across them as though they were a single data store. Mechanisms for one ontology to import and use another at run time are provided, as well as tools for the semantic alignment of ontologies. Aligned ontologies are connected via explicit mapping of the entities in one ontology via semantic relationships to entities in the other ontology. Such alignment can be human-mediated or semi-automated, using heuristics and matching algorithms.

Semantic Brokers and Services

Semantic brokers and services take advantage of semantic data interoperability to provide intelligent search and other reasoning-based services over the interlinked data. The use of ontologies supports model-driven applications to access and process executable models of the domain.

Intelligent Agents

Finally, intelligent agents can use semantic brokers to find and compose services to undertake complex tasks on behalf of users. The modularity of data, logic, and application supports the composition and redeployment of each element for new and innovative uses.

APPLICATIONS AND BENEFITS

The innovations that semantic technologies offer simplify the process of achieving interoperability between data sources, paving the way for vastly improved searching, querying, and reasoning over the amassed data.

The semantic interoperability community of practice (SiCoP) forecasts that in the near term, semantic technologies will deliver the capabilities of information integration and interoperability, intelligent search, and semantic Web services, and in the longer-term, will deliver model-driven applications, adaptive autonomic computing, and intelligent reasoning (Niemann et al., 2005). Each of these applications brings its own specific benefits.

Information Integration and Interoperability

Typically, an organization needs to work with and reconcile multiple data sources, including disparate systems within the enterprise or between different organizational systems in the supply chain, across an industry, between government organizations, or on the Web. The ability to seamlessly integrate these into a cohesive whole for search, querying, retrieval, and reasoning is clearly of great benefit. When business units or parts of the supply chain are not currently connected, or when a corporate merger takes place, the ability to connect data at a virtual semantic level, rather than having to physically merge it, is a powerful means to expedite operational efficiency and effectiveness.

Copyrighted material

it was possible to fully capture the models formally, ensure the conformance and logical consistency of implementations, and provide a basis for combining the implementations of different agencies into a unified whole.

The work of creating the ontologies was performed by TopQuadrant Consultants over a 3 month period. By creating a set of OWL ontologies to cover the five reference models, plus bridging and reference ontologies, they were able to create an ontology-based system to support an automated advisor to answer questions such as:

- Who is using which business systems to do what?
- Who is using what technologies and products to do what?
- What systems and business processes will be affected if we upgrade a software package?
- What technologies are supporting a given business process?
- Where are components being re-used or where could they be re-used?
- What are the technology choices for a needed component?
- How is our agency architecture aligned with the FEA?

An ontology graph was produced, which captured the rich relationships connecting the concepts stated across the five FEA reference models. These relationships provided a basis for understanding and reasoning over the overall model. Some of the resulting benefits included:

- Answering the listed questions through use of model querying and automated reasoning. For instance, automated graph traversal reasoning was used to infer "line-of-sight" between different enterprise entities.
- Context-specific information: a "capabilities advisor", using a semantic engine to advise different stakeholders on the capabilities available or in development to support the FEA and the U.S. presidents' e-government initiatives, was able to provide project-specific guidance for preparing business cases, ensuring project compliance with the FEA, knowledge of related initiatives and possible duplication, and candidate federal, state, and local partners for the project.
- Ability to dynamically generate cross-reference tables showing multidimensional agency relationships and capabilities, through use of a "model-browser" directly linked to the relevant data, ensuring an up-to-date view over all information gathered directly from the information source.

EXPLORING ONTOLOGIES

Broadly speaking, an ontology is any specification of a conceptualization, and, in this broad sense, can include virtually any kind of model or representation, including taxonomies, entity-relationship diagrams, flowcharts, and so on. In recent years, ontologies have drawn from the disciplines of artificial intelligence, particularly knowledge representation & reasoning, and formal logics, evolving the ability to represent more complex relationships supported by an underlying formal semantics. This section explores those capabilities. The Semantic Web ontology language (OWL) is currently the most well-developed language for building ontologies, and the examples and descriptions used may be taken to reflect OWL unless stated otherwise. Please note, however, that OWL is not confined to use on the Web: being XML-based it may be implemented as widely as XML itself.

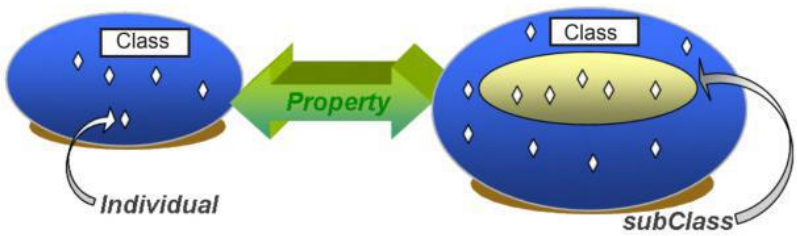
Expressing Knowledge

The typical constructs used by ontologies include classes (also known as concepts), instances (or individuals), and properties (or relations), which have a complex set of possible roles, interrelationships, and constraints.

Instances correspond to individual things that have associated properties, whilst classes are various

Copyrighted material

Figure 2. Typical ontology constructs

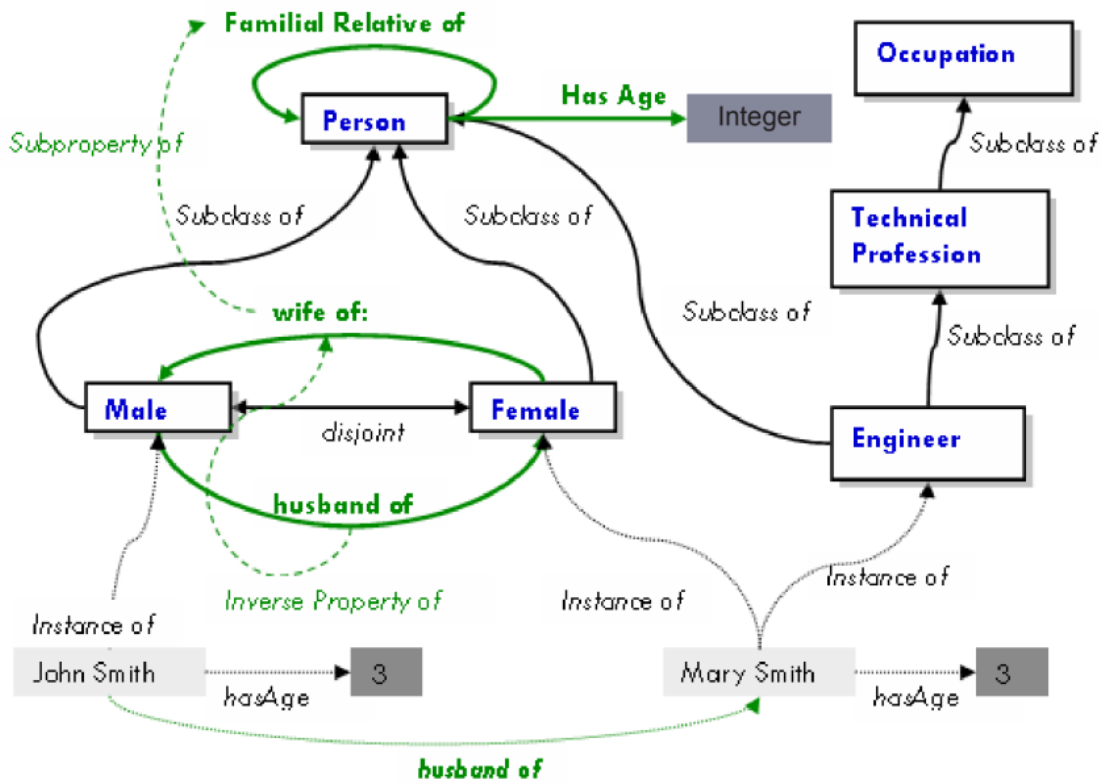


groupings over those things and properties are the connections between them. Figure 2 shows an example of an ontology illustrating these notions.

- *Classes* contain *instances*, for example, the class `Female` contains specific individual `Mary Smith`.
- *Classes* are typically related to each other by *subclass* relations, meaning that the instances in one class are a subset of another; for example, `Male` is a subclass of `Human`. Subclasses inherit the properties of all their superclasses; for example, if `Engineer` is a *subclass* of `Technical Profession`, and `Technical Profession` is a *subclass* of `Occupation`, then `Engineer` inherits all the properties of both `Technical Profession` and `Occupation`. This entails that instances of subclasses are automatically classified as instances of the classes above, for example, if `John Smith` is a `Male`, he is automatically an *instance* of the class `Human` also, inheriting any properties of `Human`.
- There can be distinct sets of class-subclass hierarchies that overlap; that is, ontologies are not just a tree (hierarchy) but a graph. For instance, `Engineer` can be a subclass of *both* `Technical Profession` and of `Person`.
- *Classes* may be *disjoint* from each other, that is, have no instances in common. For example, the class `Person` may have subclasses `Male` and `Female` defined to be disjoint from each other, so that no `Person` may be an instance of both `Male` and `Female`. A set of subclasses may also give complete coverage of the class they belong to—for instance, it can be specified that the two classes `Male` and `Female` completely cover the class `Person`, so that every `Person` must be an instance of either `Male` or `Female`; there can be no `Person` who is neither `Male` nor `Female`.
- *Classes* may have *properties* which connect them to specific literal values or individuals; for example, a `Person` may have a specific age which is a non-negative integer and have a specific relationship to other individuals. For example, a `Person` can be a familial relative of another `Person`. While the property is defined on the class, note that it applies to the individuals in the class, rather than to the class itself—that is, it is each individual `Person` who has an age value, not the class `Person` itself.
- *Properties* may have specific *domains* and *ranges*. For example, "husband of" is a property with domain `Male` and range `Female`. This means that the `Husband Of` property may *only* apply to an individual who is an instance of the class `Male` and may only connect that individual to an individual who is an instance of the class `Female`.

Copyrighted material

Figure 3. Example ontology



- *Cardinalities* and other property characteristics like *subproperty of*, *transitivity*, being the *inverse* of another property, and so on may be specified. For instance, `Has Age` is given a *cardinality* of exactly one: a `Person` has exactly one age. `Has Husband` would have a cardinality of maximum one: a `Female` may have no more than one husband, but may have no husband. `Has Wife` is the inverse property of `Husband Of`: if a certain `Male` is the husband of a `Female`, then that `Female` is the wife of the `Male`. `Has Husband` is also a *subproperty of* `Familial Relative Of`, and thus inherits from and specializes this property.

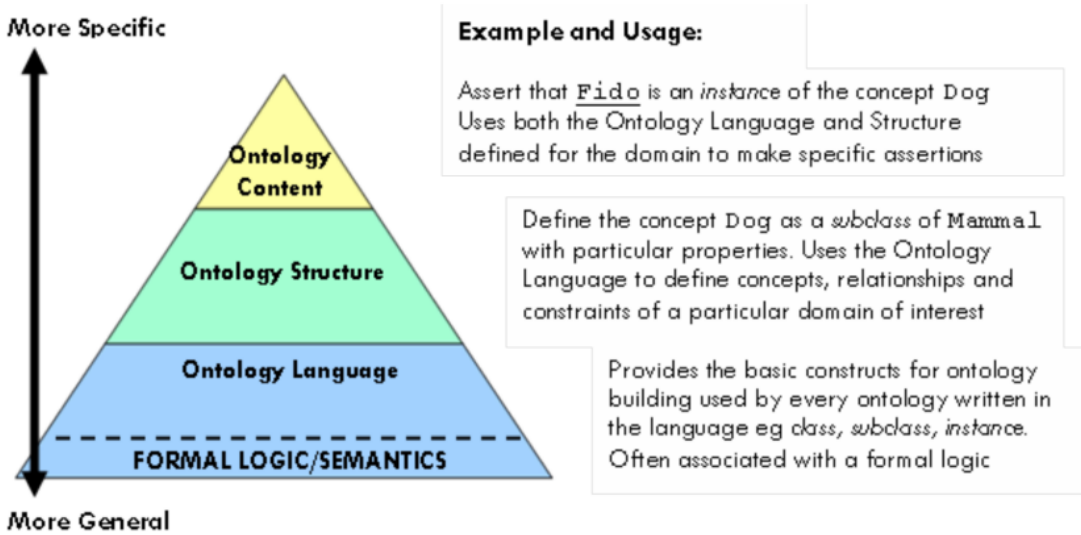
In building an ontology, there are potentially many design decisions in choosing how to represent the domain to be made to ensure the ontology will best suit the stated purpose. More than one model may be considered to be "correct", but usually some designs will provide the desired functionality more readily than others. As experience and understanding develops, ontology engineering is emerging as a research area and profession in its own right.

Components

Ontologies may be understood in terms of language, structure and content components. While closely intertwined, each component performs a separate and distinct function.

Copyrighted material

Figure 4. Ontology components



Ontology Language

The ontology language provides the fundamental modeling constructs for building specific ontologies. It includes language constructs relating to classes, properties, instances, and other formal constructs reflecting the various interrelationships and constraints these may have. Grammatical rules specify how these may be combined. Every ontology written in the ontology language uses these constructs, independently of the domain being modelled.

The particular language used is chosen by the ontology builder for its ease of use, expressivity, logical properties, and tool support. OWL is a very expressive ontology language, based on a kind of formal logic known as description logics. Several subspecies or "flavors" of OWL with different expressive and logical properties are available (OWL-Full, OWL-DL, and OWL-Lite). OWL essentially extends the constructs of the resource description framework (RDF) and RDF-schema.

Ontology Model Structure

Using the ontology language, a model is built to represent information about a domain of interest. This structure is like a template or stencil, specifying the concepts and the logical relationships and constraints they must satisfy in every specific case. For instance, the example in Figure 3 defines classes `Person`, `Male`, and `Female` and their relationships. In description logics, this part of the ontology is referred to as the T-box, as it is where the terminology is defined.

Ontology model structure is usually static in real-time processing (excepting the provisions for automated merging and importation between ontologies), but ontology authors or owners may choose to adapt and extend it as often as they wish, usually in a way that is backwards compatible with previous versions of the ontology, unless the entire conceptualization is radically changed. Ontologies for business are likely to be reasonably shallow and relatively static, whereas ones describing intricate research domains

Copyrighted material

like medical science may need progressive clarifications, extensions, and revisions as the underlying understanding of the research area evolves and the conceptual model changes.

Ontology Content

Ontology Content pertains to the object level of the ontology, corresponding to specific facts, individuals, and data values populating the ontology model structure. In the example in Figure 3, this would include specific individuals like `John Smith`, his gender, age, and relationships. In description logics this is referred to as the A-box, as it is where assertions are made. Ontology content reflects and conforms to the ontology model structure, for example, when `John Smith` is asserted to be `Male`, he is automatically a `Person`, because `Male` has been defined as a subclass of `Person` at the structural (terminological) level.

Depending on the tools that are used to construct and edit the ontology, in some cases it will not be possible to insert inconsistent or nonconformant data, and in others errors will automatically be identified. For instance, an attempt to assert that `John Smith` is both `Male` and `Female` will either not be permitted or will be flagged as an error, if the ontological structure has specified that these two classes must be disjoint, and, therefore, can have no instances in common.

In some ontology languages, the structure and content level are not kept separate: for instance, in OWL-Full, a class may also be an instance, while OWL-DL and OWL-Lite do not allow this. While it gives more freedom of expression, this has ramifications for its inferencing capabilities, as the underlying logic is no longer tractable. In contrast, the OWL-DL and OWL-Lite flavors of OWL are well-behaved in every respect.

Features of Ontologies

Virtual Structures

An ontology is a virtual conceptual structure over distributed physical resources, dynamically linking multiple data sources. Elements of the ontology language, model structure, and content can physically reside anywhere: the ontology language may reside in a W3C namespace, linked in by its URI, the ontology structure can live on an analyst's desktop in another namespace, and the data can live in a corporate data base. For instance, John Smith's age may reside in a human resources database, while the ontology contains a unique identifier providing a direct link to this data. All that is needed is a way of uniquely specifying the address/location of the data or resource, via a URI or some other mechanism. This approach ensures that data can be maintained centrally and applications always access the current information.

Ability to Import, Merge, and Align at Run-time

Additionally, ontologies can import and build on other ontologies, providing the ability to reuse and extend ontologies. This can occur at the design phase, but can also occur at run-time, merged based on matching URIs or identifiers: if two data resources have the same URI, they are assumed to be the same, and a combined ontology structure is generated on this basis. There are also language constructs within ontology languages to explicitly specify that one information resource is the same as another, even though they may have different identifiers, for example, synonymous concepts in different ontologies.

Connection to Formal Logics

Typically, ontology languages are designed to have what is called a "formal semantics," which give inference rules for drawing valid conclusions from an existing knowledge base. This ensures that starting from a knowledge base that includes only propositions that are true, and following only the specified rules of inference to deduce more propositions, will be guaranteed to generate only statements which are also deductively true. Under certain circumstances, this process can also be guaranteed to produce every possible logical

Copyrighted material

Ontologies vs. Other Data Structures and Models

Ontologies are generally compatible with other methods, and rather than replacing them, leverage their value. A brief comparison of the key differences between ontologies and other technologies follows.

Ontologies vs. Data Models

While a data model may be the outcome of a conceptual analysis and provides the design for a database, the data model itself is not directly linked to the data, whereas an ontology is an explicit map of data, directly linked to the data. Updating a data model, such as an entity-relationship diagram, does not automatically generate new knowledge about instance data or adapt the way the data links to other data sources but updating an ontology can potentially do so. Data models are not as expressive as ontologies: ontology languages are richer and treat relationships as "first-class" constructs. Data models can usually be constructed as simple ontologies in a straightforward manner.

Ontologies vs. Unified Modeling Language (UML)

UML is a specific modeling language, and compared to current ontology languages, it provides more constructs because it is intended not only for data modeling, but for modeling processes, use cases, and so on. However, it is not linked directly or dynamically to the data and does not support automated reasoning. Work is proceeding on a formal semantics for UML, a UML standard notation for ontologies, and executable capabilities, and it is likely that a convergence between ontologies and UML will be reached at some point. In the short-term, a tool for importing UML data models directly to an ontology editor is certainly feasible.

Ontologies vs. Databases

While data can be stored within an ontology, ontology tools are generally not optimized to do this, and for performance reasons it is not be advisable unless working with a very small data set. Some data base vendors such as Oracle now support some aspects of semantic technologies in conjunction with more traditional data base technology. Rather than replacing databases, ontologies are best used in conjunction with then to providing a conceptual virtual view over the data, enabling interoperability and leveraging its value.

Ontologies vs. Taxonomies

Taxonomies and ontologies are related, but whereas a taxonomy has a tree structure, ontologies have a graph structure due to their ability to support multiple inheritance and to link via properties. Also, ontologies have a much richer ability to capture relationships than the basic taxonomical "is-a" relation. Taxonomies may be viewed as very simple ontologies.

Ontologies vs. Expert Systems

Ontologies may be considered as weak expert systems, in the sense that they make knowledge explicit, can use rules, and support deductive capabilities. However, ontologies are currently more geared to capturing knowledge than to decision making per se. Intelligent agents using ontologies may, in the future, incorporate some of the capabilities envisioned for expert xystems.

THE SEMANTIC WEB

Semantic Technologies and the Semantic Web

While "semantic technologies" is an umbrella term encompassing all those technologies that seek to explicitly specify, harness, and exploit meaning for automated processing, the Semantic Web is a specific application of this idea to the World Wide Web. The World Wide Web Consortium (W3C), in collaborative association with many researchers and other organizations, has developed a suite of complementary technologies

Copyrighted material

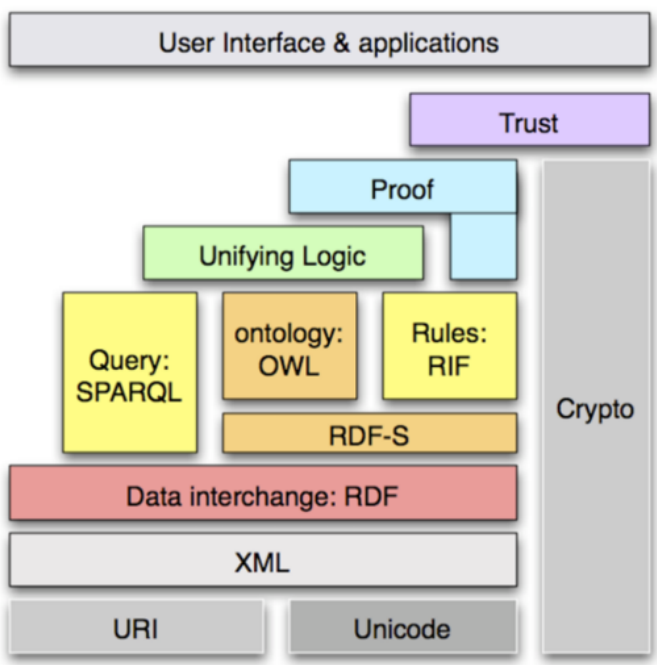
which together comprise the "Semantic Web." In one sense, the Semantic Web is narrower than semantic technologies generally, as not all semantic technologies necessarily make use of the W3C-endorsed recommendations. However, in scope, the Semantic Web is broader and probably more challenging than any other application, as it potentially interlinks data across the breadth and depth of the entire Web, and needs to handle the constant ebb and flow of available data sources across an unlimited and constantly evolving subject domain. The Semantic Web thus has to be more robust, or less brittle, than any other application of semantic technologies, as any brittleness is likely to produce cracks very quickly in such a demanding environment.

Idea

"The Semantic Web is an extension of the existing Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation" (Berners-Lee, Hendler, & Lassila, 2001).

The Semantic Web coalesced as a specific vision for the World Wide Web in 1998, initiated by Berners-Lee himself (Berners-Lee, 1998). However, many of the principles on which semantic technologies are based pre-date the Web itself, coming from diverse areas such as artificial intelligence, formal logics, database theory, information modeling, and library science. The Semantic Web has been an effective catalyst to crystallize the efforts of many research and industry groups into a cohesive and coordinated effort, and currently represents the most highly developed and complete approach for delivering Semantic Technology. The Semantic Web suite of standards is not confined for use only on the Web: it is equally applicable to enterprise systems for organizing internal data or across private data networks coordinating information between multiple participants.

Figure 6. Semantic Web "layer cake" (Berners-Lee & Swick, 2006)



Copyrighted material

Table 1. Description and status of Semantic Web 'layer cake' elements, as of August 2007

Element	Description
Unicode	The basic character set encoding (pre-existing). **Status: Operational**
URI Universal Resource Locator	Provides a mechanism for uniquely identifying and locating current and future resources on the Web. **Status: Operational**
XML Extensible MarkUp Language	XML provides a syntax for structuring data and tagging it, without specifying or constraining the structure or tags. XML Schema is a language for restricting the structure of XML documents. **Status: Operational**
RDF Resource Description Framework and **RDF-S** RDF Schema	RDF is a simple data model for referring to objects (known in RDF as resources) and specifying how they are related. An RDF-based model can be represented in XML syntax. RDF Schema is a vocabulary for describing properties and classes of RDF resources with a semantics for generalization-hierarchies of such properties and classes. **Status: Operational; significant Database vendor support implemented**
OWL Web Ontology Language	OWL adds more vocabulary for describing properties and classes, such as relations between classes (e.g., disjointness), cardinality (e.g., "exactly one"), equality, richer typing of properties, characteristics of properties (e.g., symmetry), and enumerated classes. **Status: Operational; Further extensions in development**
RIF Rule Interchange Format	Certain kinds of logical constraints cannot be implemented by OWL alone. Rule languages provide a means to implement these and are potentially very useful for encoding business rules. RIF working group currently active at W3C, developing a framework for rule interchange. **Status: Candidates are under consideration, including SWRL and Rule-ML**
SPARQL RDF Query Language	Provides the ability to query RDF. Similar in nature to SQL, but SPARQL allows for a query to consist of triple patterns (for RDF triples), conjunctions, disjunctions, and optional patterns. **Status: Candidate Recommendation**
Unifying Logic	A logical framework providing Formal Semantics for inferencing. OWL currently has a Description Logic basis. **Status: DL Formal Semantics for OWL (2004); Horn Logics proposed for RIF; continuing evolution.**
Proof	Logical conclusions by themselves are not convincing. This layer provides justification of inferences made, giving logical grounds for inferences. **Status: In development**
Trust	Once a basis of logic and proof is set up, it leads to an environment of trust for conducting transactions. **Status: A social variable, to be engendered by the technologies in development, especially Proof and Cryto**
Crypto	Support privacy and security. **Status: In development**
User Interface and Applications	Provide the semantic technology to the user through appropriate user interfaces and applications. The W3C has emphasised the need for more well-designed UIs to encourage the spread of Semantic Technologies. **Status: Mechanisms to embed RDF in existning Web are in development.**

Copyrighted material

Relationship to the World Wide Web

The Web's governing body, the World Wide Web Consortium (W3C), believes the Web can only reach its full potential when data can be shared, processed, and used by automated tools as well as people, and furthermore can be used by programs that have been designed independently of each other and the original data sources. The Semantic Web does not replace the existing Web, but builds on it, enabling better interoperability and further capabilities. While the existing Web focuses on uniquely identifying resources (URIs), displaying information using HTML, and publishing documents online, the Semantic Web focuses on data and interlinking it, ultimately supporting intelligent Web services/agents.

Semantic Web Components

The Semantic Web is comprised of several layers, summarized by Tim Berners-Lee's now famous "Semantic Web Layer Cake" shown in Figure 6, which has been revised several times since it first appeared.

The lower layers are already well-established Web standards used in the existing Web (URI, Unicode, XML), while the higher layers are specific to the Semantic Web, and build on the platform provided by the existing technologies. Table 1 contains a brief description of each element and its current status. Full details may be found by following the links at the W3C's Web site (www.w3.org).

ISSUES AND CHALLENGES

Semantic technologies range from being emergent to being quite well-developed and mature. Many of the key issues are social, rather than technological. The exposition in this chapter has concentrated on the vision of semantic technologies. While key technical components have been delivered, widespread adoption requires addressing several issues.

Large-Scale Semantic Markup of Existing Data

Semantic Technologies rely on data owners to semantically markup their data and, to date, there is no way to automate the process. There is an element of critical mass here: if only a few sources are marked up, not as much value is delivered. However, the benefits of sharing data effectively among even a few sources can be quite considerable. Whether it becomes universal remains to be seen, but key players in the IT industry are starting to embracing Semantic Technologies. The successful and widespread adoption of bottom-up tagging in Web 2.0 applications such as FlickR and del.icio.us has shown the potential of the approach and the willingness of participants to do tagging to support virtual communities. While existing tagging is essentially unorganized, Semantic Technologies can provide structure, logic, and reasoning to make tagging far more powerful.

Large Scale Data Manipulation and Querying

Tools and techniques for supporting large scale applications need further development. While ORACLE and other vendors currently support RDF triple stores, further integration with existing database technologies is needed. Some semantic techniques and algorithms need further optimization to ensure computing resources can adequately support them.

Ontology Building by NonExperts

A new initiative known as Sydney OWL Syntax helps nonlogicians to build ontologies by offering the option of using a simple English syntax for building and reading ontologies, instead of having to use formal logical or XML-based notations (Cregan, Schwitter, & Meyer, 2007). This is to be supported by a guided interface.

Copyrighted material

Standards and Methods for Resolving Meaning

Interoperability relies on representing data explicitly and mapping it with other representations. However, it is not always obvious whether it is appropriate for the elements of different ontologies to be mapped to each other and the constructs currently available in ontology languages for mapping them are somewhat limited. The ideal scenario would include a more descriptive mapping for data transformations and mediation without human intervention. One approach is to gather stakeholders to jointly develop ontologies for a domain, for use as a common standard. It is not imperative that everyone use the standard, only that they map their own ontology to it as a kind of "lingua franca". This avoids the need for pairwise mappings of every ontology needed for interoperability, as each can simply be mapped once to the common standard.

On the other hand, in order to truly automate the resolution of meaning in a way that is dynamic and adaptable, it is necessary to have a solid understanding of the underlying theory of semantics—not just formal semantics, but cognitive semantics, situated meaning, the identification of semantic primitives, and symbol grounding strategies. Work on upper level ontologies fits in this space, as well as the author's own work on symbol grounding for the Semantic Web (Cregan, 2007).

Dealing with Incomplete, Uncertain, and Probabilistic Data

Real world data tends to be imperfect and is not always suited to deductive reasoning. A W3C incubator group [URW3] has formed to investigate bridging the gap between the reasoning capabilities currently provided by Semantic Web technologies and what is needed to deal effectively with incomplete, uncertain, and probabilistic data.

Proof, Trust, and Security

The capability of accessing and dynamically lining data must be balanced with appropriate measures to handle who can access what data at what level, ensuring privacy, security, and the protection of digital rights. Challenges also include ensuring that intelligent agents will be accountable for their actions and able to provide useful and understandable explanations of the chain of reasoning underlying their decisions.

CONCLUSION

While there are issues and challenges to be addressed and further developments in the pipeline, semantic technologies are already sufficiently developed to be applied in real-world scenarios, as shown by the case study, to achieve interoperability and other benefits. Looking to the future, Semantic Technologies hold great promise for delivering more intelligent information services enabling much more effective support for finding, analyzing, and using knowledge, hopefully leading to the emergence of what might be called "pragmatic technologies" which use this knowledge as a basis for effective, informed, automated action.

ACKNOWLEDGMENT

NICTA is funded by the Australia Government's Department of Communications, Information, and Technology and the Arts and the Australian Research Council through Backing Australia's Ability and the ICT Center of Excellence program. It is supported by its members the Australian National University, University of NSW, ACT Government, NSW Government, and affiliate partner University of Sydney.

Copyrighted material

with proprietary terms and meanings, cooperation or collaboration becomes extremely difficult. As a consequence, without a semantic level of shared understanding, only very limited interoperability on the business level can be reached.

The most promising way to address the problem of common understanding, that is, a representation of agreed knowledge of actors from different partners to be used for business interaction, is to define *formal ontologies*, understood as an agreed vocabulary of common terms and meanings shared by a group of interaction participants. In recent years, the Web ontology language (OWL) has become one de facto standard for formal ontologies. Beyond supporting development of editors for ontology building, OWL is XML-serializable and therefore allows ontologies to be shared and to be represented using an interchangeable format (Smith, Welty, Volz, and McGuinness, 2003; Bechhofer et al.,2004). OWL can be used in three dialects, one of which can be made equivalent to various dialects of description logic (Baader, Calvanese, McGuinness, Nardi, & Patel-Schneider, 2003).

From a domain specific point of view, logic oriented ontology languages with interchangeable formats like OWL do not provide sufficiently precise guidelines for ontology-building from expert knowledge. A prominent case in point is the gene ontology, as Smith (2004) has shown. Being one example of a real world ontology used for enabling data interchange and data mining across heterogeneous representation standards in the life sciences laboratories of different countries and different expertise, this ontology lacks some fundamental semantic properties that are necessary to ensure consistency and correctness of logical inferences. These problems relate to issues like the distinction of sub/superclasses from that of a part/whole relationship. For details, see Smith (2004) and further references mentioned there. Relating to domain semantics, they cannot be discovered by offering mere description logic based ontology representation language.

For business interactions involving computer systems, Web services have allowed an unprecedented level of interoperability (Zimmermann, Tomlinson, & Peuser, 2003). The Web services protocol stack contains WSDL as a common language for service interface and implementation descriptions. This enables the necessary degree of common understanding in many cases. However, if services are to be composed on the basis of textual or other semantic description criteria, the interface definition is not sufficient to establish interoperability. Therefore, research into semantic Web services has proposed additional description languages. To be more specific, there are many categorization systems for business entities and business services that can be used as descriptors in UDDI (Bellwood et al., 2004). These categorizations are only a first step towards an encompassing semantic layer on top of the Web services protocol stack. OWL-S (The OWL Services Coalition, 2003) embeds service categories in service profiles that comprise input/output relations, preconditions, and effects, as well. The Web services modeling ontology WSMO (Fensel et al., 2006), proposes several metaconcepts for semantic Web services modeling, among which *goals, capabilities, and mediators* are the most prevailing. Both OWL-S and WSMO rely on *ontologies* providing agreed concepts and relationships within an interaction domain. The semantic layer defined in this way can be extended to provide semantic descriptions of business processes like those modeled in the business process execution language (BPEL). Using semantic frameworks for description of real services and their interactions can enable shared understanding on the business and on the technical level. This can be accomplished by suitably annotating Web service definitions with related business goals and by implementing mediation components that allow identification of partner services fitting a given semantic description and plug them into a given service.

In the present chapter, we will focus on ontologies for at least partially human driven business interactions. A general answer to the problem of defining languages suitable for establishing common understanding could be searched by looking at theories of concepts and definitions like in Margolis (1999), which, due to being rooted either in the analysis of natural

Copyrighted material

language semantics or of philosophical theories of meaning, provide meaningful constraints to business or service concepts and relations beyond those of mere first order logic or description logic. More specifically, we will focus on Aristotelian definition theory and its implications for ontology engineering.

The Aristotelian approach is based on the *specific differentiation* principle which focuses on differences between concepts. This theory distinguishes itself from description logic (Baader et al., 2003) by restricting classes in an interpretation that can correspond to meaningful concepts. A concept, according to Aristotle, is always defined by taking its next superconcept (or *genus*) and a defining property (*difference*) whose interpretation intersects with the superconcept class to give the subconcept class. Qualities like size or color cannot define a concept. Intersections or unions of classes interpreting a concept are usually not corresponding to concepts themselves—a sharp difference to description logic, where this is allowed in many dialects (Baader et al.). While Aristotle's view can be reconstructed using modern extensional first order logic (Berg, 1983), it restricts the way concepts may be formed much more than description logic in all its variants.

A more detailed description of the Aristotelian approach will be given. Since this approach allows us to construct classes from concepts of *natural kinds*, we argue that it greatly simplifies building a *consensual* ontology in accordance with knowledge of domain experts. See Roche (2000), which is in line with the work in Smith (2004).

In the present chapter, we will outline some of the central assumptions and consequences of the Aristotelian approach to conceptualization of a domain and building a suitable ontology. The main goal is, then, to show how we can translate Aristotelian ontologies in OWL. We will show that simple restrictions of the usage of OWL allow us to comply with the Aristotelian approach.

APPROACHES TO DOMAIN ONTOLOGIES

There are at least three traditional lines of research in which structures related to domain ontologies are built without reference to modern formal ontological methods as they are implemented in OWL tools.

Classification and taxonomy analysis has been used in biology since Linnés taxonomies. For a general overview, see Diderot (1755) and Foucault (1966), using systems of characterizing properties to define general classes (*genera*) and their specifications (*species*). Today, several methods of statistically-based cluster analysis (not assuming apriori known classes) and classification analysis (assuming apriori known classes) are used for data mining purposes. For an example, see Ester and Sander (2000).

Terminology construction is defining concepts based on *delimiting a characteristic which is an essential characteristic used for distinguishing a concept from related concepts* (Depecker et al., 2001).

Formal conceptual systems is an analysis of matrices of class properties into a conceptual lattice (Ganter & Wille, 1996). In the simplest case, the properties considered are binary and can be viewed as essential characteristic in the sense of ISO-1087.

There is still other work related to conceptual modeling, notably the theory of structural elements in UML 2 (Object Management Group, 2005), however, we will not explicitly determine the relationship of the present work to it.

The common denominator of the aforementioned lines of research is an approach to domain modeling going back to Aristotle, which we will quote following Berg (1983) and one of his best known commentators, Porphyry (1975), and refered to as Aristotelian ontological approach or simply Aristotelian ontology (where it should be noted that Aristotle's own concept of ontology was not instantiable to modeling single application domains). We will call this method alternatively epistemological, since it is derived from an approach of *theory of knowledge* rather than modeling

Copyrighted material

in terms of concepts and roles of description logic (Baader et al., 2003).

ARISTOTELIAN ONTOLOGIES

Aristotelian or epistemological ontologies construct a domain of entities by a sequence of definitions. Definitions, according to the famous formulation of Aristotle, proceed by stating the *genus proximum* and the *differentia specifica*. In this way, a concept tree is built by successive definitions of species from genders (*genera*), where defined species are used as genders for subspecies until a set of leaf species is reached. The concept tree corresponding to an Aristotelian approach is usually binary, since the very notion of difference in its traditional philosophical sense allows only dichotomous alternatives (like being an eternal being or a temporal being). An axiomatic reconstruction of Aristotelian definition theory on the basis of careful examination of all related writings by the philosopher has been given by Berg (1983), who also discusses exceptions to the strict tree structure of a definitional hierarchy.

In order to appreciate this approach, it is important to point out the distinction between *defining* and *describing* properties. Defining properties or differences are assumed to be essential predicates of a class, while properties or accidental attributes may, in general, apply to different individuals of a class in different degrees or not at all. For instance, a car can be described as having four seats (an object property); however, this does not contribute to the definition of a car, since a car-like vehicle with any number of seats greater than one still would commonly be called a car. Specific differences (or delimiting characteristics, in the parlance of terminology standards) are always defining—and therefore must be distinguished from properties or accidental attributes. A frequently used subset of describing properties are *qualities* like size (Berg, 1983). Defining properties or *differentiae* always delineate the extension of a *species* qua intersection with the extension of its closest *genus*. A *differentia* can be reused in various definitions of a conceptual hierarchy (examples of this follow).

There appear to be some principal hurdles in applying Aristotelian ontologies to practical domain modeling tasks:

- It has been discussed in the literature on concepts that often delimiting characteristics or specific differences are not readily available (Margolis, 1999). There are many examples indicating that the specific difference approach is not applicable without additional conventions.
- In Depecker et al. (2001), a distinction is made between specialization and comprehension (or part-of) hierarchies. While this distinction is important in practice, Aristotelian ontologies are based on the substance category rather than on the part-of relationship between objects. In practical solutions, part-of relationships may be used for expressing specific differences.
- Some problems arise if a defining difference of a concept is a relationship, for example, defining mother as a woman that has a child, a standard example in description logic, according to Baader et al. (2003). Here, the defining difference is not an *essential* predicate of the *definiendum*. In this case, the rigid Aristotelian approach would require not considering mother as a concept in its own right. Being a mother should be considered as a role in the sense of UML, according to the Object Management Group (2005)—that is, some individuals of the class of women implement the interface (in UML parlance) of giving birth to a child, and so forth, but that does not define a class (or a concept).
- Multiple inheritance can be realized implicitly by classifying an individual according to several ontologies or conceptual systems. So, in the Aristotelian view, concepts themselves may not have multiple generalizations, but individuals may be members of several classes (like you do not have multiple class inheritance in Java, but objects may realize several interfaces belonging to different hierarchies).

Copyrighted material

Aristotelian ontologies in our understanding are relative to a state of knowledge and a perspective of analysis. As an example, take the definition of men and women in an ontology of living beings. In this ontology, age is an attribute, since it is not a defining difference for either concept. However, in an ontology of beings in general, including eternal beings, having an age is a defining property of the non-eternal beings.

ARISTOTELIAN ONTOLOGIES FROM A LOGICAL POINT OF VIEW

Let us consider the definition approach according to Aristotelian logic in more detail. Let $C_0,\ldots, C_m$ be concepts, that is, generic notions (*genera*), and let $D_1,\ldots, D_n$ be unary predicates representing specific differences. Then, an epistemological ontology seems to be expressible for concepts $C_0,\ldots, C_m$ as follows:

$$(\forall x)C_0(x) \tag{1}$$

$$(\backslash\backslash\forall x) \quad (C_1(x) \Longleftrightarrow C_0(x) \wedge D_1(x)) \tag{2}$$

$$(\backslash\backslash\forall x) \quad (C_2(x) \Longleftrightarrow C_0(x) \wedge \neg D_1(x)) \tag{3}$$

$$(\backslash\backslash\forall x) \quad (C_3(x) \Longleftrightarrow C_1(x) \wedge D_2(x)) \tag{4}$$

Here, C_0 would be the root concept (like *thing* in OWL), and, in a full binary hierarchy, each of C_k, $k \geq 1$ would have C_j as *genus proximum* where $j = \lfloor k/2 \rfloor - (k+1) \bmod 2$. Note that in applications some of the concepts at different depths of the binary hierarchy will become leaf concepts.

A further property of the Aristotelian or epistemological ontology is that non-leaf concepts should correspond to abstract classes (in practice, not all of them do). Abstract classes in the OO sense appear as non-leaf nodes in the epistemological ontology (OO: object oriented systems). See, for example, Gamma, Helm, Johnson, and Vlissides (1995). Abstract classes may be unnamed (anonymous) in an Aristotelian ontology since in the domain of discourse targeted by the ontology there is no generally accepted name for the corresponding concept.

However, the set of formulae given above does not adequately capture the intended ontological meaning of definition by specific differences. The basic tenet of classical definition theory is that a specific difference is not applicable further up (towards C_0 in the conceptual hierarchy. E.g., the difference of *can fly* (yes or no) is not applicable to things in general. However, a straightforward representation with first order logic and unary predicates is not capable of representing this constraint due to the *tertium non datur* principle. One could only state that for individuals in, say, C_1, tautologically $D_2(x) \vee \neg D_2(x)$ holds. Instead, we need to assume some form of sorts in the individuals in order to express, for example, that $D_2(x)$ is not applicable to objects in $C_1(x)$ or $C_2(x)$. This corresponds to defining class hierarchies in OO modeling. Classes under some specific definition in such a hierarchy actually are assumed to be interpreted within some subuniverse of individuals.

Thus, in a standard first order logic framework with unary predicates only the notion of applicability of a predicate to an individual would lead to contradictions. One way to express the notion of predicate applicability is to use Hintikkas state descriptions. In the Hintikka (1974) formalization of state descriptions, a state is an exhaustive set of existence or nonexistence statements with respect to a finite set of predicates. If all predicates are unary, this corresponds to a list of possible worlds in analogy to propositional calculus. If n-ary predicates are allowed, applicability can be expressed as existence of related objects. For example, the predicate of fluidness is not applicable to human beings because they are no simple substances. In a suitable state description, a simple substance would be described as aggregate having one state (fluid, solid, or gaseous), while human beings would lack such a relationship to an individual of class aggregate state. As another example consider the notion of mortality. It is reasonable to postulate that only living beings or God can be mortal or immortal, while an molecule or a rock can neither be called mortal nor immortal—the

Copyrighted material

predicate of mortality is not applicable to simple substances. Again, this can be modeled by n-ary predicates in state descriptions. In this case we might say living beings have a slot (binary relationship) indicating their mortality, while molecules or rocks do not have such a slot. Note that this construction seems close to a dichotomous predicate; however, we are able again to express applicability, which would have been impossible without the slot in a state description. For the formulae reflecting these considerations, see Hintikka (1974) and Spies (2004).

Therefore, an Aristotelian ontology is not just a sequence of binary partitions of some universe, it is deeply related to object oriented modeling and domain ontology modeling. This led us to asking how an Aristotelian ontology might be represented in a domain ontology modeling language like OWL.

APPROACHES TO SIMULATING ARISTOTELIAN ONTOLOGIES IN OWL

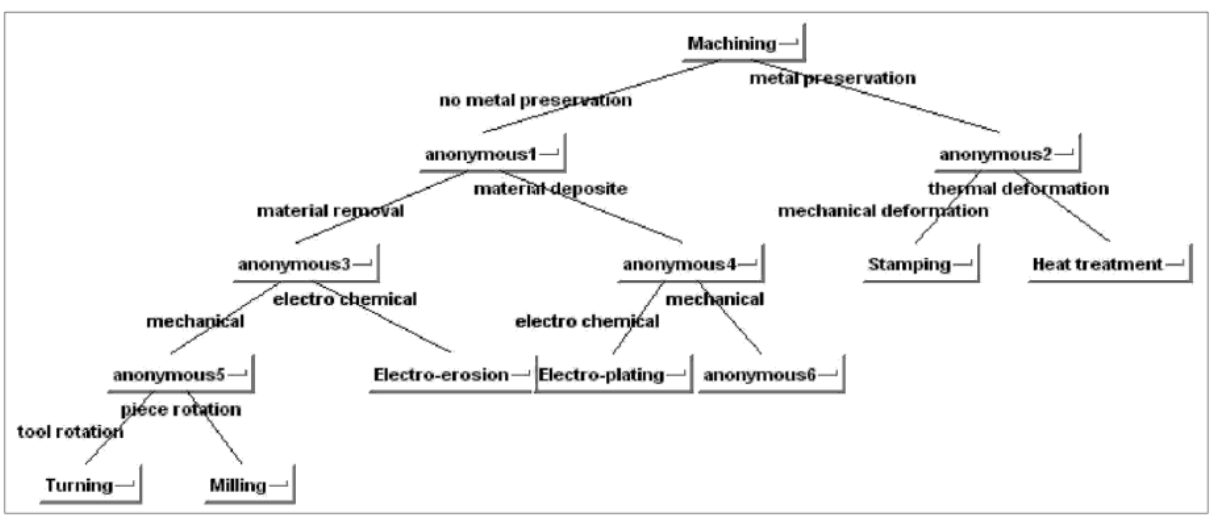
In order to examine the differences between an epistemological and a description-based approach to ontological modeling of application domains, an epistemological ontology that has been created by Roche (2000) for basic classes in machine processing of substances containing metal has been recast in OWL with the Stanford University Protégé ontology editor. For reference, we include in Figure 1 the ontology as formulated with the OCW editor of Condillac Research Group of Université de Savoie (Roche, 2000).

In this section, the principal assumptions of a working representation of an Aristotelian ontology in OWL are explained. For introduction to OWL, see Smith et al. (2003); for a reference, see Bechhofer et al. (2004). This approach has been tested on the machining ontology using Protégé 3.2 of Stanford University (http://protege.stanford.edu).

The main requirements of such an approach are:

- Concepts of an Aristotelian ontology must appear as classes in an OWL ontology.
- Classes in the OWL ontology should be related by subclass relationships according to the Aristotelian ontology.
- Non-leaf classes in the Aristotelian ontology correspond to abstract classes in an OWL model, for example, to classes without individuals. Non-leaf classes may be named or anonymous in an Aristotelian ontology.
- Leaf classes in the Aristotelian ontology must correspond to concrete or real classes in an OWL model, that is, to classes that may have

Figure 1. The machine processing Aristotelian ontology of Condillac Research Group



Copyrighted material

another property class, like anatomic or biophysical features.

Thus, in this approach, each leaf class in the class hierarchy has an object property with domain in its own class and range in a related class—here, for the sake of simplicity, we introduce only one related class, that of material. This object property is an individual of the property class modeling the specific difference applicable. The specific differences and their corresponding individual object properties are again inherited through the class hierarchy.

An important consequence of this approach to modeling epistemological ontologies with OWL concerns the multiple appearance of a specific difference in an ontology. In general, specific differences are not restricted to appearing only once in a class hierarchy. As for the property class of hasWings, it is well known that it is a specific difference for a subclass of invertebrates, as well. The same holds true for the corresponding object property class that represents this specific difference. In the present approach we can reuse the specific difference of having wings and just give it another instance for the insects' wings that allows to describe their anatomy and physiology at any desired detail. For example, our specific difference object property class hasWings can be instantiated to hasHummingWings to be applied to the class of honey bees.

Technically, multiple appearance of a specific difference in an ontology presupposes multiple inheritance of object property classes representing specific differences. In OWL, multiple inheritance is allowed for any subclass system (be it in the property or the class hierarchy). However, Aristotelian ontologies share the assumption of many modern object oriented programming systems in that they require single inheritance for the genders and species hierarchy. Thus, in the present modeling approach, the object property classes do not form a single-inheritance hierarchy, while the gender and species classes do. For a similar distinction between singly inheriting classes and a multiply inheriting hierarchy of properties, see Gamma et al. (1995), where the multiply inheriting hierarchy is referred to as type system.

In the mechanical production ontology of the Savoie University (Roche, 2000), it is the difference between electrochemical and mechanical substance change that appears both in the substance adding and in the substance removal branch of the class tree. From a statistical point of view, one might be tempted to describe such a situation as cross-classification. Note, however, that other subclasses of substance adding vs. removal evolve in the class tree without repeating distinctive features.

The application of the present modeling will now be illustrated in the context of the example ontology from Roche (2000). First, we show the class tree in OWL (see Figure 4). It should be noted that we introduced a second root class, Material, that serves as abstract domain of the specific differences in mechanical production. Basically, this amounts to defining a general object on which each mechanical production method operates. Of course, in other examples, none such reified relation domain may be necessary, while in others, more than a single additional abstract class will be required. Additionally, we depict the hierarchy of object properties used for the defining differences in the mechanical production ontology (see Figure 5).

As discussed before, real differences between individuals belonging to concrete classes are modeled as individuals belonging to some subclass of the class of difference object properties. Using this approach in Protégé will lead to all applicable individual properties (or property individuals) being offered in the individuals tab of the main interface menu. So, finally, we illustrate the use of these constructions by a screen shot that shows how individual properties evolve proceeding through the object hierarchy, see Figure 6.

It should be noted that OWL editors will need to be enabled at the OWL-full level (instead of mere description logic) in order to work with this approach. The DL analogy to subclasses are subproperties in OWL. With subproperties, however, Aristotelian difference systems can not be represented, because assigning subproperties to a class implies inheriting all property values of higher (or super-) properties. This would make reusing a difference in different

Copyrighted material

branches of the conceptual hierarchy impossible. It should be noted, however, that in all practically relevant cases we never iterate instantiation of property classes, therefore, there is no risk of undecidability of reasoning, as it can occur with OWL-full ontologies in general.

To sum up, in this approach we ontological differences as finite hierarchy of metaclasses. They comprise a finite set of individual properties as members from which to generate a suitable conceptual model.

DISCUSSION AND CONCLUSION

In this chapter, we argue that it is reasonable for practical ontology engineering to follow an approach derived from the definition theory by Aristotle. Some of our suggestions serve as an ontology design guideline in application projects in the manufacturing and human resource domains using the ontology editor by the Condillac research group at Savoie University. Our argument here, however, goes beyond these results in suggesting several methods for implementing the Aristotelian approach in the most commonly used ontology modeling language, namely OWL.

What is still missing is systematically gained empirical evidence of the practical implications of ontology editing using Aristotelian theory. However, the approaches outlined correspond exactly to design options in applications, like, for example, in building an ontology representation of taxonomies from the extensible business reporting language (for example, XBRL, see http://www.xbrl.org). The EU project MUSING (MUlty Industry Semantic based Next Generation Business Intelligence, see http://www.musing-project.eu) is building an integrated decision support platform which contains, among many other components, ontology representations for XBRL balance sheet data and metadata, regional economic indicators, and statistical models of their dependencies and predictive value for specific business decisions. In this highly demanding context, ontologies must satisfy functional requirements (integration into business applications) and still must remain understandable and

Figures 4, 5. The Protégé 3.2 class and property class hierarchies for the mechanical production ontology example

Figure 4.

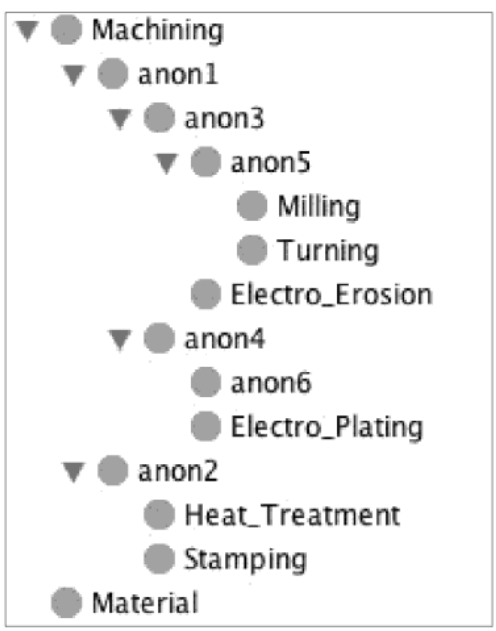
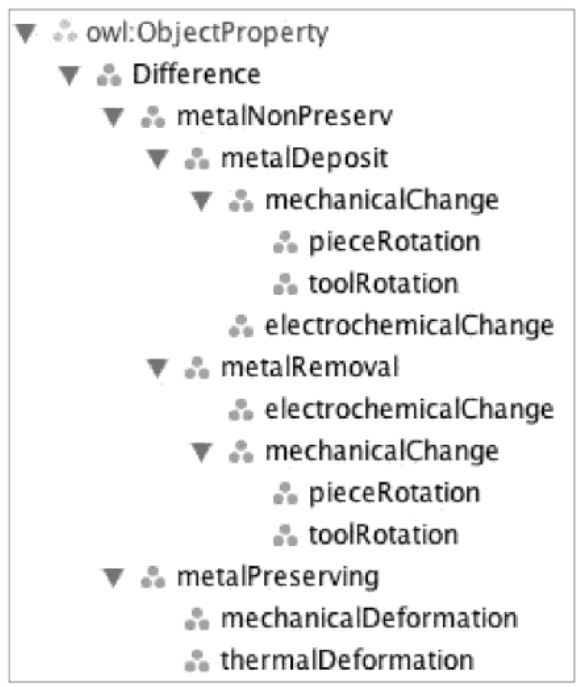


Figure 5.



Copyrighted material

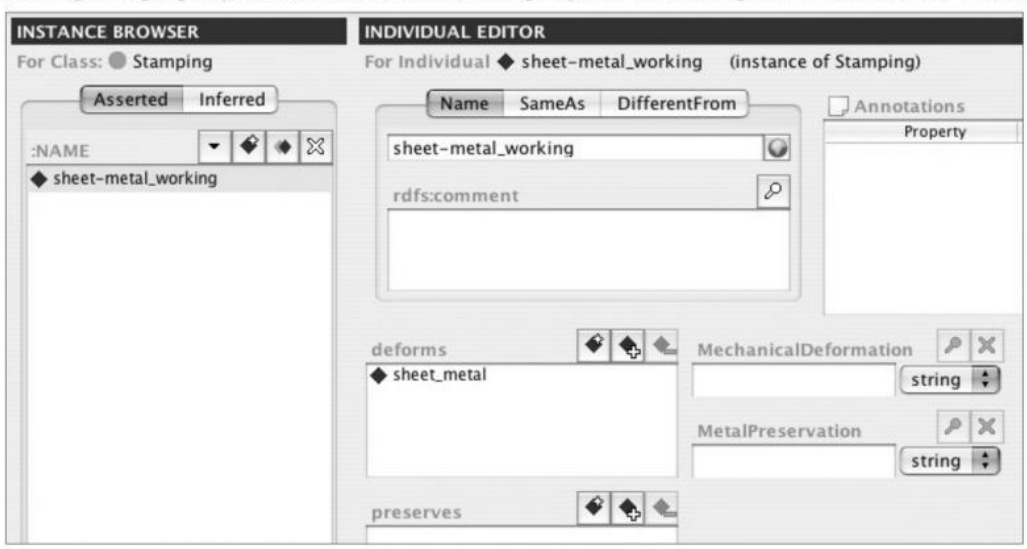
maintainable by domain experts. It appears that the Aristotelian approach outlined in the present chapter helps in fulfilling both requirements. More details on the experiences and methodologies from MUSING will be presented in subsequent publications.

To put our approach in perspective from a more philosophical point of view, it should be mentioned that the OWL approach to object modeling is, by far, not unique. For instance, Stroustrup (2003) explicitly rejected an approach reminding of ontologies in the description logic sense for his design of the C++ standard library STL. Therefore, one should be cautious in declaring any approach as generally recommendable because it is based on logic. Certainly, with the advent of a definition of concepts as functions in Frege (1973), the Aristotelian definition theory has lost some of its intuitive attractiveness. We have shown why in the first part of this chapter: It is by far not obvious how to model applicability of differences to individuals without assuming a sorted domain (i.e., by postulating domain properties that the ontology is about to introduce ...). However, as it was shown, the Aristotelian theory can elegantly be retrieved by the very means of the leading modern domain ontology languages, with a little help taken from Hintikkas' (1974) concept of state descriptions. Whether OO-flavored or epistemologically-flavored ontological modeling is applied in practice can, therefore, be decided on practical reasons alone.

There are at least three conclusions to be drawn from the results of the present chapter:

- Constructing a terminology the Aristotelian methodology provides a useful guideline because it forces us to distinguish defining features from accidental attributes. As an example, take the modeling decision in XML Schema to put information into an element item or an attribute item which is sometimes considered as an arbitrary choice. An element information item corresponds to a constituent of an entity, like an object property in OWL, while attribute information item corresponds to a property or even an accidental attribute in the Aristotelian approach. It should be helpful to think in these terms borrowed from a long philosophical tradition when working on practical modeling tasks.
- Another point is that the negation of a concept in an extensional sense often does not lead to another concept—this is similar to the situation in object oriented or entity relationship modeling. The negation of a class (defined by inverting the defining and non-defining features) is in most cases not itself a class (or a relational table or ...). Since most description logic dialects treat nega-

Figure 6. Protégé 3.2 property individuals inherited and specific to one concept (here sheet metal working)



Copyrighted material