Proceedings of the International Workshop on Resources and Tools in Field Linguistics, Las Palmas, Spain. May 2002.

A unified system for accessing typological databases

Paola Monachesi*, Alexis Dimitriadis*, Rob Goedemans†, Anne-Marie Mineur*

*Utrecht University, Uil-OTS
Trans 10, 3512 JK Utrecht, The Netherlands
{Paola.Monachesi, Alexis.Dimitriadis, Anne-Marie.Mineur}@let.uu.nl

† Leiden University
Leiden, The Netherlands
R.N.W.Goedemans@let.leidenuniv.nl

Abstract

We present the goals and architecture of the Typological Database System, a project for the creation of a unified interface to numerous independently developed typological databases. The aim of the project is to develop a software system that allows a user to simultaneously query different databases through a single interface. The challenge of the project lies in the variability of the included data. In order to overcome the diversity, the system relies on detailed formal descriptions (metadata), prepared in advance and describing in detail the structure and content of each component database. The metadata is used to match a user's query against the capabilities of the component databases.

1. Introduction

Typology, the study of the variation that language exhibits, is one of the most important and interesting fields within linguistics. Typological databases are a valuable tool for this enterprise, and numerous typological databases have been developed by researchers in the field, often for personal or small-group use. Increasingly, these databases are being made available to the linguistic community over the Internet, providing the potential for enormous increases in the power of exploratory typological investigation. However, these databases can be quite heterogeneous. A typologist seeking information on a particular subject may find parts of it scattered over several databases, organized in various forms and expressed in ways that reflect different research traditions. As the number of potentially relevant databases increases, so does the amount of effort required by a user to locate them, understand how they are organized, figure out the query system and perform a query, and interpret the results.

The aim of the *Typological Database System (TDS)* project is to facilitate this process, by developing a software system that allows a user to simultaneously query many different typological databases through a single interface. This system, which is currently under development, will reside on a server computer that a user can query over the internet by use of a standard web browser. The component databases can in principle reside in separate, remote servers, although for performance reasons the server may need to maintain local copies of some or all of them. By accessing the single gateway site, the user will gain access to the data contained in all the databases participating in the project. The goal is for the system to behave as much as possible like a single, *virtual* database.

The TDS project (*http://www-uilots.let.uu.nl/td/*) is being carried out by a research group in the Netherlands Graduate School of Linguistics (LOT), with members representing the Universities of Amsterdam, Leiden, Nijmegen and Utrecht. A number of typological databases developed by participating researchers constitute the initial components of the TDS, and have been providing us with concrete experience of the problems that need to be addressed.

2. Exploring multiple typological databases

The problem of managing and presenting information becomes ever more important as the information available on the internet grows (Abiteboul et al., 2000). As more typological databases become accessible to users other than their creators, colleagues, and others already familiar with them, the following tasks become more challenging:

1. **Resource discovery.** This is simply the step of finding a datasource that contains information on some given topic.
2. **Correct and effective use.** As we have stated, databases use varying terminology, notation, organization of the data, and search commands. Even if these are documented in detail, they can be quite difficult for a new user to assimilate and employ properly.
3. **Efficiency of resource utilization.** As the amount of online information grows, the time and effort involved in searching databases one by one and collating the results becomes an obstacle to their efficient utilization.

The first of these problems is being addressed by various initiatives that are currently developing standards for resource description and discovery, including the Dublic Core Metadata Initiative (general) (Dublin, n.d.), the Open Language Archives Community initiative (linguistics specific metadata and harvesting protocol) (OLAC, n.d.), and the International Standards in Language Engineering (ISLE, n.d.) metadata initiative.

Our project directly addresses the second and third tasks. The unified interface of the TDS will allow speedy combined searches from one set of screens, using as

$V27$	PRED ADJ AGR
Predicative adjs agree with the subj in nb and/or gender	
$V106$	ATTR ADJ AGR CASE
Attributive adjs agree with their nominal heads in case	
$V456$	VERB FLEX SUBJ
Finite verbs agree with their subjects	
$V469$	FLEX ORDER = VERB-TMA-X
In a V the morpheme order is Stem-Tense/Mood/Aspect-Agr	
$V475$	DEF ART
Finite verbs agree with their subjects	
$V469$	FLEX ORDER = VERB-TMA-X
In a V the morpheme order is Stem-Tense/Mood/Aspect-Agr	
$V475$	DEF ART
The definite article is obligatory	

Table 1: Example from the Typological Database Nijemegen

much as possible a consistent interface, terminology, and command language. The results will be summarized and perhaps transformed for more efficient presentation. The TDS project will explore the limits of combining diverse databases and presenting them as a unified virtual database.

3. The component databases

Several databases developed by participating researches constitute the core of the TDS project and have provided us with concrete experience of the problems that need to be addressed. In the rest of this section we provide a brief description of the component databases.

- **Stresstyp database (University of Leiden)**. This database encodes information about the stress system of 500 languages. It contains both analytic data and examples. Software: 4th dimension.
- **Word order database (University of Amsterdam)**. The database contains analytic data about word order for 150 languages. Software: Microsoft Access.
- **Person Agreement Database (University of Amsterdam and Lancaster University)**. The database contains analytic data about agreement for 400 languages which constitute a balanced sample of all the languages of the world. Software: Microsoft Access.
- **Typological Database Nijmegen (University of Nijmegen).** The database contains analytic data on various topics including basic word order, intransitive predication, case marking, temporal sequencing, relative clause formation, comparatives, possessive constructions, verbal morphology, tense/aspect, noun phrase coordination, manner adverb encoding, verbal derivation. The number of languages varies from topic to topic, with a minimum of 140 for all topics, and a maximum of 410 for some topics. Software: MS Access.
- **Spinoza database (University of Nijmegen/Leiden)**. The database will encode information about 100 languages. Ortographic, phonological, morphological and syntactic information is provided. It is a database of sample sentences annotated with glosses and translations. An analytical component is currently being developed. Software: Microsoft Access.
- **Anaphora database (Utrecht University)**. This database contains exhaustive descriptive information on the linguistic behavior of anaphors (i.e., reflexives and reciprocals). It includes some information on ordinary pronouns, and general descriptive information on each language. The information will be limited to a relatively small, but typologically diverse sample of approximately 50 languages. It contains both analytic data and examples. Software: SQL server.
- **Aspect database (Utrecht University)**. The database will contain in-depth information on about 25 languages (mainly Slavic, Romance and Germanic). The data consists of collected examples (which will be annotated) and text articles about (sets of) examples, covering the field of aspect and related areas like tense, quantification and argument structure. Software: SQL server.
- **Inflection/agreement database (Utrecht University)**. The database focuses on agreement morphology in a number of typologically distinct languages, and its relation to the occurrence of empty subjects, empty objects and the syntactic position of the verb in the clause. Software: SQL server.

4. Obstacles to combining the databases

The aim of the TDS project is to combine diverse databases (including the ones discussed above) and present them as a unified virtual database. The challenge lies in the great heterogeneity of the included data. The diversity can be of several types.

Diversity of content type Typological databases consist mostly of logical variables describing each language as a whole. Table 1 illustrates a fragment of the *Typological Database Nijmegen* which employes variables to encode information about various language phenomena. For example variable $V456$ pertains to agreement between the subject and the verb.

Text line:	1
Representations	
Orthographic	Njadi nuna jàka na-laku-ka i Umbu Ndilu nàhu la woka
Phonemic	ndj\adi nuna dj\Aka na=laku=ka=l Umbu Ndilu nAhu la wOka
Morphological	njadi nu-na jàka na=laku=ka i Umbu Ndilu nàhu la woka
Morphological Gloss	thus DEM-3s if/when 3sNom=go=PFV DEF lord male.name LOC garden
Idiomatic	So, that one, when Umbu Ndilu goes to the garden

Table 2: Example from the Spinoza Database

However, several of the component databases in this project contain example sentences with detailed annotations, as can be seen in table 2 which contains an example from the *Spinoza database* with several levels of description. The ultimate goal of the project is to integrate different types of content so that, for example, a single query could return both examples and logical variables as an answer.

Diversity of theoretical commitments Because there is no single, universally accepted, and exhaustive linguistic theory, the information in the various databases reflects the analytical and theoretical commitments of its creators. A linguist can recognize the descriptive content of a statement based on identifiable assumptions. The TDS project will place a high priority on preserving and making visible the framework of assumptions that will allow a (linguist) user to properly interpret any data extracted from a component database.

Diversity in form In many cases the different databases use equivalent, or near-equivalent, ways of describing data. One obvious example is the use of different abbreviations for broadly accepted linguistic notions, such as *accusative* or *plural*. There is also great variation in the choice of abbreviations used to label properties such as part of speech, gender and agreement features, etc. It is generally easy to reconcile purely notational differences, but the definitions of such notions can also differ in their details. It is thus necessary to establish guidelines for distinguishing notational variation from theoretically important differences, and to normalize the data with respect to the former but not the latter.

Different design choices Beyond purely notational issues, there are many ways to organize a given body of information into a database. The decisions made by the creators of various databases in organizing similar information introduce an additional element of variation that must be compensated for.

Different software Databases are embedded in various database management systems (DBMSs) representing different generations of software and adhering to different conventions: SPSS, SQL Server, Access, 4th Dimension, custom-made database engines, etc.

Consultations with the community of prospective users have established that the preservation of the specific claims made by the creators of the individual databases is of the utmost importance. Extensive normalization of the collected data into a common form would result in unacceptable distortion. Therefore a major focus of research will be the question of how to best strike a balance between improving usability and preserving the reliability of the collected information.

5. The architecture of the system

Figure 1 shows the general structure of the Typological Database System (TDS). Its core component is a database server that acts as an intermediary between the user and the component databases. Communication between the user and the central component is performed exclusively via a web interface. The protocol that is used to communicate with the various databases depends on the type of the database, but the user is not aware of that: the translation of the user query into the proper format is performed at the server. From the user's perspective, the system behaves as a single but very heterogeneous database.

The system relies on detailed formal descriptions (*content metadata*), prepared in advance and describing in detail the structure and content of each component database. The server stores a separate set of content metadata (CMD in the diagram) for each database, which is used to match a user's query against the capabilities of the component databases. The TDS will then transform the user's query into queries suitable to the one or more databases found to contain relevant information, submit the queries to the appropriate databases, collect their responses, and present them to the user.

Once the query is dispatched to the relevant databases, the metadata can again be used to partially transform the results into a more consistent format, or to aid various types of aggregation.

5.1. From query to answer

We now consider the steps that will take place during the processing of a user query. Their execution relies on the presence of detailed metadata (discussed in more detail in section 6.), which has been prepared in advance and describes the component databases in all the relevant respects.

1. **Generate a user query.** The user creates a query through an HTML form or applet downloaded to his or her browser. This query is then submitted to the server for processing. An interactive process of query refinement is envisioned, as discussed in section 7. The system could also provide multiple interfaces tailored to different skill levels and requirements.

2. **Process query and compute a way to resolve it.** The query is matched against the content metadata for the component databases, and the databases that contain relevant information are selected.

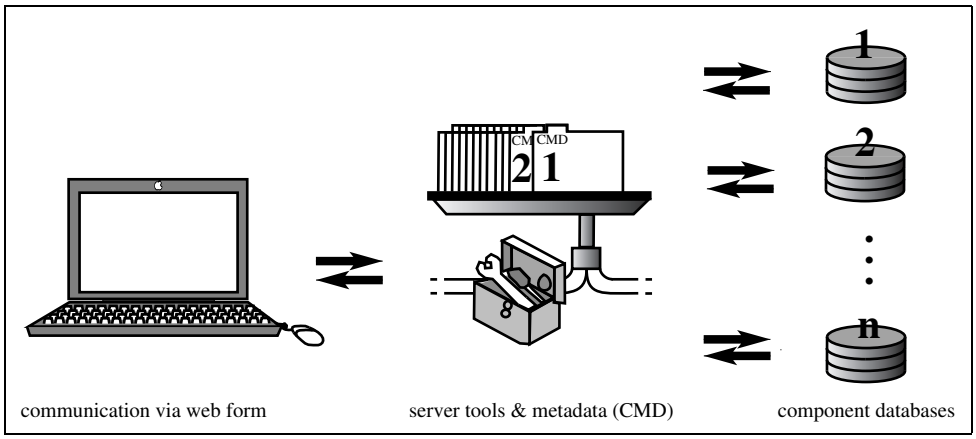

Figure 1: The Typological Database System (TDS)

3. **Translate query into forms suitable to each database.** The system needs to know where in a component database the desired information is stored; if particular values are being sought, it must also know how the desired value is represented in each database. This information is provided by the content metadata, which must describe in detail the structure and semantics of each field in each component database.

4. **Interact with the database servers.** The server will transmit the transformed queries to those servers containing relevant information, and wait for their response. This step is entirely dependent on communication protocols: For example, communication with an SQL Server database, or over an HTTP port. Although it is of course essential to the operation of the TDS, it is primarily a matter of supporting the appropriate communication protocols.[1]

5. **Translate results and present to the user.** This step might involve changing the encoding or vocabulary used in the results, as well as merging responses from different databases. In the simplest case, this step could do nothing: results are simply presented as they are received. i.e., in database format. More complicated transformations will be aided by the content metadata, which tells us, for example, how to interpret the value *0* or *1* in a certain field of some database. In order to ensure that the reliability of the retrieved information is not endangered by the transformation process, the system will be very conservative in this respect.

6. The metadata

The heart of the TDS is the content metadata, which describes the component databases to the central component and allows queries to be matched to databases. For each field in a database, the metadata must provide the following information:

1. **Its location.** This is provided by means of a key (i.e., a field name or column number) that identifies its location in the database and thus allows its retrieval.

2. **Its subject matter.** For example, the system must know that the field named *V27* contains information on whether a language allows null subjects.

3. **Its encoding.** For example, that a *0* value in *V27* means that null subjects are disallowed, and a *1* that they are allowed.

The first and third of these are more or less present in the *system catalog* (data dictionary) of the component databases, since the system catalog of every database includes information about the name, location, and data type of each variable in it. The "subject matter" of the variables is another matter: In order for the TDS to logically combine diverse databases, it must know when variables in different databases contain information about the same subject matter. To achieve this, the meaning of each variable will be defined in terms of a standardized *linguistic vocabulary,* whose development is a major part of the linguistic aspect of the TDS project.

We refer to the standard linguistic vocabulary, somewhat confusingly, as the *linguistic metalanguage.* The metalanguage will contain a general dictionary of linguistic terminology, alternative terms or near-synonyms with a description of their relationship to the preferred terms, glossing standards, and database specific terminology.

The linguistic metalanguage is the core element of the content metadata. The content metadata for each database must describe the subject matter of each field in terms of the metalanguage. We discuss this in more detail in section 6.2.

6.1. The linguistic "metalanguage"

The metalanguage needs to satisfy some important conditions:

1. It must be **general** enough to cover all the databases involved in the project.

[1] Since the TDS will interact with multiple remote databases, the system must be able to support parallel queries, and be able to respond properly even if some of the databases fail to respond. But these issues are completely independent of the linguistic content of the databases, and will not be further discussed here.

2. It must be **extensible,** so that future participation of additional databases will not necessitate the development of a new metalanguage.

3. It must be **economic,** in the sense that we will adopt already existing terminology, as long as this satisfies our needs and our quality standards.

The metalanguage will be used in the content metadata, and also for query construction. It will include precise definitions of notions and categories relevant to the component databases. These are necessary because most common linguistic terms have multiple interpretations, which occasionally differ considerably.

Consider, for example, the term *subject.* Its basic meaning is clear and understood by everyone in the field of linguistics, but the details of its application can vary considerably. Unclear cases are easy to come by: so-called "experiencer subjects," for example, have many subject-like properties but are frequently marked with oblique (non-subject) case. Whether they are coded as "subjects" in a given database depends not only on the beliefs of the creators of the database, but also on their purposes. Similar issues arise with the subjects of unaccusative verbs, which are widely believed to be underlyingly objects, and in countless other instances. The metalanguage should adopt *one* criterion for what it will consider a subject, and decide the unclear cases as consistently as possible. The chosen definition will not be (and should not be considered) the "best" definition on some theoretical grounds, but it should have the advantage of being easy to understand and apply.

A different type of complications is introduced by ergative languages, which treat the subjects of intransitive verbs as akin to the *objects,* rather than subjects, of transitives, and have led some linguists to reject the notion *subject* altogether, as incoherent. For example, (Comrie, 1989, p. 111) proposes in its place a tripartite classification that distinguishes the *sole argument of intransitive verbs (S),* the argument of transitive verbs which includes the *agent of action verbs (A),* and the argument of transitive verbs which includes the *patient of action verbs (P).* The metalanguage need not take a stand in this issue; Comrie's *S* and *A* classes can be accommodated as a subdivision of the usual notion *subject.* While the resulting collection of terms might lack theoretical coherence, it is expressive and allows a clear hierarchical relationship among its members.

Notational variation between databases is not as troublesome. For example, variation in the choice of abbreviations used in glossing agreement features and the like can be addressed by simply choosing one abbreviation as the standard, with no injury done to any of the component databases (as long as the variation is purely notational, of course). Wherever possible the TDS selects an existing system of annotation guidelines and adopts it as the standard for the metalanguage. Adopted standards include the Eurotyp standard for morphological annotation (Bakker et al., 1993), and the Ethnologue names (Grimes, 2000) and codes as the canonical names for languages in the component databases. Where applicable, the TDS project will also strive for compatibility with the Dublin Core Metadata Element Set, the OLAC extensions, and the ISLE initiative.

Clearly the metalanguage will be most successful if it is easy to understand and use for any linguist (or at least for any typologist), regardless of theoretical orientation. The reliance on existing standards helps achieve this goal.

6.2. The content metadata

The metadata describes the linguistic content of each field of the component databases in terms of the standard vocabulary ("linguistic metalanguage") developed within the TDS project. It also defines a correspondence between the values of each field and some standard data format into which the contents of that field could be converted. In other words, the content and format of the component database is mapped to our standard form by the metadata.

The correspondence between the categories defined by the TDS (e.g. a definition of *Subject*) and those used by the component databases may be only partial. In that case, the metadata will describe the nature of the overlap: e.g. a category is a subset or superset of another, etc. (cf. example below). More than set relations must be encoded: for example, the degree of similarity between two notions, and some way to cluster notions into groups, will almost certainly be important. Working out a suitable system of relationships is central to the TDS project.

The metadata, then, describes the component databases in terms of the standard notions provided by the metalanguage. In order to query a component database, we must perform a translation in the opposite direction: a query expressed in the standard vocabulary is converted into the vocabulary and format appropriate for that database. This is done by (automatically) processing the metadata for each database so that it can be used bi-directionally.

6.2.1. A simple example: Typological variable names

For a very simple example, suppose that our standard vocabulary represents the null subject parameter with the keyword *prodrop,* which may take the values *True* and *False.* Component database X might represent the same information in a column that has the identifier *NS* (for "null subject"), and is associated with the explanatory text "Absence of subject". The database uses the value 1 if a language can drop its subjects, and 0 if it cannot. The metadata for database X would then include the information shown in figure 2.

6.2.2. A more complex example: the notion "subject"

Let us now consider a more complicated, real example. Consider a simple query such as "which languages have subject-verb agreement." Component database X contains a boolean variable answering exactly this question; let us assume that the standard vocabulary includes a term *subject* whose definition is very close to the definition employed in database X. Another, database Y, includes a block of variables giving more information about subject-verb agreement, if it exists. But there is a complication: the notion *subject* is not a primitive in database Y. Instead, it recognizes four different types of agreement controllers: sole subject of an intransitive verb *(S),* and agent-like (A), patient-like (P), and recipient-like (R) arguments of a transitive verb. How can this classification be used to get information on subject-verb agreement? We must define a query

```
TYPE = "variable"                  # what this describes
LABEL = "Absence of subject"       # The documentation in the database
KEY = "NS"                         # The name to be used in queries
SUBJECT.MATTER = "prodrop"         # Our standard name for the subject matter
CORRESPONDENCE = "subset"          # The database uses a notion of prodrop that is
                                   # strictly narrower than ours
VALUES = [0 = False                # Interpret 0 as False
          1 = True                 # Interpret 1 as True
```

Figure 2: Sample (partial) metadata for an imaginary variable

in terms of the available categories. The details of how we can do this depend on how "agent like", etc., are defined in database Y. Note that the terms and abbreviations are reminiscent of those introduced by (Comrie, 1989) (see section 6.1.) If the creators of database *Y* have indeed followed Comrie's definition, then the common notion *subject* corresponds exactly to the combination of the *S* and *A* categories. But if the categories used in the database correspond to the usual, semantic rather than syntactic, notions of *agent, patient* and *recipient*, then things are not so clear-cut. The common notion *subject* probably includes all controllers of type S, all or most controllers of type A, and perhaps some controllers of type P and R. The most useful strategy, then, is probably to search for data on S and A controllers. Since the correspondence between the category "subject" and the available categories is clearly imperfect, the user must be warned about the situation so that he or she can properly interpret the results of the query.

The appropriate correspondences must be worked out by a linguist who can decide how best to map database Y's categories to the standard vocabulary. The mappings are then recorded in the metadata for database Y, and later consulted in order to map a user query about subjects into a form that asks about S and A. The degree of correspondence is also encoded in the metadata, allowing the system to warn the user that in this case the correspondence is imperfect.

It must be stressed that the appropriate action depends on the specifics of the notions used, often (as in this case) to a degree of detail that is simply unavailable in the brief documentation commonly made available with databases of this type. This is not an obstacle, since the appropriate metadata will be constructed in cooperation with the creators of the component databases.

Let us now consider the reverse situation: a linguist who subscribes to the classification of arguments in database Y wants to know which languages have agreement with the patient-like argument. Since one of the component databases does provide this information, the unified interface should make it available. In addition, a user who wants to set up this query should be informed that another database (database X once again) includes information about "object agreement," which includes most instances of "patient-like" agreement. The user interface might present the results of searching database X for information on object agreement, but it would be more useful if it presented the user with information of query terms similar to the user's desired query, and allowed the user to manually refine the final query. A user could then decide to rely exclusively on database Y, or to additionally look up information on object agreement on database X, later refining it by consulting off-line grammars or by other means.

7. The user interface

The simple examples we just considered highlight some of the challenges that a successful unified interface must address. The goal of the TDS is to provide a unified interface to diverse databases prepared on the basis of different research traditions. These databases must be made easily accessible to users who are familiar neither with the detailed content of the component databases nor with their underlying theoretical assumptions.

We have not discussed the specifics of the query interface in any detail. In part, this is because we view the user interface as conceptually separate from the core task of providing joint query functionality across the component databases. But the user interface is of course central to the success of an enterprise such as the TDS project. In this section we discuss our current design for the unified user interface.

Modern databases generally have a graphical user interface, which provides a set of screens that present "views" of the database's contents, organizing its variables in groups that are (ideally) useful and intuitive to understand. Such screens are designed by database programmers, and are tailored to the purposes of the particular database. In line with the TDS project's goal of creating an approximation of a "virtual database," the TDS will provide pre-designed query screens that present related variables in an organized fashion. Several of the component databases, in their standalone instantiation, include a graphical user interface that provides such screens; they can serve as the model for similar screens provided by the TDS. In this way the experience of using the TDS can be similar to that of using one of the component databases, even though a single TDS screen could provide access to variables from several different databases.

As much as possible, the system should allow searches defined in terms of the standard vocabulary (the metalanguage). But the specific notions used by the component databases should also continue to be available as query terms. Otherwise, inevitably situations will arise in which a user will be forced to get access to a component database directly, in order to retrieve information not available via the TDS interface. This contravenes an explicit design goal of the TDS: As much as at all practicable, the unified interface should provide access to *all* the information avail-

able via the individual interfaces.[2] For the same reason, the system should provide some means of browsing the full set of variables for more powerful query construction. This can take advantage of the available content metadata to allow searches (or browsing) over the variable lists: the user should be able to ask which variables have to do with subjects, and be referred to variables related to a non-standard category (e.g., "Agent-like argument of a transitive verb") employed by a component database.

Finally, these features can be combined with an interactive process of query refinement as suggested in the previous section: the user is presented with the set of search terms provided by the unified interface, or types in unrestricted search terms. In either case, the user's selection is then matched against the metadata describing the contents of the component databases, and the user is presented with a list of near-matches. The user then constructs a query that requests information actually present in one or more databases.

8. Conclusion

The aim of the Typological Database System project is the creation of a unified interface to numerous independently developed typological databases which will allow the user to simultaneously query them from a single gateway. By accessing the single gateway site, the user gains access to the data contained in all the databases participating in the project. This aim must be achieved while preserving the integrity of the data; among other things, the user must be able to trace how the information was obtained and what its source is.

The TDS will be part of a *Language Typology Resource Centre* (LTRC), a web-accessible electronic archive for typological description including typological databases, corpora, and grammars. The initial steps in developing such an archive are being made in the context of the research network *Language Typology Resource Centre* that is funded by the EC for 2001–2004. The LTRC will promote the development of standards in typological description, and will make available knowledge and resources for developing typological databases.

9. References

Serge Abiteboul, Peter Buneman, and Dan Suciu. 2000. *Data on the Web: From Relations to Semistructured Data and XML*. Morgan Kaufman, San Francisco.

Dik Bakker, Östen Dahl, Martin Haspelmath, Maria Koptjevskaja-Tamm, Christian Lehmann, and Anna Siewierska. 1993. EUROTYP guidelines. Technical report, European Science Foundation Programme in Language Typology.

Bernard Comrie. 1989. *Language Universals and Linguistic Typology*. The U. of Chicago Press, second edition.

Dublin core metadata initiative. Web page, *http://dublincore.org/*.

Barbara F. Grimes, editor. 2000. *Ethnologue*. SIL, Dallas, 14th edition. Online version: *http://www.ethnologue.com/*.

International standards in language engineering (ISLE). Web site, *http://www.mpi.nl/world/ISLE/*.

Open language archives community (OLAC). Web site, *http://www.language-archives.com/*.

Typological database system (TDS). Web site, *Typological Database System*.

[2] We recognize that it may not be possible to adapt all of the content of the component databases into the standardized format. The design goal is satisfied by ensuring that the unified interface provides ways to bypass the standardized format, querying directly the variables native to each database. For example, even if the standard vocabulary only provides a notion *subject*, the user should still be able to search for *S* or *A* in those databases that make use of the notion.