MLRS, a Resource Server for the Maltese Language

Mike Rosner
Department CSAI
University of Malta
Malta
mike.rosner@um.edu.m

Ray Fabri
Institute of Linguistics
University of Malta
Malta
ray.fabri@um.edu.mt

Duncan Attard
Department of CSAI
University of Malta
Malta
mike.rosner@um.edu.m

Albert Gatt
Dept Computing Science,
King's College
Aberdeen, UK
agatt@csd.abdn.ac.uk

ABSTRACT

This paper decribes the aims, objectives, and current achievements of MLRS (Maltese Language Resource Server), a two-year RTDI project that has been running for just under one year. The overall aim of the project, as suggested by the title, is the provision of such language resources, on a par with those that are available for other languages, together with a framework for their collection, development and maintenance. The paper focuses primarily on the design and current implementation of the corpus server and a lexicon development toolkit. We will conclude with plans for the coming year.

Keywords

Natural Language Processing, Machine Translation, Parallel and Cluster Computing, Grid Computing, Networking, Collaborative Systems.

ACKNOWLEDGEMENT

The work described has been supported under the RTDI initiative administered through the Malta Council for Science and Technology. We are indebted to MITTS, a partner on the project, for the supply of corpus materials from Government sources.

1. INTRODUCTION

The term “language resources” refers to a set of speech or language data and descriptions in machine readable form. They are regarded as key enablers in three main areas for developments in (i) natural language processing systems and tools, (ii) linguistic research that yields new knowledge about the language itself, and (iii) language-related industries such as software localization, translation, publishing etc.

This paper decribes the aims, objectives, and current achievements of MLRS (Maltese Language Resource Server), a two-year RTDI project that has been running for just under one year. The overall aim of the project, as suggested by the name, is the provision of such language resources, on a par with those that are available for other languages, together with a framework for their collection, development and maintenance. It shares some of the objectives and approaches of an earlier project called Maltilex as described in Rosner et. al (1998), Dalli (2001).

In the project we are interested in both *primary* language resources, which can be more or less directly obtained at source, such as newspaper text or audio recordings, and *secondary* ones, which are derived using a combination of automated tools and expert knowledge. Examples include corpora marked up for parts-of-speech or other information such as named entities.

We include in particular amongst such secondary resources a *computational lexicon* - that is, a structured collection of information about words that can be used by a whole series of “language enabled” user applicatons including spell and style checkers. The compilation of such a resource demands extensive interaction with linguists, and we have catered for this in the design of the system.

Section 2 of the paper discusses the structure of the Maltese National Corpus. Section 3 describes the morpho-syntactic tagging scheme that has been developed for Maltese. Section 4 elaborates on ODL, a metalanguage for the description of lexical information whilst section 5 concentrates on implementation issues. Section 6 discusses future work. We will conclude with plans for the coming year.

2. CORPUS

2.1 Maltese National Corpus

We are in the process of building a Maltese National Corpus. The basic philosophy behind this resource is similar to the British National Corpus (BNC), its British counterpart (see Aston and Burnard [1]). There are several characteristics that we would like such a corpus to have when complete. These include

Representativeness. By representative we mean that the content – a collection of written and spoken language resources, should have characteristics and properties that reflect the language as a whole. We expect this collection to be diverse in terms of genre and subject matter (including, for example, newspaper text, fiction, technical documents in different areas, educational materials, laws etc.) and of a certain size: the BNC comprises about 100 million words and there is no reason to believe that, given adequate resources, the MNC will end up being much smaller.

Accessibility. A second important desideratum is that the corpus should be easily accessible. It goes without saying that the primary access mechanism will be the Internet, and that where possible, web services will be used to deliver resources and tools. Different categories of user need to catered for, and we are in the process of designing such services for originators, annotators, and consumers of corpus materials. An indexing scheme is in the process of being devised.

Annotation Support. Corpus materials are likely to arrive from contributors in any form, and we wish to impose as few conditions as possible on what we deem to be an acceptable form. For example, text resources received to date have been in PDF, Word, PS, and ASCII. We do not even exclude contributions on paper. At the same time, we want the possibility to annotate individual documents or sets of documents with different kinds of information and furthermore, we want to provide support for such annotation. In some cases, the annotation itself will be manual. In others – part of speech tagging comes to mind - it will be automated. In order for annotation to be possible at all, documents must conform to certain basic standards, and for this purpose, we have adopted different levels of representation, as shown in Figure 1.The idea is that the higher the level of representation, the more refined the level of automated linguistic processing.

2.2 Corpus Contents

At the outset of the project, the corpus consisted of about 1M words of news text, collected on an opportunistic basis primarily from Maltese language newspapers. The texts represent a broad spectrum of the types of articles found in these media.

Submissions are currently managed by hand using an ftp server. This is a temporary solution pending the development of a lightweight web service that will allow upload of documents and browsing of the index up to the level of filenames

The current directory structure is highly rudimentary, being divided into the following main categories, some of which are subdivided and others which may become subdivided as the corpus grows.

- **Academic:** academic and scholarly papers.
- **Blog:** weblogs
- **Chat:** multi-party text based interaction over the internet.
- **Email:**
- **Government**. Goverrnment documents, legal notices and publications. Will probably subdivide into different government offices. EU documents will also find their place here.
- **Law**. Legalistic documents including laws and judgements.
- **Literature:** Literary works of all kinds. Dwill probably subdivide into poetry, prose, and drama.
- **Press**. Newspaper text from daily and weekly newspapers. The directory is currently subdivided by newspaper (Nazzjion; Il-Mument; Kull-Hadd; Lehen)
- **Religion**. Religious and biblical texts
- **Speech**: speech data (currently empty)
- **Miscellaneous**

Table 1 summarises the current contents, not counting some documents that have been submitted on CD ad on paper.

Item	Remarks	Date Added	Size (MB)
Press		2005	1
Government		2006	123
Law		2005	25
Religion		2006	16
TOTAL			165

Table 1. Current Level 0 Contents

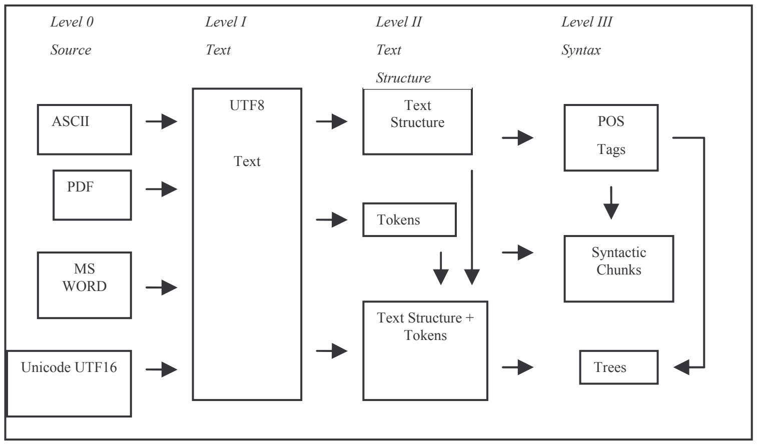

Figure 1: Levels of Representation

2.3 Corpus Representation Levels

Level 0. At the bottom of the heap, so to speak, we designate resources received in the ftp directory from the originator as being at Level 0, sometimes referred to as the source level. At level 0, very few assumptions about the nature of the resource are possible except basic information concerning author, title, date etc.

Level I. The text level is the first level at which we are in a position to define a standard.The idea here is that this represents a minimal input standard for any kind of subsequent automated processing. Whilst the mapping between level 0 and level I is undertaken on an ad-hoc basis (this includes manual conversion, use of OCR etc), we aim for completely automated mappings from level I onwards. The most important aspect is the text encoding – for us it is UTF8. Concretely, level 1 is encoded in UTF8 and includes a lightweight TEI-style (cf. TEI Consortium 2002) header identifying and certain key features of the source text including title, author, date.

Level II. This epresents a first level of actual annotation where text structure is made explicit. In contrast to level 1, we foresee several different annotation schemes operating at level II, since there is some debate about what constitutes basic text structure.

Minimally, we assume that the representation of certain characters with distinct UTF8 codes has been standardized (e.g " and " are mapped to "), tabs are mapped to spaces, consecutive whitespaces are mapped to single whitespace, and each distinguishable token appears on separate line, with blank lines indicating paragraph break. Other kinds of text structure might also be annotated at this level, e.g.

- tokenization with type information (i.e. with distinction between word, punctuation, number)
- sentence and paragraph boundaries
- higher level text structure. For example, in the LT4eL project (Monachesi et. al 2006), a "basic XML" format has been defined which includes a subset of the information present in an HTML document that is likely to be useful for the purposes of automatic content extraction: font styles, headings, lists, and so on.

There are many possible ways in which such annotations for text structure might be arrived at. One might compute them all together, or one might choose to identify the sentence and paragraph boundaries using one annotation scheme, tokenization using another scheme, finally combining the two together in a third scheme. Cristea and Butnariu (2004) describe how different annotation schemes can be regarded as points on a lattice so that paths between points automatically define pipeline computations for transforming one annotation scheme into another.

Level III Here linguistic annotation is introduced in order identify chunk structures (cf. Abney 1991) or trees.

Finally, higher levels of annotation, involving semantic information, such as named entities coreferencing etc, come in at level IV and beyong. These are not shown for the sake of clarity.

3. Morphosyntactic Annotation

This section describes a tagset that has been specially developed for Maltese. The tagset plays a crucial role in level III corpus annotation (the first level of annotation to include morpho-syntactic information). It also plays a crucial role in the lexicon, where the morpho-syntactic properties of words are actually stored. We first describe the tagset itself. This is followed by a discussion of the annotation process

3.1 Tagset

The tagset for Maltese is an extension of earlier work which drew directly on the EAGLES recommendations (Leech and Wilson, 1996). Since the original category set proposed by EAGLES proved to be somewhat limited to cover the full range of morphosyntactic phenomena in Maltese (cf. Gatt et al. 2003), this design was extended. Apart from the major grammatical categories and morphosyntactic features, tags for punctuation, as well as special particles specific to the Maltese language have also been included.

The tagset is structured such that every morphosyntactic tag is represented as a pair (C,F), where *C* is a major category, and *F* is a set of *features*, each of which is represented as an attribute-value pair. Table 2 lists the major categories, whilst Table 3 shows the full feature set.

Major categories
noun
verb
modifier
pronoun
determiner
adposition
negation
verb aspect marker
conjunction
existential
interjection
punctuation
residual

Table 2. Major categories in the Maltese tagset

Feature set	
Level 1 Features	Level 2 Features
type	dimension
person	degree
number	aspect
gender	tense
attachment	mood
negation	case

Table 3. Major categories and features in the Maltese tagset

As shown in table 3, features are divided into two levels. This division was primarily aimed at splitting the task of annotation into two stages: Level 1 features were annotated during a first pass; Level 2 features, which tend to be more limited in their distribution, were annotated during the second pass.

Within any major category, the most important feature is *type*, whose values distinguish between different subclasses of the category. Thus, elements of category *noun* can be of type *common* or *proper*, verbs fall into *main* and *auxiliary* types, and so on.

To illustrate, table 4 displays the main types of verbs and nouns (i.e. the values of the *type* attribute for these categories).

Major Category	Types
noun	1. common 2. proper
verb	2. main 3. auxiliary 4. copular 5. participle 6. modal

Table 4. Values of type for the main categories of *noun* and *verb*

The *type* for any category is meant to make distinctions that exert an influence on the morphosyntactic structure of a phrase, rather than semantic distinctions. For example, the personal pronouns in Maltese have a copular function, exemplified in by *huwa* in (1); in such cases, the pronoun is marked as copular verbs.

(1)	dak	huwa	ħija
	dem-3SgM	he-3SgM (Pro-3SgM)	brother-1Sg-Gen
	'that	is	my brother'

Clearly, not all features are defined for all categories. Thus, *aspect* pertains only to elements of the verbal category, while *case* is only annotated in two cases: that of nouns explicitly marked for the genitive (which is limited to a subset of nouns in the Construct State, exemplified in (2)), and the adposition *lil*, which functions as an non-nominative marker, as shown in (3).

(2)	Mart	Toni
	wife-gen	Tony
	'Tony's wife'	

(3)	Pietru ċempel	lil	Ġanni.
	Peter call-P3Sg	prep-*non-nom*	John
	'Peter called John'		

One of the advantages of this layout is that it is possible to define which features pertain to a particular category type. This is important, in that the application of features such as *tense* and *aspect* is not always straightforward (cf. Fabri, 1996 for discussion). Thus, it has been argued that main verbs in Maltese are only inflected for aspect, rather than tense. The exception is the auxiliary *kien*, which functions as a tense marker, distinguishing between *past* and *non-past*. A further category, that of verb aspect markers, deserves special mention. This category was introduced to accomodate particles whose function is to further mark the event structure (*aksionsart*) of verb phrases, over and above the information supplied by the inflectional morphology of the verb itself. The category could not be incorporated with verbs, because none of the features that apply to the different verb types applies to them (e.g. they are not inflected for gender or number). Moreover, the *aspect* attribute does not apply to these elements, because it is incorporated in the *type* attribute. Examples of the two types of aspect markers, *progressive* and *prospective* are shown in (4) and (5).

(4) kien	qed	jiekol
be-past	prog	eat-prog
'he was eating'		

(5) kien	se	jiekol
be-past	fut	eat-prog
'he was about to eat'		

Another category, that of *modifiers* incorporates both adverbs and adjectives. These are distinguished by the *type* attribute. The reason for this is that there is a large class of modifiers with both an adjectival and an adverbial function in the Maltese language. Distinguishing these using different categories would fail to capture a potentially useful generalisation for future users of the corpus. Of the other features, one of the most important is *attachment*. This was inherited from the original EAGLES recommendations, and distinguishes between morphosyntactically independent elements, and clitics.

3.2 Level III Annotation

Morpho-syntactic annotation of the corpus is highly labour intensive, and yet is a prerequisite of almost any semantic analysis of corpus materials. Clearly, an automatic solution needs to be devised using an appropriate part of speech tagger. Many tagging algorihms are available. We are currently using the Rule-Based Tagger developed by Brill (1996), to automatically annotate increasingly large samples of the corpus.

The work is being carried out in three stages. During the first stage, a random sample of ca. 3000 words was taken from the newspaper corpus. This was manually annotated with Level 1 features during a first pass. Following validation of the output of the first pass, Level 2 features were added. During this initial phase, the tagset was expanded to accomodate distinctions that were deemed useful because of their frequency. For instance, the category of *existentials* was introduced at this stage (following the practise adopted for Version 2 of the CLAWS tagset for the British National Corpus), to mark existential uses of demonstrative pronouns, as shown in (6).

(6) hawn	wieħed raġel
dem-3Sg/Ex	one-SgM man
'here is a man'	

Following this initial phase, a new sample, of a further 3000 words was collected, and annotated manually.

Our decision to use the Brill tagger was motivated by the fact that it encodes generalisations made from training data as rules which can be manually edited for corrections. The automatic annotation is proceeding in train-test-retrain cycles. Small samples are annotated using the tagger and manually corrected. Subsequently, the newly annotated samples are added to the training data and the tagger is retrained.

Future work on annotation is now proceeding in two directions. The first, and most urgent, is the completion of the automatic annotation for a large text sample. The main sticking point here is the provision of training data for the tagger.

Secondly, we are investigating the relationship between the tagset and the contents and organisation of entries in the lexicon. For this to be possible we need to have very clear ideas about :

- The nature and representation of tag information;
- the relationship between information carried by tags and information carried by lexical entries;
- the nature and representation of lexical information;
- the sharing of information between lexical entries.

For the purpose of answering these questions as well as developing a practical system we are developing a special-purpose description language called ODL (Object Description Language) for lexical information. The language provides a way for the linguist to specify what constitutes a possible lexical entry taking account of internal dependencies and constraints between data fields,

A key component of this investigation concerns developing a simple description language to handle inheritance of features and values within the lexicon. This is based on the kind of inheritance found in object oriented programming languages, hence the name: ODL (Object Description Language). This is discussed in the next section.

4. OBJECT DESCRIPTION LANGUAGE (ODL)

4.1 Lexical Information

ODL is a very simple object-oriented description language for lexical information. It is based on a notion of class which is very similar to that used in object oriented programming languages such as Java or C#.

The main assumption underlying the language is that lexical information, i.e. the information associated with lexical entries, is expressed in the form of classes, and that a class is a collection of attribute/value pairs. This approach has a long tradition, underlying the constraint-based unification grammar approach to language processing typified by, e.g. PATR2 (Shieber 1984), FUG (Kay 1985) and many others (GPSG, HPSG, TFS etc).

An important and useful aspect of being able to create classes (e.g. part of speech classes) is that of providing each with a general default set of attributes and values which would first of all be applied automatically to new words entering the lexicon, and secondly would list only those values which are applicable to certain attributes.

The language thus includes three classes of symbol which, for the purpose of this article, are distinguished notationally as follows:

- Classes (written in uppercase)
- Attributes (written with initial uppercase letter)
- Values (lowercase or integer)

Examples of classes might be `NOUN, PROPER_NOUN, COMMON_NOUN, VERB, TRANSITIVE_VERB`, etc. Typical attributes for these classes might be `Category, Number, Gender, Case`, and typical values of the respective attributes might be `noun, sing, masc, gen`. With these symbols we can define arbitrary sets of attributes and values.

A Maltese word like *kelb* (dog), for example, might have a lexical entry that comprises the following set of attributes and values:

```
{(Category,noun),
 (Number,sing),(Gender,masc) }
```

Clearly, only some collections of attributes and values make sense, linguistically speaking. For example, the following is linguistic nonsense

```
{(Category,verb),
 (Case,gen),(Dimension,masc) }
```

The idea behind an ODL description is (a) to precisely characterise the set of well formed classes and (b) to serve as a formal basis for the efficient compilation of the entire system of well-formed classes.

Currently we are experimenting with the possibility of flattening out classes into a bit string which is stored as a single record in a conventional database table. The result is that the database data structure, based on the ODL code, is compiled and stored prior to the building of the actual lexicon.

An ODL description contains the following parts in order:

1. Enumeration Declarations
2. Class Declarations
3. Rules (Optional)
4. Replacement Declarations (Optional)
5. Macro Definitions (Optional)

The next sections discuss each of these in turn

4.2 Enumeration Declarations

These enumerate the set of allowed values for a given attribute:

```
enum Cat      {noun, verb, adj, pronoun}
enum Type     {common, proper, main, aux}
enum Number   {sing, dual, plur. *}
enum Case     {gen, non-nom, * }
```

The value * is a special wildcard symbol which allows an attribute of a word in the lexicon to remain "unspecified". During subsequent processing, this unspecified value is consistent with the any of the normal values in the enumeration.

The absence of * in an enumeration implies that a normal value *must* be present for a well–formed class involving that attribute.

4.3 Class Declarations

These define the attributes and values that make up a class, e.g.

```
class NOUN
{
  Cat           = noun;
  Type          = common | proper;
  Number        = *;
}
```

The compiler checks that the class declaration is consistent with the enumerations already specified.

The example shown is valid because {noun}⊆Cat, {common,proper}⊆Type, and Number is unspecified.

4.4 Class Inheritance

As in object oriented programming languages, the class system supports inheritance. For instance:

```
class PRONOUN: NOUN
{
 Case = *;
}
```

states that the `PRONOUN` class extends the `NOUN` class by adding a `Case` attribute to the set of attributes already belonging to NOUN.

There is of course potential for conflict in such a system. For instance, noticing that, as a consequence of inheritance, PRONOUN will have Cat = noun, we might wish to write:

```
class PRONOUN: NOUN
{
 Cat  = pronoun
 Case = *;
}
```

for which the value of the `Cat` attribute conflicts with that inherited from `NOUN`. The solution to all such conflicts is to establish precedence rules which allow information to be overwritten in a systematic way. In this case we require that he more specific inherit<u>ing</u> class (`PRONOUN`) to overwrite information supplied by the more general inherit<u>ed</u> class (`NOUN`).

Sometimes we will want to define a class which inherits attributes and values from more than one subclass. In this case, we write

```
Class CLASS: CLASS1, CLASS2,…,CLASSN
```

Once again, the conflict can be resolved provided that we can establish unambiguous precedence relations between the classes.

This problem is not a new one, having been discussed extensively within AI/Knowledge Representation communities some 20 years ago. In the special case of language, several articles on inheritance and the lexicon appeared in a special issue of Computational Linguistics (Gazdar & Daelemans 1992).

We are still experimenting with an appropriate solution but in the first instance, propose to follow the example of Russell et. al. (1992) by using an algorithm based upon that used in Python 2.3. This is currently described on the Python website at www.python.org.

4.5 Rules

A well-known fact about Maltese is that plural entities are without gender. This fact implies that any well-formed class having a `Number` attribute equal to `plur` should not have a `Gender` attribute. ODL includes rules of the following kind to state such information:

```
if (Number == Plural) { !Gender }
```

This kind of information is used to maintain the well-formedness of defined classes in two ways:

- to prevent invalid information from entering the system in the first place (e.g the lexicon editor should disallow any attempts to provide a value for gender if the number is equal to plural) and
- to ensure that any combinatory operation that is employed by the system (e.g. class inheritance) never produces a class that conflicts with the rules.

Since these are the only ways classes can come about, we can thus guarantee that information represented by classes is completely consistent with the linguist's specification.

Currently, these rules are employed at user interface level. If a class satisfies the rule antecedent, the consequent is automatically asserted.

Where a consequent has the form !Attribute, as in the example above, we assign the distinguished symbol "~" as value of that attribute. This behaves just like any other constant, in that an attribute having it as value will only combine with another attribute having the same value. In other words, there is no possibiity of combining a non-existent attribute with one that exists.

Finally, unlike other attributes, we require that those having "~" be invisible to the user. This will be taken into account by the display algorithms which are still being designed.

4.6 Replacement declarations.

These rules are used displaying attribute values in a shorthand fashion. To illustrate, the rule

```
from Category replace Noun with n;
```

indicates that Noun is usually written as n. in a dictionary

4.7 Tag Definitions

The tagset described in section 3 is in the process of being linked to the object system. This is achieved by defining each tag using tag definitions.

An example of a tag definition would be:

```
tagdef NNSM on Noun
{ Category = Noun;
  Type     = Common;
  Number   = Singular;
  Gender   = Masculine; }
```

A tag is defined on a particular class, and thus, can only include attributes which are allowable in that class. Therefore, a tag defined for the *Noun* class cannot be applied to a *Verb* class and vice versa. Moreover, the attributes in the tag defined on the *Noun* class must exist in the *Noun* class itself. Finally, the values specified must be normal values (not wildcards and no disjunctions).

5. Implementation

The project is in an intermediate state of implementation.

A database server provides the central storage for the system, and all transactions that take place are processed on the server.

There are currently two main databases: one for handling corpora as described in the section 2 of the paper, one for the lexicon, and one for have been created were created for this purpose. The corpus database holds information about corpus documents. one which maintains the wordlist and serves as an outside contact with the external world, and the other which serves as the actual lexicon storage, which in turn, can only be modified internally.

5.1 Corpus Database

In order to support multiple levels of annotation, our corpus is organized along two dimensions. At the top level is a catalogue of Level I raw text files. File information, which currently includes author, category of text and other attributes are maintained in a database table – as depicted by the table structure in figure 2 pointing to individual untagged files (shown shaded solid). In parallel to this, every raw file is associated with an arbitrary number of Level II files representing different levels of annotation. These different versions of the same file are depicted using different kinds of stripe in the same figure.

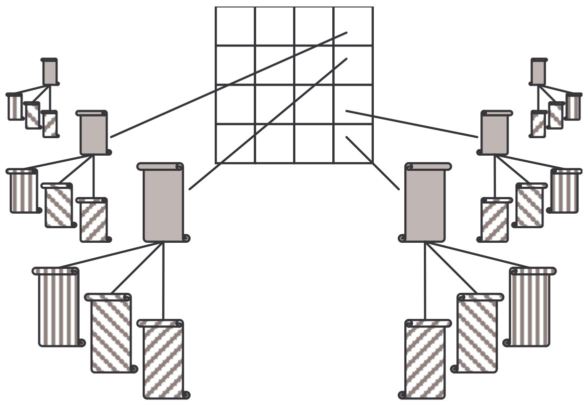

figure 2. Corpus Database

.

5.2 Lexicon Database

The lexicon itself is implemented as a database of entries which provide information about words. In order not to prejudge the kind of information that might be represented, we assume that, conceptually at least, it takes the form of attribute value pairs. We have seen how in section 3 how this works for morphosyntactic information, and subsequently, we hope to extend the approach to deal with the semantic and phonological properties of words.

The current implementation is structured around a general purpose database table containing information the word itself, its stem or root, and associated morphosyntax. The latter property was implemented as a long integer bit field, since this is extremely efficient for the purpose of when queries are made. Using a bit field to represent all the possible parts of speech implies the following assumption however: the total number of parts of speech values must not exceed 63. For simple parts of speech, this is more than enough. However, for more general attribute-value structures, it is is clearly inadequate.

Clearly, a database based on hierarchical feature structures provides for a more logical and elegant data representation. However, a disadvantage of such systems is that at the moment of writing is that they are not able to efficiently cope with large amounts of data.

A challenge is therefore going to be how to reconcile the efficiency of the relational database approach with flexibility of feature structures.

This can be fairly easily achieved by relational databases, though admittedly, these do not provide such an elegant and hierarchical structure to the data within since they are oriented towards general purpose solutions.

A issue which we will have to revisit is how to best harness the limited capacity but efficient bit string implementation with the more general, more elegant but less efficient representation offered by feature structures.

A simple list of records lying in a database can to some extent provide reasonable ‘dynamic’ structure through the use of queries. Thus, if for example, all the nouns are required, comparing a parts of speech bit fields with each other should be fairly efficient. The drawback with this approach is that queries must be computed in real time.

5.3 Lexicon Editor

The system currently includes a Lexicon Editor (Led) whose main function is to enable a linguist to create and maintain lexical entries. Led uses ODL in two ways.

At start-up, the lexicon editor parses the ODL file containing all the class, rule and macro definitions. Then, it modifies its word-entry interface to reflect the classes found in the ODL file, together with its attributes and values each one can take.

Since the database back-end receives data which is already in relational database format, the lexicon editor takes on the important role of processing the ODL definitions which then are applied on new words, thus transforming them correctly into database records.

Apart from allowing additions and deletions, the lexicon editor also caters for the editing of entries as well as the construction of queries using a query editor. It must be said that the ODL file read in the editor at start-up is verified and checked to see that it has not be modified, as such modifications can compromise the lexicon. One central ODL file, residing on the server is maintained, and any remote lexicon editor starting up compares its ODL file version with the one maintained on the server. If the former or the latter differ, the ODL file on the server is automatically downloaded on the client, parsed and used to start up the lexicon editor.

5.4 Wordlist Database

The wordlist database, which essentially is open to anyone who would like to submit words, feeds the lexicon in a controlled manner, the lexicon administrators deciding which words should go in and which should be rejected

5.5 Website and Web Services

The system resides at http://mlrs.cs.um.edu.mt The website currently provides some basic public information about the project, access to ftp services for uploading source materials and downloading updated project documents..

Since the system will in the future be inevitably used by other applications or projects, which may or may not be written for the same framework in which the system in question is written, web services have been used wherever possible to mediate database access. Currently web services are mainly provided to support lexicon construction as discussed in greater detail in section 3 of this paper.

The starting point for lexicon construction is a wordlist, and we therefore provide a web services to support a web interface for extraction of words from documents, and maintenance of wordlists. The functionality of the service is implemented as follows:

1. User submits text, files or page URLs.
2. These resources are scanned and the words extracted from them and displayed.
3. User edits the resulting lists of extracted words manually
4. User submits final version for incorporation into the wordlist database.

The wordlist submission website also provides a simple interface for submitting documents to the Corpus Database mentioned in section 5.1

6. FUTURE WORK

6.1 Web Portal

Since we are developing a national resource, we place a very high priority on the public profile of this project. The current website does not yet add up to a portal containing a an integrated suite of tools and resources. Moreover, the contents are only partially accessible to members of the public.

Our immediate goals is to remedy this shortcoming by developing some simple public web services (as opposed to raw ftp) which will allow submission of primary content, and browsing of primary content filenames. For legalistic reasons, viewing of primary content will be restricted to registered users. A legal framework within which defines the copyright of submitted materials is currently in preparation.

In the medium term, the web services available will depend largely upon the availability of the associated linguistic tools. Some of these, (e.g. POS tagging) are relatively straightforward to implement Others (e.g involving morphological analysis) raise further questions and will take longer.

6.2 Lexicon Structure and Morphological Analysis

The present lexicon is full form. This means that even though words such as *kiteb* (book) and *kotba* (books) are closely related, they have entirely separate lexical entries. Clearly, there is considerable scope for the sharing of lexical information between such entries. We want to write the lexical information just once, associate it with the most general entry (in this case the singular word, or possibly the root k-t-b) and arrange for the information to be inherited by more the more specific variant or variants.

This has two implications. One concerns lexical structure and inheritance. If the information associated with lexical entries are classes, we need to elaborate not only on the inheritance mechanisms at work, but also on various categories of lexical entry. Which are the most general forms? Are there special classes which have the role of abstract classes not attached to an actual word - for example - triconsonantal roots? We are looking for an appropriate lexical structure, that is, a *system* of classes and associated inheritance mechanisms that fit the special case of Maltese.

The second issue concerns the ability to recognize the relationship between the word forms through morphological analysis. This is particularly important for a morphologically rich language like Maltese where a word can have tens and possibly hundreds of different forms. Unfortunately, a *computationally tractable* description of Maltese morphology is not available off-the-shelf. Some preliminary work has been carried out using a finite state approach (Rosner 2003) but much work remains to be done.

6.3 Semantic Information: Wordnet

The incorporation of semantic information into lexical entries is not only important is not only a standard feature of dictionaries and thus in its own right, but essential for certain classes of application program. Examples include automatic query expansion (synonyms required), document classification (are the high frequency words associated with domain X or Y), and of course machine translation.

Fortunately, quite a lot of the work has already been done for other languages in this area by the WordNet community (Fellbaum 1998) who have succeded in creating a lexically based hierarchy of concepts called *synsets* for English together with various interfaces for accessing the information We intend in the first instance to include references to the EuroWordNet concept hierarchy (Vossen 1998) into our own lexical entries.

6.4 Linguistically Sensitive Tools

The longer term aim of the groundwork being carried out in this project is the creation of computational tools that make use of the information available in the lexicon. Many such tools are envisaged.

A tokeniser is already available. A part-of-speech tagger will become available before the project is finished, and this will enable is to tag large quantities of text.

In the immediate future, we hope to develop a chunker for recognizing noun and verb groups will be within our grasp soon afterwards. These will act as enablers for more elaborated document-processing tools including those for exploring semantic content.

In the medium term we see grammar and style checking as an achievable goal, but these depend upon the elaboration of an appropriate grammar model for Maltese. Such work involves exactly the kind of cross-disciplinary effort that the project is intended to promote.

7. CONCLUSION

This paper represents ongoing work, describing the first year of a two year project. Many of the loose ends that have been identified will be resolved before the end of the project. Others of course, remain long term research goals, and we hope to be able to participate in their achievement beyond the confines of the project.

8. REFERENCES

[1] Guy Aston and Lou Burnard. The BNC Handbook, Exploring the British National Corpus, Edinburgh University Press, 1998.

[2] Eric Brill, A Corpus-Based Approach to Language Learning, PhD Thesis, University of Pennsylvania, 1993.

[3] National Corpus with SARA. Edinburgh University Press, 1998.

[4] Dan Cristea and Cristina Butnariu. 2004. Hierarchical XML representation for heavily annotated corpora. In *Proceedings of the LREC 2004 Workshop on XML-Based Richly Annotated Corpora*, Lisbon, Portugal.

[5] Dalli, A. (2001) Interoperable Extensible Linguistic Databases, Proc. IRCS Workshop on Linguistic Databases, University of Pennsylvania

[6] Fellbaum, C, (1998) Wordnet, an Electronic Lexical Database, Cambridge: MIT Press.

[7] Gazdar G., Daelemans W. (1992) (eds), Computational Linguistics, vold 18.2 and 18.3, Special Issues on Inheritance I and II

[8] Paola Monachesi, Dan Cristea, Diane Evans, Alex Killing, Lothar Lemnitzer, Kiril Simov, Cristina Vertan. *Integrating Language Technology and Semantic Web techniques in eLearning*. To appear in *Proceedings of ICL 2006*

[9] Leech GN, Wilson A 1996 *EAGLES Recommendations for the Morphosyntactic Annotation of Corpora*. EAGLES-Guidelines EAG--TCWG--MAC/R. Final version of 3.1996. EAGLES, Istituto di Linguistica Computazionale, Pisa.

[10] Gatt, A., Vella, A., and Caruana, J. (2003). Annotating textual and speech data in Maltese. Technical Note ISO/TC 37/SC 4, International Standards Organisation Language Resource Management.

[11] Fabri, R. (1996). Kongruenz und die Grammatik des Maltesischen. Tubingen: Niemeyer.

[12] Kay, Martin (1985). "Parsing in functional unification grammar." In *Natural Language Parsing,* edited by David R. Dowty, Lauri Karttunen, and Arnold M. Zwicky, 251-278. Cambridge University Press

[13] TEI Consortium (2002), Guidelines for Electronic Text Encoding and Interchan (TEI-P4), University of Virginia Press

[14] Rosner M. (2003), Finite State Models of Linguistic Phenomena in Maltese, Proceedings of CSAW03, University of Malta.

[15] M. Rosner, J. Caruana, and R. Fabri. (1998) Maltilex: A computational lexicon for maltese. In M. Rosner, editor, Computational Approaches to Semitic Languages: Proceedings of the Workshop held at COLING-ACL98, Université de Montréal, Canada, pages 97–105, 1998.

[16] Russell, G., Carroll, J., Ballim, A ,.Warwick-Armstrong, S, (1992), A Practical Approach to Multiple Default Inheritance for Unification-Based Lexicons, Computational Linguistics, vol. 18.3, pp311-337.

[17] Shieber, Stuart M. (1984). "The design of a computer language for linguistic information." In Proceedings of 1984 International Computational Linguistics Conference, Stanford, pp 362-366

[18] Vossen, P, Introduction to EuroWordNet,, in Ide N., Greenstein D. and Vossen, P, (eds), Special Issue on EuroWordNet, Computers and the Humanities, vol 32 nos 2-3 pp 73-8