Roles in the Universal Database: Data and Metadata in a Distributed Semantic Network

Peter Lucas, Dominic Widdows, Joe Hughes, William Lucas
{lucas,widdows,hughes,lucasw}@maya.com

MAYA Design Inc.
Technical Report MAYA-05009

Abstract. This work begins with an assumption that a universal ontology is too difficult for humans to define once and for all: however, it may be possible for such an ontology to evolve in the marketplace, provided the underlying data-structure is flexible enough to permit such evolution.

In traditional ontologies, classes of objects have to be defined before individual objects can be modelled as part of those classes. In traditional databases, schemata have to be chosen for entire tables of data objects before information can be represented. Both approaches can be used to build stable closed systems, but it is very difficult make two systems with different schemata interoperable.

This paper describes an alternative, whereby all systems for data representation use a common currency of underlying data-objects called u-forms. A u-form is simply a bundle of attribute-value pairs indexed by a universally unique identifier. Any attribute can be added to a u-form at any time, so there are no fixed schemata. Instead, schemata are added by giving relations to *role* u-forms. A role is simply another u-form which defines a local attribute namespace. In this way, authors of information can state the intended interpretation of attributes, without compromising the system's fundamental ability to cope with *any* attributes and values.

We describe some of the basic roles used for representing phenomena, collections, and adjectival assertions for spatial extension and temporal scope, in a universal and scalable semantic network. An important corollary of this method is that a single u-form can play several roles, enabling widespread polymorphism. This crucial behavior can be used to model systematic ambiguity, until now a breaking point for formal representations. We consider the implications of this plasticity in the light of linguistic and philosophical considerations, including Generative Lexicon theory and Wittgenstein's objections to feature-based definitions.

1 Introduction

An acknowledged problem with ontologies is that they are often *brittle*, which is to say, they are easily broken. An ontology designed for a very specific domain may be quite successful at describing objects in that domain of discourse. However, every domain of discourse (even pure mathematics) shares concepts with

other domains, and the schemata used to model one domain may not sit well upon another. This makes the problem of transferring ontologies from one domain to another a difficult research challenge for the Semantic Web community (see e.g. [1, 2]). Even when ontologies can be unambiguously aligned, there is little guarantee that the new ontologically defined *meaning* of a piece of information corresponds accurately to the original *intention* of its author.

An alternative approach is to try and build a single ontology that is extensible to all domains, the most notable ontology with this goal being perhaps the Cyc knowledge base [3]. Projects such as Cyc demand that the model begin with a top-level description of categories of objects in the world, and many objects or relations that were not considered during the original design may be difficult to fit into the ontology. There are examples of this difficulty in the WordNet lexical database [4]: some "religions" are listed as "organizations", some as "psychological states", and some as both. This makes it difficult to use WordNet as a starting resource for compiling a dataset of "World Religions", because while it contains much of this information, it was not designed for this purpose.

Related problems and research have arisen in the relational database community. Ever since the (spectacularly successful) introduction of the relational database [5], database design has involved creating a *schema* for an entire relational table before any information can be entered into the table. This makes it extremely difficult to extend database tables when new kinds of information become available, or to integrate information from different database sources with incompatible schema, although this also is an active area of research [6]. The inflexibility of relational tables is one of the reasons for the increasing popularity of XML (eXtensible Markup Language [7]), which forms the syntactic backbone of the Semantic Web [8].

A fundamental design pattern that makes relational databases and ontologies so difficult to manage may be described as *fixed semantic binding*. That is, at its inception, a contributor must decide upon a fixed syntactic schema and semantic interpretation for a piece of information, within a predefined universe of discourse. We believe that this places too many constraints upon the information modelling challenge to enable fully tractable, future-proof solutions.

Tight coupling of syntax and semantics to permanently fix an interpretation is nowadays accepted as a fundamentally difficult problem. The philosophical science of "ontology" (deciding what kinds of objects exist in the universe) has largely given way to the construction of "ontologies", and the transition of the word "ontology" from abstract mass-noun to common count-noun syntax is a significant indicator of this difficulty. However, this grammatical shift is also indicative of the usefulness of having at least *some* ontology, whether count-noun or mass-noun — that is, *some* way in which providers of information can make statements about how that information is to be interpreted.

This paper describes a system for enabling users to supply such information, in a way that enables robust automatic interpretation and inference to be made

where possible, and enables simple display of information where appropriate. The system relies on three basic concepts.

1. A *u-form*, which is a bundle of attribute-value pairs together with a unique identifier. U-forms are basic data-containers that can be shepherded between venues in a peer-to-peer system [9].
2. An *encoder*, which is an interface device that enables a user to interact with u-forms.
3. The focus of this paper is on the third concept, which is a *role* u-form. A role u-form is simply a u-form that declares an attribute namespace for other u-forms to follow.

Effectively, roles can be used to apply a schema to a u-form so that its attributes and values can be interpreted correctly by a human user (by way of a suitable encoder), or by an artificial reasoning agent (such as a spatial or temporal reasoning component).

While roles enable ontological relationships to be expressed in a uniform manner, they are not in themselves ontologies: they are attribute namespaces that enable ontological assertions to be made along with an intention for interpretation. Since non-overlapping namespaces can be mutually compatible, a single u-form can play multiple roles. The importance of this polymorphism is one of the crucial benefits of the system. For example, it enables any object with a `latitude` and `longitude` attribute to be placed correctly on a map, provided that the *Role for Geo-Reference* is there to assert that these attribute names should be interpreted in this fashion.

The compatibility of multiple schemata has another important consequence: it enables individual roles to prosper or be rejected by groups of users, without unnecessary dependence on the rest of the information architecture. Instead of trying to choose global attribute names for the whole system, we choose only one global attribute name, `roles`, and this is enough to specify the interpretation of all of the other attribute names. All other attributes are therefore free to vary.

As this paper attempts to demonstrate, we believe that this system of layering multiple compatible schemata on top of a fundamentally extensible data object is capable of modelling persistent yet evolving data-structures. It allows a universal database system to use well-understood schemata and to model ontological assertions, without being constrained by these modelling decisions.

The rest of this paper is organized as follows. In Section 2, we introduce the crucial concepts of *u-forms* (abstract information objects), *encoders* (physical information interfaces) and *roles* (the structured namespace that mediates between a u-form and an encoder). This paper is mostly about roles, since they are used to give metadata and semantic structures to the system. Section 3 describes more clearly how the space of roles is organized, and how this organization enables the marketplace to gradually prefer some semantic structures and deprecate others. Section 4 explains how adjectival information can be added to all sorts of u-forms, enabling spatial, temporal and attribution information to be stored on the same u-forms, in parallel to any ontological categorization. In

this, multiple schemata can further enable many parts of the information architecture to prosper even if others are unsuitable. Finally, Section 5 compares the information architecture desribed in this paper with other information systems, and to certain lexical principles from linguistic theory.

2 U-forms, Roles, and Encoders

This section describes the main concepts that will be explored in this paper. It should be stressed that these concepts all exist both as abstract principles (such as the formal definition of a datatype) and as working implementations, in a large peer-to-peer database system.

2.1 U-forms

The fundamental currency of our system is the *u-form* [9]. A u-form is a bundle of attribute-value pairs (called an *e-form* by Michael Dertouzos [10]) indexed by a unique identifier or UUID. The benefit of UUID's is that they can be used to give permanent identity to *all* information objects, not just those at fixed physical locations (e.g. those referenced by URL/URI references). While it sounds daunting to give a unique identifier to all information objects, it is in fact not difficult, and may be utterly necessary to enable a world of pervasive computing rather than fixed devices (see [11]). Successful schemes for generating UUID's may be achieved by both hierarchical authority or by automatic generation. A recent proposal to incorporate automatically generated UUID's within the URN namespace of the internet may lead to many systems using this concept as a means of exchanging unambiguously referenced information objects [12].

The concept of a u-form is adequately described elsewhere [9].[1] For the purposes of this paper, the most important points are as follows:

- In the universal database system, u-forms are the basic currency of information, that is, all information is expressed in u-forms.
- U-forms have no fixed schemata and are forever extensible. That is, any new attribute can be added to any u-form at any time (provided the user has suitable write permission).
- U-forms are not specific to any one location. A request for a u-form with a given UUID will cause the system to look for that u-form in the user's current venue, if this fails, a request for the u-form goes out to the peer-to-peer network.
- UUID's can be used as values of attributes in their own right, enabling u-forms to refer to one another, creating a global semantic network.
- Network traffic, synchronization of u-forms across different venues, and the recognition (and sometimes management) of conflicts is handled by artificial agents called *shepherds*.

[1] See also `http://www.civium.net/civwiki/`

– The system is designed to be entirely distributed and self-describing. Indexes, metadata, and user session information are all stored in u-forms.

U-forms are by definition extensible, and so are bound by no fixed schema. In fact, such a restraint would violate the distributed nature of the system: in one interface to the repository of u-forms at one particular venue, it may be possible to restrict the list of attributes that certain u-forms may possess, but it is impossible to enforce this practice in other venues (or even to know that all venues have been alerted to such a constraint, since many users of the repository may be disconnected).

2.2 Roles

Since u-forms have no predefined schema, an attribute name may have many different interpretations. For example, the attribute `parent` could easily refer to the biological parent of an individual person in a genealogical tree, or to the conceptual parent of a node in a taxonomy (for example, it would be perfectly reasonable to create a u-form for the concept *thrush* whose `parent` attribute contained a relation to the u-form representing the concept *bird*). Or indeed, the attribute name string `parent` could refer to any value a user wants it to, though if they use it to refer to the latitude or color of an object they run the risk of being misinterpreted.

The appropriate meaning of an attribute name such as `parent` may thus vary from u-form to u-form, and the meaning appropriate to this particular u-form is defined by the system of *roles*.

Roles work as follows. A role is, by definition, a u-form that defines the intended meaning of certain attribute names. For example, there is a basic role called the *Role for Entity* that defines three attribute names, `name`, `label`, and `description`. As well as listing these attribute names, the role gives a type definition (to be correctly displayed, these attributes should all be strings) and a human-readable semantic description of how each attribute is to be used.

For a u-form u to assert that the attributes called `name`, `label` and `description` should be interpreted according to the specification in the *Role for Entity*, it is enough to place the UUID of the *Role for Entity* in the `roles` attribute of the u-form u. In this way, the entire database carries its own metadata: the only special attribute name that an agent must know about in order to access this metadata is the attribute name `roles`.

Just like any other attribute, the `roles` attribute of a u-form may be edited at any time: thus, u-forms may (and frequently do) become adapted to new schemata during their lifetime.

Roles are effectively a way of enabling a creator and a user of information to agree on the intended interpretation of this information. The structure of the role space and instructive examples of roles will be described in more detail in later sections.

2.3 Encoders

An *encoder* is a physical device[2] that is used to render a u-form to a user. Encoders are driven by the role system: that is, an encoder will render u-forms that play a certain role, and will ignore u-forms that do not play this role. This introduces a contract between the author of information and the system displaying it.

Years of experience designing information architecture and user interfaces has taught us (as individuals, and as a community of practitioners) that very few users care at all about metadata standards and semantic interoperability — at least, not when they are described in this fashion. In order to encourage whole communities of users to adopt recognized standards, there must be significant and immediate incentive. The number of World Wide Web pages annotated with HTML runs into the billions, whereas the number of Semantic Web pages annotated with RDF/OWL descriptions is still comparatively small. This is not because HTML is a good architecture and RDF/OWL is a bad one; it is because there are many readily available browsers in which a user can see instantly that adding a tag such as `<b>` will get the text in their webpage to appear in boldface.

Encoders are the incentive for users to use the role system correctly. For example, consider the u-form displayed in Figure 1. This encoder is rendering the u-form which contains the first chapter of the novel *Moby Dick*. For those interested in the preservation and dissemination of public-domain information, it is already worthwhile to create a pervasive datastore that can keep a single version of *Moby Dick*, and replicate this version to many different venues without ambiguity.

However, many users would be unlikely to use the system for this reason alone. By importing your data to the system using the *Role for Textscrap* and the associated attribute names, a user can take advantage of ready-made indexing and rendering tools. This incentive is likely to make many users use u-forms and roles who would otherwise "not see the point" in semantic standards and metadata.

It is quite possible for developers to create new encoders for old roles, and this polymorphism is one of the keys to success in the system. For example, u-forms playing the role for *Tabular Dataset* may be displayed as a table, a histogram, or a bar chart, depending on the user's preference. In this way, we see the way in which the marketplace can begin to decide which roles are useful and which are not: a role that are well thought out and applicable to many different u-forms is likely to prosper, because good encoders will take advantage of this role, and u-forms will use the role to take advantage of the encoders.

3 The Structure of the Role Space

This section describes the structure of the space of roles more closely, enabling the reader to understand more clearly how the structure can evolve.

[2] A computer running a piece of software is by definition a device: an algorithm is a piece of abstract information, a machine running that algorithm is a physical device.

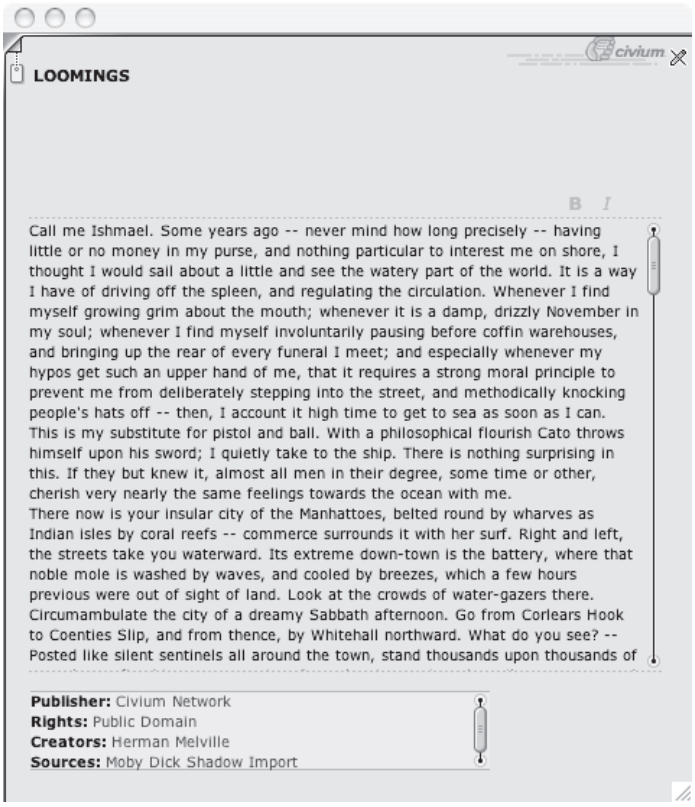

Fig. 1. Encoder guided by the *Role for Textscrap*. This encoder knows to display the value of the `text_content` attribute in the main frame area, the value of the `name` attribute at the top, and the values of attributes such as `publisher` and `rights` in the scrollbar at the bottom.

Consider again the u-form of *Moby Dick* encoded in Figure 1. As stated earlier, this u-form plays the *Role for Textscrap*. The attributes defined by this role are shown in Figure 2. Drilling down on any of these attribute names will lead to a description of the attribute, and the expected syntax and semantics of these values. Note that roles also have a special encoder, that encodes the *Role for Role*. Even the *Role for Role* itself, which lists attribute names, their types, and semantic desciptions, would not be expected to prosper in the marketplace. without a good encoder.

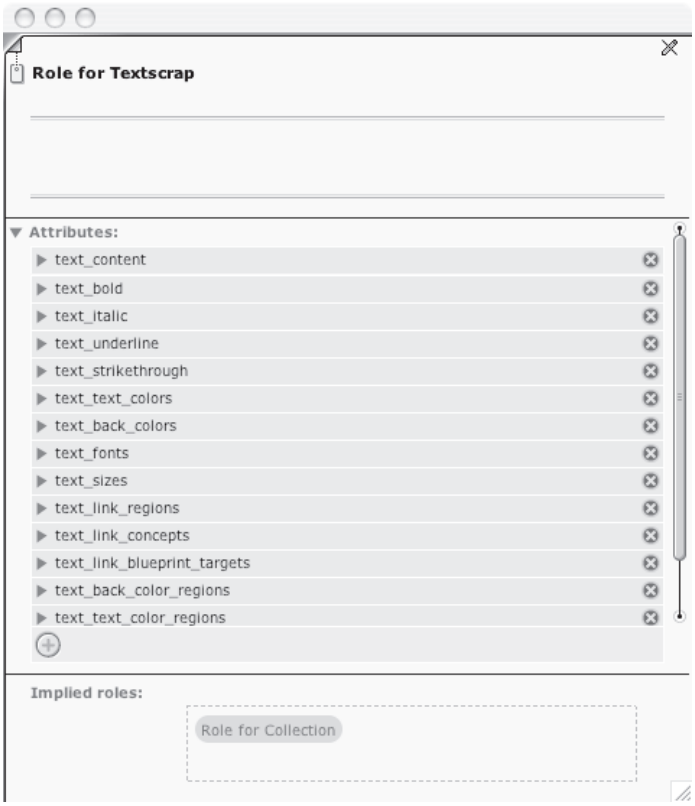

Fig. 2. The *Role for Textscrap*, showing the attribute names `text_content`, `text_bold`, etc. The pane at the bottom shows that this role inherits from *Collection*.

At the bottom of this frame is a pane entitled *Implied Roles*, which contains a link to the *Role for Collection*. The *Role for Collection* defines only one attribute, called `members`, and asserts that the value of the `members` attribute is simply a list of UUID's. This role is shown in the encoder in Figure 3. The *Role for Collection*

is used in any situation where one u-form contains links to many others: as such, it is the principle way to implement Boolean set theory in u-forms. One benefit of this is that optimized set theoretic operations (union, intersection, difference, containment) need to be implemented only once, and applied as predicates to any pair of u-forms that play the *Role for Collection*. This role is also the main way of accessing drill-down features in interfaces. Again, the interplay between u-forms and encoders, mediated by the role system, comes into its own: users can quickly learn that some encoders have a drill-down feature, and that by using the *Role for Collection*, they can give any dataset a nested structure that can be explored recursively using this feature: not only by themselves, but by any user who has the same encoder and the UUID of the head of the nested collection.

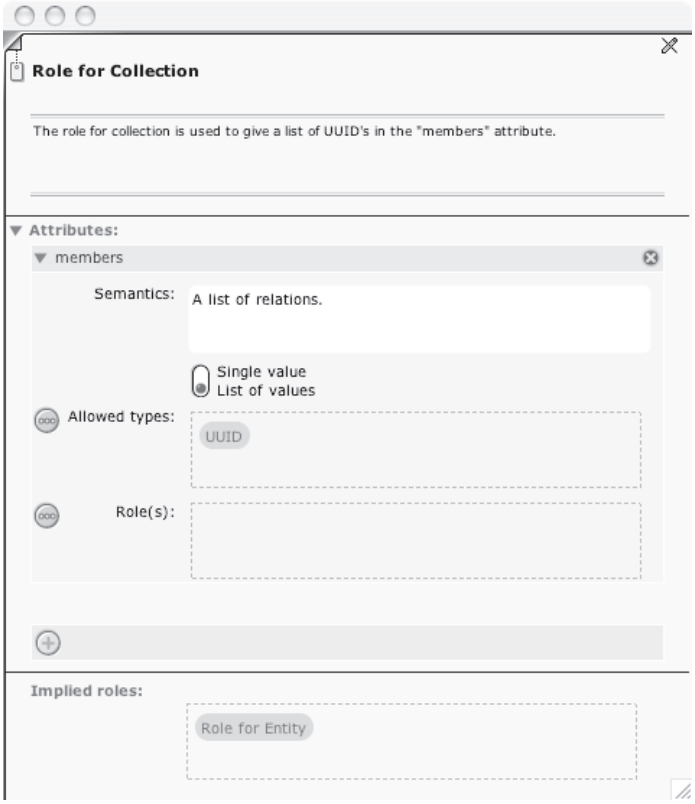

Fig. 3. The *Role for Collection*, showing the single attribute name `members`, and that this attribute takes a list of UUID's as its value. This role inherits from one role, the *Role for Entity*.

The fact that the *Role for Textscrap* has a link to the *Role for Collection* in its `implied_roles` attribute means that the *Role for Textscrap* inherits all of the attributes covered by the *Role for Collection*. In this way, one role can be used to extend and specialize the namespace defined by another. This has powerful consequences, especially when polymorphism is considered (i.e. that one u-form can play several roles, and a role can have several encoders).

In this case, the fact that *Role for Textscrap* inherits the attribute `members` from collection means enables a textscrap to be interpreted as simply a collection of hyperlinks. An smart encoder that knows how to format these `members` along with the other textscrap attributes can intersperse these links in the correct parts of the text. However, a dumb encoder that knows about the *Role for Collection* but not the *Role for Textscrap* can at least enable a user to drill down from a textual object to reach any of the other objects cited therein.

The feature whereby one role can extend the namespace of another is termed *Role Inheritance*, because of its obvious similarity with paradigms such as Object-Oriented Programming and Formal Concept Analysis [13]. However, role inheritance is strictly a namespace extension, not an ontological statement. The fact that *Role for Textscrap* inherits from *Role for Collection* is not a statement that every piece of text is a collection of hyperlinks (many are not). It is merely a statement that, if the `members` attribute of a textscrap cannot be given a specialist interpretation by a textscrap encoder, it should be given a general interpretation by a collection encoder.

However, while it is not normally used for making ontological assertions, there are some similarities between the role space and other semantic networks. As long as roles are simply describing an attribute namespace, they are purely metadata, not ontology. However, roles may also express preferences concerning the *values* of their attributes, and once roles are used to make statements about *values* as well as attributes, we begin to encroach upon ontological territory. Some roles are *type-enhanced*, which means that they can express the desired types (int, float, string, UUID/relation, list) of their values. Some roles take this a step further, becoming *constraint-enhanced*. Constraint enhanced roles can state a preference concerning the targets of any relations contained in their attributes. For example, the role for *Event* has an attribute `location`, and in order for the event to be placed on a map, it may be specified that the value of this attribute should play the *Role for Geo-reference using global coordinate system*.

At the extreme, a few roles fully specify the values of certain attributes. For example, the *Role for Country* inherits from the *Role for Geo-political subdivision*, since countries have some attributes that other geo-political subdivisions do not (such as 2-letter internet codes such as `uk` for the United Kingdom and `de` for Germany. The *Role for Country* inherits the attribute `administrative_level` from the *Role for Geo-political subdivision*, which is normally set to 1 for a country, 2 for a region or state within a country, and so on. The *Role for Country* asserts in its definition that the value of this attribute should always be set to 1, and by this stage we have definitely started to build an ontology.

Another similarity with inheritance structures such as ontologies is that role inheritance must never be cyclic. In practice, this should never happen, because if a role A extends the namespace defined by a role B, then it is not possible for a role B also to extend the namespace defined by A.

One of the most important reasons for role inheritance is to enable a comparatively few good encoders to prosper near the root of the role-tree. One of the most basic roles is the *Role for Entity*, described earlier in Section 2.2. This role defines 3 attributes, `name`, `label` and `description`. Many encoders know about the *Role for Entity*: for example, all of the encoders depicted so far in the paper use the *Role for Entity* to place the `name` attribute at the top of a frame, and the `description` attribute (where present) in the box just underneath. The `label` attribute is used to contain shorter abbreviations (e.g. 'PA' instead of 'Pennsylvania'), which can be useful for encoders with limited space-per-item, such as map encoders.

Such general encoders often encourage users to supply missing attributes if these attributes are described by standard roles (provided there are good tools for doing this). Again, while indexing agents and reasoning agents may use role-mediated information behind the scenes, it is important to create systems where users can *see* the benefits of adopting standards in order to get users to commit to a structured system. Role inheritance is one of the main tools for enabling the marketplace to test roles and encoders and to keep those ideas that are genuinely useful. For example, it is possible that a developer may believe that the *Role for Textscrap* (Figure 2) is badly structured, and may design a competing role and encoder for representing text and hypertext. If many users prefer this new encoder, they will structure their u-forms accordingly. However, we believe that it is much less likely that any developer will ever decide that the *Role for Entity* is a mistake, and that the attribute names `name`, `label` and `description` should be changed. If the *Role for Entity* remains and the *Role for Textscrap* falls by the wayside, then that is a perfectly acceptable outcome. Even if this is frustrating for the original developers of the textscrap encoder, we believe that such an architecture is a significant step forward over other relational and ontological models. Roles provide a namespace organization that can be used to mediate everything from simple formatting guidelines to full semantic networks, and they do it in such a way that many parts of the system will survive and be reused, even if other parts of the system fail to stand the test of time.

4 Adjectival Roles

Many roles are normally used to give attribute namespaces for representing objects in the real world. U-forms that represent objects in the real world are called *phenomenal* u-forms. Phenomenal u-forms correspond closely to the category of *substance* as described by Aristotle (c.f. *Categories*, Ch 5). The roles played by phenomenal u-forms almost always inherit from the *Role for Entity* (possibly indirectly, since role inheritance is by definition a transitive operation). In this part

of the modelling process, roles occasionally arise that correspond quite closely to ontological templates.

Some of the most important roles are not 'ontological' in this sense, and are not used to represent phenomena, but to represent attributes that many u-forms playing a variety of roles may possess. These often correspond closely to Aristotle's categories of *quantity*, *quality*, and *relation* (c.f. *Categories*, Ch 6, 7, 8), and the roles that describe them are called *adjectival roles*.

Because u-forms are extensible and can play several roles, it is easy to add such adjectival information to u-forms. This is done by adding the adjectival role's UUID to the list in the u-form's `roles` attribute, and by adding the desired adjectival attributes to the u-form itself. Important adjectival roles include the following.

Role for Geo-reference using Global Coordinate System Many classes of physical objects (countries, buildings, rivers, roads) have fixed locations on the planet Earth, and building a common language for this location information is becoming increasingly important in a world populated by smart mobile devices.

To enable the locations and extents of all such objects to be expressed, there is a single *Role for Geo-reference using Global Coordinate System* (or just *Role for Geo-reference* that describes the attributes `latitude`, `longitude`, and `geo_extents`, the latter being a list attribute that contains the latitude and longitude coordinates of the bounding box of the object.

Any u-form playing the *Role for Geo-reference* can be placed on a map by a map encoder. Any such u-form can also be placed in a spatial index by an indexing agent. Again, the modular design of the role space comes into its own: it is possible for a simple encoder to display many objects on the map even if it does not know how to represent their other attributes. It is even possible for developers to disagree on what these other attributes should be, but still agree on the standard use of the attribute names `latitude` and `longitude`.

Role for Temporal Reference As with the *Role for Geo-reference*, many different kinds of objects have temporal extents. In fact, even more objects have temporal extents than spatial extents: for example, literary works such as *Moby Dick* and organizations such as the *United Nations* may be said to have come into existence during a certain time, but can not really be located to any one place (even though copies of a book an facilities owned by the institution may be located).

The *Role for Temporal Reference* uses four temporal attributes, each of which takes a value along a global time axis (currently defined by the W3C time standard [14]). These are as follows:

- `starttime_min`Indicates a lower bound on the time at which the event began. If this u-form is phenomenal, this value must represent the earliest time at which the phenomenon could have come into existence.

- `starttime_max` Indicates an upper bound on the time at which the event began. If this u-form is phenomenal, this value must represent the latest time at which the phenomenon could have come into existence.
- `endtime_min` Indicates a lower bound on the time at which the event ended. If this u-form is phenomenal, this value must represent the earliest time at which the phenomenon could have ceased to exist.
- `endtime_max` Indicates an upper bound on the time at which the event ended. If this u-form is phenomenal, this value must represent the latest date at which the phenomenon could have ceased to exist.

It may at first seem strange to use four attributes for expressing a temporal event, whereas normally 1 is normally considered sufficient to represent a point event and 2 points are considered sufficient to represent an interval. The problem with such schemes is that they presume that we know times exactly (or at least, that our uncertainty about time measurement is normally distributed about a central point). However, this is rarely the case: often, when dealing with temporal assertions, the information we have coming in to the system is scoped to a given interval such as a year or a day. For example, it is easy in our system to express that the composer J.S. Bach was born in 1685 and died in 1750, by setting the attributes

`starttime_min` = 1685-01-01T0000
`starttime_max` = 1686-01-01T0000
`endtime_min` = 1750-01-01T0000
`endtime_max` = 1751-01-01T0000

For many indexing purposes, it is only necessary to set the `starttime_min` and `endtime_max`. However, in many cases it is extremely useful to have the option of using the other attributes. We believe that this system contrasts favorably with that proposed by the DAML time ontology [15], in which temporal granularity can only be easily expressed for intervals that are a notational prefix in the underlying time format, any other intervals requiring a complex machinery involving creating equivalence classes and evaluating predicates in first-order logic.

Like the *Role for Geo-reference*, the *Role for Temporal Reference* and its accompanying attributes can be applied to many different sorts of u-forms, and the information architecture of temporal assertions is decoupled from any ontological modelling carried out using attributes of phenomenal roles.

Role for Attributed U-form It is extremely important to record the sources of information, especially when trying to create a public information space using peer-to-peer technology. The *Role for Attributed U-form* defines the attributes `publisher`, `creator`, `source`, `date`, `language`, and `rights`, enabling a user to assert (for example) that the text of *Moby Dick* is in the Public Domain, that it was created by the author Hermann Melville, that this particular u-form was

published by the Civium Network,[3], and that the source text for this version is the public domain text obtained from Project Gutenberg.[4]

Many encoders use the scrollbar at the bottom of a frame to display the attributes from the *Role for Attributed U-form* (see Figure 1).

Any u-form can use the *Role for Attributed U-form*, which is used by shepherding agents to make sure that u-forms with copyrighted material (or whose publishers are not willing to vouch for the non-infringing nature of their content) are not distributed.

Injective and Projective Adjectival Roles The three adjectival roles we have described so far are *injective*. Such a role can be used to insert ('inject') attributes directly onto a phenomenal u-form. This is possible because any attributes can be added to any u-form, and because it makes sense to add these particular attributes *only once*. For example, it is not necessary to be able to assert that an object has more that one `latitude` attribute.

On the other hand, there are some adjectival attributes that may occur many times in relation to the same u-form. An individual or an organization may have many telephone numbers, and a whole range of objects may be offered for sale at many different prices under many different terms and conditions.

To incorporate such cases into the information architecture, *projective* adjectival u-forms are defined. That it, instead of having a single adjectival attribute `cost` that can be added to a u-form, there may be a list of UUID's in a single `cost` attribute that contains relations to several u-forms, each of which expresses the `price` that a particular object is offered for and under what conditions. For example, this enables a museum's price-list to express the cost of a adult admission, child admission, family admission and an annual membership admission. The phenomenal role played by the museum (for example, *Role for Service Offering*) needs only to know about this single `cost` attribute, instead of having to know about separate attributes for `adult_admission`, `child_admission`, etc.

The creation of several projective u-forms that are referred to by a single attribute of a phenomenal u-form is a desirable architecture, because it enables the schemata of phenomenal u-forms to remain agnostic as to how many different adjectival assertions can be made about them. It would be highly undesirable for the *Role for Service Offering* to give an attribute describing every way in which a service can be priced, because such a list would doubtless be messy, difficult to construct, and difficult to map items into unambiguously.

This concludes our survey of types of roles and their uses.

[3] The Civium Network is a non-profit consortium tasked with overseeing the development of the universal database architecture into a global Information Commons.

[4] `http://www.gutenberg.org`

5 Engineering and Linguistic Comparisons

This final section compares the architecture of u-forms, roles, and encoders with other similar architectures, and with certain linguistic and philosophical positions which we believe to be related.

5.1 Comparable Engineering Frameworks

Some readers may have already considered a resemblance between u-forms and XML. U-forms are certainly more similar to XML than to relational databases, at least along the dimension of extensibility. That is, any tags can be added to XML documents provided that they are syntactically well-formed (for example, a `<TEXT>` tag needs to be closed by a `</TEXT>` tag). Similarly, and attribute can be added to any u-form. However, the correspondence is not exact, because XML is largely *document centric* whereas u-forms are largely *information centric*. For example, XML encourages users to nest tags and create hierarchical structures within a single document, whereas u-forms encourage users to create nested structures comprising of several u-forms, by making use of the *Role for Collection*. Whereas the ability to refer to one another is built into u-forms, XML cannot make out-of-document references without using some external namespace, such as the URI namespace of the Semantic Web. In current systems, this still assumes that the information targeted by a URI reference can be found at a single location, which does not encourage the development of a distributed information network. However, modulo this important difference, the concepts of u-form / attribute / value and subject / property / object (see [8]) are analogous.

Given this analogy, there is also a rough correspondence between roles in the universal database architecture and the DTD's of the XML language. A DTD is a separate document used as metadata for interpreting an XML document, to check that the tags in the XML document conform to some semantic schema and behave as expected [7, Ch. 3].

A less well-known but much more closely similar system is the BibTex system used to typeset bibliographic references in TEXdocuments (including the references at the end of this paper). In BibTeX, each index entry is described by a list of attributes and values, and an identifier that is unique (at least, which is supposed to be unique within the scope of the local BibTeX file). For example, a typical BibTeX entry may look something like:

```
Book{einstein-relativity,
   author = {Albert Einstein},
   title = {Relativity: the Special and General Theory},
   publisher = {Holt and Company (English edition, 1920)},
   year = {1916},
   note = {Republished by Dover, 2001},
   }
```

In order to cite this work, the writer of a TEXdocument has simply to type "`\cite{einstein-relativity}`". When the document is typeset, this pulls the

entry out of the database, and typesets a formatted description in the "References" section at the end of the document, depending on the attributes contained in the BibTeX record and a program that reads and parses this record using a *bibliography style* file.

This is quite analogous to the system with u-forms, roles, and encoders. It is possible to add any attributes to a BibTeX record, but this does nothing to make sure that these attributes are typeset correctly unless a user is willing to conform to a well-known schema. Developers are free to create new bibliography styles, and publish new schema to enable people to use their style convention. (For example, several styles in recent years have been created or adapted to interpret a "`\url`" attribute.)

If I believed that I had found a completely new and improved way to represent bibliographic data, I would have to provide a package for interpreting and typesetting this data: convincing other BibTeX users that my new model is ontologically superior is not in itself going to convince anyone to *use* the model. At the same time, some parts of the traditional model (such as fields for `year` and `title`) have stood the test of time so well that they are recognized by *all* bibliographic style definitions.

At the same time, it is not necessary to claim that BibTeX developers have created an ontology as such, even though many of the attributes in a BibTeX database have semantic interpretations. What the community has achieved is more of an operational consensus: if information is presented in a certain way, it will be treated in a fashion desirable to the user. In designing the system of roles, u-forms and encoders, we are hoping to generalize this pattern of loosely organized collaboration to a much wider universal (but still only loosely organized) domain.

5.2 Linguistic and Philosophical Motivation

While many of the design patterns promoted in this paper are practical in nature. we believe that they are consistent with some underlying philosophical and considerations that are too often neglected when trying to create formal models for information.

Words in a natural language are not fixed in meaning, but adaptable to new contexts. There have been many traditional attempts by scholars to define the meanings of words and concepts, but in practice, these attempts never keep pace with the shifting consensus that enables speakers and listeners to understand one another. This point is argued in detail by [16, Ch 4], who devotes a whole chapter to words as "slippery customers." However, some of the ways in which words slip from one meaning to another are becoming increasingly well-understood. Ambiguity is no longer considered as an occasional aberration: it is the systematic extension of word-meanings that allows us to use words like *bank*, *university*, and *church* to refer to institutions and buildings / locations interchangeably. Similarly, many words can be interpreted as both a living thing and a foodstuff. Much of this research is associated with the idea of the *Generative Lexicon*, whose theory is developed in [17]. Polymorphism is not only natural in such a lexicon, it

is vital. Multiple schemata are not an aberration, they are part of the nature of human communication, and the unfortunate fixing of schemata is part of the brittleness of formal communication. We believe that this brittleness has been an unnecessary part of information engineering for far too long, and that the widespread opportunities for collaboration provided by today's communication infrastructure should enable people, and the marketplace, to evolve naturally flexible rather than breakable architectures.

Finally, we are tempted te recall Wittgenstein's famous criticism of feature-based classification, from the *Philosophical Investigations* [18, §66].

> Consider for example the proceedings that we call "games". I mean board-games, card-games, ball-games, Olympic games, and so on. What is common to them all? – Don't say: "There must be something common, or they would not be called 'games' " – but look and see whether there is anything common to all. – For if you look at them you will not see something that is common to all, but similarities, relationships, and a whole series of them at that. To repeat: don't think, but look! –
>
> Look for example at board-games, with their multifarious relationships. Now pass to card-games; here you find many correspondences with the first group, but many common features drop out, and others appear.
>
> When we pass next to ball-games, much that is common is retained, but much is lost.– Are they all 'amusing'? Compare chess with noughts and crosses. Or is there always winning and losing, or competition between players? Think of patience. In ball games there is winning and losing; but when a child throws his ball at the wall and catches it again, this feature has disappeared. Look at the parts played by skill and luck; and at the difference between skill in chess and skill in tennis.
>
> Think now of games like ring-a-ring-a-roses; here is the element of amusement, but how many other characteristic features have disappeared!

We are forced to agree with Wittgenstein that there are no features that are common to *all* games, that only games possess (though note that all of the games Wittgenstein mentions have well-recognized names and descriptions, which we would have no hesitation in describing using the *Role for Entity*). However, in addition to these, there are many attributes that are possessed by *some* games but not others. A game may perhaps have `participants`, involve `equipment`, or have `rules` and an `outcome`.

In using roles to provide a model for information, we are not concerned with finding attributes or features that are necessarily common to all games. We are, however, concerned with finding enough common attributes to justify the building of an encoder that knows how to display these attributes, or an artificial agent to reason about them. For example, it is easy to imagine an artificial agent that counts the number of `participants` wishing to play a card game, and to search a dataset of card games to recommend appropriate games for this number.

Our goal is not create complete and uniform information system: as Wittgenstein shows, this is impossible, because it is contrary to the nature of the world

we are modelling. However, we should no longer be building information systems that break when they lack complete information. Information systems that behave creatively, even with incomplete information, are already possible, and will thrive in the marketplace.

References

1. Stumme, G., Maedche, A.: FCA-MERGE: Bottom-up merging of ontologies. In: 17th International Joint Conference on Artificial Intelligence (IJCAI-01). (2001) 225–234
2. Gal, A., Modica, G., Jamil, H., Eyal, A.: Automatic ontology matching using application semantics. AI Magazine **26** (2005) 21–30
3. Lenat, D.B., Guha, R.V.: Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project. Addison-Wesley (1990)
4. Fellbaum, C., ed.: WordNet: An Electronic Lexical Database. MIT Press (1998)
5. Codd, E.F.: A relational model of data for large shared data banks. Comm. ACM **13** (1970) 377–387
6. Doan, A., Halevy, A.Y.: Semantic-integration research in the database community. AI Magazine **26** (2005) 83–94
7. Harold, E.R., Means, W.S.: XML in a Nutshell. 3rd edn. O'Reilly (2004)
8. Minola, F., Miller, E.: RDF primer. (2004)
9. Lucas, P., Senn, J.: Toward the Universal Database: U-forms and the VIA Repository. Technical Report MTR02001, MAYA Design (2002)
10. Dertouzos, M.: The Unfinished Revolution. Harper Collins (2001)
11. Lucas, P.: Mobile devices and mobile data: Issues of identity and reference. Human Computer Interaction **16** (2001) 323–336
12. Leach, P., Mealling, M., R.Salz: A UUID URN namespace. Technical report, The Internet Society (2004) Current draft, awaiting approval.
13. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer (1999)
14. Wolf, M., Wicksteed, C.: Date and time format. Technical report, World Wide Web Consortium (1997)
15. Hobbs, J.R., Ferguson, G., Allen, J., Fikes, R., Hayes, P., McDermott, D., Niles, I., Pease, A., Tate, A., Tyson, M., Waldinger, R.: A daml ontology of time. Technical report, DAML (2002) `http://www.cs.rochester.edu/~ferguson/daml/daml-time-nov2002.txt`.
16. Aitchison, J.: Words in the Mind: An Introduction to the Mental Lexicon. 3rd edn. Blackwell (2002)
17. Pustejovsky, J.: The Generative Lexicon. MIT press (1995)
18. Wittgenstein, L.: Philosophical Investigations. Blackwell, Blackwell (1953) 3rd edition, 2001.