Ontology Properties

The following properties are necessary for something in order to be considered as an ontology (specifications possessing these properties will be referred to as “simple ontologies”):

- Finite controlled (extensible) vocabulary
- Unambiguous interpretation of classes and term relationships
- Strict hierarchical subclass relationships between classes

The following properties are considered as typical but not mandatory:

- Property specification on a per-class basis
- Individual inclusion in the ontology
- Value restriction specification on a per-class basis

The following properties may be desirable but are not mandatory nor typical:

- Specification of disjoint classes
- Specification of arbitrary logical relationships between terms
- Distinguished relationships, such as inverse and part-whole

Simple Ontologies and Their Uses

First, simple ontologies provide a *controlled vocabulary* for their domain. This by itself can provide great impact, since users, authors, and databases can all use terms from the same vocabulary. In addition programs can generate interfaces that encourage usage of the controlled terms an ontology contains. The result is that people share the same set of terms.

Second, a simple taxonomy may be used for *site organization and navigation support*. Many Web sites today expose the top levels of a generalization hierarchy of terms as a kind of browsing structure. The categories are typically hot, and a user may click on them to expand the subcategories.

Third, taxonomies may be used to support *expectation setting*. It is important as a user interface feature that users be able to have realistic expectations of a site. If they may explore even the top-level categories of the site’s hierarchy, they can quickly determine if the site might have content (and/or services) of interest to them.

Fourth, taxonomies may be used as *“umbrella” structures from which to extend content*.

Some freely available ontologies are attempting to provide the high-level taxonomic organization from which many efforts may inherit terms.

The Universal Standard Products and Services Classification (UNSPSC) is one such categorization scheme. It was aimed at providing the infrastructure for interoperability of terms in the domains of products and services. It provides a classification scheme (with associated numbers) for products and services.

For example, Category 50 (Food, beverage, and tobacco products) has a subclass family 5010 (Fruits and vegetables and nuts and seeds), which in turn contains a subclass 501015 (Vegetables), which in turn has a subclass commodity 50101538 (Fresh vegetables). The numbers provide a unique identification for each term and also encode the hierarchy.

A number of e-commerce applications today are looking for such umbrella organization structures, and in fact many have chosen to be compliant with the UNSPSC. Most applications will need to extend these ontologies with their specific hierarchies of categories, but if applications need to communicate among a number of content providers, it is convenient to use a shared umbrella or upper-level ontology.

Fifth, taxonomies may provide *browsing support*. Content on a site may be tagged with terms from the taxonomy. This may be done manually in the style of Yahoo or automatically (possibly using a clustering approach). Once a page (or service) is metatagged with a term chosen from a controlled vocabulary, then search engines may exploit the tagging and provide enhanced search capabilities.

Sixth, taxonomies may be used to *provide search support*. A query expansion method may be used to expand a user query with terms from more specific categories in the hierarchy.

The experience shows that under certain conditions (such as short document length and limited content areas), query expansion can radically improve search results.

Seventh, taxonomies may be used to *sense disambiguation support*. If the same term appears in multiple places in a taxonomy, an application may move to a more general level in the taxonomy in order to find the sense of the word.

For example, if an ontology contains the information that Jordan is an instance of BasketballPlayer and also an instance of s country, an application may choose to query a user searching for Jordan if he is interested in basketball players or countries.

Structured Ontologies and Their Uses

Up to this point, we have focused on simple taxonomies for usage in applications. As ontologies begin to have more structure, however, they can provide more power in applications. Once ontologies have more structure than simple generalization links, their property information enables them to be used in many forms.

First, these more structured ontologies can be used for simple kinds of *consistency checking*. If ontologies contain information about properties and value restrictions on the properties, then type checking can be done within applications.

Second, more structured ontologies may be used to provide *completion*.

An application may obtain a small amount of information from a user, such as the fact that he is looking for a high-resolution screen on a PC, and then have the ontology expand the range of the number of pixels that the user expects. This can be accomplished simply by defining what the term “High-Resolution PC” is with respect to a particular pixel range on two dimensions: “verticalResolution” and “horizontalResolution”.

Similarly, information can be reused. For example, a medical system may obtain information from an ontology that if a patient is stated to be a man, then the gender of the patient is “male”, and that information may be used to determine that a question concerning whether or not the patient is pregnant should not be asked, since there could be information in the system that things whose gender is male are disjoint from things that are pregnant.

Third, more structured ontologies may be able to provide *interoperability support*.

Controlled vocabularies enhance interoperability support, since different users and applications are using the same set of terms. In simple taxonomies, we can recognize when one application is using a term that is more general or more specific than another term and facilitate interoperability.

In more expressive ontologies, we may have a complete operational definition for how one term relates to another term, and thus we can use equality axioms or mappings to express one term precisely in terms of another and thereby support more “intelligent” interoperability.

For example, an ontology may include a definition that a “StanfordEmployee” is equal to a “Person” whose “employer” property is filled with the individual “StanfordUniversity”. This definition may be used to expand the term “StanfordEmployee” in an application that does not understand either “StanfordEmployee” or “Employee” but does understand the terms “Person”, “employer” and “StanfordUniversity”.

Fourth, more structured ontologies may be used to support *validation and verification testing* of data (and schemas).

If an ontology contains class descriptions, such as “StanfordEmployee”, these definitions may be used as queries to databases to discover what kind of coverage currently exists it data sets.

For example, if one was going to expose the class “StanfordEmployee” on an interface to some application, it would be useful to know first if the data set contained any instances of “Person” whose “employer” property was filled with the value “Stanford University”.

Additionally, if in a simple data model, we stated that a "Person" had at most one "employer", then we could use that information to check to see if any current information on "Person"s in the data set contained more than one "employer" value. Similarly, checks could be conducted to determine if there were currently "Person"s in the data set that were known to be "Employee"s yet did not have a value for the "employer" property (thereby showing that the data set was not complete).

Fifth, more structured ontologies containing markup information may *encode entire test suites*.

An ontology may contain a number of definitions of terms and some instance definitions, then include a term definition that is considered to be a query: find all terms that meet the following conditions. Markup information could be encoded with this query to include what the answers should be, thus providing enough information to encode regression testing data.

Sixth, more structured ontologies can provide the foundation for *configuration support*.

Class terms may be defined so that they contain descriptions of what kinds of parts may be in a system. Additionally interactions among properties can be defined so that filling in a value for one property can cause another value to be filled in for another slot.

Seventh, more structured ontologies can support structured, comparative, and customized search.

For example, if one is looking for televisions, a class description for television may be obtained from an ontology, its properties may be obtained (diagonal, price, manufacturer, etc.), and then a comparative presentation may be made of televisions by presenting the values of each of the properties for each television. Those properties can also be used to provide a form for users to fill in so that they may provide a detailed set of specifications about the items they are looking to find.

This also provides the foundation for providing a number of different search interfaces: a simple text box in which the user is expected to type a textual query, as well as search interfaces exposing important properties of products that can provide a structured search query. More sophisticated ontologies may be generated that mark which properties are most useful to present in comparative analyses so that users may have concise descriptions of products instead of comparisons offering complete details. Thus, ontologies with markup information may also be used to prune comparative searches.

Eighth, more structured ontologies may be used to exploit *generalization/ specialization information*.

If a search application finds that a user's query generates too many answers, it might dissect the query to see if any terms in it appear in an ontology, and if so, then the search application may suggest specializing that term.

For example, if one did a search for concerts in the San Francisco Bay area and obtained too many answers, a search engine might look up “concert” in an ontology and discover that there are subclasses of concert (and it may also discover that there are specific concert locations in the Bay area). The search engine could then choose to present the user with the option of looking for a particular kind of concert (say, rock concert), which would restrict the search, thereby returning fewer answers.

These are just some of the ways in which more structured ontologies may be used to refine search queries. We could also look at the ontology to provide alternative values (by looking at siblings in the ontology) for terms specified in the search query.

We have not claimed to present an exhaustive list of the ways in which ontologies may be used in applications. The above lists are illustrative of some ways that ontologies have been used to support intelligent applications.