IDEF5 Ontology Description Capture Method

Concept Paper

Christopher P. Menzel
Richard J. Mayer

Knowledge Based Systems Laboratory
Texas A&M University

1990

Cooperative Agreement NCC 9-16

Research Activity No. IM.06:
Methodologies for Integrated
Information Management Systems

NASA Johnson Space Center
Information Systems Directorate
Information Technology Division

ricis

Research Institute for Computing and Information Systems
University of Houston-Clear Lake

TECHNICAL REPORT

The RICIS Concept

The University of Houston-Clear Lake established the Research Institute for Computing and Information Systems (RICIS) in 1986 to encourage the NASA Johnson Space Center (JSC) and local industry to actively support research in the computing and information sciences. As part of this endeavor, UHCL proposed a partnership with JSC to jointly define and manage an integrated program of research in advanced data processing technology needed for JSC's main missions, including administrative, engineering and science responsibilities. JSC agreed and entered into a continuing cooperative agreement with UHCL beginning in May 1986, to jointly plan and execute such research through RICIS. Additionally, under Cooperative Agreement NCC 9-16, computing and educational facilities are shared by the two institutions to conduct the research.

The UHCL/RICIS mission is to conduct, coordinate, and disseminate research and professional level education in computing and information systems to serve the needs of the government, industry, community and academia. RICIS combines resources of UHCL and its gateway affiliates to research and develop materials, prototypes and publications on topics of mutual interest to its sponsors and researchers. Within UHCL, the mission is being implemented through interdisciplinary involvement of faculty and students from each of the four schools: Business and Public Administration, Education, Human Sciences and Humanities, and Natural and Applied Sciences. RICIS also collaborates with industry in a companion program. This program is focused on serving the research and advanced development needs of industry.

Moreover, UHCL established relationships with other universities and research organizations, having common research interests, to provide additional sources of expertise to conduct needed research. For example, UHCL has entered into a special partnership with Texas A&M University to help oversee RICIS research and education programs, while other research organizations are involved via the "gateway" concept.

A major role of RICIS then is to find the best match of sponsors, researchers and research objectives to advance knowledge in the computing and information sciences. RICIS, working jointly with its sponsors, advises on research needs, recommends principals for conducting the research, provides technical and administrative support to coordinate the research and integrates technical results into the goals of UHCL, NASA/JSC and industry.

IDEF5 Ontology Description Capture Method

Concept Paper

RICIS Preface

This research was conducted under auspices of the Research Institute for Computing and Information Systems by Dr. Christopher P. Menzel and Dr. Richard J. Mayer of Texas A&M University. Dr. Peter C. Bishop served as RICIS research coordinator.

Funding has been provided by the Air Force Armstrong Laboratory, Logistics Research Division, Wright-Patterson Air Force Base via the Information Systems Directorate, NASA/JSC through Cooperative Agreement NCC 9-16 between the NASA Johnson Space Center and the University of Houston-Clear Lake. The NASA technical monitor for this research activity was Robert T. Savely of the Information Technology Division, NASA/JSC.

IDEF5 Ontology Description Capture Method

Concept Paper

Christopher P. Menzel
Richard J. Mayer

Knowledge Based Systems Laboratory
Department of Industrial Engineering
Texas A&M University
College Station TX 77843

Reviewed by
Michael K. Painter, Capt, USAF
Armstrong Laboratory
Logistics Research Division
Wright-Patterson Air Force Base, Ohio 45433-6503

This report was prepared under subcontract between the University of Houston - Clear Lake and Texas Engineering Experiment Station, Texas A&M, on RICIS Research Activity No. IM.16. (NASA Cooperative Agreement NCC 9-16).

Preface

This report describes the research accomplished at the Knowledge Based Systems Laboratory of the Department of Industrial Engineering at Texas A&M University. Funding for the Laboratory's research in Integrated Information System Development Methods and Tools has been provided by the Air Force Armstrong Laboratory, Logistics Research Division, AFWAL/LRL, Wright-Patterson Air Force Base, Ohio 45433, under the technical direction of USAF Captain Michael K. Painter, under subcontract through the NASA RICIS Program at the University of Houston. The authors and the design team wish to acknowledge the technical insights and ideas provided by Captain Painter in the performance of this research as well as his assistance in the preparation of this report. Special thanks goes to the IDEF5 research team whose names are listed below:

Dr. Christopher P. Menzel
Dr. Richard J. Mayer
Dr. Paula S.D. Mayer
Mike Futrell

Summary

This report presents the results of research towards an ontology capture method refered to as IDEF5. Viewed simply as the study of what there is in a domain, ontology is an activity that can be understood to be at work across the full range of human inquiry prompted by the persistent effort to understand the world in which it has found itself---and which it has helped to shape. In the context of information management, ontology is the task of extracting the structure of a given engineering, manufacturing, business, or logistical domain and storing it in an usable representational medium. A key to effective integration is a system ontology that can be accessed and modified across domains and which captures common features of the overall system relevant to the goals of the disparate domains. If the focus is on information integration, then the strongest motivation for ontology comes from the need to support data sharing and function interoperability. In the correct architecture, an enterprise ontology base would allow the construction of an integrated environment in which legacy systems appear to be open architecture integrated resources. If the focus is on system / software development, then support for the rapid acquisition of reliable systems is perhaps the strongest motivation for ontology. Finally, ontological analysis has been demonstrated to be an effective first step in the construction of robust knowledge based systems.

An IDEF5 description of an ontology is a computationally tractable representation of what exists in a given domain. IDEF5 provides the means to identify the primary classes, or kinds, of objects there are within the domain by isolating the properties that define the members of those kinds, and the characteristic relations that hold between domain objects (see below). IDEF5 allows such representations to be purposely structured in a way that closely reflects human conceptualization of the domains in question. In IDEF5, differing perspectives on the same domain (e.g., varying levels of granularity) and their interrelations are also supported. Finally, IDEF5 supports the identification of complex kinds (system kinds) and the properties and relations that characterize members of those kinds.

1.0 Background, Motivation and Informal Foundations

Any organized system---a business, a university, a manufacturing plant--can be thought of as the resultant of three vectors:

(i) the system's ontology, i.e., the basic entities that populate the system--personnel, equipment, manufacturing systems, etc.;
(ii) the structure those entities jointly exhibit--the relations they bear to one another; and
(iii) the processes they undergo--the changes that take place in the organization over time. An accurate representation of such a system will thus reflect the information within all three vectors.

Currently, existing IDEF methodologies are geared chiefly toward information of the second and third types: IDEF1 and IDEF1X capture primarily structural information, IDEF0 and IDEF3 various types of process information. Of course, since both structural information and process information involve objects in a system, there is the capacity for limited ontology representation within the existing methodologies. But, as noted below, there are several important kinds of ontological information that are not representable in those methodologies. Furthermore, those methodologies do not include techniques specifically designed for eliciting and capturing system ontologies. This suggests that there is a need for a separate methodology. We intend to substantiate this suggestion in this report and begin laying the groundwork for the needed methodology, IDEF5.

Like other IDEF methodologies being worked on at the Knowledge Based Systems Laboratory, IDEF5 will have two components: (1) a rigorous, formal foundation for the methodology and (2) an accompanying documented software implementation designed for practical information capture and information modeling. The software tool is designed for use by domain experts--people attuned to the way a specific system works. The basic question faced by any domain expert, or by a knowledge engineer working with such an expert, is how to describe the things he or she knows about. A good methodology will reveal the appropriate sorts of general structures that classify the knowledge being sought smoothly and flexibly--the formal foundation--and then provide a rich, powerful, user-friendly environment for eliciting that information from the expert, and which then stores and integrates the garnered information efficiently and effectively.

In the following sections we describe the nature of ontology and ontological information, sketch the proposed IDEF5 formal and methodological foundations for capturing that information, and discuss the general proposed features of an IDEF5 software environment.

1.1 Philosophical Foundations: The Nature of Ontology

In Western thought, ontology has chiefly been thought of as an attempt to divide the world at its joints. In a word, it can be thought of as the study of what there is. Historically, ontology arose as the major component of the branch of philosophy known as metaphysics, which deals with the nature of reality generally. Metaphysics is perhaps most often associated with questions typically taken to be beyond the reach of physical science, such as the nature of the soul or the mind, the existence of God, or whether or not we have free will.1 However, there is no necessary connection between ontology and pure, nonempirical philosophical speculation. Viewed simply as the study of what there is, ontology is an activity that can be understood to be at work across the full range of human inquiry prompted by humanity's persistent effort to understand the world in which it has found itself---and which it has helped to shape.

Natural science, in particular, can be viewed as an example of ontology par excellence. Perhaps the chief goal of subatomic physics, for example, is to develop a taxonomy of the most basic kinds of objects that exist within the natural world--electrons, protons, muons and their fellows. At the other end of the spectrum, astrophysics, among other things, seeks to discover the range of objects that exist in its domain: quasars, black holes, gravity waves, etc. Similarly, the so-called life sciences seek to categorize and describe the various kinds of living organisms that populate the planet. Such examples can be multiplied, of course, from geology to psychology, chemistry to sociolinguistics.

This sort of inquiry is not limited to the natural sciences, however. The abstract sciences as well---mathematics, in particular---can be thought in part at least as an attempt to discover and categorize the domain of abstract objects: prime numbers, transfinite ordinals, Hilbert spaces, continuous nondifferentiable functions, polynomial algorithms, commutative groups, and so on.

[1] Unfortunately, in contrast to these deep, important--albeit often ultimately unresolvable--questions, in the popular consciousness the term 'metaphysics' has come to be associated with such pseudo-intellectual bilge as astrology, astral projection, occult "science," and similar nonsense.

The natural and abstract worlds, however, do not exhaust the applicable domains of ontology. For there are vast, human designed and engineered systems---manufacturing plants, businesses, military bases, etc.--in which the task is just as relevant, and just as pressing. Here, though, the ontological enterprise is motivated not so much by the search for knowledge for its own sake, as---ideally---in the natural and abstract sciences, but by the need to understand, design, engineer, and manage such systems effectively.

Ontology, then, is a basic research task common to the natural and abstract sciences on the one hand, and the information sciences, on the other. In the next section we lay out the nature of ontological information in greater detail, and discuss its application to the information sciences.

1.2 Kinds and Instances

Ontology can be understood to involve several subtasks; four are especially worth discussing here: (i) providing an inventory of the kinds of objects that exist within a given domain according to our best sources of information regarding that domain (e.g., a theory or a domain expert), (ii) for each kind of object, providing a description of the properties that are common to all and only instances of that kind, (iii) characterizing the particular objects that in fact instantiate the kinds within a system, and (iv) providing an inventory of the associations that exist within a given domain between (and within) kinds of objects.

The first two tasks are common in the physical sciences. Thus, for example, in microphysics, one finds the subatomic world grouped into basic kinds---at the grossest level (in the context of subatomic physics!), leptons and quarks, and beneath them the large variety of subkinds of each of those overarching kinds. And along with each kind, one finds the properties common to all and only members of the kind, including the specific property values of such attributes as mass, charge, spin, and so on the members all share. Again, in biology, one finds perhaps the foremost example of classifications into kinds and subkinds and characterizations of the distinctive properties associated with each kind.

The third task of ontology becomes more relevant in contexts where we want to be able to characterize specific individual objects, to speak specifically of them and their properties. A basic metaphysical distinction is especially useful in this regard, viz., the distinction between essential and accidental properties. An essential property of an object S is a property that S could not possibly have lacked. An accidental property of S, by contrast, is a property that S in fact has, but nonetheless might not have. For example, the number 17 has the property of being prime essentially; it could not possibly have been evenly divisible by anything other than 1 and itself. On the other hand, it has the property of being my favorite natural number accidentally; if I hadn't existed, or if my affections had been directed toward the number 43 instead, it would have lacked it (and no doubt would have been none the worse for it). Again, human beings are usually thought to have the property of being human essentially -- no one could have been, say, a donkey or a stone instead of a human. On the other hand, all of us could have been (and indeed, have been) a different height, for example, and so one's height is an accidental property.

Now, the usual notion of a kind is that of a class of objects all of which share a common nature, i.e., a set of properties that belong essentially to all and only members of the kind. On this conception, then, the properties in virtue of which a thing is a member of a kind are also those which define its nature as an entity. This definition is for the most part quite appropriate in the context of natural science and mathematics. For example, the most natural properties for delimiting biological kinds involve having a certain DNA structure, then clearly, this will also be an essential property of the animals in question (on the reasonable assumption that no particular animal could have been a member of a different species). Similarly, the most natural properties for delimiting kinds of subatomic particles--e.g., a certain mass, charge, spin, etc.--will be in terms of analogous underlying structural properties that are essential to the instances of those kinds.

As we will argue, though, this definition is rather too restrictive for use in the context of human designed systems. However, there is a closely related conception--alluded to briefly in the first paragraph in this section--that is somewhat more flexible and more applicable in the context of information modeling. On this conception, the properties that define a kind are not necessarily essential properties of the members of the kind. Rather, the membership conditions only specify what properties it takes to be an instance of that kind, irrespective of whether or not those properties are essential to the members. Thus, on this broader conception, a kind K is a class of objects consisting of all and only those things that exhibit a certain set of properties, which we can call the defining properties of K.

An example will help to show how this conception of a kind is the more useful one in the context of human designed systems, and will also help to clarify one way in which an ontology might function in the course of information management. Consider the following representation of the basic ontology of a manufacturing cell composed of five entities; objects enter the cell and encounter a cutter, then a drill, an inspection station, and two cleaners:

KINDS	DEFINING PROPERTIES
A: Cutter	{Has diamond tool,...}
B: Drill	{Has high speed motor,...}
C: Inspector	{Has high intensity lens light...}
D: Cleaner and Painter	{Has dust filters, high gloss paint,...}
E: Cleaner	{Has liquid cleaners,...}

This shows the kinds of objects that populate the system, and lists the defining properties of each kind; a representative defining property or two is listed for each kind. It thus provides an abstract representation of the general structure that the manufacturing cell must exhibit at any given time. Now, the property having a diamond cutting tool is a defining property of the kind Cutter. However, suppose the cutter that is in fact instantiating this kind has the capacity of switching from diamond cutting tools to carbide. Then even though having a diamond cutting tool is a defining property of the kind Cutter, it is nonetheless an accidental property of the cutter; it would lack the property if someone were to swap out the diamond tool for a carbide tool.2 The fact that it is a defining property of the kind thus means only that at any given time, it must be the case that whatever is playing the role of the cutter in the manufacturing cell

[2] There are of course some significant philosophical issues involved in the the nature of artifacts; some philosophers, for example, argue for the view--known as *mereological essentialism*--that every part of an artifact, or physical object generally, is essential to it, so that if we swap out one cutting tool for another in a cutter, the cutter with the replaced tool ceases to be, and a new cutter comes to exist. The puzzle here goes back to Greek times in the guise of the Ship of Theseus: if we bit by bit replace the planks of a ship with new planks, and simultaneously bit by bit build a new ship from the old planks, then which ship is which? Is the new ship identical with the original ship because it has the same parts? Or is the rebuilt ship identical with the original ship because of the insignificance of each plank individually to the identity of the whole? Thankfully, we needn't address, or at least we needn't answer, such questions. The chief purpose of ontology modeling, and information modeling generally, is not so much to divide the world at its ontological joints, to discern its ultimate nature, but rather simply to categorize it in the most useful way for the purposes at hand. And the fact is that, in our ordinary ways of thinking about such matters, ordinary objects do not cease to exist if we change relatively insignificant parts. As a matter of fact, however, our theory will remain neutral on this question, and will indeed permit, though not require, mereological essentialism should it prove useful in some contexts, as it conceivably might.

has a diamond cutting tool, irrespective of whether or not the cutter that is in fact playing the role has a diamond tool essentially or accidentally.

The general point here is that things can belong contingently to important kinds of objects within human designed systems. The reason for this is that the kinds within such a system are usually artifacts, human constructions, and hence it can often turn out that an object of one kind might "mutate" into an object of another kind simply in virtue of undergoing some nondestructive change, e.g., the exchange of cutting tools. Compare this with, e.g., a case in which an electron decays into two pions. This is a case of destructive change more typical in natural systems; the original object does not survive, but is rather replaced by two distinct objects of a different kind.

Put another way, the reason we use the broader notion of a kind is that when we build an ontology for a certain human designed system we are not necessarily setting out to discover and classify the world as it is in itself, but rather to divide up and categorize the objects within the system in useful and informative ways. An ontology's categorization scheme is justified only insofar as it is useful to organizing, managing, and representing information in the system so categorized. If objects of a certain kind K play a useful role in the system, that is all the justification one needs for admitting them into the system's ontology, irrespective of whether or not the defining properties of K are essential to its members.

The third subtask of ontology is operative in the above example as well. For in addition to listing and characterizing the kinds that define the manufacturing cell, we have also discussed the natures of some of their possible instances, e.g., whether or not they have a certain property essentially, whether or not that property is a part of their nature. This is no mere philosophical exercise. It might well be crucial to be able to distinguish the essential from accidental properties. For the essential properties of a thing S put inviolable bounds on what is possible within a system containing S. For if S has a property P essentially, it cannot fail to have it. Hence, for example, a design that specifies a kind that includes a property that precludes P among its defining properties cannot use S as one of its instances, regardless of how well it might meet the remaining specifications.

There is more to characterizing the objects in a system than listing their properties, though. For in the context of a given system it is equally important to detail the relations that objects in the system can, and do, bear to one another. Considerations such as those above lead us to distinguish system-essential from system-accidental relations. A system-essential relation relative to two (or more) kinds K1, K2 is a relation that must hold whenever there are instances of K1 and K2. A system-accidental relation relative to K1 and K2, by contrast, is one that needn't hold between all possible instances of those kinds. For example, the nature of the manufacturing cell depicted above might require a certain sort of informational link to be established between the cutter and the drill that informs the drill of the type of operation the cutter has performed on a given piece of material. In ontological terms, this would then be characterized as a system-essential relation relative to the kinds Cutter and Drill. On the other hand, the spatial relationship between cutter and drill may well be irrelevant; e.g., the drill might just as well be three feet north of its actual location in the cell. In this case, we say that the de facto spatial relationship between cutter and drill is system-accidental. (Though of course certain facts about the configuration of the drill or cutter could require that the two be oriented in one and only one way. In this case the relation would be system-essential. Note that, just as defining properties of kinds needn't be essential to their instances, in the same way entities that stand in system-essential relations don't necessarily stand in those relations essentially; though being spatially oriented in a certain way might be essential within the system, the drill and the cutter don't necessarily have to stand in that relation in any possible system in which they might exist.)

An interesting example of a system-essential relation is the part-of relation that often holds between a complex object and some of its parts. Consider an engine of a specific design. The engine can itself be viewed as a complex system, made up of many smaller parts. Each of these parts can be classified as instances of a kind, as can the engine itself. Call its kind E. Given some kind of part P that is necessary to the design of the engine, then, relative to P and E, the part-of relation is system-essential. Note also, though, that, given an instance e of E and the instance p of P within e, some other instance p* of P would have done just as well. Hence, the part-of relation does not hold essentially between the instances p and e.

As this example shows, entire systems can themselves be considered as further objects in yet larger system, and can be characterized as possessing certain properties, e.g., in the case of the manufacturing cell, comprising five machines. This means that an adequate ontology tool will have the capability of examining and characterizing the system from the coarsest to the finest levels of detail.

1.3 Accumulation of Domain Ontologies

What, exactly, is ontology good for? What role can it play in the design and development of information systems? In what sorts of information modeling contexts will it be useful?

One of the most important aspects of the general development and use of the IDEF5 methodology will be the accumulation of a wide range of domain ontologies. Among the greatest problems in information management generally is inefficiency. Redundant effort is expended capturing or recreating information that has already been recorded elsewhere. Consider the analogy with programming. Very often the same kinds of routines, e.g., in the design of user interfaces, are used again and again in different programs by (in general) different programmers. Enormous amounts of time and effort have thus gone into reinventing the wheel over and over again. Recognition of this problem has led to the development of vast libraries that have been collected over time that contain often used routines which a programmer can simply call straight into his or her program, rather than having to duplicate the function of existing code.

Information management across similar settings faces the same sort of problem. Manufacturing domains, for example, share many common features; and the more similar the domains, the more features they share. Rather than have to encode this information all over again in every new setting, our idea is to develop an analogue of the concept of a programming library by collecting this common information into ontology libraries, i.e., large revisable databases of structured, domain specific ontological information where it can be put to several uses in the IDEF5 environment. We envision numerous advantages to such libraries, two of which especially stand out. First, domain experts developing an IDEF5 ontology for a specific system will be able to import relevant portions of the general ontology database for the type of system they are describing directly into their IDEF5s. This will save them the trouble of having to record the information directly. This information will of course be malleable, so that a given expert can modify it in light of features unique to his or her system. Second, the information can be used to construct general techniques for aiding the domain expert in extracting domain knowledge. For example, by isolating and analyzing general patterns or features of ontologies within certain domains one can develop productive strategies for eliciting and structuring the sorts of knowledge one is likely to find in those domains. For instance, if a certain common type of machine varies in certain details from location to location, the background ontology database can import the common information directly, and then lead the user through a series of questions to elicit the specifications that are unique to his domain. Again, an expert may not know how a certain object should be classified. By searching on a list of essential properties of the object, the tool could return a set of kinds in which the object would most naturally be included.

With an array of ontology databases in use across a wide variety of engineering, manufacturing, business, and logistical systems, the task of information modeling could be revolutionized. The construction of such databases, of course, is an enormous--though, we believe, quite realizable--task. However, there is an even more basic task. Before one can build any complex physical objects--a bridge, say--there must be an appropriate methodological and theoretical foundation. This is no less true for abstract objects like information models. That is, before we can think about the structure of a domain specific ontology database, we need formal theoretical foundations for ontology proper---e.g., the appropriate representational medium---and methodological foundations for the capture and storage of ontological information. To those issues we now explicitly turn.

1.4 Ontology and Existing Methodologies

The goal of IDEF5 is not to define yet another methodology to do something a little better or a little different than some other existing methodology. We have no interest, and see no point, in instigating another skirmish in the methodology wars. Rather, our goal, first, is to point out a gap in the existing set of methodologies: there is, we believe, a type of information--ontological information--that has not been directly targeted by any existing methodology; our second goal is thus to make some preliminary suggestions for filling that gap, both theoretically and practically.

Thus far we have outlined the nature of ontological information. The importance of this sort of information should be clear. What is perhaps less clear is the need for a new methodology for capturing this information. In this section, we take up this issue.

For those familiar with other IDEF methodologies, the idea of capturing information about kinds and their associated properties will no doubt suggest both IDEF1 and IDEF1X. For a kind has been defined above as a certain sort of class, and this might then suggest that a kind is like an IDEF1 entity class or an IDEF1X entity. Furthermore, associated with each entity class (entity) is a list of associated attributes which assign property values to the members of the entity class. So perhaps we the makings of an ontology modeling method is right under our noses in one of these two methodologies.

Let's begin with IDEF1. Right at the outset we can say that it would be a serious error to think of IDEF1 as an ontology modeling tool. The central reason for this is that ontology modeling is real world modeling; that is to say, the members of kinds are real world objects, the actual instances of those kinds that exist within the system being modeled. The members of an IDEF1 entity class, by contrast, are information objects --they are objectified clusters of information that need to be kept about a system, the various "information images" of the real world objects within a system. Such objects are defined by the information they encode. Thus, all the property values associated with an IDEF1 information object are essential to that object; altering a value results in a new object.

This view has two consequences relevant to ontology. First, there will in general not be a one-to-one correspondence between the information objects within an IDEF1 model and the real world objects being modeled. For instance, within an IDEF1 model of a certain business there might be an entity class MANAGER and another entity class EMPLOYEE. These will be different entity classes since they keep different kinds of information. An employee of the business who is also a manager would thus generate two distinct information objects, one for each class--one for the employee in her role as an employee, and another for that same real world employee in her role as a manager. It would thus be a confusion to think of IDEF1 as an ontology modeler; it is simply not designed to represent that kind of information. Second, since all the properties of an information object are essential to it, there is no room for the distinction between essential and accidental properties; that latter have no purchase in the context.

We can press the issue farther. Suppose, against all better judgment, we overlook the above problems. Suppose we are determined to use IDEF1 as an ontology modeling tool and hence to represent kinds as entity classes. Here then is another difficulty. Suppose that some of the members of a certain kind of engine widget come with an additional, removable part—a FRAMMITZ—that, depending on its location on the widget, makes them suitable or not for use in engines of various sizes. Then having a frammitz, and its being located at a certain place on a given widget, are accidental properties associated with the kind WIDGET—members can either have them or lack them, and members that have them can come to lack them—but nonetheless they are properties of which it is important to be aware and to keep track. We've already noted that the inapplicability of the notion of accidental properties in IDEF1. But, further, in IDEF1, by the "No Null" rule, every attribute associated with a given entity class must yield a corresponding value for every member of the entity class. Thus, returning to the example, location_of_frammitz cannot be a legitimate attribute in an IDEF1 representation of the kind WIDGET, since not every widget has a frammitz, i.e., the value of location_of_frammitz for some widgets is null.

Now, in IDEF1 one can capture the information in question without violating the No Null Rule by inventing a new class of entity—WIDGET_WITH_FRAMMITZ. But in the context of ontology there are several problems with this. First, just as a matter of ontological aesthetics, to paraphrase Ockham's Razor, one shouldn't be forced to multiply entity classes beyond necessity; one shouldn't be forced to represent the information in question by introducing an entirely new entity class. But second, more importantly, despite the significant degree of freedom one is allowed in constructing an ontology for a human designed system, one is still constrained to make natural and useful divisions into kinds. But a class like WIDGET_WITH_FRAMMITZ does not represent such a division. From the perspective of ontology, it is an artifact foisted upon the modeler by the given modeling tool. The information in question is more accurately and appropriately captured by identifying the class of frammitz bearing widgets as a mere subclass of widgets whose members belong to the class contingently, than by identifying a separate, overlapping kind.

One might suppose, then, that we will fare better with IDEF1X. For although there is some disagreement about the exact semantics of IDEF1X diagrams, it is clear that the members of an IDEF1X 'entity' (IDEF1X's spectacularly ill-advised term for a class of similar objects in a system) are to be thought of as real world objects, not information images of those objects as in IDEF1. Thus, an IDEF1X model of the business in the above example would be thought of as containing the same real world object in both the EMPLOYEE and the MANAGER entities. In an ontology model of the same business, the kind EMPLOYEE and the subkind MANAGER would be thought of in the same way. Furthermore, with its capacity for expressing the subclass relation, the recommended analysis of the WIDGET example in the previous paragraph could be expressed in IDEF1X. So maybe IDEF1X is all we need.

However, there are deeper limitations. Chief among these is that IDEF1 and IDEF1X are purposely designed with certain expressive limitations built in in order to constrain the structure of the information that they represent. This makes for very clear, uncluttered, and efficient information and data models. But it also limits the applicability of IDEF1 and IDEF1X outside of their intended domains. IDEF1's inability to distinguish essential from accidental properties was illustrated above. The problem is shared by IDEF1X. Return to our manufacturing cell example above. Suppose for security reasons we want it to be impossible to swap out the diamond tool in the cutter; that is, suppose that we want to specify in the list of defining properties of the kind Cutter that any instance has to have a diamond tool essentially. Without the capacity to express modal information, this is not possible; in particular, it is not possible to express this in IDEF1X. But as the example illustrates, it may be of singular importance to be able to express such information.

Further examples abound. For instance, in both IDEF1 and IDEF1X it is not possible to name individual objects in an ontology and assert things specifically about them. Rather, one can only say things that hold of every member of a given class of entities in general. This is a crucial limitation in cases where there is a distinguished member of a given kind with special properties. And, more germane to the current context, it effectively rules out the possibility of carrying out the third task of ontology. If one can't say anything about specific objects, one cannot in particular talk about what properties they have. Again, the two methodologies can express only a limited variety of general propositions about the structure of the entities within a given class. For instance, one might want to note that for every member of class A with property P, there is another member with property Q. This is a straightforward quantificational statement, easily expressed, say, in predicate logic; once again, this proposition is beyond the expressive capabilities of IDEF1 and IDEF1X. But, as with the previous examples, this is the sort of thing that one might well need to say in giving a thorough characterization of the nature of the objects within a system.

The overarching point here is that the existing IDEF methodologies were simply not designed to do ontology modeling; they were designed with other goals in mind. Granted, especially with IDEF1X, we could probably nail on an addition here, bang on it until it fits our needs there, ad infinitum. But what would be the point? Why force a tool designed for one type of job to perform another? Why add such a burden to an already demanding task? Again, the claim is not that there is something wrong with or inadequate about the existing IDEFs. They were simply not designed to be tools for ontology modeling, and hence should not be expected to meet the requirements of such a tool.

1.5 Increasing Expressive Power

First and foremost among the requirements of an ontology tool, then, is greater expressive power. This need will be met in the theoretical foundations of IDEF5 by imbuing its underlying formal knowledge representation language with the full power of first order modal logic. The power of first order logic is well known, and greatly exceeds the expressive power of IDEF1. (Nonmodal first-order logic is developed and discussed in some detail in the KBSL report [...].) Modal logic extends first-order logic by introducing modal operators for necessity and possibility and a corresponding set theoretic semantics. This extension, among other things, gives one the power to express facts about essential and accidental properties in a very natural way. An essential property of x, recall, is a property that x could not fail to have, i.e., a property that is not possible for x to lack.

The standard set theoretic semantics for modal logic is discussed in terms of the heuristic concept of a 'possible world'. The idea goes back to the philosopher/mathematician Leibniz. Most of us believe that there are many ways the world could be other than the way it is in fact. These ways the world could be can be thought of as other possible worlds. One way the world could be, of course, is the way the world actually is. Thus, the actual world is one of the possible worlds. Unlike them, though, it is actual, not merely possible. An object S is said to exist in a possible world W just in case S would have existed if W had been actual. Now, it was noted that an essential property of an object S is a property that S couldn't have lacked. On the possible worlds picture, this can be defined as follows: property p is essential to S just in case S has p in every possible world in which S exists. Correspondingly, p is accidental to S if there is some world in which S exists and fails to have p.

It is often illuminating to think of systems in terms of possible worlds. In importing the enterprise of ontology into the information modeling domain, we noted that our concern was not with the world per se, but rather with the world of an organized system. Accordingly, in this context, possible worlds should be thought of not as alternative states of the world per se, but rather as alternative states of the system. Thus, a relational database model could be thought of as modeling in one fell swoop all the possible states of the database being modeled, all the different possible relations that could populate the database. Thinking in these terms often helps one to design more breadth and flexibility into the model in anticipation of possible but unlikely or previously unconsidered states. In the context of ontology, in addition to providing a definition of the notion of essential and accidental properties, the possible worlds picture helps one to anticipate or consider all possible natural kinds that might appear within the system, and thus to define a sufficiently broad ontology.

A caveat is in order here to head off a potential misconception. The intuitive concept of a possible world might suggest the idea of completeness or totality: a world, after all, is a total system, complete in every detail. However, the use of worlds in our formal apparatus might suggest that, in order for us to have an acceptable model of a given system, we must capture every piece of information within the system down to its last detail. But then informationally incomplete models like the simple manufacturing ontology model M above will not be do; we will have to fill in all the informational details before we have an acceptable model. For example, in a system represented by M, each machine consists of parts that were not mentioned explicitly in the model; and each part meets certain specifications that were not mentioned, and has a certain origin (e.g., a particular vendor) that was not mentioned; and so on. But practically speaking, this descending chain of information is unending. Similarly, any two objects within the system can in principle be regarded as a further object. There is often call for such representations -- suppose, for example, that in a system represented by M the cutter and the drill are integrated in such a way that it is useful to regard them jointly as a single object, Yet no such object is represented in M. Hence, the notion of a world seems to put far too to great a demand on the modeling enterprise.

Fortunately, this is not a genuine problem. The notion of a world should not be taken too literally. Formally speaking, worlds are just indexed structures that (in a modeling context) represent possible or successive states of a system. These structures themselves can be as sparsely or as richly detailed as the modeler desires, depending on how much detail he or she wishes to capture. In particular, a formalized version of the model M, with just that much detail, would be a fully acceptable 'world'. Since there is no finite upper bound the amount of detail that can be stored within this framework, one can add detail or new objects whenever it is deemed appropriate, and in whatever fashion is deemed appropriate.

The efficacy of the framework of possible worlds is witnessed by the fact that it is more or less the framework chosen by the members of the ISO working group for characterizing the notion of a conceptual schema: a conceptual schema consists of all the necessary propositions that hold in a given system, those that hold in all possible worlds, or all possible states of the system. Our use of the framework here thus ties in naturally with our work on the development of the three schema architecture.

2.0 Methodological Foundations

Our methodological experience in ontology development is based on practical industrial applications with Chrysler, Sematech and our work on the emerging Air Force IDEF5 ontology description capture method. (IDEF5 encapsulates the best practice experience in ontology development of the information management community at large to date.) The work with Sematech took place in the manufacturing and engineering domain; the work with Chrysler was in the product design domain. The experiences at both companies in developing ontologies was found to be remarkably similar. The still formative methodology sketched below is based on this experience. Broadly stated, the procedure consists of the following five steps (brief annotations follow the statement of each step):

Step 1 - Scope Domain and Collect Raw Data: This task is responsible for: 1) determination of the boundaries of a domain, 2) performing interviews with the domain experts, 3) collecting samples of data representative of the inputs, controls, policies, knowledge, and products of the domain.

Step 2 - Development of Initial Proto-kinds: This task is responsible for the analysis of raw data to generate a tentative relation-poor ontology of proto-kinds, proto-situations, and proto-situation types. By a relation-poor ontology we mean that system-essential relations of kinds are not yet considered in detail at this point (see the annotation to Step 4 below). By a proto-kind (-situation, -situation type) we mean a tentative kind (situation, situation type) generated from observation and/or a cursory analysis of existing sources of information. This provides a very useful, albeit defeasible, "rough draft" ontology to guide further inquiry and analysis.

Step 3 - Refinement of Initial Analysis: This task is responsible for the validation of the initial protokinds and the generation of a more stable (but still relation-poor) ontology from tentative ontology. Further inquiry and analysis guided by the tentative ontology gradually yields a revised and more stable ontology. Stability is of course a relative notion. Our experience confirms, however, that careful analysis can come close to the ideal.

Step 4 - Addition of Relations: This task is focused explicitly on the addition of system-essential relations to the ontology. The chief reason for this is that, if a significant number of relations are introduced into the tentative ontology, it can become an extremely messy task to untangle, reassess, and refine the initial relational connections. Furthermore, adding relations early on can be misleading, since the ostensible occurrence of a relation involving a nongenuine kind can prejudice a modelers assessment of the reality of that kind.

Step 5 - Validation of Stable Ontology Using Raw Data: This task is responsible for validation of the stable ontology by taking the initial raw data and attempting to "instantiate" it, i.e., model it within the stable ontology. Where this doesn't prove possible, or where it proves inordinately awkward, the ontology is modified appropriately.

At each step in the above described process the results will be distributed to our coalition for peer review and comment. In our experience, Steps 1, 3, and 4 were found to work very well in team contexts. Step 2--the move from a tentative ontology to a stable one, involves some fairly refined attunement to certain patterns within the system which only seem to appear when one develops the tentative ontology oneself from the raw data.

3.0 The IDEF5 Description Development Environment

3.1 Levels of Data Entry

First-order logic is powerful and efficient, but it does take a good bit of experience to master the art of translating ordinary language into it. Thus, we envision an environment that will permit several levels of data entry. Those familiar with logic should be able to enter information in that format directly. A level up from direct entry will be the possibility of graphical entry. There are several graphical representations of first-order logic that have been developed, several of them explicitly for the end of knowledge representation [Sowa, NETL, Burch]. We will be drawing on this work to develop our own graphical representation of first order modal logic. The modal component in particular will require work beyond what is currently available. The KBS research team counts modal logic among its stronger areas of expertise.3

In conjunction with the graphical language, we will also build in a facility for guided, structured text entry. The form of such entries will be midway between straight first-order modal logic and unconstrained natural language. We are fully cognizant of the severe, perhaps intractable, difficulties of full natural language processing (NLP), and we don't in any way pretend that we will be able to develop a full blown NLP component to the IDEF5 environment. (Though it will certainly be capable of incorporating the current state of the art at any point.) However, our own experience, and the experience of others, in developing constrained natural language environments has shown that users can with relative ease learn to express their thoughts within certain syntactic guidelines.4 Developing such guidelines in the IDEF5 environment will then permit entry of data in a manner that is relatively natural and easy to learn, but which is immediately processable by the software, or at least easily converted into processable form. The facility will include online guidance for proper entry, and an appropriate amount of built in syntax checking so as to assist the user without confusing or defeating him.

Finally, the IDEF5 environment will also allow straight text entry for those unfamiliar with the graphical or first-order languages, and for quick collection of domain knowledge that can be analyzed more formally at a later time.

3.2 Hooks to Other Methodologies

Our chief goal in developing and extending the suite of methodologies is data integration. Thus, we envision the IDEF5 environment itself to be smoothly integrated with the other IDEF methodology tools, as well as

[3] Cf. C. Menzel, "The True Modal Logic," forthcoming in the *Journal of Philosophical Logic*; also C. Menzel, "Actualism, Ontological Commitment, and Possible World Semantics," forthcoming in *Synthese*.
[4]Cf. P. Mayer, "A Computational Approach for Processing Locative and Temporal Information in Clinical Medical Records," unpublished Ph.D. dissertation, Department of Computer Science, Texas A&M University, 1989; also P. Mayer, et al., "Locative Inferences in Medical Texts," *Journal of Medical Systems* **11**, 68-85, (1987).

with tools developed for other, related methodologies such as ER and NIAM. Our efforts are thus geared toward the development of a comprehensive information modeling/knowledge engineering environment capable of storing, integrating, and reasoning with information across various types of domains.

4.0 Formal Foundations

In this section we provide a formal language and model theory for ontology. We will indicate along the way the roles of the various elements of the formalization, i.e., to what aspects in the informal development above they correspond.

4.1 Model Theory

We begin with the notion of a basic ontology model structure (boms). A boms is a representation of a system ontology (at some level of development and detail). More precisely, a boms M is an 8-tuple ·D,W,@,d,£,K,R,pÒ, where D and W are mutually disjoint nonempty sets, @ŒW, d : W Æ Pow(D) (i.e., d is a function from W into the power set (set of all subsets) of D). Intuitively, D is the set of all possible individuals, W is the set of all possible worlds or, more relevantly, all possible states of a given system, and @ is the actual world, or actual system state. d is then to be thought of as a function which assigns to every possible world wŒW the subset of D that consists of the possible individuals that exist in w. d(w) is called the domain of w.

The last four elements of M need a little more discussion. First, for all natural numbers n, let Fn be {f | f : W Æ Pow(Dn)}, i.e., the set of all functions from W into the set of all sets of n-tuples of elements of D. Fn is the standard possible world semantical definition of the set of all n-place relations; in particular, F1 is the definition of the set of all properties. The idea behind this definition is that, whatever properties ultimately are, it is intuitively clear that corresponding to each property in any given world is the set of all the things in that world that have the property. Thus, for example, corresponding to the property redness in the actual world is the set of all the things that actually are red, and in another world there is a different set. This suggests, rather than seeking any deeper analysis, that we simply identify redness with these varying sets, or more precisely, that we identify it with a function that, in each world w, picks out exactly the red things in w. To have the property redness in a given world w is thus simply to be in the set of things (the red things, of course) picked out by the property in w.[5]

Note that on this account of properties and relations, the function d in M which assigns a domain of objects to each world wŒW is a property, viz., the property existence: it is a function which assigns to each world w the set of objects that exist in w. Note also that the distinction between essential and accidental properties is captured straightforwardly in this framework. As noted above, intuitively, an object has a property p essentially just in case it has it in every possible world in which it exists, and it has p accidentally just in case there is some world in which it exists but lacks p. This translates as follows: an object a has the property pŒF1 essentially just in case, for all wŒW such that aŒd(w) (i.e., for all worlds in which a "exists") aŒp(w); and for relations generally, objects a1, ..., an stand in the relation rŒFn essentially just in case, for all w such that a1, ..., anŒd(w), ·a1, ..., anÒŒr(w). Similarly, a has the p accidentally just in case there is a wŒW such that aŒd(w) but aœp(w).

Given the definition of the Fn we can specify the character of the remaining elements of M. First, we stipulate that £ŒF2--i.e., that £ is a two-place relation on possible individuals--and that for each wŒW, £(w) is a reflexive partial ordering on the domain d(w) of w. That is, writing a£wb for ·a,bÒŒ£(w), for all aŒd(w), a£wa (reflexivity), and for all a,b,cŒd(w), if a£wb and b£wc, then a£wc (transitivity). Intuitively, £ represents the part-whole relation; thus, for all wŒW, £(w) is the set of pairs ·a,bÒ Œ d(w) such that a is a part of b in world or system state w. Thus, a£wb can be read asa is a part of b in w. We write a<wb if a£wb and aπb, and say that a is a proper part of b in w if a<wb. We also say that a is simple in, or relative to, w if a has no proper parts in w. If a is not simple in w, then we say that a is complex, or a system, in w.[6]

[5]This is the standard "possible worlds" definition of properties and relations. The account has suffered much criticism from philosophers and linguists of late because it is *coarse-grained*, i.e., properties and relations that pick out the same sets in all possible worlds are identical. However, intuitively, the objection goes, properties and relations can be necessarily coextensive without being identical, e.g., the properties **triangularity** and **trilaterality**. Though important, it is our belief that these issues to not typically effect ontology or information modeling generally, and hence the complexities of finer-grained accounts can be avoided. Cf. e.g., J. Barwise and J. Perry, *Situations and Attitudes* (Cambridge, MIT Press/Bradford Books, 1983), ch. 2; G. Bealer, *Quality and Concept* (Oxford, Oxford University Press, 1980), ch. 2.

[6]The idea of adding additional algebraic structure on each worlds domain of individuals to capture the part-whole relation was inspired by the work of Godehard Link on the semantics of plurals. Link imposes a full-blown boolean algebra on the individuals to provide interpretations for a wide variety of plural phenomena in natural language, and this seems to be far more structure than is necessary for present

As stressed above, part-whole relations are crucial for the accurate representation of physical systems, especially manufacturing and engineering systems, and this additional structure imposed on the objects of each possible world (possible system state) captures those relations in a simple but powerful way. Note that since the relation is partial, it can be as elaborate or as sparse as required: everything from the empty relation to a linear well-ordering counts as a partial ordering. The requirements of reflexivity and transitivity guarantee only that every object is a part of itself, and that the parts of the parts of an object a are also parts of a. In particular, because models needn't be complete descriptive representations, the part-whole relations between objects in a model can be as detailed or as sparse as one desires. This makes for great flexibility in the development of models, since it allows one to add part-whole information incrementally in the construction of a model to whatever extent is deemed necessary. Note also that the part-whole relation needn't hold essentially between two objects. That is, it is perfectly consistent within a model for a to be a part ofb in one world w and for a not to be a part of b in another. This implements the idea discussed above (see footnote ??) that, intuitively, most complex objects don't have all of their parts essentially.[7]

The sixth element K $\subseteq$ F1 is a set of properties that represent the kinds within a system, and hence the members of K are called the M-kinds, or the kinds of M. In our informal development above kinds were identified with classes, which are usually taken to be collections of some kind. However, kinds cannot be thought of as mere collections, since they transcend their members: the nature of a kind is not altered if its instances change. This is precisely the feature of properties noted above that distinguishes them from sets. Thus, in our more precise development, kinds are best identified with certain distinguished properties, and hence K is stipulated to be a subset of the set of properties F1.

The seventh element R of M represents the system essential relations, and hence we stipulate that R $\subseteq$ »n$\geq$ 2Fn, i.e., that R is a subset of the set of all 2-or-more-place relations. The final element p is defined to be a function

purposes here. Link also restricts his attention to nonmodal contexts. See G. Link, "The Logical Analysis of Plurals and Mass Terms: A Lattice Theoretic Approach," in R. Bauerle *et al.* (eds), *Meaning, Use, and Interpretation* (Berlin, De Gruyter, 1983).

[7] Note, however, that having a filter might be a defining property of the kind **cleaner**, so that in any state w of a given system, all cleaners must have filters. We have dwelled on this point already above, but if an ontology is not to be muddled, it is crucial that the distinction be clear.

on K»R such that p : K Æ Pow(F1-K), and p : R Æ »n≥2Kn such that for all n-place relations rŒR (n≥2), p(r)ŒKn. The role of p, then, intuitively, is to map each kind k to the set of its defining properties of k, and to map each relation r in R to the kinds relative to which r is system-essential in w. That no kind is the defining property of some other kind (or itself, for that matter) is ensured by the stipulation that p maps K into Pow(F1-K), rather than Pow(F1) simpliciter. The stipulation that p be one-to-one assures that no two distinct kinds have precisely the same defining properties. Note though that the defining properties of one kind k might constitute a (proper) subset of the defining properties of another kind k¢, so that every instance of k¢ is an instance of k. In such a case we say that k¢ is a subkind of k. One might, for example, wish to define a general kind cutter, and two separate subkinds diamond-tool cutter and carbide-tool cutter obtained simply by adding additional properties to the more inclusive kind. By defining a kind's defining properties independent of any world, however, we build in the idea that a kind's defining properties are essential to it. One's conception of a particular kind might change over time, of course, but this can be represented in terms of a series of several distinct but related kinds.

Further stipulations about p's behavior must be made in order to assure that defining properties and system essential relations are represented correctly in M. Specifically, we add in addition two conditions on p. First, if kŒK, then for all pŒp(k), k(w) ⊆ p(w), for all wŒW, i.e., in any world w, every member of the kind k in w must have the property p. Second, in the same manner, for any n-place relation rŒR such that p(r) = ·k1, ..., knÒ, for any wŒW, if ki(w)π∆ for all i such that 1£i£n, then there are a1, ..., anŒd(w), aiŒki(w), 1£i£n, such that ·a1, ..., anÒŒr(w). What this condition does is to capture the system-essentiality of system-essential relations; specifically, the condition says that for any world w, whenever each of the kinds relative to which r is system essential has at least one member in the domain of w, then r in fact holds between members of those kinds in w.

An important relation that can obtain between models is that one can be embedded in another, in the sense that all the information in one model is preserved in another model which contains more information. If a model M is so embedded in another M¢ we say that M is a submodel of M¢ . This sort of situation can arise in at least two ways. First, it is an essential fact of the modeling enterprise that models evolve over time. One of the circumstances under which this happens is when an existing model must be augmented in light of new information. Another is when one might purposely filter out information in order to obtain a simpler, more coarsely-grained model--not all available information, after all, is useful in all contexts; one might thus freely filter the information in a given comprehensive model in a variety of ways to obtain many different submodels.

Formally, then, to begin with, say that M = ·D,W,@,d,£,K,R,pÒ is a substructure of M¢ = ·D¢,W¢,@¢,d¢,£¢,K¢,R¢,p¢Ò if and only if D ⊆ D¢, W ⊆ W¢, @ = @¢, and d(w) = d¢(w)«D, for all wŒW. Suppose then M is a substructure of M¢, and let rŒ»nFn, be an n-place relation of M, and r¢ an n-place relation of M¢. Then we say that r is the restriction of r¢ to M, written r¢≠M, just in case, for all wŒW, r(w) = r¢(w)«D. M is a submodel of M¢ just in case M is a substructure of M¢; £ = £¢≠M; for each kŒK there is a k¢ŒK¢ such that k = k¢≠M (such a k¢ is called a correlate of k in M, ¢), and similarly for R and R¢; for each kŒK, and for each pŒp(k), there is some p¢Œp¢(k¢) such that p = p¢≠M, where k¢ is a correlate of k in M¢; and for each n-place rŒR, p(r) = ·k¢i, ..., k¢nÒ, where k¢i, 1£i£n, is a correlate of k in M¢.

Roughly, then, in English, M is a substructure of M¢ if the individuals and worlds of M¢ include those of M , they share the same actual world, and the individuals that exist in a world of M are exactly those that individuals of M that inhabit that world in M¢. Thus, all the individuals that inhabit that world according to M also inhabit it according to M¢, though M¢ may include new individuals in that world as well. The remaining conditions that must be met in order for M to be a full-blown submodel of M¢ simply spell out the idea that the properties and relations of M--in particular, the part-whole relations, the kinds, and the system-essential relations of M--can only change in M¢ in ways that increase information, i.e., such that none of the information of M is lost. Thus, for example, if a is a part of b in w relative to M, then a is a part of b in w relative to M¢--though there may be some part c of a in w relative to M¢ that was not recognized in M because c is not among the individuals M; this corresponds, e.g., to a situation in which M¢ represents a finer-grained representation of a system also represented by M. Again, a kind k may have more defining properties in M¢ than it had in M, but those in M¢ that have correlates in M will still be true of all the objects in M¢ that they were true of in M (plus perhaps some that were not among the individuals of M). It may be, however, that certain defining properties of k in M¢ were not recognized in M because they only appear at a finer level of granularity, or because of some other shift in perspective not captured by M.

4.2 Languages for Ontology

In this section we present the formal language for ontology and discuss the development of more user-friendly, graphical languages for use in the IDEF5 description development environment.

As noted the formal IDEF5 language L will will be a modal extension of first-order logic. It will thus consist of the usual possibly infinite store of individual constants c1, c2, ..., individual variables v1, v2, ..., n-place first-order predicate constants P^n_1, P^n_2, ..., and n-place predicate variables F^n_1, F^n_2, ..., for any or all n as desired, though it is required that L at least contain all variables and the predicates P^2_1, which will ordinarily be written as =, as well as the predicate P^2_2, which will be written as e. e will express the part-whole relation in L. In addition, L will contain the standard logical operators $ (existential quantifier), ¬ (negation), and & (conjunction), as well as the modal operator ‡ (possibility). L differs from typical first-order modal languages in that it contains predicate variables as well as distinguished higher-order predicates KIND, DP, and SERn (n≥2) that express the property of being a kind, the relation between a kind and its defining properties, and the relation between a system-essential relation and the kinds relative to which it is such, respectively.

Then syntax of the formal language will also be standard, modulo the special higher-order predicates. Specifically:

- If P is an n-place first-order predicate (constant or variable) and t1, ..., tn any n terms (i.e., constants or variables). Then Pt1, ..., tn is a (first-order atomic) formula (of L).

- If P is a one-place first-order predicate, then KIND(P) is a (second-order atomic) formula.

- If P and Q are 1-place first-order predicates, then DP(P,Q) is a (second-order atomic) formula.

- If P is a 2-place first-order predicate and P1, ..., Pn (n≥2) are 1-place first-order predicates, then SERn (P,P1, ..., Pn) is a (second-order atomic) formula.

- If j and y are formulas, so are ¬f, ‡f, and (f & y).

- If j is a formula and c is any variable (individual or predicate), then $aj are formulas.

We define the other standard logical operators in the usual way, viz.,

- f/y =df ¬(¬f & ¬y), f Æ y =df ¬(f & ¬y), f ´ y =df (f Æ y) & (y Æ f)

- ∀af =df ¬$a¬f, []f =df ¬‡¬f

4.3 Interpretations

Given a model M and a language L for ontology we now want specify how L is interpreted in M. This is done in terms of an interpretation function V which maps elements of L to appropriate semantic objects of M. The general notion of an interpretation function is discussed at length in the final technical report for the IISEE project, so we will not dwell on details.8 That said, an interpretation function V for L and M = ·D,W,@,d,£,K,R,pÒ is a function such that

- If t is a term (of L), then V(t) Œ D;

- If P is n-place first-order predicate (constant or variable), and wŒW, then V(P,w) Œ √(Dn); in particular, $V(P^2_1,w)$ = {·a,aÒ | aŒd(w)}, and $V(P^2_2,w)$ = £(w). (P^2_1, recall, is to be the identity predicate, and P^2_2 the predicate that expresses the part-whole relation.)

[8]C. Menzel and R. Mayer, "Theoretical Foundations for Information Representation and Constraint Specification," final technical report, IICEE project, AFHRL/LRL, WPAFB, Ohio, March 1991.

- V(KIND) Œ √(K); V(DP) = {·k,pÒ | k Œ K and p Œ p(k)}; and V(SERn) = {·r,k1, ..., knÒ | ·k1, ..., knÒ Œ p(r)}.

Interpretations for formulas of L will be defined recursively in terms of V in the usual way. Specifically, we define V′ to be a total extension of V such that V′ also maps the formulas of L into the set {T,F} (truth and falsity) in the following way:

- If j is a first-order atomic formula Pt1, ..., tn , then V′(j,w) = T iff ·t1, ..., tnÒ Œ V(P,w).

- If j is KIND(P), then V′(j,w) = T iff V(P) Œ K.

- If j is DP(P,Q), then V′(j,w) = T iff V(P,w) Œ K and V(Q,w) Œ p(V(P,w)).

- If j is SERn (P,P1, ..., Pn), then V′(j,w) = T iff V(P) Œ R and ·V(P1,w), ..., V(Pn,w)Ò Œ p(V(P,w)).

- If j is ¬y, then V′(j,w) = T iff V′(j,w) = F.

- If j is (y & q), then V′(j,w) = T iff both V′(y,w) and V′(q,w) = T.

- If j is $ay, then V′(j,w) = T iff there is a total extension V″ of V differing from V′ at most in what it assigns to a such that V″(y,w) = T.

- If j is ‡y, then V′(j,w) = T iff there is a w′ Œ W such that V′(y,w′) = T.

4.4 Axioms for Ontology

A proper axiomatic basis that captures the logic of our ontology models will be needed as a basis for developing computational tools with a capacity for automated reasoning. In this section we will describe an appropriate axiomatic basis, though we will not explore the issues of computational implementation, which will be a task for the next phase of IDEF5 development.

The basis for the system will be a fairly weak second-order logic modal logic. That is, in addition to the usual basis propositional tautologies, and axioms for quantifiers and identity,9 we also have the usual axioms of the modal logical system S5:

K: [](j ⊃ y) ⊃ ([]j ⊃ []y)

T: j ⊃ ‡j

5: ‡j ⊃ []‡j

and the rule of inference of necessitation:

Nec: If ˜æ j, then ˜æ []j

i.e., if a formula j is provable, then the proposition []j that it is necessary is also provable. This rule captures the intuition that anything provable is a truth of logic and hence should be true in all possible worlds, or for all possible system states.10

The last thing we need are axioms that capture the logical content of the distinguished predicates of the language of ontology-- i.e., KIND, DP, and the predicates SERn--that was given them in the definition of an interpretation above. Thus we have the following:

O1: DP(F,G) ⊃ (KIND(F) & ¬KIND(G))

O2: DP(F,G) ⊃ ∀x(Fx ⊃ Gx)

O3: SERn (F,F1, ..., Fn) ⊃ (KIND(F1) & ... & KIND(Fn))

O4: SERn (F,F1, ..., Fn) ⊃ [($xF1x & ... & $xFnx) ⊃ ($x1F1x1 & ... & $xFnxn & Fx1...xn)]

O5: DP(F,G) ⊃ []DP(F,G)

9 See again the IISEE report. It should be noted that the logic developed in the latter report is not second-order; however, the quantifier axioms for the logic in the present report will work in exactly the same way, regardless of whether the quantified variable is first- or second-order.
10 There is some doubt about the soundness of necessitation as a general modal rule of inference however; cf. C. Menzel, "The True Modal Logic," forthcoming in the *Journal of Philosophical Logic*.

O6: SERn (F,F1, ..., Fn) $\supset$ []SERn (F,F1, ..., Fn).

O1 says that if the relation DP holds between two properties F and G, then F must be a kind, and G must not be a kind. This captures the idea that DP holds between a kind and any of its defining properties, which includes the idea that no kind is a defining property of any kind. O2 expresses the thesis that a member of a kind must have all of the defining properties of the kind--note, however, that it does not say that it must have them essentially, in line with our earlier discussion of the notion of a kind. O3 captures the idea that system essential relations are such relative to some finite collection of kinds. O4 expresses the thesis that a system essential relation must hold between representatives of the kinds it is relative to whenever those kinds are nonempty. O5 and O6 capture two important modal properties of kinds and systems essential relations, viz., that if G is a defining property of a kind F, then it is necessarily a defining property of F; and if F is system essential relative to some collection of kinds, then it is necessarily system essential relative to those kinds. Note that these properties are enforced in the model theory by the fact the values of the function p on a member of K or R was defined independent of W. (p, recall, determines the defining properties of a given kind, and the kinds relative to which a given relation is system essential.) Note also that few properties of kinds and system essential relations follow from the restriction that p be one-to-one; however, these are not expressible in the language as it stands, since it requires that we be able to express identity between properties and relations, and this requires a second-order identity predicate. Addition of such a predicate will be explored in the next phase of IDEF5 development.

5.0 Bibliography

[Barwise 83] Barwise, J. and Perry, J., Situations and Attitudes, The MIT Press, Cambridge, 1983.

[Devlin 91] Devlin, K., Logic and Information, Volume I: Situation Theory, Cambridge University Press.

[Hobbs 87] Hobbs, J., Croft, W., Davies, T., Edwards, D., and Laws, K., The TACITUS Commonsense Knowledge Base, Artificial Intelligence Research Center, SRI International.

[Link 83] Link, G., "The Logical Analysis of Plurals and Mass Terms: A Lattice Theoretic Approach," in R. Bauerle et al. (eds), Meaning, Use, and Interpretation, Berlin, De Gruyter, 1983.

[Menzel 90] Menzel, C., "Actualism, Ontological Commitment, and Possible World Semantics," Synthese 85 (1990), 355-389

[Menzel 91a] Menzel, C., and R. Mayer, "Theoretical Foundations for Information Representation and Constraint Specification," final technical report, IISEE project, AFHRL/LRL, WPAFB, Ohio, March 1991.

[Menzel 91b] Menzel, C., Mayer, R., and Edwards, D., "IDEF3 Process Descriptions and Their Semantics," forthcoming in Kuziak, A., and Dagli, C., Knowledge Base Systems in Design and Manufacturing, Chapman Publishing, forthcoming 1991.

[Webster 88] The Merriam-Webster Dictionary, Simon & Schuster, New York, NY, 1986.