Exploiting OWL Ontologies in the Multilingual Generation of Object Descriptions

Ion Androutsopoulos*, **Spyros Kallonis***+ and **Vangelis Karkaletsis**+

*Department of Informatics
Athens University of Economics and Business
Patission 76, GR-104 34 Athens, Greece

+Institute of Informatics and Telecommunications
National Centre for Scientific Research "Demokritos"
P.O. Box 60228, GR-153 10 Aghia Paraskevi, Greece

Abstract

We present three ways in which a natural language generator that produces textual descriptions of objects from symbolic information can exploit OWL ontologies, using M-PIRO's multilingual generation system as a concrete example.

1 Introduction

A strand of work in Natural Language Generation (NLG) has been devoted to the generation of textual descriptions of objects from symbolic information in ontologies and databases. An example of such work is ILEX [O'Donnell *et al.*, 2001], which was demonstrated mostly in the museums domain, where it could produce personalised English descriptions of exhibits; the Power system [Dale *et al.*, 1998] is another example from the same domain. More recently, the M-PIRO project [Isard *et al.*, 2003] developed a multilingual extension of ILEX, which has been tested in a variety of domains, including museum exhibits and items for sale.[1] A major problem in this and many other NLG subareas is the difficulty of obtaining source symbolic information in forms compatible with the requirements of the language generators. This issue has mainly been addressed so far by extracting source information from structured and semi-structured data [Dale *et al.*, 1998], and by developing authoring tools that help in the creation of source information and domain-dependent linguistic resources. Such tools were developed, for example, in GIST [Power and Cavallotto, 1996], DRAFTER [Hartley and Paris, 1997], ITRI's WYSIWYM systems [Van Deemter and Power, 2003], and M-PIRO [Androutsopoulos *et al.*, 2002].

In recent years, considerable effort has been invested in the Semantic Web, which can be seen as an attempt to develop mechanisms that will allow computer applications to reason more easily about the semantics of the resources (documents, services, etc.) of the Web. A major target is the development of standard representation formalisms, that will allow ontologies to be published on the Web and be shared by different computer applications. The emerging standard for specifying ontologies is OWL, an extension of RDF.[2] In NLG systems that describe objects, pre-existing OWL ontologies can provide much of the required source information, reducing the authoring effort and providing a common standard representation to generate from.[3] We discuss the role that OWL ontologies can play in M-PIRO's authoring process, and report on progress we made towards extending M-PIRO's authoring tool to support OWL. We argue that the benefit from using OWL would be greater, if the ontologies included the domain-dependent linguistic resources and user modelling information that NLG systems need. This would allow content to be published on the Sematic Web in the form of OWL ontologies, with different NLG engines acting as browsers responsible for rendering the content in different natural languages and tailoring it to the interests and interaction history of the users. A challenge for the NLG community, then, is to agree upon standards on how linguistic resources and user modelling information should be embedded in OWL ontologies.

Section 2 below introduces briefly M-PIRO and its authoring tool. Section 3 then shows how M-PIRO's ontologies can be expressed in OWL, and presents facilities we have added to the authoring tool to export ontologies in OWL. Among other benefits, this allows machine-generated texts to be published on the Web along with the ontology they were generated from, and to be annotated with OWL entries that express their semantics in terms of the ontology, making the semantics accessible to computer applications. Section 4 subsequently discusses how existing OWL ontologies can be imported into the authoring tool, and the benefits that this brings. Our import facilities currently support only a subset of OWL; part of section 4 is devoted to problems that remain to be solved. Section 5 focuses on the need to establish standards to embed linguistic resources and user modelling information in OWL ontologies, and how this would allow NLG engines to become the browsers of the Semantic Web. Section 6 concludes and summarises directions for future research.

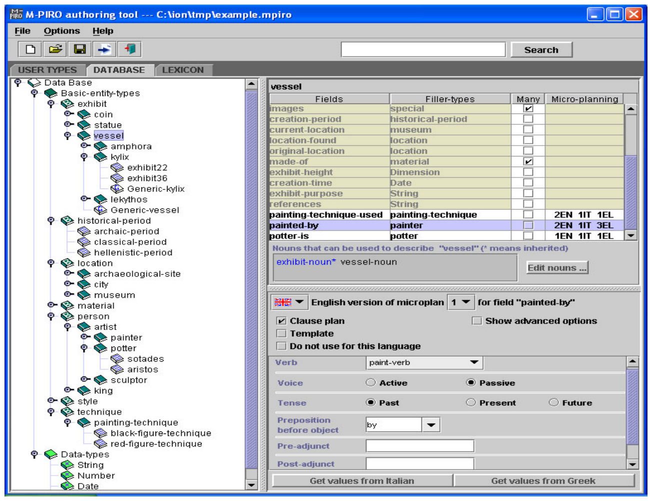
2 M-PIRO's authoring tool

M-PIRO's authoring tool allows *authors*, i.e., persons responsible for porting M-PIRO's technology to new application do-

[1] M-PIRO was an IST project of the European Union. It ran from 2000 to 2003. Its partners were: the University of Edinburgh, ITC-irst, NCSR "Demokritos", the National and Kapodistrian University of Athens, the Foundation of the Hellenic World, and System Simulation. This paper includes additional work, carried out at the Athens University of Economics and Business and NCSR "Demokritos".

[2] Consult http://www.w3.org/TR/owl-guide/.

[3] See also [Wilcock, 2003], [Bontcheva and Wilks, 2004].



vessel

Fields	Filler-types	Many	Micro-planning
images	special	✓	
creation-period	historical-period		
current-location	museum		
location-found	location		
original-location	location		
made-of	material	✓	
exhibit-height	Dimension		
creation-time	Date		
exhibit-purpose	String		
references	String		
painting-technique-used	painting-technique		2EN 1IT 1EL
painted-by	painter		2EN 1IT 3EL
potter-is	potter		1EN 1IT 1EL

Nouns that can be used to describe "vessel" (* means inherited)

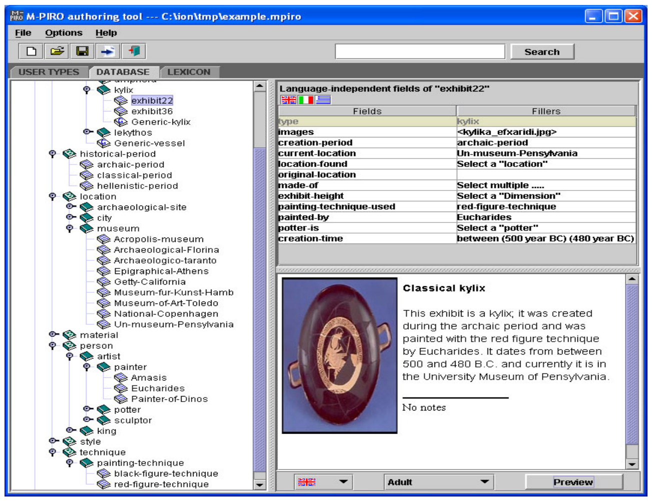
exhibit-noun* vessel-noun

English version of microplan 1 for field "painted-by"

[✓] Clause plan
[ ] Template
[ ] Do not use for this language
[ ] Show advanced options

Verb: paint-verb
Voice: Active [● Passive]
Tense: [● Past] Present Future
Preposition before object: by
Pre-adjunct:
Post-adjunct:

Figure 1: An M-PIRO ontology and a clause plan.



Language-independent fields of "exhibit22"

Fields	Fillers
type	kylix
images	<kylika_efxaridi.jpg>
creation-period	archaic-period
current-location	Un-museum-Pensylvania
location-found	Select a "location"
original-location	
made-of	Select multiple .....
exhibit-height	Select a "Dimension"
painting-technique-used	red-figure-technique
painted-by	Eucharides
potter-is	Select a "potter"
creation-time	between (500 year BC) (480 year BC)

Classical kylix

This exhibit is a kylix; it was created during the archaic period and was painted with the red figure technique by Eucharides. It dates from between 500 and 480 B.C. and currently it is in the University Museum of Pensylvania.

No notes

Figure 2: Source information and the resulting English text.

mains, to modify the domain-dependent resources: the ontology, some language resources, and the end-user stereotypes.

M-PIRO generates texts from an ontology that provides information on the *entities* of a domain (e.g., the statues and artists in a museum), the *relationships* between the entities (e.g., the association of statues with their artists), and the entities' *attributes* (e.g., their names or dimensions). Entities are not necessarily physical objects; they may be abstract concepts (e.g, historical periods). They are organized in a taxonomy of *entity types*, as illustrated in Figure 1, where 'exhibit' and 'historical-period' are *basic entity types*, i.e., they have no super-types. The 'exhibit' type is further subdivided into 'coin', 'statue', and 'vessel'. The latter has the sub-types 'amphora', 'kylix', and 'lekythos'. Each entity belongs to a particular type; e.g., 'exhibit22' belongs to 'kylix', and is, therefore, also a 'vessel' and an 'exhibit'. For simplicity, M-PIRO adopts single inheritance, i.e., a type may not have more than one parents, and an entity may not belong to more than one types.[4] This introduces some problems when importing OWL ontologies; related discussion follows.

Relationships are expressed using *fields*. It is possible to introduce new fields at any entity type, which then become available at all the entities of that type and its subtypes. In Figure 1, the fields 'painting-technique-used', 'painted-by', and 'potter-is' are introduced at the type 'vessel'. (The top right panel shows the fields of the type selected in the left panel.) Hence, all entities of type 'vessel' and its subtypes, i.e., 'amphora', 'kylix', and 'lekythos', carry these fields. Furthermore, entities of type 'vessel' inherit the fields 'creation-period', 'current-location', etc., up to 'references', which are introduced at the 'exhibit' type. (The 'images' field is used to associate images with entities.) The *fillers* of each field, i.e., the possible values, must be entities of a particular type. In Figure 1, the fillers of 'potter-is' are of type 'potter'; hence, the entities 'sotades' and 'aristos' are the only possible values. To represent that a particular 'vessel' entity was created during the classical period by 'aristos', one would fill in that entity's 'creation-period' with 'classical-period', and its 'potter-is' with 'aristos'. Figure 2 shows the fields of entity 'exhibit22', and the resulting English description. M-PIRO supports English, Greek, and Italian; descriptions can be generated in all three languages from the same ontology.

The 'Many' column in Figure 1 is used to mark fields whose values are *sets* of fillers of the specified type. In the 'made-of' field, this allows the value to be a set of materials (e.g., gold and silver). It is, thus, possible to represent many-to-one (e.g., only one material per exhibit) and many-to-many relationships (many materials per exhibit), but not one-to-one relationships (e.g., a unique social security code per person). OWL, in contrast, supports one-to-one relationships.

Fields are also used to represent *attributes* of entities (e.g., their names or dimensions). Several built-in data-types are available ('string', 'number', 'date', etc.), and they are used to specify the possible values of attribute-denoting fields. The 'Many' column also applies to attributes. In Figure 1, the values of 'references' and 'exhibit-purpose' are strings. The two fields are intended to hold canned texts containing bibliographic references and descriptions of what a particular exhibit was used for; e.g., "This statue honours the memory of Kroissos, a young man who died in battle". Information can be stored as canned text in string-valued fields when it is difficult to represent in symbolic form. The drawback is that canned texts have to be entered in all three languages.

The authoring tool also allows the authors to specify *user types*, i.e., types of end-users the texts are intended for (e.g., 'average-adult', 'child'), and *stereotypes*. The latter assign, for each user type, values to parameters that control, for example, the length of the texts, or the extent to which aggregating clauses to form longer sentences is allowed. The stereotypes also specify how interesting each field is for each user type; this allows the system to tailor the content of the descriptions to the users' interests. M-PIRO employs additional *personal user models*, where it stores the interaction history of each particular end-user, allowing, for example, the system

[4] M-PIRO's core language generator actually supports some forms of multiple inheritance, but the authoring tool does not.

to generate comparisons to previously seen objects.

M-PIRO uses systemic grammars, one for each language, to convert sentence specifications to surface text. The grammars can be used in a variety of object description applications without modifications, and, hence, can be treated as domain-independent for M-PIRO's purposes. However, a part of the lexicon that the grammars employ, known as the *domain-dependent lexicon*, has to be filled in by the authors when the system is ported to a new application. The domain-dependent lexicon contains entries for nouns and verbs; when moving to a new application, it is initially empty. The authors enter the base forms of the nouns and verbs they wish the system to use, and there are facilities to generate the other forms automatically. Noun entries are linked to entity types, to allow, for example, the system to generate referring noun phrases; in Figure 1, the entity type 'vessel' is associated with the lexicon entry 'vessel-noun' (see the area next to 'Edit nouns'). The entries are trilingual; e.g., 'vessel-noun' contains the nouns "vessel", "$\alpha\gamma\gamma\epsilon\acute{\iota}o$", and "vaso" of the three languages.

For each field and each language, the authors have to provide at least one *micro-plan*, that specifies how the field can be expressed as a clause in that language. Following ILEX, M-PIRO's primary form of micro-plans are *clause plans*, where the author specifies the clause to be generated in abstract terms, by selecting the verb to be used (from the domain-dependent lexicon), the voice and tense of the resulting clause, etc. As with nouns, verb-entries are trilingual; e.g., the 'paint-verb' entry of the clause plan of Figure 1 contains the base verb forms "paint", "$\zeta\omega\gamma\rho\alpha\phi\acute{\iota}\zeta\omega$", and "dipingere". By default, the entity that carries the field becomes the subject of the resulting clause, and the filler of the field the object. The clause plan of Figure 1 leads to clauses like "This vessel was painted by Eucharides". Appropriate referring expressions, e.g., "Eucharides", "a painter", "him", are generated automatically. Alternatively, micro-plans can be specified as simplistic *templates*, i.e., sequences of canned strings and automatically generated referring expressions; see [Androutsopoulos *et al.*, 2002] for details.

Unlike ILEX, M-PIRO allows multiple micro-plans to be specified per field, and this allows greater variety in the generated texts. Furthermore, the user stereotypes can be used to indicate that particular micro-plans are more appropriate to particular user types, and this allows the system to tailor the expressions it produces. When planning the text, M-PIRO attempts to place clauses that convey more interesting fields towards the beginning of the text. It is also possible for the authors to specify particular orderings; otherwise, M-PIRO's text planner is domain-independent.

3 Exporting M-PIRO ontologies to OWL

M-PIRO's ontological assumptions are very similar to those of OWL. As with M-PIRO, OWL assumes there are entity types, called *classes*, and entities, called *individuals*. M-PIRO's fields correspond to OWL's *properties*. Relationships between entities are expressed by defining *object properties*, that map entities to other entities, while attributes of entities are expressed via *datatype properties*, that map entities to literals of specific datatypes. It is, thus, relatively straightforward to export an M-PIRO ontology to OWL, as sketched below. There are actually three different versions of OWL, called OWL LITE, OWL DL, and OWL FULL, with increasing sophistication. The mapping from M-PIRO's ontologies to OWL produces ontologies in OWL LITE, which can be thought of as a subset of OWL DL and OWL FULL.

When exporting M-PIRO ontologies to OWL, entity types give rise to class definitions; e.g., the 'vessel' entity type of Figure 1 leads to the following OWL class:

```
<owl:Class rdf:ID="Vessel">
 <rdfs:subClassOf>
  <owl:Class rdf:about="#Exhibit" />
 </rdfs:subClassOf>
</owl:Class>
```

Fields are exported as OWL properties; e.g., the 'painted-by' field of Figure 1 leads to the following object property that associates vessels with painters:

```
<owl:ObjectProperty rdf:ID="painted-by">
 <rdfs:domain rdf:resource="#Vessel" />
 <rdfs:range rdf:resource="#Painter" />
</owl:ObjectProperty>
```

The 'exhibit-purpose' field of Figure 1 leads to the following datatype property, that associates exhibits with strings:

```
<owl:DatatypeProperty rdf:ID="exhibit-purpose">
 <rdfs:domain rdf:resource="#Exhibit" />
 <rdfs:range rdf:resource=
  "http://www.w3.org/2001/XMLSchema#string" />
</owl:DatatypeProperty>
```

Entities map to OWL individuals, as with statue 'exhibit42' below. String-valued fields, like 'exhibit-purpose', lead to properties with separate values per language.

```
<Statue rdf:ID="exhibit42">
 <current-location rdf:resource="#acropolis-museum" />
 <creation-period rdf:resource="#archaic-period" />
 <exhibit-purpose xml:lang="EN">This statue honours the
  memory of Kroissos, a... </exhibit-purpose>
 <exhibit-purpose xml:lang="IT">Questa...</exhibit-purpose>
 <exhibit-purpose xml:lang="GRC">...</exhibit-purpose>
 ...
</Statue>
```

One problem we have encountered is that OWL provides no mechanism to specify default values of properties. In M-PIRO, it is possible to introduce a *generic entity* per entity type, and the values of its fields are used as default values of all the entities in that type. For example, one could specify that kouroi, a kind of statue, were made in the archaic period, by introducing a 'generic-kouros' entity, similar to the 'generic-kylix' of Figure 1, and filling its 'creation-period' with 'archaic-period'. This would save us from having to specify the creation period of each individual kouros; their 'creation-period' fields would be left empty. It is also possible to override default information: to specify that a particular kouros was created during the classical period, perhaps the art of an eccentric classical sculptor, one would fill its 'creation-period' with 'classical-period', and this would licence texts like "Kouroi were created during the archaic period. However, this kouros was created during the classical period". We export generic entities as ordinary OWL individuals, but use a special prefix in their identifiers, which allows M-PIRO's system to assign them special status when reloading the ontology. Another system, however, that relies only on OWL's official semantics would have no way to realize that such individuals should be assigned special status.

A second problem is that some of M-PIRO's datatypes (e.g., dates) do not correspond exactly to OWL's recommended datatypes. We have defined new datatypes in OWL, using XML SCHEMA, that correspond exactly to M-PIRO's datatypes, and we currently use those in the exported ontologies instead of the recommended OWL datatypes. We hope to modify M-PIRO's datatypes to correspond exactly to the recommended ones in future versions of M-PIRO's system.

The mapping from M-PIRO ontologies to OWL that we sketched above has been fully implemented, and it now allows the authoring tool to export its ontologies in OWL. Apart from allowing other systems to reuse M-PIRO's ontologies, the mapping also opens up the possibility of generating object descriptions in both human-readable and machine readable forms. Every natural language description that M-PIRO produces can in principle also be rendered in a machine-readable form consisting of OWL individuals, this time using the mapping to translate into OWL the parts of the ontology that the system has decided to convey. For example, the English description of Figure 2 can be rendered in OWL as:

```
<Kylix rdf:ID="exhibit22">
 <creation-period rdf:resource="#archaic-period />
 <painting-technique-used
  rdf:resource="#red-figure-technique />
 <painted-by rdf:resource="#eucharides />
 ...
</Kylix>
```

M-PIRO's generator might have also included in the resulting text information deriving from the fields of the painter, e.g., the city the painter was born in, or other entities mentioned in the text. In that case, the OWL rendering of the description's content would include additional individuals, such as:

```
<Painter rdf:ID="eucharides">
 <painter-city rdf:resource="#athens" />
 ...
</Painter>
```

In the machine-readable forms of the descriptions, the OWL individuals would include only properties corresponding to fields the generator has decided to convey, unlike when exporting the full ontology. That is, the OWL individuals may not include properties corresponding to fields deemed uninteresting for the particular end-user, or fields that have already been conveyed; e.g., the painter's city may have already been conveyed when describing another work of the same artist.

It is thus possible to annotate the generated texts with OWL individuals representing their semantics. This would allow computer applications (e.g., Web agents visiting the site of a retailer that generates product descriptions using M-PIRO's technology) to reason about the semantics of the texts (e.g, locate items of interest). Alternatively, it is possible to define user types for both human users (e.g., 'expert', 'average-adult') and artificial agents acting for users of different interests and expertise (e.g., 'agent-expert', 'agent-average-adult'), and produce human-readable or machine-readable descriptions depending on the user type (in M-PIRO's demonstrators, there is a login stage where visitors select their types). The OWL ontology without its individuals (classes and properties only) can also be published on the Web to help the agents' developers figure out the structure and semantics of the OWL individuals their agents may encounter.

4 Importing OWL ontologies

When porting M-PIRO's system to a new domain, much of the authoring effort is devoted to defining entity types, and the fields that express attributes and relationships. This is a time-consuming process, partly because the ontology often has to be reshaped as more experience about the domain is gained. If a well-thought OWL ontology about the domain already exists, as will be the case with the gradual expansion of the Semantic Web, the authoring can be accelerated by importing the existing ontology into the authoring tool. Thereafter, the authors can focus on adding the necessary domain-dependent linguistic resources (micro-plans, lexicon entries, etc.), setting up the user stereotypes, and populating the ontology with entities that were not already present in the imported one. For the latter, we have developed software that allows the authoring tool to construct entities automatically from data in relational databases via ODBC; the authors only need to establish a mapping between the fields of the entity types and the attributes of the database's relations.

As already mentioned, there are three versions of OWL (OWL LITE, OWL DL, OWL FULL) with increasing sophistication. The mapping from M-PIRO's ontologies to OWL of the previous section uses only a subset of OWL LITE. Hence, importing an arbitrary OWL ontology, as opposed to an OWL ontology exported by the authoring tool, is not simply a matter of following the inverse mapping of the previous section. Below we highlight the problems that arise when importing arbitrary OWL LITE ontologies, to offer a taste of the work that remains to be carried out to make M-PIRO's system fully compatible with OWL LITE. We also point to some additional problems that arise when one moves on to OWL DL and OWL FULL. The discussion is based on experiments we conducted with more than a dozen of existing OWL ontologies.[5]

One of the main difficulties is that OWL (all versions) allows multiple inheritance, while M-PIRO does not (section 2). Importing an ontology with multiple inheritance currently causes the process to fail. The need for multiple inheritance has also been noted by authors, who often encounter cases where, for example, a person has to be categorized as both painter and potter. We hope to support multiple inheritance in future versions; this requires, among others, modifications in how the ontology is presented in the authoring tool.

Another problem is that OWL (all versions) supports property inheritance. For example, there may be a property 'is-player-of', used to represent the relationship between soccer players and their teams, and another property 'is-goalkeeper-of', that associates goalkeepers with their teams. The latter is a subproperty of the former, in the sense that if X is the goalkeeper of Y, then X is also a player of Y. The import facilities of the authoring tool currently ignore subproperty inheritance, because there is no corresponding notion in M-PIRO's ontologies; i.e., the two properties would be treated as unrelated. Subproperty inheritance, however, could help the generator avoid expressing information that follows from other information it has already conveyed; e.g., if a user has been told that X is the goalkeeper of Y, avoid saying that X is also a player of Y. We hope to extend M-PIRO's model

[5] See http://protege.stanford.edu/plugins/owl/ontologies.html.

with subproperty inheritance in future work.

A further complication is that OWL LITE allows the range of possible values of a property to be the intersection of several classes, while in M-PIRO's model the values of each field must come from a single, named entity type. A possible solution is to create automatically a new entity type in M-PIRO's ontology for each intersection in the OWL ontology, but this leads back to the single inheritance problem, because the intersection has to inherit from all the intersected types. This problem is more acute in OWL DL and OWL FULL, where several set operations (e.g., union, complement) between classes are allowed when specifying the ranges of properties.

In OWL it is also possible to refine a property's range. For example, an ontology may specify that individuals of the class 'product' have a property 'made-by', which associates them with individuals of the class 'manufacturer'; there would be an `rdfs:range` in the definition of 'product' setting the range of 'made-by' to 'manufacturer'. We may then wish to specify that individuals of 'automobile', a subclass of 'product', accept as values of 'made-by' only individuals of 'automobile-manufacturer', a subclass of 'manufacturer'. There are mechanisms in OWL (all versions) to state this (`allValuesFrom` tag), but there is no equivalent mechanism in M-PIRO's ontological model. We currently ignore range refinements when importing OWL ontologies, but this has the risk that authors may violate refinements (e.g., when adding individuals), creating ontologies that are no longer compatible with the imported ones.[6] Additional work is needed to support OWL's (all versions) `someValuesFrom`, which allows stating that in set-valued properties (cf. M-PIRO's 'Many' column) at least one of the elements of each set-value should belong to a particular class. A further mechanism in OWL DL and OWL FULL (`hasValue` tag) allows specifying that all the individuals of a class have a particular value at some of their properties; e.g., that all wines of class 'burgundy' have 'dry' taste. Such information can be imported into M-PIRO's generic entities (Section 3), though the correspondence is not exact, as generic entities carry default information that may be overridden.

As already pointed out (Section 1), M-PIRO does not allow relationships or attributes to be declared as one-to-one. In contrast, OWL (all versions) provides appropriate facilities, as well as facilities to declare properties (relationships or attributes) as transitive, symmetric, or the inverse of another one. All such declarations are currently ignored when importing OWL ontologies; again, this has the risk that the authors may modify the ontologies in ways that are incompatible with the ignored declarations. An additional problem in OWL FULL is that classes can be used as individuals, allowing the use of relationships to associate classes, as opposed to individuals; this violates M-PIRO's current ontological model.

It should be clear, then, that there are still issues to be resolved in M-PIRO's ontological assumptions to make M-PIRO fully compatible with OWL LITE, and there are additional difficulties with OWL DL and OWL FULL. As discussed above, however, most of the necessary improvements appear to be within reach, at least for OWL LITE. Overall, it appears reasonable to conclude that future versions of NLG systems like M-PIRO's will be able to exploit fully OWL ontologies.

[6]ILEX and M-PIRO's core generation engine provide some support for such refinements, but M-PIRO's authoring tool does not.

5 Towards semantic browsers

We have so far proposed two ways in which OWL ontologies can be exploited in systems like M-PIRO's: first, the generated texts can be accompanied by OWL specifications of their semantics, with an OWL ontology establishing the semantic vocabulary; and, second, existing OWL ontologies can be imported, to accelerate the authoring. In both cases, the ontologies are linked to domain-dependent language resources (micro-plans, lexicon entries, etc.) and user stereotypes (the interest of each field per user type, etc.), but these additional resources are not parts of the OWL ontologies: when exporting M-PIRO ontologies to OWL, the authoring tool produces additional proprietary XML files that contain the domain-dependent language resources and stereotypes; and when importing OWL ontologies developed by others, the additional resources have to be filled in by the authors. We argue below that agreeing upon standards on how the additional resources could be embedded in OWL ontologies would allow NLG systems like M-PIRO to play a central role in the Semantic Web.

Note, first, that it is possible to represent in OWL M-PIRO's domain-dependent linguistic resources and user stereotypes. For example, micro-plans could be treated as individuals of a class 'Microplan' with subclasses 'ClausePlan' and 'Template'. In a similar manner, there would be a class 'Voice' with individuals 'active' and 'passive', and similarly for tenses, genders, supported languages, etc. There would also be a class 'LexiconEntry' with subclasses 'VerbEntry' and 'NounEntry', and individuals corresponding to the entries of the domain-dependent lexicon. (Classes corresponding to language resources could be grouped under a 'LinguisticResource' super-class.) Then, for example, the English micro-plan of Figure 1 would roughly be represented in OWL as:

```
<ClausePlan rdf:ID="painted-by-mp1-en">
 <for-property rdf:resource="#painted-by" />
 <for-language rdf:resource="#english" />
 <use-verb rdf:resource="#paint-verb" />
 <use-voice rdf:resource="#passive" />
 <use-tense rdf:resource="#past" />
 <use-preposition>by</use-preposition>
</ClausePlan>
```

Similarly, the English part of the trilingual lexicon entry 'vessel-noun' could roughly be represented in OWL as:

```
<NounEntry rdf:ID="vessel-noun-en">
 <lexicon-entry-id>vessel-noun</lexicon-entry>
 <for-language rdf:resource="#english" />
 <refers-to-class rdf:resource="#vessel" />
 <base-form>vase</base-form />
 <has-gender rdf:resource="#neuter" />
 ...
</NounEntry>
```

One complication is that we need to establish mappings from micro-plans to the properties (fields) they can express, and this requires using property names as values of other properties. This can be seen in the micro-plan above, where we used the property (field) name 'painted-by' as the value of property 'for-property' to signal that the micro-plan can express 'painted-by'. Using property names as values of properties, however, requires OWL FULL. There is a similar problem with noun entries, which have to be associated with classes (entity types) they can refer to: in the noun entry above, we used the class name ‘vessel’ as the value of property ‘refers-to-class’. Using class names as values of properties again requires OWL FULL. Similar problems arise with stereotypes.

We are currently exploring how M-PIRO’s domain-dependent language resources and stereotypes can be best embedded in OWL ontologies. This embedding will lead to ‘language-enabled’ ontologies, that will include all the resources a system like M-PIRO needs to render the ontologies in several natural languages. This opens up another possibility for publishing content on the Semantic Web: a site could publish only its language-enabled ontology (including the individuals that correspond, for example, to the items it sells), and the NLG technology to render the ontology in natural language could take the form of a browser plug-in. When visiting a site of this kind, a human user would be initially presented with an inventory of objects that can be described (e.g., product thumbnails). Selecting an object would transmit to the browser the ontology or its relevant parts, and it would be the responsibility of the NLG plug-in to produce an appropriate description in the user’s language and tailor it to the user’s type and interaction history. If the NLG community could establish standards for language-enabled ontologies, there could be different NLG plug-ins by different makers, perhaps each specialising in particular languages and user types, in the same way that there are different browsers for HTML. There could also be a market for developers of language-enabled ontologies for particular sectors (e.g., museums, retailers of computer equipment), who would sell their ontologies to organisations wishing to publish content in those sectors. The client organisations would only need to populate the ontologies with their own individuals (e.g., exhibits, products), possibly by reusing databases, and publish them at their sites. Artificial agents would interact directly with the ontologies of the various sites, invoking their own NLG plug-ins to report their findings in natural language.

Establishing standards is, of course, far from trivial. For example, different NLG systems may require very different domain-dependent language resources, or make different assumptions on which resources are domain-dependent or independent. Nevertheless, we believe it is worth trying to move towards this direction, as there are large potential gains for both the NLG community and the users of the emerging Semantic Web. Furthermore, the effort to establish standards should proceed in cooperation with other fields that could exploit language-enabled ontologies. For example, the association between entity types and noun entries can be used for query expansion in information retrieval; and the association between micro-plans and ontology fields can be useful in information extraction systems that populate ontologies.

6 Conclusions and future work

We have presented three ways in which an NLG system that generates object descriptions from symbolic information can exploit OWL ontologies, using M-PIRO’s system as an example. First, the NLG system’s source symbolic information can be exported in the form of an OWL ontology. Apart from enabling other OWL-aware systems to reuse the source information, this allows the generated texts to be accompanied by OWL descriptions of their semantics, with the OWL ontology establishing the semantic vocabulary. Thus, the semantics of the generated texts become fully accessible to computer applications, such as Web agents. Second, when porting the NLG system to a new domain, it is possible to import a pre-existing OWL ontology, saving a significant amount of effort. Third, it is possible to embed in OWL ontologies all the domain-dependent language resources and user modelling information that NLG systems like M-PIRO’s need. This would allow Web sites that carry information about objects to publish their content solely in the form of OWL ontologies, passing the responsibility of generating natural language descriptions to NLG browser plug-ins. The latter requires the NLG community to develop appropriate standards.

We hope that future work will address the remaining incompatibilities between M-PIRO’s technology and OWL. We also plan to explore more fully how NLG engines could become central components of the Semantic Web’s browsers, and release prototypes that will demonstrate these ideas.

References

[Androutsopoulos *et al.*, 2002] I. Androutsopoulos, D. Spiliotopoulos, K. Stamatakis, A. Dimitromanolaki, V. Karkaletsis, and C.D. Spyropoulos. Symbolic authoring for multilingual natural language generation. In *Methods and Applications of Artificial Intelligence*, number 2308 in Lecture Notes in Artificial Intelligence, pages 131–142. Springer, 2002.

[Bontcheva and Wilks, 2004] K. Bontcheva and Y. Wilks. Automatic report generation from ontologies: the MIAKT approach. In *Proc. of the 9th International Conference on Applications of Natural Language to Information Systems*, pages 324–335, Manchester, U.K., 2004.

[Dale *et al.*, 1998] R. Dale, S.J. Green, M. Milosavljevic, C. Paris, C. Verspoor, and S. Williams. Dynamic document delivery: generating natural language texts on demand. In *Proc. of the 9th International Conference and Workshop on Database and Expert Systems Applications*, pages 131–136, Vienna, Austria, 1998.

[Hartley and Paris, 1997] A. Hartley and C. Paris. Multilingual document production – from support for translating to support for authoring. *Machine Translation*, 12(1–2):109–129, 1997.

[Isard *et al.*, 2003] A. Isard, J. Oberlander, I. Androutsopoulos, and C. Matheson. Speaking the users’ languages. *IEEE Intelligent Systems*, 18(1):40–45, 2003.

[O’Donnell *et al.*, 2001] M. O’Donnell, C. Mellish, J. Oberlander, and A. Knott. ILEX: an architecture for a dynamic hypertext generation system. *Natural Language Engineering*, 7(3):225–250, 2001.

[Power and Cavallotto, 1996] R. Power and N. Cavallotto. Multilingual generation of administrative forms. In *Proc. of the 8th International Workshop on Natural Language Generation*, pages 17–19, Herstmonceux Castle, U.K., 1996.

[Van Deemter and Power, 2003] K. Van Deemter and R. Power. High-level authoring of illustrated documents. *Natural Language Engineering*, 9(2):101–126, 2003.

[Wilcock, 2003] G. Wilcock. Talking OWLs: towards an ontology verbalizer. In *Proc. of the workshop “Human Language Technology for the Semantic Web”, 2nd International Semantic Web Conference*, pages 109–112, Sanibel Island, Florida, U.S.A., 2003.