Natural Language Engineering **13** (3): 191–233. © 2006 Cambridge University Press
doi:10.1017/S1351324906004268 First published online 19 June 2006 Printed in the United Kingdom

Source authoring for multilingual generation of personalised object descriptions

I. ANDROUTSOPOULOS
Department of Informatics, Athens University of Economics and Business, Patission 76, 104 34 Athens, Greece

J. OBERLANDER
School of Informatics, University of Edinburgh, 2 Buccleuch Pl., Edinburgh EH8 9LW, U.K.

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

(*Received 2 November 2004; revised 29 December 2005*)

Abstract

We present the source authoring facilities of a natural language generation system that produces personalised descriptions of objects in multiple natural languages starting from language-independent symbolic information in ontologies and databases as well as pieces of canned text. The system has been tested in applications ranging from museum exhibitions to presentations of computer equipment for sale. We discuss the architecture of the overall system, the resources that the authors manipulate, the functionality of the authoring facilities, the system's personalisation mechanisms, and how they relate to source authoring. A usability evaluation of the authoring facilities is also presented, followed by more recent work on reusing information extracted from existing databases and documents, and supporting the OWL ontology specification language.

1 Introduction

The ability to produce texts in several languages from a single, language-neutral symbolic source is one of the main advantages of natural language generation, as it can reduce dramatically translation costs; see, for example, Hartley and Paris (1997) for related discussion, and Reiter and Dale (2000) for an introduction to the field. Even in the monolingual case, natural language generation can lead to texts of higher quality, compared to simplistic methods that employ inventories of predetermined texts with slots to be filled in, and it allows the resulting texts to vary dynamically, depending on the reader's profile and interaction history, as demonstrated, for example, by Coch (1996) and O'Donnell *et al.* (2001).

This paper is based on experience gained from developing a natural language generation system that produces descriptions of objects in multiple languages from

language-neutral symbolic information and pieces of canned text. The symbolic information is drawn from an ontology and a database that provides information on the ontology's instances; the pieces of canned text are also stored in the database. The system has been tested in Web-based and virtual reality applications, ranging from descriptions of museum exhibits to presentations of computer equipment for sale. The underlying technology was developed in the M-PIRO project (Isard *et al.* 2003; Calder *et al.* 2005), using the ILEX system (O'Donnell *et al.* 2001) as a starting point.[1] We believe it is fair to say that prior to M-PIRO, ILEX reflected the state of the art in the sub-area of natural language generation that is concerned with producing personalised descriptions of objects. Work on ILEX, however, had focused mostly on the generation of English descriptions of museum exhibits for Web-based interaction.[2] In contrast, M-PIRO targeted *multilingual* generation, which required a careful separation of language-specific processes and resources from language-independent ones, along with facilities to keep the generation capabilities across the supported languages aligned; the system currently supports English, Italian, and Greek. ILEX's personalisation mechanisms were also extended, as will be explained in the following sections, and the new system is much easier to port to new application domains where object descriptions are needed.

In this paper, we focus on the source authoring facilities that were developed in M-PIRO, which allow people with no previous experience in natural language generation, hereafter called *authors*, to configure the system for new application domains. The configuration involves defining the domain's ontology, populating the database with information on the ontology's instances, creating domain-dependent linguistic resources, linking them to the ontology, and adjusting parameters related to user modelling. We will concentrate on the case where the authors have a computer science background. Usability evaluation results indicate that after receiving a relatively short introductory course, third-year computer science undergraduates with no previous experience in natural language generation can successfully configure the system for new applications without major problems. We believe that the expertise of our evaluation's subjects is a good lower boundary of the expertise of the employees that would be assigned the task of configuring M-PIRO for e-commerce applications (e.g., generating descriptions of items for sale) in a corporate environment, and, hence, the results are encouraging. Furthermore, feedback from the project's partners indicates that in a museum context M-PIRO's authoring process would most likely be assigned to curators with a cultural informatics background, or to computer scientists who would interact with curators to obtain the information to be presented at each

[1] M-PIRO (Multilingual Personalised Information Objects) was a project of the Information Societies Programme of the European Union. The project 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 Ltd. This paper includes additional work on M-PIRO's authoring facilities, carried out at the Athens University of Economics and Business.

[2] See also Dale *et al.* (1998) for information on a similar generation system for museums.

exhibit. Hence, we believe that our usability evaluation will also be of interest to readers involved in museum applications.

M-PIRO's authoring facilities are not intended to be used by natural language processing experts to create or modify large-scale domain-independent linguistic resources, such as grammars, unlike, for example, the user interface of the KPML generation engine (Bateman 1997). Thus, M-PIRO's notion of authoring is closer to that of DRAFTER (Paris *et al.* 1995; Paris and Vander Linden 1996; Hartley and Paris 1997) and AGILE (Hartley *et al.* 2001), two generation systems that provide authoring facilities to enter symbolic source knowledge, from which software manuals are generated in multiple languages. Those systems, however, aim at the generation of instructional texts, and, hence, their source knowledge representations and some of the generation techniques they employ are significantly different from M-PIRO's, and this is reflected on the functionality of the respective authoring facilities. Similar comments apply to the authoring mechanisms of the GIST system (Power and Cavallotto 1996), which aims at the multilingual generation of texts describing administrative (e.g., form-filling) procedures, and the semantic editor of Biller *et al.* (2005), which is used in a multilingual generator that produces cooking recipes. In terms of genre of the generated texts, M-PIRO is closer to the system of Brun *et al.* (2000), which generates drug descriptions in multiple languages.

The facilities of the systems mentioned above allow the authors to create instances of pre-existing concepts, but not new concepts. That is, the authors can manipulate the population of a pre-defined ontology, but not the ontology itself, and this limits their ability to port the systems to new domains. Furthermore, the authors have no means to edit the systems' domain-dependent linguistic resources, such as the domain-dependent parts of the lexicons. In contrast, M-PIRO's authors have full control over the domain's ontology, they can edit all the domain-dependent linguistic resources, and they can also tune user modelling parameters that affect the content and form of the generated texts. In these respects, M-PIRO is closer to ISOLDE (Paris *et al.* 2002; Colineau *et al.* 2002), which allows the authors to define new concepts, not just instances, to manipulate domain-dependent lexicon entries, and to adjust some parameters that affect the style of the generated texts. As with DRAFTER and AGILE, however, ISOLDE aims at the generation of instructional texts for manuals, as opposed to M-PIRO whose target is descriptions of objects.

M-PIRO's authors manipulate mainly symbolic representations, but previews of the resulting texts provide natural language feedback. As we explain in the following sections, it is possible to envisage extensions where M-PIRO's authors would interact more directly with natural language renderings of the semantic representations they manipulate, moving towards the WYSIWYM approach (Power and Scott 1998; Van Deemter and Power 2003). Another approach is to require the authors to enter text snippets in multiple versions for different reader types, annotated with information such as rhetorical relations between the snippets, co-references, and criteria that have to be satisfied for the snippets to be included in the generated texts. This type of authoring is used in the HEALTHDOC system (Hirst *et al.* 1997; DiMarco and

Foster 1997) and the macronodes approach (Not and Zancanaro 2000).[3] It has the disadvantage, however, that the snippets have to be entered and be annotated in all of the supported languages and in multiple versions, which becomes impractical when generating in several languages and for several types of readers.

M-PIRO's technology will often have to be ported to application domains where a large volume of information is already available in databases (e.g., museum databases, databases containing information on products) or where established (e.g., medical) ontologies exist. M-PIRO allows data to be imported from existing databases, with special support provided for relational databases. It is also possible to export M-PIRO's ontologies to OWL, an ontology specification language designed for the Semantic Web, and there is partial support to import existing OWL ontologies.[4] Furthermore, we have experimented with an approach where M-PIRO's ontology is populated with information extracted from existing documents (e.g., Web pages, product catalogues), using information extraction techniques. In these respects, M-PIRO follows the example of ISOLDE, which has demonstrated in the context of software manuals that it is possible to extract knowledge from, among other sources, existing UML software models and documentation texts.

To summarise this section, M-PIRO improves upon ILEX, which so far reflected the state of the art in generating personalised object descriptions, by adding support for multiple languages, extended personalisation mechanisms, and authoring facilities. This paper focuses on the latter, but we also discuss issues where multilingualism and personalisation interact with the authoring process. Previously presented authoring facilities for other natural language generation systems differ from M-PIRO's in that they either target different text genres or provide less functionality.

The remainder of this paper is structured as follows. Section 2 describes the system's overall architecture and highlights the possible applications of M-PIRO's technology. Section 3 discusses the domain-dependent resources the authors manipulate and the functionality of the authoring facilities. It also presents M-PIRO's personalisation mechanisms and their relation to the authoring process. Section 4 reports on the usability evaluation of the authoring facilities. Section 5 moves on to more recent work, discussing how existing databases and document collections can be exploited, followed by Section 6, where we discuss issues related to supporting OWL. Section 7 concludes and proposes directions for future work.

2 System architecture and applications

Let us first get a better view of the possible applications of M-PIRO's technology. We will be using the term *visitor* to refer to the end-users the descriptions are generated for. In the Web-based system of figure 1, one of M-PIRO's demonstrators, a visitor can request information on a particular museum exhibit by clicking on the exhibit's icon

[3] M-PIRO investigated to some extent ways to integrate macronodes with full natural language generation, but we do not discuss this issue here; see Calder *et al.* (2005).

[4] Consult `http://www.w3.org/TR/owl-guide/` for information on OWL.

Fig. 1. A Web-based public access system for a museum that uses M-PIRO's technology.

in one of the available electronic showcases; alternatively, it is possible to locate an exhibit by typing its name in the search box, or follow one of the routes around the exhibits that the museum suggests.[5] Once an exhibit has been selected, the system produces a Web page containing a picture and a dynamically generated description of the exhibit, as shown in figure 1. As in ILEX, each description ends with pointers to other related exhibits, such as exhibits from the same historical period or works of the same artist; we call these *forward pointers*. The entire object description of figure 1, including the forward pointers, was generated dynamically, without using any canned text.

M-PIRO extends ILEX's personalisation mechanisms in several ways. Most importantly, *visitor stereotypes*, i.e., settings that are sensitive to the visitor's type, are extended to allow the system to tailor the language expressions of the generated descriptions, not just their semantic content, to visitor type; for instance, children can be served with simpler sentence structures. *Personal models*, i.e., models of individual visitors, are also augmented to record, apart from the semantic content that has been conveyed to each visitor, information on the language expressions that were used, allowing the system to reduce repetitions of the same expressions, which remained an issue with ILEX. The personal models are also made persistent over multiple sessions via a *personalisation server*, which stores both the personal models and the visitor stereotypes.[6] In the demonstrator of figure 1, an initial log-in

[5] The demonstrator of figure 1 was developed by System Simulation Ltd. and the Foundation of the Hellenic World, using content from the Foundation's electronic collections.

[6] The personalisation server was developed by NCSR "Demokritos", and has been used in several different systems. It is implemented in Java as a Web server, that communicates

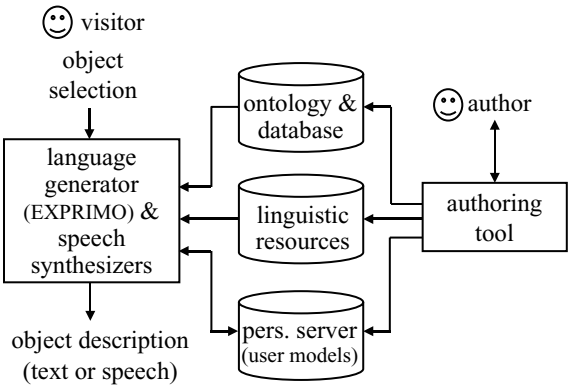


Fig. 2. Overview of M-PIRO's architecture.

screen associates each visitor with a unique identifier, allowing the visitor's personal model to be retrieved from the personalisation server in subsequent visits. When they log-in for the first time, visitors also select the language they prefer and their type (e.g., child, adult, expert); these preferences are stored in their personal models, and can be changed at any time.

In a second, virtual reality M-PIRO demonstrator, there is a similar log-in stage, and visitors can examine three-dimensional reconstructions of ancient vessels. In this case, M-PIRO's technology produces a personalised description of the vessel being examined, which is passed on to a speech synthesizer for the selected language.[7] The generated texts are actually marked up in APML (de Carolis *et al.* 2004), a form of XML designed to couple speech synthesis and character animation.[8] The additional markup reflects information that affects prosody and is known to the generator, such as syntactic structures that are needed to compute prosodic boundaries, and new vs. given semantic information; consult Pan and McKeown (1997) and Theune *et al.* (2001) for examples of related work. A similar form of M-PIRO's technology could also be used in virtual reality reconstructions of ancient cities, or applications where information on the landmarks of a city is sent to the mobile phones or PDAs of tourists; see, for example, Staab *et al.* (2002).

Figure 2 provides an overview of M-PIRO's architecture. When a visitor requests a description of an object, the identifiers of the visitor and the object are passed on to the natural language generator. M-PIRO's language generator, called EXPRIMO, retrieves information from the ontology and database (e.g., facts about the exhibit) and the personalisation server (e.g., preferences of the visitor, interaction history),

with its clients via an HTTP-based protocol; see `http://www.iit.demokritos.gr/skel/en/Projects/PServer.htm`. All the other components of M-PIRO's system are also implemented in Java.

[7] M-PIRO's virtual reality demonstrator was developed by the Foundation of the Hellenic World. It uses Festival (Clark *et al.* 2004) for English and Italian speech synthesis, and DEMOSTHENES (Xydas and Kouroupetroglou 2001) for Greek synthesis.

[8] The APML language was mainly developed in the European IST MagiCster project; consult `http://www.ltg.ed.ac.uk/magicster/`.

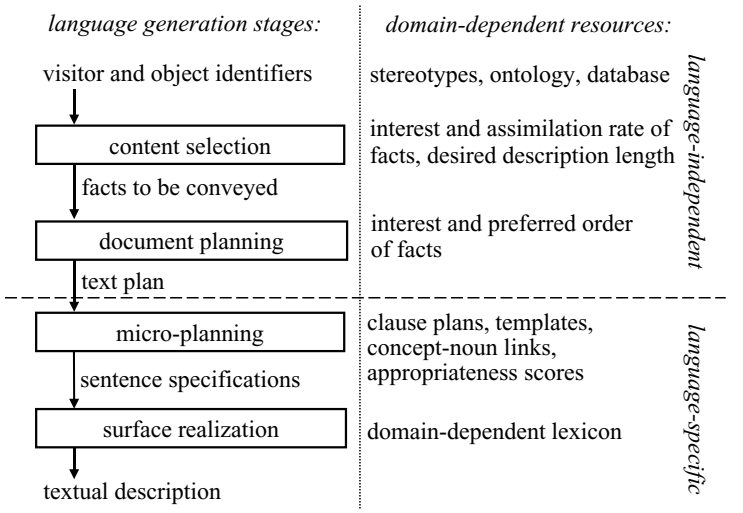


Fig. 3. Generation stages in EXPRIMO and domain-dependent resources.

and produces a description in textual form using both language-independent and language-specific linguistic resources. If spoken output is required, the description is then passed on to a speech synthesizer.

Any information that pertains to particular visitors is kept separately from EXPRIMO in the personalisation server. It is, thus, possible to send to the same EXPRIMO instance a stream of requests to produce descriptions for different visitors, a situation that arises in applications with concurrent visitors. In applications with large numbers of concurrent visitors, it is also possible to use a farm of EXPRIMO servers, with each incoming request being directed to the first available server. In this approach, which was used in the Web-based demonstrator of figure 1, all EXPRIMO servers are equipped with copies of the same application ontology, database, and linguistic resources, and share a common personalisation server. The same farming approach can be used with speech synthesizers.

M-PIRO's authoring facilities, collectively known as the *authoring tool*, allow the authors to configure the domain-dependent resources of the system, namely the application ontology and database, the domain-dependent linguistic resources, and the visitor stereotypes of the personalisation server. Figure 3 shows EXPRIMO's main processing stages and the domain-dependent resources that are used at each stage, i.e., the resources that the authors have to configure; for simplicity, domain-independent resources are not shown. EXPRIMO's architecture is based on a typical generation pipeline, but with four main stages, rather than three (Reiter 1994) or two (Thompson 1977), much as in Reiter and Dale (2000). To support multilinguality and domain portability, the processing stages and resources of the pipeline were divided into domain-independent vs. domain-dependent, and language-independent vs. language-specific ones. There is a tradeoff between what can be generated and how easily a system can be ported to new languages and application domains. To be able to generate perfect natural text in any language and application domain, one

may well have to treat *all* the generation stages and resources as language-specific and domain-dependent, and also abandon the pipeline architecture in favour of a more complex, fully connected one. However, this makes it more difficult to develop practical systems, and port them across languages and domains. In the context of generating object descriptions, the architecture of figure 3 produces texts of reasonable quality, requiring an acceptable amount of effort from the authors and the language engineers who may need to add support for additional languages.

The first processing stage, *content selection*, is concerned with the selection from the ontology and database of facts to be conveyed to the visitor. The selected facts must be both *relevant* to the selected object and *appropriate* for the particular visitor. By relevant we mean that the facts must refer to the selected object either directly (e.g., they may specify the creator of the object, or the historical period the object was created in) or indirectly (e.g., specify the nationality of the object's creator, or provide background information about the historical period the object was created in). By appropriate we mean that the selected facts must not repeat previously conveyed information that we believe the visitor has assimilated (i.e., facts that have become part of the visitor's knowledge by having been conveyed one or more times), and they must match the visitor's presumed interests; for example, one should avoid including references to scientific articles when describing museum exhibits to children. Furthermore, the desired size of the description limits the number of facts that can be selected. Hence, content selection depends on: the application's ontology and database; the visitor stereotypes, which specify among other things the desired description length per visitor type, the interest of the various facts, and how difficult it is to assimilate them (how many times they have to be conveyed before we can assume they are part of the visitor's knowledge); and the personal models of the visitors, which show, among other things, their interaction history and which facts the system believes they have assimilated.

While the content selection process is directly based on that in ILEX (O'Donnell *et al.* 2001), the second stage, *document planning*, is a significant simplification. It outputs an overall document structure, which specifies the sequence of the facts in the description to be generated. Unlike ILEX, there is no attempt to deal with the rhetorical relations between facts (Mann and Thompson 1988) – for example, whether a fact contrasts with another. EXPRIMO's document planner is largely domain-independent. It consults the personal models to obtain information on whether the various facts can be treated as partly known or entirely new information, and the relative interest of the facts, attempting to place the most interesting facts at the start of the description. The preferred fact order can also be specified by the author, as will be discussed later on.

For each fact in the document plan, *micro-planning* then specifies in abstract terms how the fact can be expressed as a clause in the selected language; for example, which verb to use, in what tense, and which argument of the fact to use as subject or object. The authoring tool allows this information to be specified in the form of *micro-plans*, of which there are two types, *clause plans* and *templates*, to be discussed below. The visitor stereotypes associate each micro-plan with an appropriateness score per visitor type; these scores may lead the system to prefer, for example,

a clause with the verb 'to show', as in "It shows Perikles.", when generating for children, instead of 'to depict', which may be more appropriate for adults. Micro-planning also includes the generation of referring expressions, and processing that determines which clauses can be aggregated in single sentences (Melengoglou 2002). Both rely on domain-independent algorithms. However, the algorithm for referring expressions, which is the same as that of ILEX, requires the authors to establish links between the concepts in the application's ontology and the nouns that can be used to refer to them (e.g., a link showing that the Italian noun "statua" can be used with statues); we discuss this issue further later on.

The last stage, *surface realization*, generates the final textual form of the descriptions. It includes the generation of appropriate word forms (e.g., verb tenses) based on the sentence specifications output by micro-planning, placing the various constituents (e.g., subject, verb, object, adverbials) in the correct order, accounting for number and gender agreement, etc. M-PIRO's surface realization is based on large-scale systemic grammars (Halliday 1994). The Greek (Dimitromanolaki *et al.* 2001) and Italian grammars parallel ILEX's English grammar, which was based on WAG (O'Donnell 1996). Following Bateman (1997), the three grammars share parts for common linguistic mechanisms, which allows faster development and easier maintenance.[9] While the grammars are domain-independent, a part of the lexicon they employ, the *domain-dependent lexicon*, has to be tuned by the authors when the system is ported to a new domain.

Note that EXPRIMO can generate a description for any entity in its database (e.g., artists or historical periods), not just entities that correspond to exhibits. In the demonstrator of figure 1, the hyperlinks correspond to entities. Clicking on "the classical period" generates a description of that period, which appears in a pop-up window. As with other descriptions, the period's description is generated dynamically, and it is tailored to the visitor's type and interaction history.

3 Source authoring and user modelling

We now take a closer look at M-PIRO's authoring tool and the domain-dependent resources the authors manipulate. We examine at the same time M-PIRO's personalisation mechanisms and how they relate to the authoring process. As already noted, we assume that the authors have a computer science background, but no previous experience in natural language generation. Nevertheless, we assume that the authors will attend a short training course on the functionality of the tool before attempting to use it. Usability evaluation results that will be presented in the following sections indicate that the training course could fit in one working day.

3.1 Ontology and database

M-PIRO's database contains information about *entities*, like statues and artists, *relationships* between entities, such as the relationship that associates each statue

[9] See Hartley *et al.* (2001) for a discussion of how the same approach was used to produce grammars for Bulgarian, Czech, and Russian from a broad coverage English grammar.

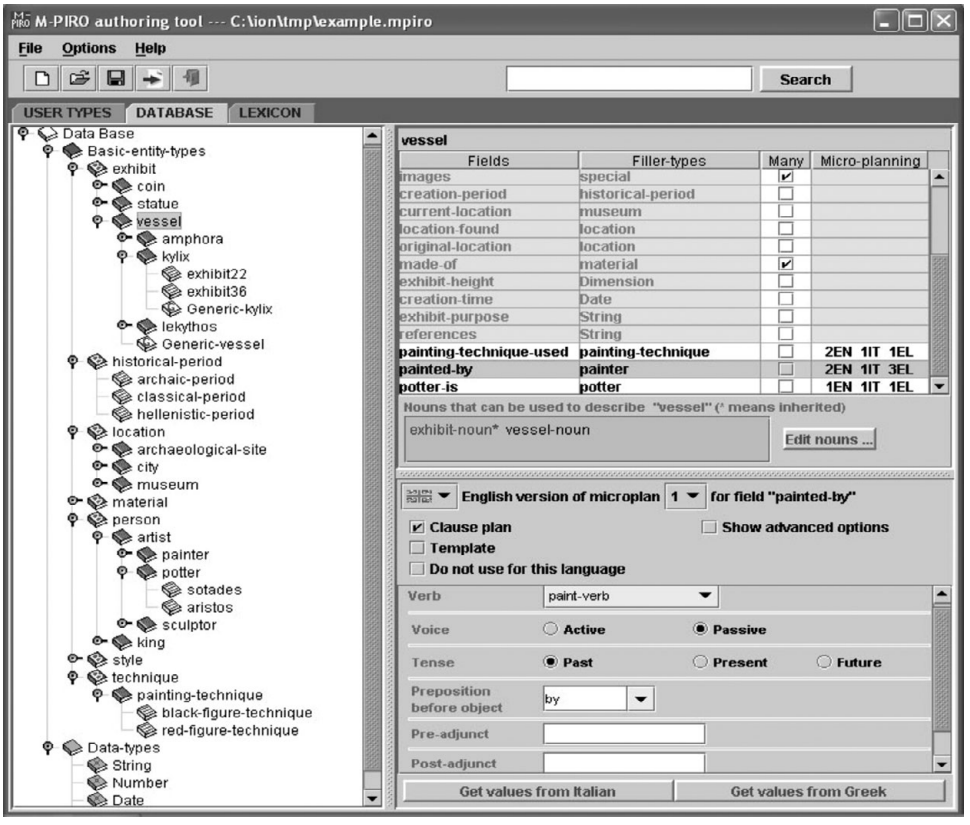


Fig. 4. The author's view of the ontology and database, and a clause plan.

with the artist that created it, and *attributes* of entities, for instance their names or dimensions. These notions are common in natural language generation, and also in the design of databases, where the entity-relationship model is often used.

Entities are not necessarily physical objects; they may be abstract concepts, like historical periods or painting techniques. They are organized in a taxonomy of *entity types*, as illustrated in the left panel of figure 4. In this example, '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 entity type; for example, 'exhibit22' belongs to the 'kylix' type, and is, therefore, also a 'vessel' and an 'exhibit'. To make the authoring tool easier to use, we have opted for a single-inheritance taxonomy, although EXPRIMO can handle multiple inheritance. In other words, an entity type may not have more than one parent, and an entity may not belong to more than one entity type. Right-clicking on entities or entity types in the taxonomy allows the authors to rename them, delete them, insert new entities or types, etc.

The authoring tool allows basic entity types to be subordinated to common entity types of Penman's Upper Model, a linguistically motivated domain-independent

ontology that has been used in several natural language generators (Bateman 1990).[10] This licenses the generator to treat the basic entity types as subtypes of the corresponding Upper Model types. Previous work in natural language generation has demonstrated that such a subordination has the advantage that some of the algorithms of the generator can be designed for domain-independent concepts of the Upper Model, and, hence, they can be decoupled from the domain-dependent ontology. We return to some possible uses of this subordination in the following sections.

Relationships are expressed in M-PIRO using *fields*. At any entity type, it is possible to define new fields, which then become available at all the entities that belong to that type and its subtypes, much as in frames and public fields of object-oriented programming languages. In figure 4, the fields 'painting-technique-used', 'painted-by', and 'potter-is' are defined at the type 'vessel'. Consequently, all the 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 defined at the 'exhibit' type. (The 'images' field is a special built-in field that allows authors to associate images with entities. Inherited and special fields are shown in a different colour in the authoring tool.) The *fillers* of each field, i.e., the possible values of the field, must be entities of a particular type. In figure 4, the fillers of 'potter-is' are declared to be of type 'potter'; hence, the entities 'sotades' and 'aristos' are the only possible values of 'potter-is'. To represent the fact that a particular 'vessel' entity was created during the classical period by 'aristos', one would fill in that entity's 'creation-period' field with 'classical-period', and its 'potter-is' field with 'aristos'. We use the term *fact* to refer to the information that the field of an entity carries, for example the information that the potter of 'exhibit22' is 'aristos', or information about the type of an entity, for example that 'exhibit22' is a 'kylix'.

The 'Many' column in figure 4 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., an exhibit may be made of both gold and silver). It is, thus, possible to specify that a relationship is many-to-one (only one material per exhibit) or many-to-many (many materials per exhibit), but not one-to-one (e.g., a unique showcase per exhibit). One-to-one relationships have to be specified as many-to-one or many-to-many, which does not guard sufficiently against errors (e.g., assigning the same showcase to multiple exhibits, in applications where this is forbidden). This limitation could be overcome by providing an option to signal that a relationship is one-to-one and corresponding checks when updating M-PIRO's database.

Fields are also used to represent attributes of entities, for instance their names or dimensions. Several built-in data-types are available, like 'string', 'number', 'date', and 'dimension', and they are used to specify the possible values of attribute-denoting fields; the 'Many' column also applies to attributes. In figure 4, the values of 'exhibit-purpose' and 'references' are declared to be strings. The two fields are intended to hold canned sentences describing what a particular exhibit was used

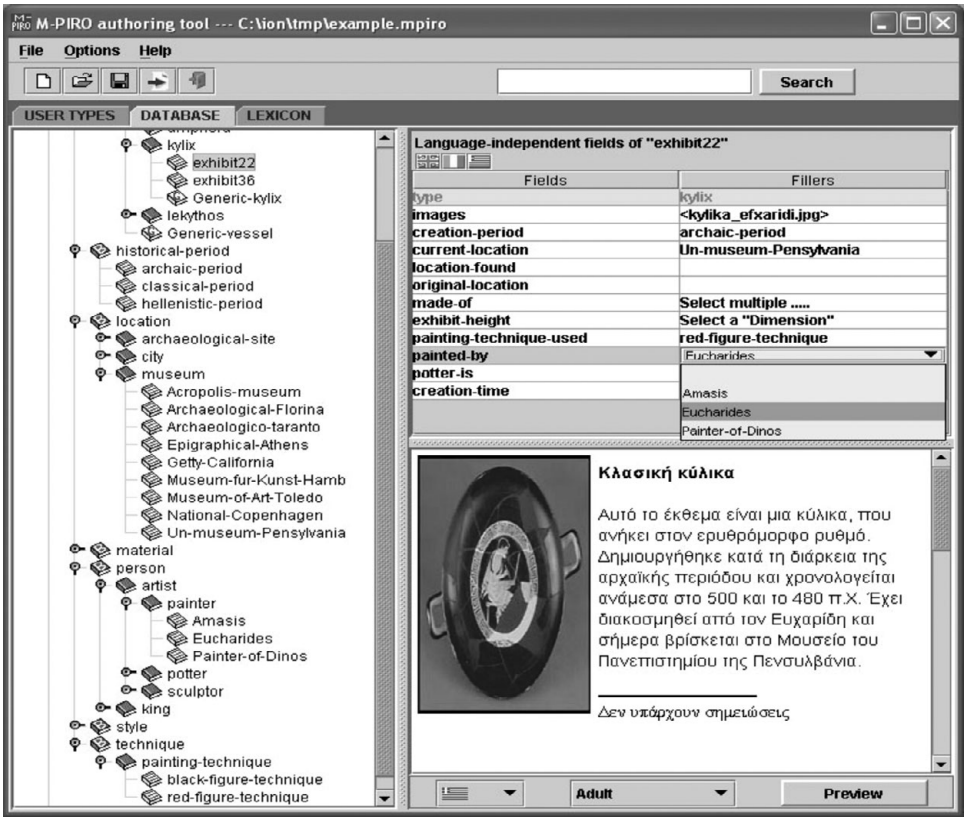
[10] See also `http://www.fb10.uni-bremen.de/anglistik/langpro/webspace/jb/gum/`.

for, and bibliographic references; for example, "This statue honours the memory of Kroissos, a young man who died in battle." and "Boardman, J., Athenian red figure vases, Thames and Hudson, 1975", respectively. Information is stored as canned text in string-valued fields when it is too difficult to represent in symbolic form. One could actually specify all the information about the entities using canned texts, but this has the drawback that the canned texts have to be provided in all of the supported languages, and they have to be updated manually whenever the information they express changes, which is costly when supporting many languages. Notice, however, that some of the benefits of natural language generation are still available with canned texts stored in string-valued fields. For example, as will be explained below, the authors can specify how interesting the information of a canned text is per visitor type, allowing the generator to decide if it should be included in the object descriptions; the document planner can also be instructed to place canned texts at appropriate positions in the descriptions; and it is possible to enhance the canned texts with dynamically generated referring expressions. Similar support for canned texts is provided, for example, by DRAFTER and ISOLDE.

We use the term *ontology* to refer to the taxonomy of entity types and the fields that are available at each type, and the term *database* to refer to the entities that populate the entity types and the field values of the entities. Once the ontology has been defined, it is possible to fill in the database, as illustrated in figure 5; pull-down menus and forms guide the authors to select among the possible values of the fields. Provided that appropriate lexicon entries and micro-plans have been defined, as will be discussed below, previews of the resulting object descriptions can also be generated. Figure 5 shows a description in Greek and the corresponding description in English. Notice that EXPRIMO has generated referring expressions where appropriate, and it has aggregated several clauses.

The values of language-dependent fields (e.g., the values of the 'exhibit-purpose' field discussed above) are entered in separate tables, one per supported language. The tables are displayed by clicking on the flags in the upper right part of figure 5. Figure 6 shows the table with the English values of the language-dependent fields of the entity 'amasis'. The 'name' and 'shortname' fields specify the entity's name and a shorter form, if available (e.g., "Alexander the Great" and "Alexander"). The 'notes' field holds an optional string that will appear as a footnote. The three fields are available at all entities, and the same applies to 'gender' and 'number'. In this particular example, the 'person-information' field is defined at the type 'person' as string-valued. Its English value at the entity 'amasis' is "is thought to have been both a maker and a painter of pots". This value would not be used only in direct descriptions of Amasis; EXPRIMO could also use it when describing an exhibit painted by Amasis, as in "This kylix was painted by Amasis. Amasis is thought. . . ".

To capture default information about all the entities of a type, *generic entities* can be introduced, much as in ILEX. For example, to specify the purpose of all entities of type 'kylix', one could introduce a generic entity of type 'kylix', shown as 'Generic-kylix' in the taxonomy of figure 5, and fill in its 'exhibit-purpose' field with "Kylikes were used as wine cups." in English, and equivalent strings in the other languages. This would licence EXPRIMO to generate texts like "This exhibit is



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 Pennsylvania. – No notes.

Fig. 5. Symbolic source information and the resulting text in Greek and English.

a kylix. Kylikes were used as wine cups. This kylix was created during the archaic period". In a similar manner, one could specify that kouroi, a kind of statue, were made in the archaic period, by filling in the 'creation-period' of 'Generic-kouros' accordingly. This would save us from having to specify the creation period of each individual kouros; the 'creation-period' fields of the individual kouroi would be left empty. It is also possible to override default information. For example, to specify that a particular kouros was created during the classical period, perhaps the art of an eccentric classical sculptor, one would set the 'creation-period' of that kouros to 'classical-period', and this would licence texts like "Kouroi were created during the archaic period. However, this kouros was created during the classical period".[11]

[11] This mechanism is currently not fully supported by EXPRIMO, and the same applies to the 'Many' column that was mentioned earlier, but we hope that these problems will be solved in the next EXPRIMO version.

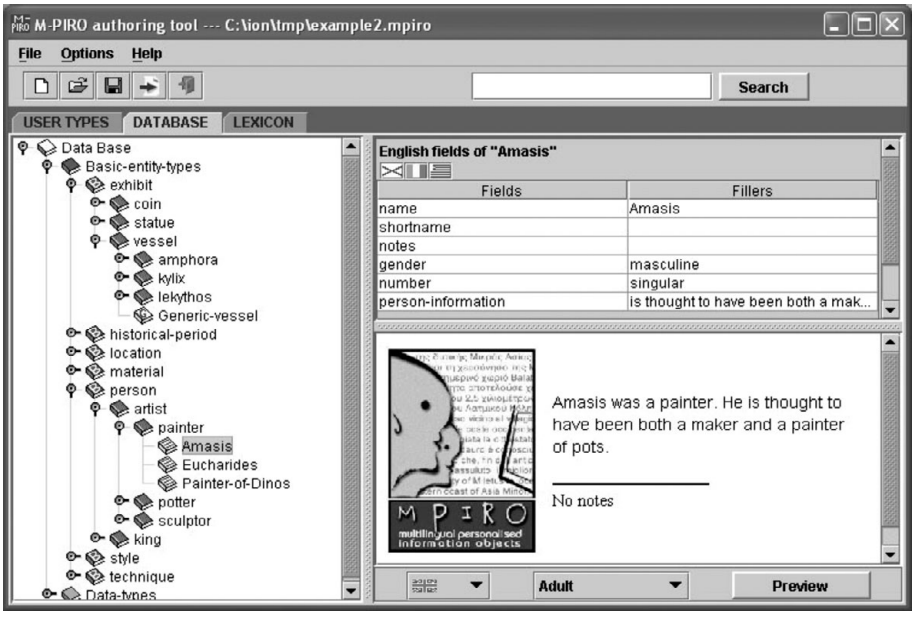


Fig. 6. Language-dependent fields.

3.2 Visitor stereotypes and personal models

The authors can define one or more types of visitors, as shown in figure 7, where 'user types' refers to visitor types. For each visitor type, M-PIRO's personalisation server maintains a stereotype, i.e., a fixed assignment of values to a set of user modelling parameters. The right panel of figure 7 shows some of these parameters and their values for the visitor type 'expert'; all the parameters are explained below.

Each fact that has been selected to be conveyed gives rise to a separate clause. This would lead to texts like: "It was created during the archaic period. It was painted with the red figure technique. It was painted by Eucharides.". The *maximum facts per sentence* parameter specifies the maximum number of clauses that can be aggregated in a single sentence. A value of 3 or greater licences the aggregated sentence "It was created during the archaic period and was painted with the red figure technique by Eucharides." in the English text of figure 5. Larger values lead to longer and more complicated sentences. A value of 4 (as in figure 7) usually leads to reasonable sentences for adults, but when generating for children shorter sentences may be more appropriate.

The *facts per page* parameter in effect controls the length of the resulting descriptions. It is often the case that the system knows a large number of facts about the entity to be described. A long text that conveys them all may be too boring for the visitor. When delivering the generated text on mobile devices or in spoken form, there may also be limitations imposed by the size of the device's display or the speed of the speech synthesizer. M-PIRO's content selection stage ranks the relevant and appropriate facts and selects the most highly ranked ones, so that

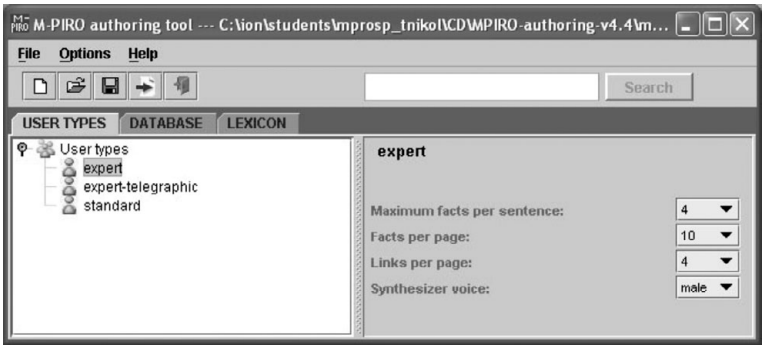
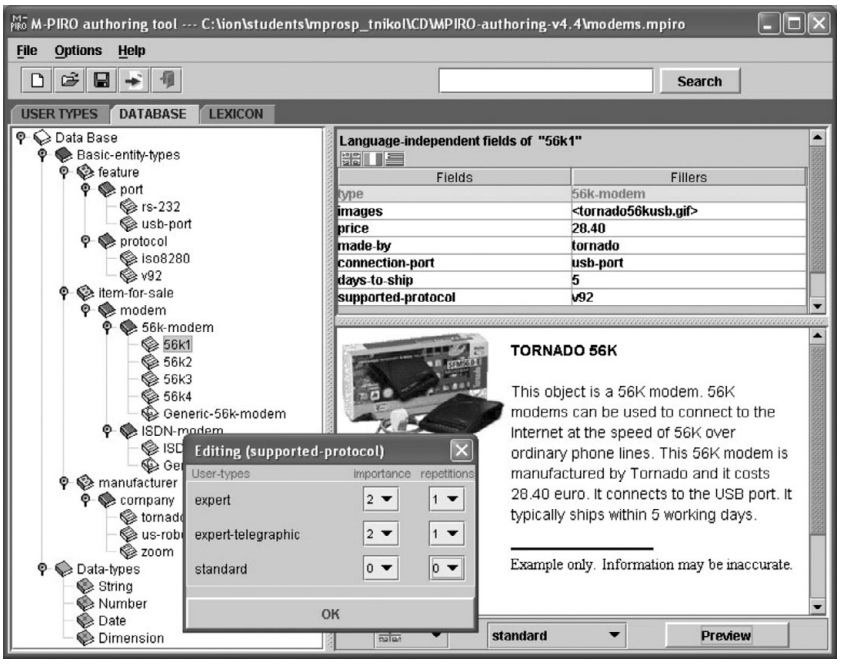


Fig. 7. Visitor types and some of the parameters of the visitor stereotypes.

their number does not exceed the value of 'facts per page'. The remaining facts can be conveyed if the visitor asks for more information, each time generating an additional piece of text, or *page* in M-PIRO's terminology. In the demonstrator of figure 1, additional pages can be generated by clicking on the "Tell me more" link or by revisiting an exhibit. There are options in the authoring tool to preview all the pages of an entity.

The idea that the number of facts expressed per sentence or page should be open to personalisation is inspired by work in the psychology of text comprehension, especially by Kintsch and van Dijk (1973). The technical notion of a fact used here corresponds reasonably closely to an 'idea' or simple proposition, while an entity corresponds to an argument. It has been argued that texts with high densities of arguments, or high quantities of ideas, are harder to process (Kintsch and Keenan 1973). Hence, less skilled readers should benefit from texts with fewer facts per sentence, and fewer facts per page. The other two parameters of figure 7, *links per page* and *synthesizer voice*, are more a matter of taste: they control the number of forward pointers that are provided at the end of each page, and the preferred synthesizer voice, respectively; both can be set to different values per visitor type.

The current system supports multidimensional user models, because it takes into account abilities, interests, and level of knowledge (Zukerman and Litman 2001). While figure 7 focuses on ability-related parameters at a general level, figure 8 shows the manipulation of values for interest and knowledge, while authoring specific content. Both figures are from a sample application that generates descriptions of modems for sale. The 'expert' and 'standard' types correspond to visitors that are familiar or not, respectively, with the functionality and features of typical modems. In distinguishing users by their level of expertise, we follow in the tradition pioneered by Paris and collaborators (Paris 1988; Bateman and Paris 1989). In museums, curators can offer advice informed by experience, and possibly visitor surveys, concerning what general assumptions can be made about different types of visitors, in terms of abilities and interests; indeed this is how the stereotypes of M-PIRO's museum demonstrator were constructed. In the modem domain, however, we currently rely on our own intuition only. Figure 8 shows an example description for a 'standard' visitor, along with the corresponding text for an 'expert' visitor.



Corresponding description for 'expert': TORNADO 56K – This object is a 56K modem; it is manufactured by Tornado and it costs 28.40 euro. It supports the v.92 protocol and connects to the USB port. It typically ships within 5 working days.

Fig. 8. A description for a 'standard' visitor (screenshot), and the corresponding text (below the screenshot) for an 'expert' visitor. Also shown is a pop-up window that allows the author to set the importance and repetitions scores of the field 'supported-protocol'.

As the example illustrates, it is possible to convey different content to different types of visitors. In the description for 'standard' visitors, the text does not mention the protocol, information presumed to be too technical for non-experts, unlike the text for 'expert' visitors. This is achieved by adjusting the *importance* scores of the field 'supported-protocol', as shown in figure 8. Importance scores range from 0 to 3, and show how interesting the corresponding fact is for visitors of the various types.[12] During content selection, preference is given to unassimilated facts with high importance scores. In figure 8, the importance of 'supported-protocol' for 'standard' visitors is zero, signalling that the corresponding fact should only be conveyed when there is nothing more interesting to say. The 'repetitions' scores show how many times the corresponding fact has to be mentioned, possibly in different pages, before presuming that the visitor has assimilated it. It therefore marks the author's

[12] ILEX and EXPRIMO distinguish between 'interest' and 'importance'. The former shows how interesting a fact is presumed to be to visitors of each type, while the latter shows how important the authors believe it is to convey it to the corresponding visitors. The usability evaluation, which will be presented later on, found that this distinction confused the authors. The authoring tool now hides the distinction, by showing only the importance scores and assigning to interest the same value as importance. We, therefore, use the two terms as synonyms.

judgment concerning the relative difficulty of a domain concept, and this is taken into account in both content selection and surface realization (Cawsey 1990). A repetitions value of zero signals that the fact should never be mentioned.

Specifying the importance and repetitions scores of a field at each individual entity is tedious. Instead, they can be specified at the entity type that defines the field. In our example, 'supported-protocol' is defined at the entity type 'modem'. Using a similar pop-up window as that of figure 8, the author can specify at the type 'modem' that the importance and repetitions of 'supported-protocol' for 'standard' users are zero, and this will apply by default to all entities that belong to that type and its subtypes. It is possible to override the default scores of a field by specifying different scores at a particular entity. For example, we may wish to specify that the material of statues should generally not be reported to average adult visitors, but in the case of a particular statue that is made of gold, we may wish to override the default and report the material.

Notice, also, that the text for the 'standard' visitors in figure 8 includes a general sentence on 56K modems, which is absent from the corresponding text for 'expert' visitors. This is a canned sentence, stored as the value of the string-valued field 'item-type-description' at the 'Generic-56k-modem' entity; the field (not shown in figure 8) is defined at the type 'item-for-sale', and its importance and repetitions are set to zero for 'expert' visitors. Furthermore, generating a description of another 56K modem for 'standard' visitors in the authoring tool, for example '56k3', after having generated the description of figure 8, produces the following text:

ZOOM 56K – This object is another 56K modem, made by Zoom. Its price is 75.20 euro. This 56K modem connects to the RS-232 port. It typically ships within 3 working days.

The new description does not repeat the general sentence on 56K modems, because it is presumed that the visitor has assimilated it. As already mentioned, the personalisation server maintains a personal user model for each visitor, which records the facts that have been conveyed to the visitor and the degrees to which the system believes they have been assimilated, allowing EXPRIMO to avoid repeating assimilated facts. The personal models also store information on the objects that have been described and the micro-plans that have been used, and this allows EXPRIMO to generate comparisons and avoid repeating the same expressions; notice, for example, the "another 56K modem" in the description of Zoom 56K above, and the fact that the manufacturer and cost are expressed with different clauses, compared to the texts of figure 8.[13] It is in principle possible to employ decay mechanisms in the personalisation server, that would decrease the assimilation scores over time and would gradually delete old parts of the interaction history of each user, to model the process of forgetting and to avoid comparing to objects examined too far in the past, but we have not explored this issue further. Instead, we adopted the simplistic approach of clearing personal models a few days after the last visit of the corresponding visitors, causing the system to treat all previously conveyed facts as unassimilated by the corresponding visitors and all objects as never examined.

[13] Like ILEX, M-PIRO generates comparisons to previously examined objects, but it does so more frequently and informatively; see Calder *et al.* (2005).

The authors do not interact directly with personal models, but the authoring tool silently maintains a sample personal model for each visitor type, which allows the authors to see the effects of personal modelling on the generated texts. There is an option to clear the sample personal models, which the authors can use to view the texts that would be delivered to visitors with no previous interaction history.

3.3 Document planning settings

M-PIRO's document planner is largely domain-independent. It is a simplified version of ILEX's planner (O'Donnell *et al.* 2001), lacking the second stage where rhetorical relations can re-order ILEX's selected content. There are only two ways in which the authors can influence the simple planner's behaviour. First, by setting the importance of the various facts, since the planner, if not instructed otherwise, attempts to place the most important facts (of those content selection has decided to convey) towards the beginning of the text, subject to coherence constraints (e.g., facts about a statue being described cannot be interleaved randomly with other facts about the sculptor of the statue, that content selection has also decided to convey). In a second, more direct manner, the authors can specify explicitly the preferred fact order by manipulating a list that contains all the available types of facts. In the application of figure 8, the author has specified explicitly that the type of the object should be mentioned first, followed by the facts that correspond to the fields 'item-type-description', 'made-by', 'price', 'connection-port', 'supported-protocol', and 'days-to-ship', in this order.[14] In more complex domains, like the museum domain of figures 4–6, the list the authors order includes fact types corresponding to fields of several entity types (e.g., fields of exhibits, persons, locations). If an explicit order has been specified, EXPRIMO expresses the facts (that content selection has decided to convey) in the specified order, again subject to coherence constraints.

Note that there is a single fact order for all visitor types. A possible extension would be to allow different fact orders per visitor type, though we have not encountered cases where this was necessary. In related work, Dimitromanolaki and Androutsopoulos (2003) investigated a machine learning approach, where the order of the facts of each description is decided by a chain of classifiers; the classifiers are trained on example texts whose facts have been re-ordered by domain experts.

3.4 Micro-plans

For each field of the ontology and each language, the authors have to specify at least one micro-plan, that specifies how the field can be expressed as a clause in that language. Following ILEX, M-PIRO supports two forms of micro-plans: *clause plans* and *templates*. In clause plans, the author specifies the verb to be used, the voice and tense of the resulting clause, the preposition, if any, to be included between

[14] The list is currently specified in an EXPRIMO startup file. We hope that future versions of the authoring tool will allow the authors to specify the fact order by re-ordering the rows of the tables that show the fields of each entity type.

the verb and the object, any desired adverb, and strings to be concatenated as adjuncts at the beginning or end of the clause. The verb has to be chosen from those available in the domain-specific lexicon; if the desired verb is not in the domain-specific lexicon, it has to be added first, as will be discussed in the following section. The clause plan in figure 4 leads to clauses like "This vessel was painted by Eucharides". Appropriate referring expressions, for example "a painter called Eucharides", "Eucharides", or "him", are generated automatically. The algorithm for constructing noun phrases is that used in ILEX (O'Donnell *et al.* 1998); it takes into account both what is known about an entity, and the context in which it is mentioned. The 'show advanced options' tick-box of figure 4 allows the authors to control parameters such as the mood of the clause, the case of the referring expressions, and whether or not the clause can be aggregated; the default values of these parameters are usually adequate.

Templates provide stricter control over the surface form of the resulting clauses than do clause plans. A template is a sequence of slots, the values of which are simply concatenated to produce a clause. Figure 9 shows a template for the 'price' field of 'item-for-sale' entities, that leads to clauses of the form "Its/This modem's/Tornado 56K's price is 28.40 euro". Each slot can be filled by: an expression referring to the entity that owns the field, i.e., the 'item-for-sale' entity in our example, as in slot 1 of figure 9; a canned string, as in slots 2 and 4; the value of the field, if the values of the field are built-in data-types, such as strings or numbers, as in slot 3; or an expression referring to the value of the field, if the values of the fields are entities, as would be the case with the 'made-by' field in figure 9. With slots that are filled by referring expressions, the author also specifies the case of the referring expression. Setting the type of a referring expression to 'auto', as in slot 1, allows EXPRIMO to decide whether it will use a pronoun, a noun phrase like "the modem", or the entity's name. It is also possible to specify that a particular type of referring expression should be used; for example, that a pronoun should always be generated.

Templates carry less linguistic information than clause plans, which does not allow EXPRIMO to exploit its full potential. For example, EXPRIMO's aggregation rules are designed to manipulate clause plans, and, hence, cannot be applied to clauses generated by templates, like the last sentences of the texts in figure 8. However, templates are the only option when facts need to be rendered in forms other than those clause plans generate. With clause plans, for example, the subject of the generated clause always refers directly to the owner of the field, as in "this modem", "it", or "Tornado 56K", which is why a clause plan cannot generate the same clauses as the template of figure 9, where the subject has to refer to the price of the modem. Templates can also be used to add dynamically generated referring expressions to canned strings. In figure 6, for example, the value of 'person-information' is a sentence without a subject. The resulting text "He is thought to. . . " is produced with a template, whose first slot asks for an 'auto' referring expression for the owner of the field, and whose second slot is filled by the string value of the field.

Another use of templates is to produce telegraphic text, which may be preferable when delivering text on devices with small screens, for example, on mobile phones via SMS. In figure 7, the 'expert-telegraphic' type is for expert visitors who prefer

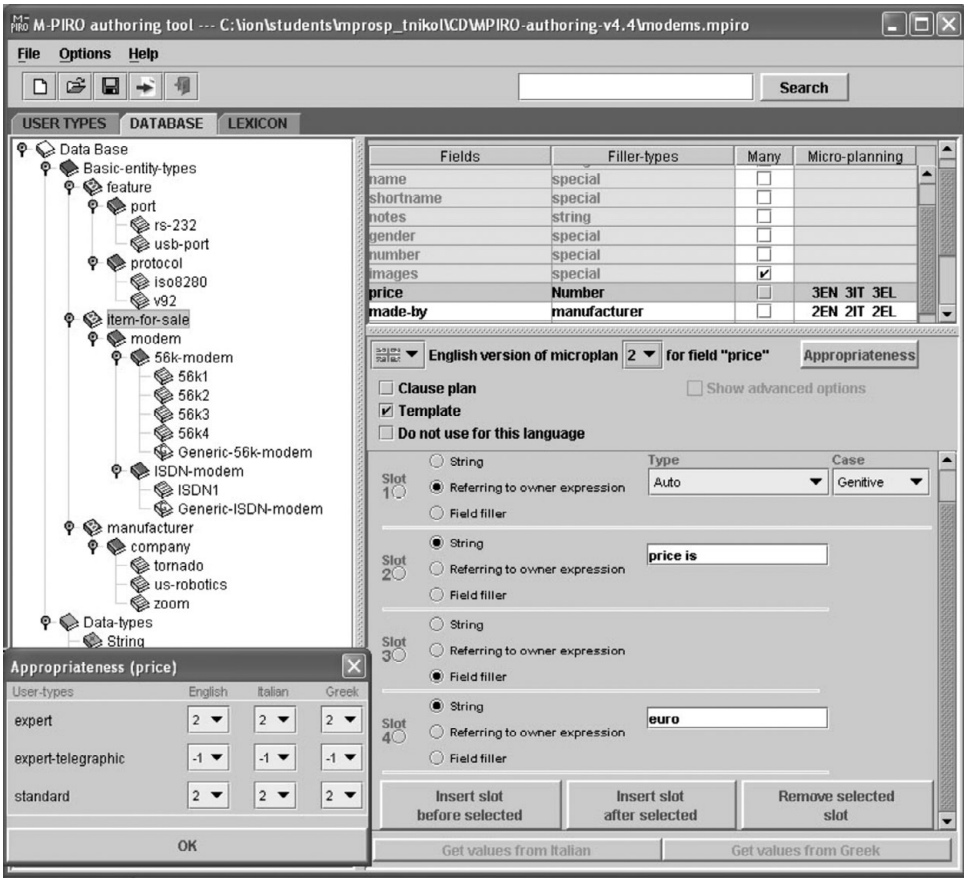


Fig. 9. A template, and specifying the appropriateness of a micro-plan.

telegraphic text. They receive approximately the same content as 'expert' visitors, but with more condensed language expressions, generated via templates. The corresponding 'expert-telegraphic' description for the modem in figure 8 is:

TORNADO 56K – 28.40 euro. Supports the v.92 protocol. Connects to the USB port. 5 days to ship.

EXPRIMO uses similar built-in templates of the form *field-name*:*value* when no micro-plan has been provided for a field. For example, if no micro-plan had been specified, the text above would be:

TORNADO 56K – PRICE: 28.40. SUPPORTED-PROTOCOL: the v.92 protocol. CONNECTION-PORT: the USB port. DAYS-TO-SHIP: 5.

This allows the authors to inspect the facts that EXPRIMO selects to convey, and possibly adjust the parameters of the visitor stereotypes, for example the importance scores or the number of facts per page, before specifying micro-plans. It also acts as a reminder that no micro-plan has been provided for a field.

Unlike ILEX, M-PIRO allows multiple micro-plans to be specified for the same field and language. This can be used to avoid repeating the same expressions in texts for the same visitor, and to tailor the expressions per visitor type; for related work on micro-planning choices for low-skilled readers, see Williams and Reiter (2005). Here, we have already noted the fact that the manufacturer and cost are expressed using different clauses in figure 8 and the description of Zoom 56K that follows it; this is the effect of having defined multiple micro-plans for the corresponding fields.

To keep the generation capabilities across the supported languages aligned, the authoring tool encourages the authors to specify each micro-plan in all of the languages, so that equivalent clauses can be produced. For example, in the second micro-plan of the field 'price' in figure 9, the author can easily see and fill in the corresponding micro-plans in the other two languages, by clicking on the flag. The 'Micro-planning' column shows how many micro-plans have been provided for each field in the three languages; in the case of 'painted-by' in figure 4, there are three Greek micro-plans, but only two English ones and one Italian, which indicates that there are micro-plans for which their counterparts in the other languages have not been entered. The lexicon entries are also kept aligned, as will be discussed in the following section; for example, the 'paint-verb' in figure 4 is the identifier of a trilingual verb entry in the lexicon, that contains the verbs "paint", "dipingere", and "ζωγραφίζω" of the three languages. The 'get values from' buttons in figure 4 speed up the authoring process by setting, where possible, the parameters of a clause plan to the same values as those of its counterparts in the other languages; for example, the same lexicon entry for the verb, the same tense, voice, etc.

For each micro-plan, *appropriateness* scores specify how appropriate it is to use the micro-plan in descriptions delivered to each type of visitor. In the pop-up window of figure 9, the author has specified the appropriateness scores of the English, Italian, and Greek versions of the second micro-plan for the field 'price'. Positive scores indicate appropriate micro-plans. EXPRIMO uses in turn all the micro-plans of a field that have positive appropriateness, starting with the micro-plan with the highest appropriateness. Negative appropriateness indicates micro-plans that should be avoided. EXPRIMO uses micro-plans with negative appropriateness only when there is no other micro-plan with a positive score, and in that case, it always selects the micro-plan whose appropriateness is the least negative. The appropriateness scores in the pop-up window of figure 9 signal that the micro-plan should not be used in descriptions for 'expert-telegraphic' visitors. There are more suitable, telegraphic micro-plans for those visitors, and their appropriateness is positive for 'expert-telegraphic' visitors and negative for other types of visitors.

The number of micro-plans the authors have to specify can be reduced if the authoring process starts with a basic ontology providing entity types that are common in M-PIRO applications, their fields, and micro-plans to express them, rather than starting from an empty ontology. For example, there could be a basic ontology providing types such as 'physical-object' or 'person', with fields like 'weight' and 'place-of-birth', respectively, and appropriate micro-plans. Then, in a museum application, the authors would simply define 'exhibit' and 'artist' as sub-types of 'physical-object' and 'person', respectively, which would cause the two new types

to inherit the fields 'weight' and 'place-of-birth', respectively, and their micro-plans. We return to this issue when presenting the usability evaluation of the authoring tool below. In a similar manner, common fields (e.g., 'price') and the corresponding micro-plans could be inherited from high-level entity types of the Upper Model via the entity type subordination mechanism of section 3.1, or by subordinating fields (that represent relationships or attributes) to corresponding notions of the Upper Model. The Upper Model has not been exploited in M-PIRO as systematically as in other generation systems; see, for example, Bateman (1990) for related discussion. While this increases the number of micro-plans the authors have to specify, it also reduces the degree to which the authors are required to have mastered the (mostly linguistically motivated) concepts of the Upper Model (e.g., NON-DIRECTED-ACTION, ADDRESSEE-ORIENTED-VERBAL-PROCESS, STATIVE-QUALITY).

M-PIRO's clause plans can produce a much smaller range of surface expressions, compared to the expressions that can be generated when specifying sentence plans in formalisms like the commonly used SPL language (Kasper and Whitney 1989; Bateman 1997). On the other hand, the simplicity of M-PIRO's clause plans makes them much easier to explain to authors, who we assume have no previous experience in natural language generation, unlike the effort and background that is needed to master SPL and the linguistic concepts it employs. M-PIRO's templates are also simpler than those of some other systems. For example, unlike the templates of Buseman *et al.* (1999) and McRoy *et al.* (2003), M-PIRO's templates do not allow default values, conditionals, or invoking recursively other templates; see also van Deemter *et al.* (2005) for a discussion of how templates can be enriched with syntactic and other information, blurring the distinction between templates and clause plans. Again, however, the simplicity of M-PIRO's templates makes them much easier to explain to authors.

3.5 Domain-dependent lexicon

The *domain-dependent lexicon* contains entries for nouns and verbs, as shown in the left part of figure 10. The entries for function words, such as articles and prepositions, are domain-independent and are kept separately. As explained in the previous section, verb entries are trilingual, and they are used when specifying clause plans. Noun entries are also trilingual. The right panel of figure 10 shows the Greek part of the noun entry 'coin-noun'; the Greek noun is "*νόμισμα*".

EXPRIMO's algorithm for generating referring expressions requires the authors to establish links between entity types and the noun entries that can be used to refer to the entity types. In figure 4, the entity type 'vessel' is linked to the noun entry 'vessel-noun' (this can be seen in the area next to the 'Edit nouns' button), which contains the nouns "vessel", "vaso", and "*αγγείο*" of the three languages. This allows EXPRIMO to produce expressions like "this vessel" when referring to entities of type 'vessel' in English. As with verbs, if the appropriate noun entry is not present in the domain-dependent lexicon, it has to be added first.

An entity type can be linked to multiple noun entries. As with micro-plans, noun entries carry appropriateness scores, which may indicate that a noun-entry is

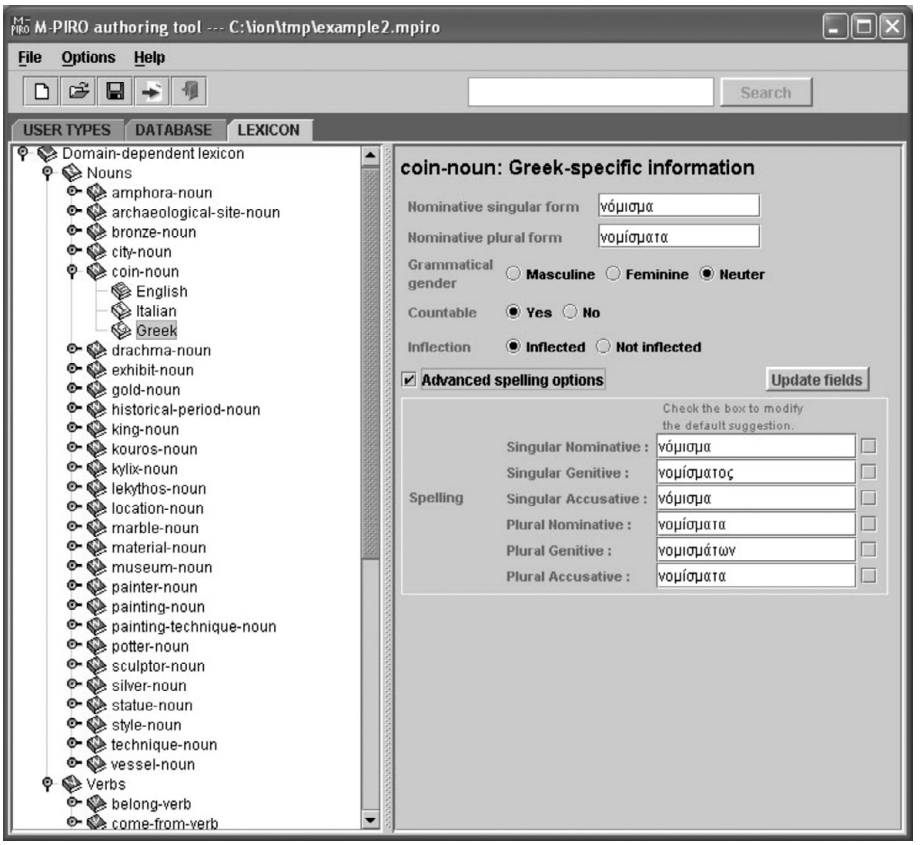


Fig. 10. Editing the domain-dependent lexicon.

more appropriate than others when generating texts for particular types of visitors. Furthermore, each entity type inherits the noun entries that have been linked to its super-types. Hence, in figure 4, the entity type 'vessel' also inherits the lexicon entry 'exhibit-noun', which has been linked to the 'exhibit' entity type; this licences EXPRIMO to refer to a 'vessel' entity as "this vessel" or "this exhibit".[15]

Like most natural language generation systems, M-PIRO's domain-dependent lexicon is typically rather small. For example, there are approximately 50 noun and 30 verb entries in the museum demonstrator of figure 1, many of which, like 'kouros' and 'kylix' are unlikely to be found in general purpose dictionaries. Hence, instead of attempting to reuse existing large-scale electronic dictionaries, we have opted for facilities that simplify entering new nouns and verbs in M-PIRO's domain-dependent dictionary. In the case of Greek nouns, for example, EXPRIMO's systemic grammar requires several features pertaining to the inflectional pattern of the noun, the position of the stress in the noun's various forms, etc. The authoring

[15] EXPRIMO currently ignores the noun inheritance mechanism, and when an entity type is linked to many noun entries, it always selects the first one. Supporting mechanisms are already in place in the authoring tool and the personalisation server, and we hope that appropriate support will also be added to the next EXPRIMO version.

tool incorporates facilities that determine and add automatically these features by examining the nominative singular and nominative plural forms of the noun that the author enters in the top right part of figure 10. Morphology rules are also available, which generate automatically the remaining inflectional forms of the nouns from the two forms that the author enters; similar facilities are available for verbs. The 'advanced spelling options' button in figure 8 displays in the bottom right part of the window all the inflectional forms of the noun (or verb), allowing the author to inspect and correct the forms that have been generated automatically, when text previews indicate that some of them have been generated wrongly. In most cases, however, the generated forms are correct, and the author does not need to consider other inflectional forms than those entered in the top right part of the window. As with micro-plans, nouns carry appropriateness scores, which may indicate, for example, that 'vase' is more appropriate than 'vessel' when generating for children.

One limitation of M-PIRO's current linguistic coverage is that it does not support directly noun modifiers, such as adjectives or prepositional phrases, as in "*archaeological* site" or "cable *for serial connection*". To generate nouns with modifiers, and to link them to entity types, one has to create pseudo-noun entries, that contain both the noun and its modifiers, like the entry 'archaeological-site-noun' in the left panel of figure 10. The main problem with this approach is that it confuses the morphology rules for nouns, and, hence, the authors have to correct manually many of the automatically generated forms of the pseudo-nouns; this is more tedious in Greek and Italian, where nouns and adjectives have several inflectional forms.

3.6 Domain authoring vs. exhibit authoring

The authoring process can be divided in two separate stages, which we call *domain authoring* and *exhibit authoring*. Domain authoring defines the application's ontology, the visitor stereotypes, and the domain-dependent linguistic resources. To ensure that all the domain-dependent resources have been defined correctly, domain authors will typically also create entries for a few entities of various types, and preview their natural language descriptions. Thereafter, exhibit authoring is concerned with populating the database with more entities. It is a data entry task that can be assigned to people whose familiarity with the authoring tool is limited to creating entities of the appropriate types, filling in their fields, and previewing the resulting texts, unlike the domain authors who need to have mastered the full functionality of the authoring tool. Both stages can be semi-automated, to exploit existing ontologies and databases; we return to this point in the following sections.

4 Usability evaluation of the authoring tool

Throughout the development of the authoring tool, there was continuous feedback on its usability from a curator of the Foundation of the Hellenic World (FHW) who has a background in cultural informatics; the curator also entered approximately half of the exhibits of M-PIRO's museum domain. The tool was also subjected to two formative usability evaluations, that helped shape its following versions. One

of them involved a group of FHW curators, while the second one was conducted with graduate informatics students at the University of Edinburgh. Here we focus on a third, summative evaluation, that assessed the usability of the authoring tool, as delivered at the end of the M-PIRO project, in the case where the authors have a computer science background but no expertise in natural language generation.[16] The main goal of this evaluation was to investigate the degree to which authors with this background can use the tool to maintain existing M-PIRO applications or create new ones, after receiving a short training course. A secondary goal was to identify possible further improvements to the tool. Some of the improvements that were proposed were subsequently implemented during additional follow-up work at the Athens University of Economics and Business (AUEB) (Nikolaou 2004; Prospathopoulou 2004). All the screenshots of the tool in this paper are from the improved version, with the exception of figure 12, which is from the version that was used during the evaluation.

The summative evaluation involved a group of 10 third-year computer science undergraduates from AUEB, who had no prior experience in language generation.[17] As already noted, we believe that the expertise of our subjects is a good lower boundary of the expertise of the employees that would be assigned the task of configuring M-PIRO for e-commerce applications (e.g., generating descriptions of items for sale) in a corporate environment. A further advantage of the group of students we used is that they were drawn from a Human-Computer Interaction (HCI) course, which allowed them to employ evaluation methods that require familiarity with HCI concepts. Although the feedback we had from the FHW curators during the formative evaluations was quite positive, we chose not to experiment further with museum curators, because their computing skills vary considerably, and, hence, it is difficult to reach solid conclusions on how easily they can use the tool. Also, feedback from our cultural informatics partners indicates that in a museum context it is reasonable to assume that M-PIRO's authoring process would be assigned to a curator with a cultural informatics background, or to a computer scientist (e.g., from the museum's information technology department) who would interact with curators to obtain the information they wish to present at each exhibit. Hence, we believe there is value in our summative evaluation for museum contexts too.

The students first attended a six-hour introductory course on the use of the authoring tool. They were then given the course's slides and 11 homework tasks to perform with the tool. The tasks were designed to cover most of the tool's functionality, and they ranged in difficulty from correcting information about an

[16] Karasimos and Isard (2004) have also carried out end-user evaluations to measure the acceptability of the resulting texts, and factual recall from them. Results indicate that aggregating clauses and including comparisons leads to higher factual recall and higher acceptability of the generated texts.

[17] All of the students had attended a single, one-semester Artificial Intelligence course, whose content follows closely that of ACM/IEEE recommended curricula, i.e., it included elementary knowledge representation (e.g., propositional logic, first-order predicate logic) and basic natural language processing techniques, such as parsing with a DCG grammar. We believe that such an AI course is typical of most computer science curricula, and, hence, other authors with a computer science background would have attended at least one similar course.

	Task type	Scores
i.	Correcting database information for existing exhibits.	**10 A**
ii.	Adding more exhibits of existing types.	4 A, **5 B**, 1 C
iii.	Adjusting user modelling parameters to affect the semantic content and surface form per visitor type.	**6 A**, 3 B, 1 C
iv.	Previewing texts in different languages.	**6 A**, 1 B, 1 C, 2 D
v.	Correcting errors in the surface form, caused by errors in the lexicon or micro-plans.	1 A, **4 B**, **4 C**, 1 D
vi.	Adding new types of exhibits and corresponding micro-plans and lexicon entries.	4 B, **6 C**
vii.	Creating a new M-PIRO application.	2 B, **6 C**, 2 D

Fig. 11. Task types of the usability evaluation of the authoring tool and their scores.

existing exhibit to creating a new mini M-PIRO application. The modems application of figure 8 is based on an application created by one of the students during the evaluation. Other students came up with applications for cars, motorcycles, music CDs, pets, etc. Based on their experience from the 11 tasks, the students were then required to fill in a questionnaire that asked their opinion on the degree to which authors with the same background as theirs would be able to perform tasks of various types, after attending the same introductory course. For each type of task, there were six possible replies: A (they would succeed with no trouble at all), B (success with only minor problems), C (they would have trouble, but they would probably succeed), D (they would have trouble and success would be dubious), E (unlikely to succeed), and F (no chance to succeed). The types of tasks are shown in figure 11, along with the replies of the students.

Overall, the majority replied that the authors would succeed in all cases, albeit in most cases after first encountering difficulties of various degrees. The rationale behind the replies is more informative. The students were instructed to base their replies on a type of *cognitive walkthrough* (Preece 1994) of the 11 tasks they had to perform. In each task, they were asked to consider how easily an author would figure out which sequence of actions offered by the tool had to be chosen, the degree to which the actions offered by the tool matched the goals the author would have in mind, the probability that the author would select actions leading to an outcome other than the intended one, and how easily the author would realize the latter mismatch and take appropriate action. The students had to summarize their cognitive walkthrough analysis per task in a separate part of the questionnaire, which reveals the rationale behind the scores of figure 11. They were also asked to propose possible improvements, based on their cognitive walkthrough analysis and a form of *heuristic evaluation* (Nielsen 1992), where they had to examine the degree to which the tool complied with established usability guidelines explained during the HCI course. We discuss below the rationale behind the scores of figure 11, along with some of the improvements that were proposed. A group discussion with the 10 students was also conducted after they had handed in the questionnaires, during which some of the replies were clarified.

No student reported serious problems in task types (i) and (ii), which is compatible with our claim that exhibit authoring is simpler than domain authoring. In task type (ii), the scores were slightly lower partly because of a bug in the graphical user interface (GUI) of the authoring tool, that caused newly entered strings to be lost when the author moved to another field without pressing 'enter'; this has now been fixed. The students also reported that showing language-dependent fields in separate screens per language (figure 6) causes problems when adding new exhibits, because authors often forget to assign values to the language-dependent fields in some or all of the languages. A possible solution is to show the language-dependent fields in the same table as the language-independent ones (figure 5), and use the preview panel to display the values of a language-dependent field across all languages when the author clicks on that field. Facilities to create copies of existing entities were also requested, as this helps entering similar entities, as well as mechanisms to move entities to more general or specific entity types.

In task type (iii), a distinction between 'interest' (how likely it is that visitors of each type will find a fact interesting) and 'importance' (how important the authors believe it is to convey a fact to visitors of each type), which was inherited from ILEX and was still used at the time of the experiment, was found to be confusing. The students found it difficult to distinguish between interest and importance in practice, and they all specified the same values for both in all facts. The distinction has now been abolished (see footnote in Section 3.2). Note that the students were not given the option to specify the order of the facts explicitly (Section 3.3). Hence, adjusting the importance scores was the only means to influence the order of the conveyed facts; this does not appear to have caused any problems. There were also difficulties caused by the fact that the authors were required to activate manually a facility to export user modelling information to the personalisation server whenever necessary, which they often forgot to do; this has now been automated.

The surprisingly low scores that some students assigned to task type (iv) were due to technical problems and bad GUI design. The tool required the authors to remember to activate manually a facility that exported the domain-dependent resources to EXPRIMO's XML format (discussed below) whenever they made changes to them. They often forgot to do so, which caused the previews to be based on outdated resources. This process has now been fully automated. There were also some bugs in the communication between EXPRIMO and the authoring tool, which sometimes caused no preview to be generated; most of these bugs now also appear to have been fixed. A third problem was that the GUI made it difficult to figure out and remember which actions had to be performed to generate a preview (see figure 12). The previewing process involved selecting a language from the options menu, right-clicking in the ontology on the entity to be described, and then selecting the preview option and the visitor type from cascaded menus. Again, this process has now been made much more intuitive, as can be seen in figure 5.

The scores of task types (v) and (vi) primarily reflect the problems that the students encountered during the manipulation of domain-dependent linguistic resources, especially micro-plans. The micro-plans and lexicon entries mention some grammar concepts, such as verb aspects, and pronoun cases, that authors with a computer

science background are not always acquainted with; this issue had also been noted in the Edinburgh formative evaluation, mentioned earlier in this section, that involved informatics graduates. Apart from introducing uncertainty and, hence, discomfort, these concepts also led some of the students to select default or random values of the corresponding options, generate previews to check their effect, and then revisit the micro-plans or lexicon entries to try other values. This could have actually been a successful form of exploratory learning, where users learn the functionality of the user interface by trying out the effects of the various options offered. However, returning to the appropriate micro-plans and lexicon entries is often time consuming, because there is no quick way to be taken automatically from a word or phrase in a preview text to the domain-dependent language resources that were used to generate it.

At the time of the evaluation, the problems of the previous paragraph were amplified by the complicated series of actions that the authors had to perform in the GUI to reach particular micro-plans; this involved right-clicking on a field of an entity type, and then navigating through cascaded menus to select the language and micro-plan number. The actions have now been simplified and made more intuitive; see figure 4. However, the editing of domain-dependent linguistic resources could be simplified further by providing mechanisms to allow the author to click on a word or phrase in a generated text and be taken to the corresponding lexicon entry or micro-plan. DRAFTER provided a similar mechanism that linked a generated text back to the database entries that carried its semantic content (Hartley and Paris 1997), and this functionality could also be added to M-PIRO's authoring tool to help authors reach database fields that need to be corrected. A similar approach could be used to locate user modelling parameters that need to be modified. Although not implemented, this kind of *active previewing* is feasible in M-PIRO, because EXPRIMO can easily markup each word or phrase in the output texts with the language resources, database entries, and user modelling parameters that were used to generate them. Previewing mechanisms of this kind would take M-PIRO closer to the WYSIWYM approach, where authors interact with the system entirely via generated texts reflecting the current content of the database and the options that are available to update it (Power and Cavallotto 1996; Van Deemter and Power 2003), although to the best of our knowledge WYSIWYM has so far been demonstrated only in cases where the authors manipulate only the contents of the database, not the ontology, language-dependent resources, and user modelling parameters.

To a large extent, the scores of task type (vii) were due to the cumulative effect of problems encountered in the previous task types. However, the scores also reflect the amount of work that has to be carried out to generate a new M-PIRO domain from scratch. Modifying an existing or similar application is much easier than generating an entirely new application, and many students felt that the authoring tool should include a basic ontology with suitable linguistic resources; for example, common entity types such as 'exhibit', 'artist', and 'historical-period' in the case of museum applications, with associated fields, micro-plans, and lexicon entries. This proposal had also surfaced in the FHW formative evaluation, and suggests that it would be easier to market specialised versions of M-PIRO's technology for particular families of

applications, for example, museum applications, or e-commerce applications, with basic ontologies and vocabularies built-in. Our more recent work on OWL ontologies (discussed below) can be seen as a step towards that direction.

As a measure of user satisfaction, the questionnaires also asked the students to indicate if they would be willing to undertake the task of using the evaluation version of the tool in a work environment to create a 50-exhibit domain, assuming that other colleagues had received the same training and could undertake the task instead. Among the 10 students, 2 would volunteer, 5 would accept, 3 would try to avoid the task, and none would refuse to undertake it. In a similar question where an employee would have to test a new improved version of the tool, 7 of the students would volunteer, 3 would accept, and none would try to avoid or deny the task.

Overall, the results indicate that it is feasible for authors with a computer science background but no expertise in natural language generation to configure M-PIRO's system for new applications, after attending an introductory course that could fit in one working day. Many of the problems that were encountered during the summative evaluation were due to problems in the design of the GUI, and most of them have now been solved; this also emphasises the importance of HCI principles in the design of such tools. The evaluation also revealed some deeper issues, namely: the difficulty to distinguish between 'interest' and 'importance'; the need for active previewing; and the need for a basic built-in ontology, possibly in different versions for different families of applications, with accompanying linguistic resources. We hope that future work will also explore efficiency issues, for example how long it takes for authors to perform various tasks.

5 Using information extraction and external databases

We have so far been concerned mostly with cases where M-PIRO's ontology and database are created from scratch by the authors. In practice, however, M-PIRO's system will often have to be ported to applications where a large volume of information is already available in databases (e.g., museum databases, retailers' databases) or collections of documents (e.g., Web pages, product catalogues). The latter case, where information exists in textual form, calls for a synergy between information extraction and natural language generation, to populate M-PIRO's database with symbolic information and snippets extracted from texts by an information extraction system; this presupposes that M-PIRO's ontology is compatible with that employed by the information extraction system. This synergy can lead to applications, where, for example, personalised real estate advertisements or job offers are generated automatically in several languages from monolingual newspaper entries.

In the remainder of this section we present work we have carried out to couple M-PIRO to an information extraction system, and to allow the authoring tool to import information from existing external databases. The facilities that we present here and in the next section, where M-PIRO's support of OWL is discussed, are less developed than those we described in the preceding sections, and they have not

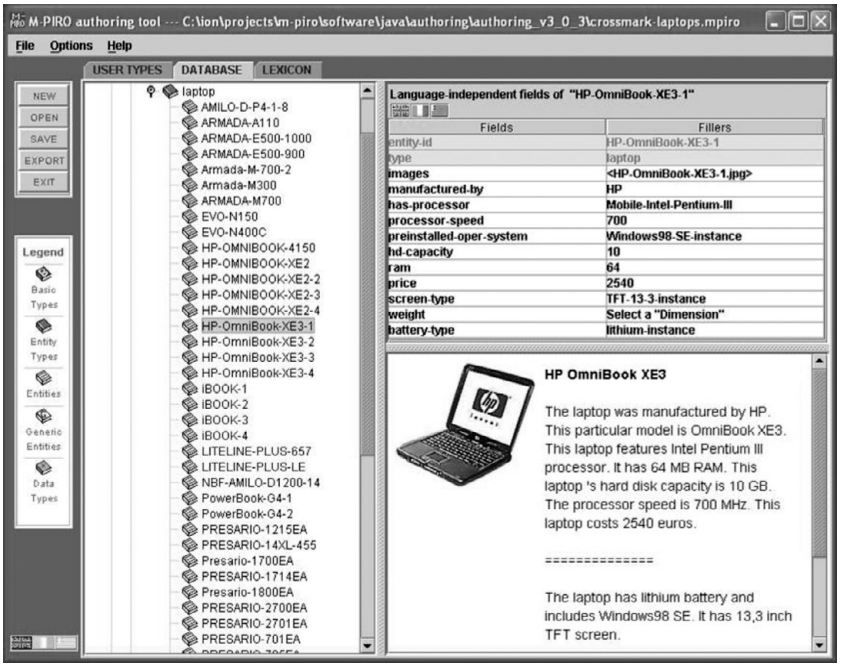


Fig. 12. An M-PIRO application that uses information extracted from Web pages.

been subjected to formal usability evaluation. Nevertheless, we believe that they are particularly interesting, as they illustrate future directions of research.

5.1 Coupling M-PIRO to an information extraction system

As a first proof of concept towards coupling M-PIRO to an information extraction system, we used M-PIRO's authoring tool to create an application for laptop advertisements (figure 12), using information that was extracted from Web pages of electronic retailers during the European IST CROSSMARC information extraction project (Hachey *et al.* 2003; Karkaletsis *et al.* 2004).[18] CROSSMARC targets multilingual information extraction, and, hence, it is a particularly good match for M-PIRO. Its ontology and templates, however, are fairly typical of information extraction systems, and, hence, resources from other information extraction systems could have been used instead. In each CROSSMARC application, information extraction specialists define in XML a background taxonomy of entity types and entities, which shows, for example, that a TFT screen is a kind of screen, that 'HP' is a manufacturer, and that 'MOBILE-INTEL-PENTIUM-III' is a processor. The specialists also define templates, that show for each entity type the fields that have to be filled in with extracted information (e.g., manufacturer and processor of each laptop), and the admissible types of the fillers. Figure 13 shows a CROSSMARC template for entities of type 'laptop'. There is also a multilingual lexicon, which lists the names of known entities in the supported

[18] See also `http://www.iit.demokritos.gr/skel/crossmarc/`.

```
<xsd:element name="laptop">
   <xsd:complexType mixed="true">
      <xsd:all>
         <xsd:element name="manufactured-by" type="manufacturer" />
         <xsd:element name="has-processor" type="processor" />
         <xsd:element name="processor-speed" type="processor-speed-type" />
         <xsd:element name="preinstalled-oper-system" type="op-system" />
         ...
      </xsd:all>
      <xsd:attribute name="language" type="document-language" />
   </xsd:complexType>
</xsd:element>
```

Fig. 13. An information extraction template from CROSSMARC.

languages, and provides terms that can be used to refer to entity types, much as M-PIRO's linking between nouns and entity types. Information extracted from documents is then stored in XML as filled-in templates.

Creating the corresponding M-PIRO application involved: reconstructing within the authoring tool CROSSMARC's laptop taxonomy; defining fields at the various entity types to mirror CROSSMARC's templates; adding entities in M-PIRO's database for entities that are mentioned in CROSSMARC's taxonomy (e.g., known manufacturers); filling in their names using CROSSMARC's lexicon; adding noun-entries (or pseudo-noun entries) from CROSSMARC's lexicon to M-PIRO's domain-dependent lexicon; linking the noun-entries to the corresponding entity types; adding an entity for each CROSSMARC filled-in template; adding micro-plans and tuning user modelling parameters. All but the last stage could in principle be automated: the authoring tool already exports all the domain-dependent resources in XML files, which are used as inputs to EXPRIMO, and one could use tools such as XSLT to convert CROSSMARC's XML resources into EXPRIMO's format. We are working on an enhanced version of the authoring tool, which will be able to read CROSSMARC XML resources, leaving to the authors only the tasks of filling in micro-plans, providing additional noun entries, and tuning user modelling parameters. Our experience from the laptops domain and a second CROSSMARC domain for job offers so far indicates that this process is feasible.

5.2 Importing information from external databases

A similar approach can be used to import information from existing external databases. In this case, one can use the authoring tool for domain authoring, i.e., to define M-PIRO's ontology, the visitor stereotypes, and the domain-dependent linguistic resources, and then replace exhibit authoring (the data entry stage) by a process that will insert into EXPRIMO's corresponding XML files new XML entries with information on individual entities (e.g., museum exhibits) obtained from an existing database via XSLT transformations.

For relational external databases, the authoring tool provides an easier to use facility, which allows entities to be created automatically from information in the external database (Kallonis 2005). The authors first right-click on an entity type in

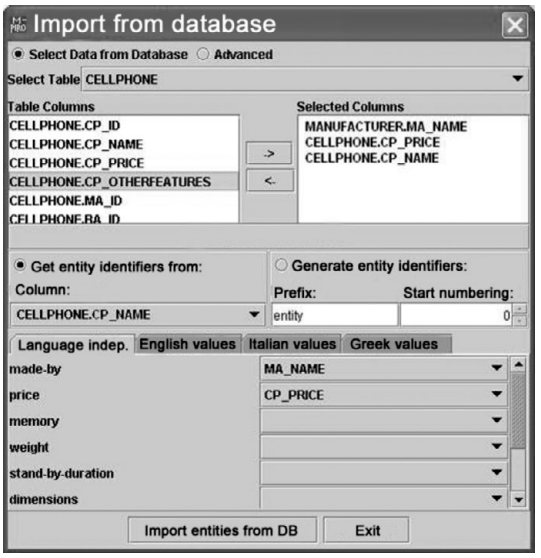


Fig. 14. Importing entities from a relational database.

M-PIRO's ontology, indicating they wish to create entities of that type. The window of figure 14 then appears. In the example of figure 14, which is from an application on cellular phones, the selected entity type was 'cellular', and each new entity represents a cellular phone model. The author selects the database tables (in our example, 'CELLPHONE' and 'MANUFACTURER') that contain information on the entities to be created, and from those tables the columns that correspond to fields of the entities (e.g., 'MANUFACTURER.MA_NAME', 'CELLPHONE.CP_PRICE').[19] In the lowest part of the window, the author specifies the mapping between the entity fields and the selected columns. In our example, the field 'price' is mapped to the column 'CP_PRICE' of the table 'CELLPHONE', i.e., the new 'cellular' entities should get their 'price' fillers from that column. The 'made-by' field is mapped to the column 'MA_NAME' of the table 'MANUFACTURER', which provides strings that are used as identifiers of the corresponding manufacturers in M-PIRO's database (in this example, the entities for the manufacturers had already been created using the same facility). In the case of language-dependent fields, it is possible to specify a different database column per language. In the middle of the window, the author specifies how the identifiers of the new entities should be created; they can be obtained from a database column (in our example, the column 'CP_NAME' of the table 'CELLPHONE'), or they can be generated automatically using a prefix and a counter (e.g., 'entity0', 'entity1', etc).

The facility of figure 14 obviously requires some familiarity with relational database concepts, though no expertise in SQL. Before it can be used, an ODBC connection to the database has to be established, but this will typically be performed once per database by its administrator. Advanced authors who are familiar with SQL

[19] Joins between the selected tables are established automatically using foreign keys.

can also issue arbitrary SQL queries, via the 'Advanced' option, and use the results of those queries instead of the database's tables.

One difficulty is that databases, especially for museum exhibits, often contain valuable information in long text fields. Information extraction techniques may again be necessary to extract names, dates, etc. from those fields, before the required information can be imported; see also Dale *et al.* (1998) for related discussion.

6 Supporting OWL

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 publishing ontologies is OWL, an ontology specification language based on XML and RDF. OWL is increasingly popular, and, as a result, publicly available OWL ontologies for many domains are beginning to appear. Tools that support the development of OWL ontologies are also becoming available, with PROTEGE being a prime example.[20]

PROTEGE is a general knowledge modelling and acquisition system, that provides a GUI for editing ontologies and allows the resulting ontologies to be exported in a variety of formats, including OWL. PROTEGE's ontology editor is similar to the 'database' tab of M-PIRO's authoring tool (figure 4). It provides a similar representation of the hierarchy of entity types, and allows its users to define fields (called *slots*) of entity types, to enter information about particular entities, etc. Authors who are already familiar with a tool like PROTEGE could use it to define the ontology of an M-PIRO application, possibly starting from a similar existing OWL ontology, and then import the resulting OWL ontology in M-PIRO's authoring tool. There are, however, some complications caused by incompatibilities between M-PIRO's ontological model and that of OWL, that the authors would have to be aware of; we discuss them below.

Note that PROTEGE and similar ontology editors cannot replace M-PIRO's authoring tool, because they do not allow the authors to specify the necessary domain-dependent linguistic resources (domain-dependent lexicon and micro-plans), tune user modelling parameters, preview texts, and iteratively modify the ontology, linguistic resources and user models until the resulting texts are acceptable. A possible target for future work, however, would be to investigate how the additional functionality of M-PIRO's authoring tool could be embedded in tools like PROTEGE, possibly as a plug-in, along with appropriate mechanisms to invoke EXPRIMO to obtain previews. Of interest is also the KAON knowledge management system, that provides an ontology editor similar to PROTEGE's, and mechanisms to associate multilingual lexicon entries with elements of the ontology.[21] Furthermore, KAON provides mechanisms to extract terms, ontology concepts, and instances from existing texts.

[20] See `http://protege.stanford.edu/`.
[21] Consult `http://kaon.semanticweb.org/`.

In the remainder of this section we report on work we have carried out to explore how M-PIRO's authoring tool can support OWL.[22] We first discuss how M-PIRO's ontologies can be exported to OWL. This allows M-PIRO's ontologies to be imported in tools like PROTEGE and to be published on the Semantic Web. As will be explained, it also opens up the possibility of generating texts that will be accompanied by machine-readable OWL entries specifying their semantics; thus, the content of the generated texts becomes accessible to computer applications (e.g., Web agents), which is a main goal of the Semantic Web. We then explore the reverse direction, how OWL ontologies, which may have been developed using other tools, can be imported in M-PIRO's authoring tool.

6.1 Exporting M-PIRO ontologies to OWL

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

When exporting M-PIRO ontologies to OWL, entity types map to class definitions. For example, the 'vessel' entity type of figure 4, a subtype of 'exhibit', 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. The 'painted-by' field of figure 4 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>
```

while the 'exhibit-purpose' field of figure 4 leads to the following datatype property, which 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>
```

[22] See Androutsopoulos *et al.* (2005) for more details on this work.

Finally, entities become OWL individuals, as in statue 'exhibit42' below. String-valued fields, like 'exhibit-purpose', lead to properties with separate values per language (in our case, English, Italian, and Greek).

```
<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...</exhibit-purpose>
 <exhibit-purpose xml:lang="IT">Questa statua...</exhibit-purpose>
 <exhibit-purpose xml:lang="GR">...</exhibit-purpose>
 ...
</Statue>
```

Note that in the preceding sections we have been using the term 'ontology' in a sense that does not include information on particular entities; but the term can also be used in a broader sense that includes entity-specific information, the information we have been treating as the contents of the database. OWL adopts the latter sense, which is why information about particular entities, like 'exhibit42' above, is included in the exported OWL ontology; this information is retrieved from M-PIRO's database.

One problem we have encountered is that OWL provides no mechanism to specify default values of properties. In contrast, M-PIRO allows default values of fields to be specified in generic entities. We export generic entities as ordinary OWL individuals, but use a special prefix in their identifiers, which allows the authoring tool to assign them special status when reloading the OWL ontology. Another system, however, that would rely only on OWL's official semantics would have no way to realize that such individuals carry default information. 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. We hope to modify M-PIRO's datatypes to correspond exactly to the recommended ones in future work.

The mapping from M-PIRO's ontologies (and databases) to OWL that we sketched above has been implemented as a facility of the authoring tool. Consequently, M-PIRO's ontologies can be published on the Semantic Web as OWL ontologies, and they can be used in other OWL-compatible systems; for example, they can be loaded in PROTEGE. More interestingly, the mapping opens up the possibility for M-PIRO to generate object descriptions in both human-readable and machine-readable forms. Every natural language description that M-PIRO produces could also be rendered in a machine-readable form consisting of OWL specifications of individuals, this time using the mapping to translate into OWL the parts of M-PIRO's ontology and database that EXPRIMO's content selection stage has decided to convey. For example, the English description of figure 5 would 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>
```

EXPRIMO might have also decided to include in the resulting text information deriving from the fields of the exhibit's painter (e.g., the city the painter was born in) or fields of other entities to be mentioned in the text. The OWL rendering of the description would then include additional specifications of individuals, such as:

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

In the machine-readable forms of the descriptions, the OWL specifications of individuals would include only properties corresponding to fields EXPRIMO has decided to convey, unlike when exporting the full ontology. That is, the OWL specifications would not include properties corresponding to facts deemed uninteresting for the particular visitor type, or facts that have been assimilated.

It is, thus, in principle possible to annotate the generated texts with OWL specifications of their semantics.[23] 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 visitor types for both human visitors (e.g., 'expert', 'average-adult') and artificial agents acting on behalf of humans of different interests and expertise (e.g., 'agent-expert', 'agent-average-adult'), and produce human-readable or machine-readable descriptions depending on the visitor type; as in the demonstrators of section 2, visitors would select their types during a login stage. 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 the agents may encounter.

6.2 Importing existing OWL ontologies

When porting M-PIRO's system to a new application, much of the authoring effort is devoted to defining the ontology of the application's domain. 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 of the domain already exists, as will increasingly be the case in the Semantic Web, the authoring process 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 and lexicon entries), tuning user modelling parameters, and populating the ontology with entities that were not already present in the imported OWL ontology.

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

[23] The mechanism to generate OWL renderings of the descriptions is currently not implemented in EXPRIMO, but it is straight-forward to do so, as it is a matter of applying the same mapping as the one used in the authoring tool to export M-PIRO's ontologies and databases, this time to only a subset of the ontology and database.

of the previous section produces ontologies in OWL LITE, but it does not exploit all the available mechanisms of OWL LITE. Hence, importing an arbitrary OWL ontology, as opposed to an OWL ontology exported by M-PIRO's authoring tool, is not simply a matter of following the inverse of the mapping of the previous section. We discuss this issue in the remainder of this section. Even with OWL LITE, we have encountered significant difficulties, which require modifications in M-PIRO's ontological model and currently force us to ignore some aspects of the imported ontologies. We hope that by reporting on these difficulties we will give the reader a taste of the complexities that future systems wishing to produce texts from OWL input will have to address.[24] In related work, Bontcheva and Wilks (Bontcheva and Wilks 2004; Bontcheva 2004) discuss how patient reports can be generated from ontologies specified in DAML + OIL, a predecessor of OWL, and RDF instance descriptions, using facilities developed within the GATE platform; Wilcock (2003a; 2003b; 2003c) discusses how a pipeline of XSLT transformations can generate dialogue responses from information represented in DAML+OIL and RDF; and Mellish and Sun (2005) investigate content selection when generating natural language renderings of OWL DL class specifications, rather than descriptions of particular instances.

One problem we encountered is that OWL (all versions) allows multiple inheritance, while the authoring tool does not. Attempting to import an ontology with multiple inheritance currently causes the import to fail. The need for multiple inheritance had also been noted by FHW curators during the formative evaluations (Section 4), who had encountered cases where, for example, a person had to be categorized as both painter and potter. As noted earlier, EXPRIMO already supports multiple inheritance; the decision not to support it in the authoring tool was taken mainly to depict the ontology in a simpler tree-like form. It appears that a better option would be to support multiple-inheritance, but offer it as an option only to advanced authors and authors wishing to import OWL ontologies.

Another problem is that all versions of OWL support property inheritance. For instance, there may be a property 'is-player-of', used to represent the relationship between soccer players and their teams, and a 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 process currently ignores 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 EXPRIMO avoid expressing information that follows from other information it has already conveyed; for example, 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 address 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 the values of each field must come from a single, named, entity type; consequently, properties of this kind are currently not imported. 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

[24] The discussion is based on experiments we conducted with more than a dozen of existing OWL ontologies; see `http://protege.stanford.edu/plugins/owl/ontologies.html`.

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 (all versions) 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 all versions of OWL (`allValuesFrom` tag) to state this, but there is no such mechanism in M-PIRO's ontological model. We currently ignore range refinements when importing OWL ontologies, though this runs the risk that authors may violate refinements (e.g., when adding individuals), creating ontologies that are no longer compatible with the imported ones. We also ignore the `someValuesFrom` tag, which is available in all OWL versions and allows stating that in set-valued properties (corresponding to M-PIRO fields with 'Many' selected) at least one of the elements of each set-value should belong to a particular class. Another mechanism in OWL DL and OWL FULL (`hasValue` tag) specifies that all the individuals of a class have a particular value at some of their properties (e.g., all wines of class 'burgundy' have 'dry' taste). This information can be included in M-PIRO's generic entities, but the correspondence is inexact, since the default information of generic entities may be overridden.

Unlike M-PIRO, all versions of OWL allow declarations of one-to-one relationships and attributes. These are currently imported as many-to-many. As discussed earlier, this problem can be solved easily by providing in M-PIRO support for one-to-one declarations. More work is needed to support the facilities that all versions of OWL provide to declare that a relationship is transitive, symmetric, or the inverse of another. All such declarations are currently ignored when importing OWL ontologies. Again, this runs the risk that the authors may modify the ontologies in ways that are incompatible with the ignored declarations. It also does not allow EXPRIMO to infer new facts (e.g., in the case of transitive relationships) or to avoid conveying equivalent facts (e.g., conveying both a relationship and its inverse).

A further problem in OWL FULL is that classes (entity types) can also be used as individuals (entities), allowing, for example, classes to participate in relationships. This is not allowed in M-PIRO (nor OWL LITE and OWL DL), and, hence, attempting to import an OWL FULL ontology that uses classes as individuals causes the import to fail. Overall, OWL FULL relaxes several constraints of OWL DL; for instance, it also allows the ontology to modify OWL's built-in vocabulary. However, this complicates OWL FULL's syntax and semantics, and removes OWL DL's guarantee of decidability. Consequently, it is considered highly unlikely that Semantic Web applications with reasoning capabilities will ever support the entire functionality of OWL FULL, and, hence, pursuing full support for OWL FULL in M-PIRO appears to be unnecessarily over-ambitious. Targeting full support for OWL LITE and subsequently OWL DL seems to be more reasonable for future work.

To summarise, OWL's semantic model is richer than M-PIRO's, which currently forces us to ignore some of the specifications of the imported OWL ontologies, or, in the case of multiple inheritance and classes used as entities, to abort the import. A reasonable target for future work is to provide full support for OWL LITE and subsequently OWL DL, but both steps require modifications in M-PIRO's ontological model. Note that OWL DL corresponds to description logics, and, hence, the required modifications in effect amount to making M-PIRO's ontological model (which has been influenced by ILEX and practical concerns, such as the desire to reuse components of ILEX) fully compatible with the more theoretically motivated ontological assumptions of description logics. Consult, for example, Baader *et al.* (2002) for an introduction to description logics and their applications in several areas, including natural language generation.

7 Conclusions and future directions

We presented the source authoring facilities of M-PIRO's natural language generation system. The system produces descriptions of objects in several languages from symbolic, language-neutral information retrieved from an ontology and a database, and pieces of canned text. The system has been tested in a variety of application domains, ranging from museum exhibitions to presentations of computer equipment for sale. The authoring facilities are intended to allow people with no previous experience in natural language generation to port the system to new applications, by modifying not only the contents of the database, but also the ontology, the domain-dependent linguistic resources of the system, and user modelling parameters that affect both the semantic content and the surface form of the generated texts. Usability evaluation results indicate that after receiving a relatively short introductory course, authors with a computer science background can successfully port M-PIRO's system to new application domains. Aspects of the authoring process can be automated by extracting information from existing external databases and document collections, and by importing existing OWL ontologies. Other OWL-aware ontology editors can also be used during the authoring process, though they cannot replace M-PIRO's authoring tool. It is also in principle possible to annotate the generated texts with machine-readable OWL entries that specify the semantics of the texts, producing descriptions that are both accessible to humans and computer applications, which is a major target of the Semantic Web.

We have pointed to several possible improvements of the system in previous sections. Active previewing, the ability to select parts of generated texts and be taken to the corresponding database entries, linguistic resources, or user modelling parameters that were used to generate them, is among the most useful possible additions, as it would speed up significantly the authoring process. The gain in efficiency could be studied in a more detailed usability evaluation. Making M-PIRO's system fully compatible with OWL LITE and subsequently OWL DL is also an important future target, as it would allow the system to exploit more fully existing ontologies of the Semantic Web. It would also be particularly interesting to explore how the additional functionality of M-PIRO's authoring facilities could be embedded

in popular ontology editors, along with facilities to invoke M-PIRO's generator to produce text previews; this would bring natural language generation closer to the knowledge-bases community and the Semantic Web. Finally, exploring further the synergy between information extraction and natural language generation may give rise to appealing applications that will transform monolingual Web pages describing products and other objects into multilingual, personalised, and thus more accessible and appealing descriptions.

Acknowledgments

We are indebted to Kostas Stamatakis, Dimitris Spiliotopoulos, Theofilos Nikolaou, Maria Prospathopoulou, and Spyros Kallonis, who implemented successive versions of the authoring tool, and Yannis Stavrakas, who developed the personalisation server and its interface to the authoring tool. Special thanks are due to Aggeliki Dimitromanolaki, who produced the Greek linguistic resources, helped test and debug the authoring tool, and played a central role in the development of the museum prototypes. We also wish to thank Jo Calder and Amy Isard, who were the main developers of EXPRIMO and helped integrate it with the authoring tool, as well as Mick O'Donnel, who created M-PIRO's initial ILEX-based generation engine and much of the Italian linguistic resources. We are grateful to all of our collaborators from M-PIRO's consortium, especially Charles Callaway, George Giannoulis, Vaki Kokkinaki, George Kouroupetroglou, George Mallen, Colin Matheson, Alexander Melengoglou, Marc Moens, Elena Not, George Paliouras, Fabio Pianesi, Maria Roussou, Michael Selway, Constantine Spyropoulos, and Gerasimos Xydas.

References

Androutsopoulos, I., Kallonis, S. and Karkaletsis, V. (2005) Exploiting OWL ontologies in the multilingual generation of object descriptions. *Proceedings of the 10th European Workshop on Natural Language Generation*, pp. 150–155, Aberdeen, UK.

Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D. and Patel-Schneider, P. F. (eds.) (2002) *The Description Logic Handbook: Theory, Implementation and Application.* Cambridge University Press.

Bateman, J. A. (1990) Upper modeling: organizing knowledge for natural language processing. In *Proceedings of the 5th International Workshop on Natural Language Generation*, pp. 54–61, Pittsburgh, PA.

Bateman, J. A. (1997) Enabling technology for multilingual natural language generation: the KPML development environment. *Natural Language Engineering* **3**(1): 15–56.

Bateman, J. A. and Paris, C. L. (1989) Phrasing a text in terms the user can understand. *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, pp. 1511–1517, Detroit, MI.

Bontcheva, K. and Wilks, Y. (2004) Automatic report generation from ontologies: the MIAKT approach. *Proceedings of the 9th International Conference on Applications of Natural Language to Information Systems*, pp. 324–335, Manchester, UK.

Bontcheva, B. (2004) Open-source tools for creation, maintenance, and storage of lexical resources for language generation from ontologies. *Proceedings of the 4th International Conference on Language Resources and Evaluation*, Lisbon, Portugal.

Biller, O., Elhadad, M. and Netzer, Y. (2005) Interactive authoring of logical forms for multilingual generation. *Proceedings of the 10th European Workshop on Natural Language Generation*, pp. 24–31, Aberdeen, UK.

Brun, C., Dyteman, M. and Lux, V. (2000) Document structure and multilingual authoring. *Proceedings of the 1st International Natural Language Generation Conference*, pp. 24–31, Mitzpe Ramon, Israel.

Busemann, S. and Horacek, H. (1999) A flexible shallow approach to text generation. *Proceedings of the 9th International Workshop on Natural Language Generation*, pp. 238–247, New Brunswick, NJ.

Cawsey, A. (1990) Generating explanatory discourse. In: Dale, R., Mellish, C. and Zock, M. (eds.), *Current Research in Natural Language Generation*, pp. 75–102, Academic Press.

Calder, J., Melengoglou, A. C., Callaway, C., Not, E., Pianesi, F., Androutsopoulos, I., Spyropoulos, C. D., Xydas, G., Kouroupetroglou, G. and Roussou, M. (2005) Multilingual personalized information objects. In: Stock, O. and Zancanaro, M. (eds.), *Multimodal Intelligent Information Presentation*, pp. 177–201, Springer.

Colineau, N., Paris, C. and Vander Linden, K. (2002) An evaluation of procedural instructional text. *Proceedings of the International Natural Language Generation Conference*, pp. 128–135, New York, NY.

Clark, R. A. J., Richmond, K. and King, S. (2004) Festival 2 – Build your own general purpose unit selection speech synthesiser. *Proceedings of the 5th ISCA workshop on speech synthesis*, pp. 173–178, Pittsburgh, PA.

Coch, J. (1996) Evaluating and comparing three text-production techniques. *Proceedings of the 16th International Conference on Computational Linguistics*, Copenhagen, Denmark.

Dale R., Green, S. J., Milosavljevic, M., Paris, C., Verspoor, C. and Williams, S. (1998) Dynamic document delivery: generating natural language texts on demand. *Proceedings of the 9th International Conference and Workshop on Database and Expert Systems Applications*, pp. 131–136, Vienna, Austria.

de Carolis, B., Pelachaud, C., Poggi, I. and Steedman, M. (2004) APML, a mark-up language for believable behavior generation. In: H. Prendinger (ed.), *Life-like Characters. Tools, Affective Functions and Applications*, pp. 65–85. Berlin: Springer.

DiMarco, C. and Foster, M. E. (1997) The automated generation of Web documents that are tailored to the individual reader. *Proceedings of the AAAI Spring Symposium on Natural Language Processing on the World Wide Web*, pp. 44–53, Stanford, CA.

Dimitromanolaki, A., Androutsopoulos, I. and Karkaletsis, V. (2001) A large-scale systemic functional grammar of Greek. *Proceedings of the 5th International Conference on Greek Linguistics*. Paris: L'Harmattan.

Dimitromanolaki, A. and Androutsopoulos, I. (2003) Learning to order facts for discourse planning in natural language generation. *Proceedings of the 9th European Workshop on Natural Language Generation, 10th Conference of the European Chapter of ACL*, pp. 23–30, Budapest, Hungary.

Hachey, B., Grover, C., Karkaletsis, V., Valarakos, A., Pazienza, M. T., Vindigni, M., Cartier, E. and Coch, J. (2003) Use of ontologies for cross-lingual information management in the Web. *Proceedings of the Ontologies and Information Extraction International Workshop*, EUROLAN 2003, Bucharest, Romania, 2003.

Halliday, M. (1994) *Introduction to Functional Grammar*, 2nd edition. London: Edward Arnold.

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

Hartley, A., Scott, D., Bateman, J. and Dochev, D. (2001) AGILE – A system for multilingual generation of technical instructions. *Proceedings of the 8th Machine Translation Summit*, pp. 145–150, Santiago de Compostella, Spain.

Hirst, G., DiMarco, C., Hovy, E. H. and Parsons, K. (1997) Authoring and generating health-education documents that are tailored to the needs of the individual patient. *Proceedings*

of the 6th International Conference on User Modeling, pp. 107–118, Chia Laguna, Sardinia, Italy.

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

Kallonis, S. (2005) *Using Existing Ontologies and Databases in the Natural Language Generation System of the M-PIRO project* (in Greek). MSc thesis, Department of Informatics, Athens University of Economics and Business.

Karasimos, A. and Isard, A. (2004) Multilingual evaluation of a natural language generation system. *Proceedings of the 4th International Conference of Language Resources and Evaluation*, Lisbon, Portugal.

Karkaletsis, V., Spyropoulos, C. D., Grover, C., Pazienza, M. T., Coch, J. and Souflis, D. (2004) A platform for cross-lingual, domain and user adaptive web information extraction. *Proceedings of the European Conference in Artificial Intelligence*, pp. 725–729, Valencia, Spain.

Kasper, R. and Whitney, R. (1989) SPL: a sentence plan language for text generation. Technical report, Information Sciences Institute, University of Southern California.

Kintsch, W. and Keenan, J. (1973) Reading rate and retention as a function of the number of propositions in the text base of sentences. *Cognitive Psychology* **5**: 257–274.

Kintsch, W. and van Dijk, T. A. (1978) Towards a model of text comprehension and production. *Psychological Review* **85**: 363–394.

Nielsen, J. (1992) Finding usability problems through heuristic evaluation. *Proceedings of Human Factors in Computing Systems*, pp. 373–380, Monterey, California.

Mann, W. and Thompson, S. (1988) Rhetorical structure theory: towards a functional theory of text organization. *Text* **3**: 243–281.

McRoy, S. W., Channarukul, S. and Ali, S. S. (2003) An augmented template-based approach to text realization. *Natural Language Engineering* **9**(4): 381–420.

Melengoglou, A. (2002) *Multilingual aggregation in the M-PIRO system*. MSc thesis, School of Informatics, University of Edinburgh.

Mellish, C. and Sun, X. (2005) Natural language directed inference in the presentation of ontologies. *Proceedings of the 10th European Workshop on Natural Language Generation*, pp. 118–124, Aberdeen, UK.

Not, E. and Zancanaro, M. (2000) The MacroNode approach: mediating between adaptive and dynamic hypermedia. *Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-based Systems*, pp. 166–178, Trento, Italy, 2000.

Nikolaou, T. (2004) *Improvements in the User Interface and Other Functions of the M-PIRO Authoring Tool* (in Greek). Final-year project report, Department of Informatics, Athens University of Economics and Business, Greece.

O'Donnell, M. (1996) Input specification in the WAG sentence generation system. *Proceedings of the 8th International Workshop on Natural Language Generation*, pp. 41–50, Herstmonceux Castle, UK.

O'Donnell, M., Cheng, H. and Hitzeman, J. (1998) Integrating referring and informing in NP planning. *Proceedings of the COLING-ACL '98 Workshop on the Computational Treatment of Nominals*, pp. 46–55, Montreal, Canada.

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

Pan, S. and McKeown, K. R. (1997) Integrating language generation with speech synthesis in a concept to speech system. *Proceedings of the Workshop on Concept-to-Speech Generation Systems, 35th Annual Meeting of ACL and 8th Conference of the European Chapter of ACL*, pp. 23–28, Madrid, Spain.

Paris, C. (1988) Tailoring object descriptions to a user's level of expertise. *Computational Linguistics* **14**: 64–78.

Paris, C., Vander Linden, K., Fischer, M., Hartley, A., Pemberton, L., Power, R. and Scott, D. (1995) A support tool for writing multilingual instructions. *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pp. 1398–1404, Montreal, Canada.

Paris, C. and Vander Linden, K. (1996) DRAFTER: an interactive support tool for writing multilingual instructions. *IEEE Computer* **29**(7): 49–56.

Paris, C., Vander Linden, K. and Lu, S. (2002) Automated knowledge acquisition for instructional text generation. *Proceedings of the 20th Annual International Conference on Computer Documentation* of the ACM Special Interest Group for Design of Communication, pp. 142–151, Toronto, Canada.

Power, R. and Cavallotto, N. (1996) Multilingual generation of administrative forms. *Proceedings of the 8th International Workshop on Natural Language Generation*, pp. 17–19, Herstmonceux Castle, UK.

Power, R. and Scott, D. (1998) Multilingual authoring using feedback texts. *Proceedings of the 17th International Conference on Computational Linguistics and the 36th Annual Meeting of ACL*, pp. 1053–1059, Montreal, Canada.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S. and Carey, T. (1994) *Human-Computer Interaction*. Addison-Wesley.

Prospathopoulou, M. (2004) *Improvements in the Exporting, the Microplans, and Other Functions of the M-PIRO Authoring Tool* (in Greek). Final-year project report, Department of Informatics, Athens University of Economics and Business, Greece.

Reiter, E. (1994) Has a consensus NL generation architecture appeared, and is it psycholinguistically plausible? In *Proceedings of the 7th International Workshop on Natural Language Generation*, pp. 163–170, Kennebunkport, Maine, USA.

Reiter, E. and Dale, R. (2000) *Building Natural Language Generation Systems.* Cambridge University Press.

Staab, S., Werthner, H., Ricci, F., Zipf, A., Gretzel, U., Fesenmaier, D. R., Paris, C. and Knoblock, C. (2002) Intelligent systems for tourism. *IEEE Intelligent Systems* **17**(6): 53–64.

Theune, M., Klabbers, E., De Pijper, J. R., Krahmer, E. and Odijk, J. (2001) From data to speech: a general approach. *Natural Language Engineering* **7**(1): 47–86.

Thompson, H. (1977) Strategy and tactics: A model for language production. *Papers from the 13th Regional Meeting of the Chicago Linguistics Society*, pp. 651–668, Illinois.

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

K. van Deemter, E. Krahmer and Theune, M. (2005) Real versus template-based Natural Language Generation: a false opposition? *Computational Linguistics* **31**(1): 15–24.

Wilcock, G. (2003) Integrating natural language generation with XML Web Technology. *Proceedings of the 10th Conference of the European Chapter of ACL*, pp. 247–250, 2003, Budapest, Hungary.

Wilcock, G. (2003) Generating responses and explanations from RDF/XML and DAML+OIL. *Proceedings of the Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, of the *18th Joint Conference on Artificial Intelligence*, pp. 58–63, Acapulco, Mexico.

Wilcock, G. (2003) Talking OWLs: towards an ontology verbalizer. *Proceedings of the Workshop on Human Language Technology for the Semantic Web*, of the *2nd International Semantic Web Conference*, pp. 109–112, Sanibel Island, FL.

Williams, S. and Reiter, E. (2005) Generating readable texts for readers with low basic skills. *Proceedings of the 10th European Workshop on Natural Language Generation*, pp. 140–147, Aberdeen, UK.

Xydas, G. and Kouroupetroglou, G. (2001) The DEMOSTHeNES speech composer. *Proceedings of the 4th ISCA Tutorial and Workshop on Speech Synthesis*, pp. 167–172, Perthshire, UK.

Zukerman, I. and Litman, D. (2001) Natural language processing and user modeling: synergies and limitations. *User Modeling and User-Adapted Interaction* **11**: 129–158.