Agents on the Web

Ontologies for Agents

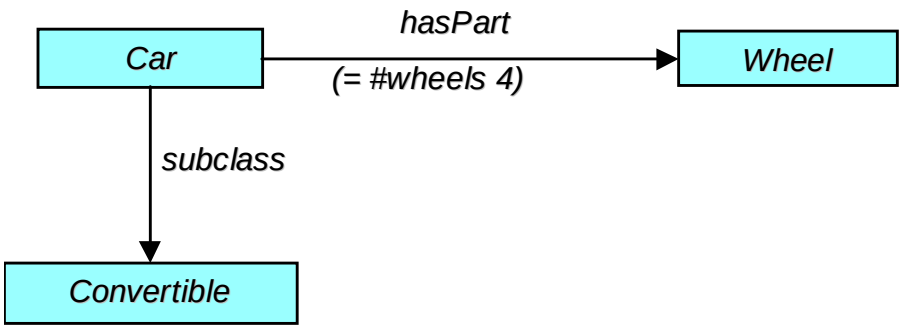
Michael N. Huhns and Munindar P. Singh

November 1, 1997

When we need to find the cheapest airfare, we call our travel agent, Betsi, at Prestige Travel. We are able to communicate with Betsi because we all speak the same language, English, and we all understand the basic elements of the subject matter under discussion, such as tickets, planes, destinations, departure times, and fares. But, suppose it is after hours and we are all busy, so we would instead like our software agent to contact Betsi's software agent and arrange our flights. None of our agents understands English, so how can they communicate? Ontologies may be the answer.

"What are ontologies?"

An ontology is a computational model of some portion of the world. It is often captured in some form of a semantic network—a graph whose nodes are concepts or individual objects and whose arcs represent relationships or associations among the concepts. Properties and attributes, constraints, functions, and rules that govern the behavior of the concepts augment the semantic network.



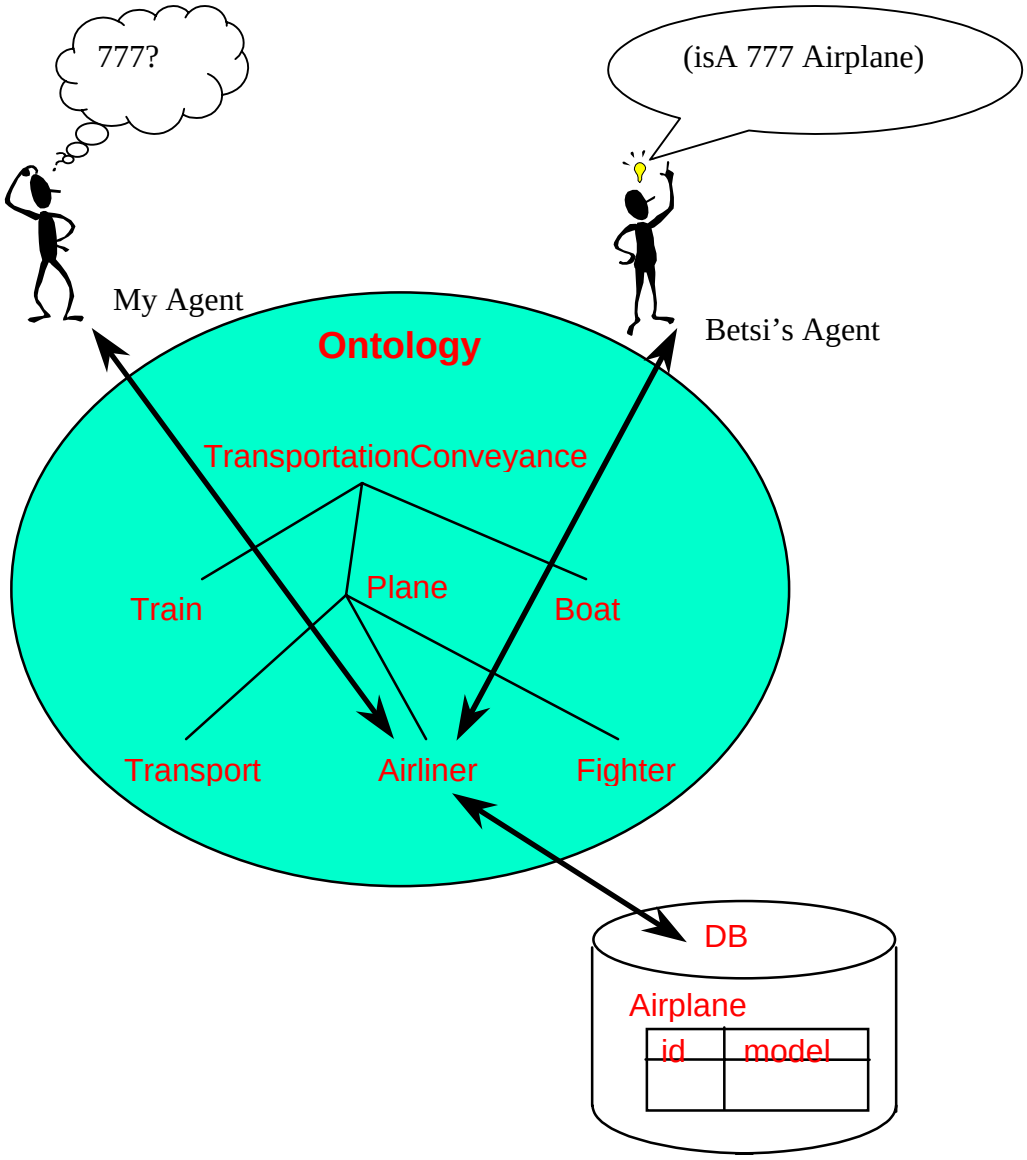
Formally, it is an agreement about a shared conceptualization, which includes conceptual frameworks for modeling domain knowledge and agreements about the representation of particular domain theories. Definitions associate the names of entities in a universe of discourse (e.g., classes, relations, functions, or other objects) with human-readable text describing what the names mean, and formal axioms that constrain the interpretation and well-formed use of these names. For information systems, or for the Internet, ontologies can be used to organize keywords and database concepts by capturing the semantic relationships among the keywords or among the tables and fields in a database. The semantic relationships provide users with an abstract view of an information space for their domain of interest.

Using an Ontology

Now, how can such an ontology help our software agents? It can provide a shared virtual world in which each agent can ground its beliefs and actions. When we talk with Betsi on the phone, we rely on the fact that we all live in the same physical world containing planes, trains, and automobiles. We know, for example, that a 777 is a type of airliner that can carry passengers such as us to our destination. When our agents talk, the only world they share is one consisting of bits and bytes—not a very interesting subject of discussion! An ontology gives the agents a richer and more useful domain of discourse.

We have written in an earlier column (see *IEEE Internet Computing*, vol. 1, no. 2) about a communication protocol, such as KQML, by which agents can exchange messages. The protocol specifies the syntax, but not the semantics of the messages. However, it also allows the agents to state which ontology they are presuming as the basis for their messages.

Now, suppose our agents have access to an ontology for travel, with concepts such as airplanes and destinations, and suppose that Betsi's agent tells our agent about a flight on a 777. Suppose further that the concept "777" is not a part of the travel ontology. How could our agent understand Betsi's agent? Betsi's agent could explain that a "777" is a kind of airplane, which is a concept in the travel ontology. Our agent would then know the general characteristics of a 777. This is illustrated in the figure below.



Features of Ontologies

A few of the most important relationships represented and supported in most ontologies are the following.

Generalization

Generalization and inheritance are powerful abstractions for sharing similarities among classes, while preserving their differences. Generalization is the relationship between a class and one or more refined versions of it. Each subclass inherits the features of its superclass, and then adds other features of its own. Generalization and inheritance are transitive across an arbitrary number of levels.

Aggregation

Aggregation is the part-whole or part-of relationship, in which classes representing the components of something are associated with the class representing the entire assembly. Aggregation is also transitive, as well as antisymmetric. Some of the properties of the assembly class propagate to the component classes.

Other Relationships

Some of the other relationships that occur frequently in ontologies are *owns, causes*, and *.contains. Causes* and *contains* are transitive and antisymmetric, and *owns* propagates over aggregation.

Meta Content Format

A recent development for the World-Wide Web is MCF, which stands for Meta Content Format. It is an open format language for representing a wide range of ontological information about content. Targeted content includes web pages, gopher and ftp files, desktop files, email, and structured (i.e., relational and object-oriented) databases. The corresponding metacontent includes indices such as Yahoo!, gopher**,** and ftp directory structures, email headers, data dictionaries, etc. There are now sites on the web that organize their information according to MCF, producing what are known as MCF Information Spaces (or Xspaces), and there are viewers available for your browser that let you "fly" through the Xspaces. <http://mcf.research.apple.com/>

Alternatives

There are many efforts underway to devise classification schemes and to use the schemes to build and populate classification structures. For the purpose of providing semantics for messages among agents, the following are all considered types of classification schemes of varying classification power: key words, thesauri, taxonomies, and ontologies. These classification schemes have potentially great utility when associated with various aspects of agent communication.

There are several additional purposes for developing classifications for concepts, including:

- assisting users in finding a particular concept from among many
- facilitating the administration of information systems
- through inheritance, conveying semantic content that is often only incompletely specified by other attributes, such as names and definitions
- deriving and formulating abstract and application concepts
- ensuring appropriate attribute and attribute-value inheritance
- deriving names from a controlled vocabulary
- disambiguating communicated information
- recognizing superordinate, coordinate, and subordinate concepts
- recognizing relationships among concepts
- assisting in the development of modularly designed names and definitions.

Each type of classification scheme mentioned above has particular strengths and weaknesses,

and provides the foundation upon which particular capabilities can be built. Keywords, for example, are a quick way to provide agents some assistance in locating potentially useful information. A thesaurus provides a more structured approach, arranging descriptive terms in a structure of broader, narrower, and related classification categories. A taxonomy provides a classification structure that adds the power of inheritance of meaning from generalized taxa to specialized taxa. Ontologies, with associated epistemologies, can provide rich, rigorously defined structures (e.g., directed acyclic graphs with multiple inheritance) that can convey information needed by software, such as intelligent agents and mediators, that are useful in the provision of intelligent information services.

Systems of the Bimonth

To experiment with the creation of ontologies, we suggest you try the Java Ontology Editor (JOE) <JOE> from the University of South Carolina. JOE is a graphical user interface that has the following two major parts: (1) an ontology editor and (2) a query formulation tool. The ontology editor provides a user interface where a user can create a new or edit an old ontology by adding new concepts (entities), attributes for that concept and relationship between two or more concepts. The query formulation tool is also a user interface that allows a user to build queries on the information space that is displayed by the ontology editor.

An alternative way to construct an ontology is to use the editor developed as part of the Ontolingua Project at Stanford <Ontolingua>. This site also has a number of ontologies contributed by other developers.

An example of a large ontology for a healthcare domain can be found at <OOHVR>. The Object-Oriented Healthcare Vocabulary Repository (OOHVR) Project at the New Jersey Institute of Technology has ~5000 concepts organized in a semantic network and stored in an object-oriented database. It is accessible on the web via any browser.

The largest and most comprehensive ontology is Cyc, developed at MCC and Cycorp <CYC>. Cyc has ~50,000 concepts and ~4 million constraints and relationships among the concepts. The upper level of the ontology is available at the Cycorp web-site.

Check these out!

References

<CYC> Cyc ontology <http://www.cyc.com/cyc-2-1/toc.html>
<JOE> Java Ontology Editor: <http://www.ece.sc.edu/Labs/HIIT/html/imts-new.html/>
<Ontolingua> The Ontolingua Project <http://www-ksl-svc.stanford.edu:5915/>
<OOHVR> Object-Oriented Healthcare Vocabulary Repository:
<http://object.njit.edu:2000/~newoohvr/JBI/INTERMED/InterTerms.html>