Interface-guided Ontology Design

Gary H. Merrill
Emerging Technologies Group
Glaxo Wellcome Inc.
Five Moore Drive, PO Box 13398, Research Triangle Park
North Carolina 27709-3398 USA
ghm48805@glaxowellcome.com

Abstract

An application-oriented methodology, focusing on the role of the user interface, is proposed to address a set of classical problems faced in the design and development of ontologies. The set of problems and their consequences are discussed in detail, and the value of an interface-guided approach to ontology design is demonstrated on the basis of this analysis and experience using such a methodology.

Goals and Perspective

My goal here is to describe a particular approach to ontology design offering as advantages and benefits:

1. A framework that imposes a clear focus and realistic constraints on the content and scope of the ontology.
2. A resulting ontology that is immediately useful relative to well defined goals and expectations.
3. Minimization of unused parts of the ontology, reducing both effort and cost in ontology development.

The methodology I recommend here is what I have come to think of as *interface-guided ontology design*, and it arises from a practical engineering perspective that is fundamentally product-oriented. It is this product-orientation together with the central role played by the product's interface that provides the foundation and focus for the efficient and effective design of an ontology.

I begin with a consideration of various problems and issues we encounter in the design of ontologies.

The Purpose of Ontology Design

What is the purpose of ontology design? Even the briefest survey of the literature is sufficient to suggest the following range of answers:

- The representation of knowledge
- The organization of knowledge
- The organization of knowledge in a particular domain
- The representation and organization of knowledge for the purpose of solving a particular problem or for the purpose of implementing a useful application
- The enhancement of searching methods
- The facilitation of data-mining.

Each of these answers is quite broad and general – as is the question being addressed – and they arise from a perspective that I refer to as *the search for grand ontologies*. A *grand ontology* is broad, deep, and hoped to be all-encompassing. Even if the ontology architect has decided to restrict his efforts to a particular domain (such as mathematics, particle physics, medicine, or molecular biology) the ontology being sought is intended to characterize that domain fully and to be a formal representation of our accumulated knowledge in the domain. A well-known example of such an attempt is the Cyc project (http://www.cyc.com) where the overall goal has been to create an ontology capable of capturing and supporting "common sense reasoning". A more recent project has been started by a working group in the ontology of molecular biology, and indeed it would appear that a number of participants in the High Performance Knowledge Base project (http://www.teknowledge.com/HPKB) are involved in the design of grand ontologies at least to some degree.

Pursuit of a grand ontology is an ambitious and noble goal, and one that requires immense time, effort, and resources. But the benefits of a successful grand ontology are also substantial. Only such an ontology offers the generality, power, and scope to fully capture a non-trivial domain. But there are problems with such a goal as well.

First, at any given point in design and development the grand ontology will be *incomplete* to some degree – in fact, to a significant degree. This immediately raises the question of how we can or should decide what portions of the ontology to favor at its various stages and how even to identify these portions. A deeper reflection raises more fundamental questions concerning what *completeness* even means in this context, the degree to which it can and should be pursued, and how we can determine the degree to which it has been achieved.

Second, even though we are thinking in terms of constructing the ontology for a particular domain, a closer examination reveals that in fact the scope is *unbounded* (or at least *indeterminate*). Suppose, for example, we begin by desiring to construct an ontology in the general area of commercial businesses and their organizational structures. How soon are we likely to realize that this ontology must creep into the areas of law, government, insurance, transportation, and a host of other domains? For more complex areas in the natural sciences the situation is even worse. If we begin by thinking that our goal is the development of an ontology for human genetics, a complex web of relationships will soon compel us to expand into areas such as molecular biology, biochemistry, biophysics, clinical medicine, and perhaps even quantum mechanics.

In the arena of grand ontologies, the set of problems surrounding the issues of incompleteness and unboundedness give rise to a paralyzing condition that can inhibit or even prohibit success. This is the realization (or at least the fear) that we cannot do anything until we can do everything. I think this problem has been felt most acutely in the area of natural language processing and the "deep" understanding of text where success with even the simplest real-world examples has seemed to require broad and general solutions to an array of difficult problems – where handling an example such as "He is not tall" requires addressing at least the general problems of negation, anaphora, and the underlying ontological issues to be used in support of these. But likewise in, for instance, the area of diseases and their treatments, how can we proceed in a fruitful way without having an ontology of proteins and genes, and thence of genetic abnormalities, alterations, and mutations, and thence of molecular biology more generally, and thence of biochemistry, and so on? Where do we stop? How do we stop? How can we tell where and how to stop? How much will be enough?

Such questions are important because whether we seek support from government agencies, from private foundations, or from management within commercial organizations this support will not be forthcoming if those questions cannot be answered – if we cannot impose a bound on our investigation and development, and justify that bound on a non-arbitrary foundation. I have focused on grand ontologies in order to expose these problems and their importance, but the problems are not restricted to grand ontologies alone.

Ontology Design: A Common Approach

Frequently the decision to create, design, or develop an ontology is driven by a desire to provide the ontology as part of an "enabling technology" for potential use in a variety of applications. This was fundamentally the original approach we took at Glaxo Wellcome in embarking on the initial development of an ontology in the general area of genetics and molecular biology.[1] Envisioned applications can include (for example) enhanced information retrieval, database integration, and data mining.

This approach engenders a kind of "top down" approach to ontology design where the development of the ontology is considered to be primary in both a logical and a temporal sense. The attitude of the ontology designers is that *first* we will design the ontology in sufficient generality and completeness, and *then* this ontology will serve as the foundation of a number of applications. At the stage where the ontology is being designed and tested, the applications are only vaguely described, typically in terms of their intended goals and capabilities, and relatively little thought is put into their specific design or precisely how they will be related to the ontology.

In our case, the primary application we had in mind can best be described as one in which a number of heterogeneous databases would be integrated semantically into a Cyc knowledge base and a graphical user interface would be written to provide improved access to this integrated knowledge. At an early stage we spent substantial time discussing the likely form of the interface (about which more later), but any real work and thought on both this and the specific functionality of the application were intentionally postponed until we had the ontology in place.

Although we did not think of ourselves as embarking on the construction of a grand ontology – for example, an ontology that could serve as a broad basis for a wide range of applications in molecular biology – the approach we took was similar to that taken with grand ontologies: the ontology comes first, it must be developed in sufficient generality to meet a variety of anticipated and even unanticipated needs, and any specific applications for the ontology will follow this development and will be in a significant sense independent of it. While this is not quite an attitude of "ontology for ontology's sake", there is something of this flavor in the approach.

---

[1] In this area work was done as a cooperative effort between the author and Dhiraj Pathak of the Bioninformatics department who did most of the original knowledge representation and who currently is expanding the content of the knowledge base. Design and implementation of the user interface for our project as well as subsequent alteration to parts of the original representation were performed by the author.

Problems and Dangers

In following our "top down" or *ontology first* approach, things went smoothly for a while as we put in place the basic concepts, assertions, and rules of our knowledge base. But then problems began to appear.

The Problem of Unbounded Revision

The most frequent and irritating problem was *the problem of unbounded revision.* Since we were oriented towards designing an ontology with no very specific application in mind, we found ourselves constantly revising our representation in order to extend its generality to handle situations we had previously not considered. This situation was terribly disruptive to the development effort, and a great deal of time was lost in analyzing what our needs might be and in revising what we had in order to meet these newly anticipated needs. Of course this is a problem that is to be expected in any ontology design project, but it is critical to minimize its effect on the project.

The Problem of Completeness

The second problem (and one contributing to the problem of unbounded revision) pertained to the issue of completeness. How much should we represent? How "deep" should the representation be? Without some quite specific guide in terms of how the ontology is to be *used,* such questions are impossible to answer, and informal or arbitrary methods must be employed. I know of one case, for example, where a designer was told that in the domain he was representing he would be permitted to introduce and represent only a certain fixed number of constants. And this decision was based on the amount of time and funding available to complete the project, together with some experience in estimating how much time was typically required for representations of the sort in question. Of course this appears arbitrary, but decisions of this kind *must* be made in genuine projects (as opposed to open ended research) since limits *must* be imposed. It is not the imposition of limits that is the danger, but rather it is the absence of any principled way in which to decide what those limits should *be.* In the end, if you lack a well defined target, then you can never know if you have hit it and you can take no rational steps towards achieving higher accuracy.

It might be thought that appeal to domain experts can solve the problem of unbounded revision – that such an expert can circumscribe for you the limits of your ontological task. Unfortunately, the problem with a domain expert is that this person is an expert in the domain in question – and consequently tends to think that *everything* in it is of critical importance. So while domain experts are of fundamental importance in your task, they cannot usually be depended upon to define or limit it.

The Problem of Testing and Adequacy

A third problem centered around testing methodology – or rather the lack of it. How do you test the adequacy of an ontology? And the immediate follow-on question is "Adequacy relative to *what*?" If the approach you take to designing and developing the ontology is to identify a certain domain (e.g. anatomy) that you wish to represent, and you have some sort of specification or measure of how much of that domain or what portions of that domain are to be represented, then you can achieve some demonstration of adequacy by showing to what degree your ontology corresponds to the pre-systematic specification.[2] But in the end this is quite a sterile notion of adequacy since it implies nothing about the *usefulness* of the resulting ontology. And it leaves open the question of justifying the measure you have settled upon. For example, does anyone really feel that listing the number of concepts or relations in an ontology is a measure of the adequacy of that ontology? Is the number of terms in your lexicon a valuable measure of the usefulness of your natural language processing system? Such measures are useful (and often seen) in progress reports submitted to funding agencies, but they do not begin to address the real issues of adequacy or success.

In our case, we attempted to test the ontology (as it appeared within our knowledge base) by developing a set of queries and expected responses to those queries. Failure of the response to be what we expected was taken as an indication of a flaw in the representation – and this was usually the case, though sometimes it turned out to be a flaw in our formulation of the query itself. This technique, though quite informally guided, did provide aid in discovering certain fundamental problems with the ontology and pointing the way to their correction; but it did not provide much confidence in the true adequacy and usefulness of the ontology since it was totally divorced from any particular use or application.

The Problem of Usability

The final problem we encountered with our *ontology first* approach was in fact the most disappointing one. When we ultimately reached the stage where we had at least some confidence in the degree of completeness (relatively to our still vaguely specified goals), had stopped (often somewhat arbitrarily) the process of revision, and were reasonably well satisfied with our initial testing, it was then time to make use of the ontology in a genuine application. But how?

[2] See, for example, (Gladwell 1999) and the *Current Status* section of (Lehmann and Foxvog 1998).

I still recall the point at which I decided the ontology was sufficiently complete and tested to freeze its development and begin serious work on the application that would make use of it. At this point the representation seemed to be quite coherent and we had been able to use it to absorb a significant amount of data (over 50,000 new constants and 150,000 assertions) from several databases into our knowledge base, to pose some very interesting queries, and to see the results of these queries (often involving inferencing within the knowledge base).

Putting aside the ontology itself I then turned to a design of the application, and in particular to the user interface. Very early in our project we had discussed at some length how the ultimate application might appear to a user. In particular, we were concerned about how the user would specify a query to the system. Overall, the goal was for the user to be able to retrieve useful and pleasingly presented information pertaining to genes, proteins, and various properties and relations that these bear to one another (disease relationships, protein family information, etc.). We had discussed a wide variety of possible approaches including a natural language query interface, using CycL as the query language, some kind of purely graphical interface, and so on. Each of these had its advantages and its disadvantages.

In order to get a prototype system working in a relatively short amount of time, to present an application which would appear at least vaguely familiar to our expected user group, and to get useful feedback from that group I decided that a relatively simple web interface in HTML was in order. Working now purely from the perspective of what a user would want and expect, I designed such an interface centered around a table of pull-down lists from which the user could select appropriate values for a number of relevant categories (Species, Gene Function, Disease Association, etc.).[3] And having thus arrived at an acceptable design and implementation of the application's interface, the time had come to connect it with the knowledge base (i.e., the ontology). But I could not.

Strictly speaking, it is not correct to say that I *could* not, but rather that doing so would obviously require a great deal of unanticipated effort and addition or modification to either the ontology, the interface, or both. Having done the ontology first (from the point of view simply of representing a certain amount of knowledge in a particular domain), and then having done the interface from the user's point of view, there turned out to be no very direct correspondence between these! And after all this time and effort the ontology had turned out not to be directly or easily useable in an application.

[3] The interface (through two additional browser windows) provides a bit greater functionality than the simple tabular query interface, but the details are not relevant in this context.

The problem here, of course, is the fairly obvious one that if an ontology is not designed with a particular use in mind, then it will be difficult to use it in the desired way. An alternative choice at that point is to allow the ontology to "force" the features and capabilities of the interface or application, but this is a very poor methodology to adopt since it will result in applications that will be rejected by users.

A Summary of the Problems

We should pause now for a brief summary of the problems we have seen illustrated thus far, and these are:

- The problem of unbounded revision
- The problem of completeness
- The problem of testing and adequacy
- The problem of usability

And to each of these corresponds a question we have encountered difficulty in answering:

- How do we bound the domain for our ontology?
- How do we characterize completeness and determine the degree to which it has been achieved?
- What does it mean to say that the ontology is adequate, and how do we test for adequacy?
- How do we ensure that the ontology will be useful and reasonably usable?

We should also note that our failure to deal effectively with these points has a number of unfortunate consequences when it comes to developing an ontology, using it as a fundamental component in a project, and justifying the costs and resources required by ontology development. A number of factors may be negatively affected by this failure:

- The time to complete the product or project
- Estimates of required resources and the actual resources ultimately required
- The cost of developing the ontology
- The usefulness of the resulting ontology, and hence the value of work done in creating that ontology

Interface-guided Design: A Software Engineering Perspective

The answer I propose in addressing these problems is to take what is fundamentally a product-oriented software engineering approach to ontology design. We must begin by recognizing that an ontology should never be thought of as (to borrow from Kant) a thing in itself but always as a *part* of some application or set of applications. It is dangerous to think that if you first build the ontology in isolation from some quite specific use of it, someone (usually someone *else*) will be able to employ it to advantage. This is perhaps the fundamental fallacy of the grand ontology. Ontology in the pursuit of application is no vice; ontology divorced from application has no virtue.

More specifically, my recommendation is, where possible, to employ the user interface to guide design of the ontology. Unless we regard the design of an ontology as an exercise in pure art (which even the most esoteric philosophers have been disinclined to do), then the purpose of constructing an ontology is to help in the solution of some problem or in the construction of some application. And the point of creating such an application is to provide it to a community of users. And for most users and most applications, the interface *is* the application.

Focusing on the end product (and in particular on the user interface) has the following consequences:

➤ *The domain will be bounded by the extent and content of the interface.*

In our own case, for example, we had wrestled at some length about how to represent (and how much to represent) the wide variety of transformations that can take place in proteins and genes: deletions, duplications, inversions, transpositions, etc. And this in turn led to a consideration of how much of the underlying structure needed to be represented: introns, exons, sequences, secondary structure, etc. The possibility of creating an adequate and sufficient ontology for such gene and protein events, and then formulating the appropriate rules, seemed overwhelming. But this is how things will seem if your goal is a non-specific one pertaining to representing a certain area of molecular biology. Once the interface was designed I realized that such detail was not necessary and that what really was needed was a much simpler (sub)ontology of genomic abnormality. Similarly, what initially had appeared as a requirement for a hopelessly complex ontology of gene expression turned out to be satisfied by a very simple and much more "surface" representation of that subdomain.

➤ *The ontology will be immediately useful.*

Having been guided by the interface in designing the ontology, hooking up the interface with the ontology (through knowledge base queries and information retrieval) turns out to be both natural and straightforward. You are no longer faced with the situation of a solution in search of a problem, but rather your ontological solution has arisen from a precise statement of the problem and a partial (interface) solution to it.

➤ *Unused parts of the ontology will be minimized.*

One of the significant dangers in taking an *ontology first* approach is that it is difficult to know where to stop, and as I mentioned previously such decisions are generally made on grounds that are largely arbitrary rather than being principled or problem-oriented. As focus on the extent of the interface bounds the domain, it thereby ensures that little beyond what is necessary will be represented. And this has a significant effect on the resources needed for ontology development and the time to completion.

➤ *There will be a clear methodology for testing the adequacy of the ontology in relation to the interface and application as a whole.*

The interface (and its interaction with the ontology) serves as the arbiter of adequacy and completeness that we previously saw was missing in the *ontology first* approach. "Completeness" and "adequacy" now acquire a specific meaning as "completeness and adequacy relative to the documented functionality of the interface". And this degree of specificity can be of major value in project proposals and funding requests.

Criticisms, Oversights, and Conclusions

The advantages to interface-guided ontology design are clear. The approach solves a number of problems otherwise plaguing us in the creation of ontologies and the delivery of products based on those ontologies. But do we lose anything in adopting this methodology?

One of the first questions that must come to mind is "Whither grand ontologies now?" If in successfully accomplishing the design of useful, efficient, and cost-effective ontologies we are to narrow our focus to the task at hand – even to the point of using the application interface as our primary guide – how can we expect ever to achieve the kind of broad (indeed *grand*) ontologies that truly are required for knowledge representation in complex domains?

Reflecting on this question, we can go on to ask whether following the path of interface-guided design is not merely a formula for producing "quick and dirty" little ontologies that accomplish limited tasks and are impossible to integrate with one another or into a more comprehensive conceptual scheme.

In answer to these two questions I must point out that although the substantial benefits of interface-guided design cannot be ignored, I have not offered it as an exclusive approach to the design of ontologies. And it is not in fact incompatible with other broader, less narrowly focused, approaches. But it can serve quite effectively to anchor the results of such approaches to a pragmatic foundation. In constructing formal systems we ought always be concerned with the questions of adequacy: what does this mean and how can we demonstrate it?

In the area of formal logic, for example (see Merrill 1978), we can attempt to proceed by developing our proof theory first and our semantics second, but we will run into the same sort of problems I have discussed above. The proper methodology for formal system development is to consider both proof theoretic and semantic issues in developing the system as a whole. It is the same in the arena of ontology design more broadly construed, and this is what interface-guided design helps us to do.

Moreover, practicing limited or bounded ontology development through the interface-guided methodology does not preclude *good* ontology design or the introduction of other criteria for evaluating the adequacy of the ontology – and this may include criteria pertaining to the integration of the current ontology with others, including some of grander scope. Following this methodology should allow us to take advantage of certain ontological phenomena of locality in the sense that we are focused on a limited, if not minimal, cluster of concepts required by our specific goal, and relatively little additional effort should be required to ensure that the resulting ontology is modular relative to its relations to other (perhaps broader) ontologies.

References

Gladwell, Martin. 1999. Moving Ontological Engineering Into The Marketplace, World Wide Web pages, accessed March, 1999: http://wwwis.cs.utwente.nl:8080/kbs/EcaiWorkshop/gladwell.html

Lehmann, F., and Foxvog, D., 1998. Putting Flesh On The Bones: Issues That Arise In Creating Anatomical Knowledge Bases With Rich Relational Structures. In *Papers from the 1998 Workshop on Knowledge Sharing Across Biological and Medical Knowledge Based Systems*, 41-50. Menlo Park, Calif.: AAAI Press.

Merrill, G. H. 1978. Formalization, Possible Worlds and the Foundations of Modal Logic. *Erkenntnis* 12: 305-327.