TRENDS IN COMPUTATIONAL PHILOLOGY -- AN ITALIAN OVERVIEW
International Conference, Venice May 22-24 2008

Presentation[*]

Trends in Computational Philology Conference (May 21-23, Venice and Padua) was aimed at joining the protagonists of the Italian research on digital classical philology with special guests of the American and European scene. Digital philology develops along two different dimensions: breadth and depth.

Digital collections about Greek and Latin texts are constantly enlarged. New corpora of primary sources, such as *Poeti d'Italia in Lingua Latina*[1], extend the temporal scope of previous collections constituted by texts extracted from a single, canonical edition. But massive initiatives of digitization (e.g. *Internet Archive*[2] or *JSTOR*[3]) are shifting the focus from canonical text to multiple editions. Furthermore, the digital libraries contain also commentaries, journals, dictionaries, studies about ancient authors, providing the secondary literature that is necessary to the real development of the ePhilogy[4]. Anyway, the general purpose approach aimed to provide many millions of books on the web as soon as possible, raises questions to philologists. One issue is the integrity of the documents (some pages are lost or repeated in the scanning process). The quality and fidelity of the page images are variable. The optical character recognition (OCR) performed on the page usually is unsupervised, with acceptable accuracy for Latin modern editions and largely unacceptable accuracy for Greek editions. Finally, the bibliographical reference and other metadata concerning the scanned books has inconstant precision and completeness. But the main question addressed by the first special guest of the conference, Gregory Crane (Perseus Project[5] - Tufts University, Boston) is: "what do you do with a million books?"[6]

Text retrieval on single canonical editions is based on a very simple structure: the textual sequence of words in larger units (e.g. lines, paragraphs, sections, works). For a digital library that contains primary and secondary sources, enriched by more complex digital objects, such as treebanks[7], geographical thematic maps, repertories of metrical analyses, repertories of conjectures etc., it is necessary to rethink the structures in terms of interoperability and reuse, in order to build the

[*] All the links have been visited last time on Sept. 15, 2009.
[1] http://www.mqdq.it/mqdq/poetiditalia/home
[2] http://www.archive.org
[3] http://www.jstor.org
[4] G. Crane, D. Bamman and A. Jones, *ePhilology: When the Books Talk to Their Readers*, in S. Schreibman and R. Siemens, *A Companion to Digital Literary Studies*, Oxford 2008, also available at http://www.digitalhumanities.org/companionDLS
[5] http://www.perseus.tufts.edu
[6] http://www.dlib.org/dlib/march06/crane/03crane.html
[7] For a general survey, see A. Abeillé, *Treebanks: Building and Using Parsed Corpora*, Dordrecht 2003.

cyberinfrastructure for classical philology[8].

Paolo Mastandrea ("Ca' Foscari" University of Venice) presented the *Musisque Deoque* Project[9], that aimed to provide minimalist critical apparatus to a large corpus of Latin poetry, in order to promote the study of intertextuality with digital tools, taking into account not only the reference edition but even the most relevant variants.

Manlio Pastore Stocchi (University of Padua) told about the new improvements in *Poeti d'Italia in Lingua Latina*[10] (the Italian Poetry in Latin).

Emiliano Degli Innocenti (SISMEL, Firenze) illustrated the large database of Medioevo Latino[11] and Fabio Ciotti told about the structure of the *Biblioteca Italiana*[12] (the Italian Library), the large collection of literary Italian texts, marked in TEI xml.

Francesca Tomasi[13] (University of Bologna) explained how to annotate digital texts with the *Text Encoding Initiative*[14] (TEI) tags, a standard in the domain of digital philology, showing case studies based on manuscripts.

Andrea Scotti (CNR of Pisa) presented the *Pinakes* Project[15], which is conceived as a container of subprojects that share digital objects, structured in ontologies, for the sake of interoperability.

Matteo Romanello ("Ca' Foscari" University of Venice) illustrated the principle of microformats[16], a new technology that embeds ontological informations, such as the unique identifiers of authors and works, inside simple html tags, which can be visualized with a common web browser.

Anna Maria Tammaro (University of Parma) showed the results of a survey on the access to digital libraries in humanities[17]. She pointed out how the Italian researchers have a very limited knowledge of the digital resources actually available online.

The second day of the conference was devoted to the second dimension of

[8] G. Crane and M. Terras, *Changing the Center of Gravity: Transforming Classical Studies Through Cyberinfrastructure*, Digital Humanities Quarterly, 3 (1) 2009. Consultabile on-line: http://www.digitalhumanities.org/dhq/vol/003/1/index.html

[9] http://www.mqdq.it

[10] http://www.mqdq.it/mqdq/poetiditalia

[11] http://www.sismelfirenze.it/mel/ita/infomel.htm

[12] http://www.bibliotecaitaliana.it

[13] F. Tomasi is the editor in chief of Griselda online: Portale di letteratura, http://www.griseldaonline.it

[14] http://www.tei-c.org

[15] http://www.pinakes.org

[16] M. Romanello, *A Semantic Linking System for Canonical References to Electronic Corpora*, in P. Zemánek (ed.), *Proceedings of the International Conference on Electronic Corpora of Ancient Languages*, Prague 2008, 155-174.

[17] A. M. Tammaro, *L'utenza della Biblioteca Digitale: Risultati dell'indagine sui bisogni, le aspettative e le capacità*, online at: http://www.rinascimento-digitale.it/documenti/conference2006/tammaro-ita.pdf

development: depth. This part of the conference was focused on projects aimed at studying a singular author or on projects aimed at enriching the levels of linguistic and stylistic analyses with features peculiar to the computational philology, such as the metrical analysis, the comparison of manuscripts and the evaluation of variants and conjectures.

Christopher Blackwell (Furman University, Greenville, South Carolina) and Neel Smith (College of the Holy Cross, Worcester, Massachussetts), the second special guests of the conference, illustrated the *Multitext Homer* Project[18] (Center for Hellenic Studies, Harvard University). High quality images of manuscripts, management of variants and canonical references to the classical texts[19] employed in this project should constitute the model for analogous projects focused on a singular author.

Federico Boschetti (University of Trento) presented the system to map the information extracted from the repertories of conjectures on the reference editions for the *Digital Aeschylus* Project[20].

Dino Buzzetti afforded the topic of textual structures in presence of variant readings from a theoretical point of view, illustrating the relations between digital representation and text model[21].

Rodolfo Delmonte (University of Venice) focused his talk[22] on advanced techniques to detect subjectivity from discourse structure, shifting the topic from textual structure issues to tools and techniques for linguistic and stylistic analysis, crossing the bridge between computational linguistics and computational philology.

Marco Passarotti ("Cattolica" University of Milan) presented the *Index Thomisticus* Treebank, based on a dependency grammar model.

Daniele Fusi (University of Rome "La Sapienza") illustrated an expert system for metrical analysis of classical texts, showing how it is possible to analyze automatically large amounts of ancient Greek poetry.

Renzo Orsini and Marek Maurizio ("Ca' Foscari" University of Venice) showed the features of a script language called *Manuzio* that manages the structural representation and querying of annotated literary texts.

[18] http://chs.harvard.edu/chs/homer_multitext

[19] For an explanation of the Classical Text Service, see C. N. Smith, *Citation in Classical Studies*, Digital Humanities Quarterly, 3 (1) 2009, http://digitalhumanities.org/dhq/vol/3/1/000028.html

[20] F. Boschetti, *Digital Aeschylus: Breadth and Depth Issues in Digital Libraries* in R. Bernardi, S. Chambers and B. Gottfried (edd.), *Proceedings of the Workshop on Advanced Technologies for Digital Libraries*, Trento 2009. See also F. Boschetti, *Saggio di analisi linguistiche e stilistiche condotte con l'ausilio dell'elaboratore elettronico sui Persiani di Eschilo*, Trento, Phd Thesis, available on-line at http://documents.univ-lille3.fr/files/pub/www/recherche/theses/BOSCHETTI_FEDERICO.pdf

[21] D. Buzzetti, *Digital Representation and the Text Model,* New Literary History, 33 (1) 2002, 61-88.

[22] R. Delmonte, *Detecting Subjectivity from Discourse Structure with GETARUNS* is available on Lexis' website: http://www.lexisonline.eu

Passarotti, Fusi and Orsini's contributions are published in the next pages of this issue.

The poster session was constituted by four works aimed at covering emerging aspects of the e-philology.

Stefano Minozzi (University of Verona) presented LatinWordNet, the Latin section of MultiWordNet[23], that is a multilingual lexical database.

Marion Lamé (“Ca' Foscari” University of Venice) showed a digital representation, tagged in xml, of epigraphic texts.

Giampaolo Galvani (University of Trento) suggested a system to represent the different colometrical layout of Aeschylean manuscripts.

Alessandro Valitutti (University of Trento) showed a technique to evaluate stylistic variations in the affective lexicon used in different translations in English of the Aeschylus' tragedies.

Poster session's contributions are published on *Lexis*' website.

During the round table, the third special guest from the University of Zagreb, Neven Jovanović, has proposed a long list of *desiderata* to meet the needs both of the creators of new philological instruments and of the users.

Last interventions by Guido Avezzù (University of Verona) and Vittorio Citti[24] (University of Trento) shifted the attention from the technological aspects to the complex relation between the traditional methods of classical philology and the new approaches. Computational philology should dialogue with the traditional philology, inheriting its precision, complexity and attention to the history of textual problems.

CIMeC – Università degli studi di Trento Federico Boschetti

[23] http://multiwordnet.itc.it/english/home.php

[24] See Vittorio Citti, *Filologia computazionale e filologia formale*, Lexis 26, 2008, 1-4.

THEORY AND PRACTICE OF CORPUS ANNOTATION IN THE *INDEX THOMISTICUS* TREEBANK

1. *Introduction*

Corpus linguistics is nowadays a well established field of research, where collaborative work with both computational and theoretical linguistics is required. As a matter of fact, computational linguistics makes use of corpus data to train probabilistic Natural Language Processing (NLP) tools, such as taggers and parsers; on the other hand, in empirical approaches to the study of language, theoretical linguistics refers to corpus evidence. On its side, corpus linguistics, as a discipline in itself, uses NLP tools to (semi)automatically build annotated corpora, and refers to linguistic theory as the backbone for the design of annotation guidelines.

The creation of a linguistically annotated corpus is, therefore, an excellent opportunity to apply to real data (and potentially revise) linguistic theories which have been designed in a pre-corpus era. This is an even more attractive challenge if a language like Latin is involved. Indeed, while the language-dependent computational processing of Latin is today limited to automatic morphological tagging[1], a number of available language-independent methods and tools of analysis can be applied to it.

2. *Corpora and annotation*

In the definition provided by Sinclair (2004, 20), a corpus is a «collection of pieces of language text in electronic form, selected according to external criteria to represent, as far as possible, a language or language variety as a source of data for linguistic research». This definition mentions some of the main themes in corpus linguistics, such as the following: (a) corpora are today in electronic form and, usually, available on the Internet; (b) the texts of a corpus should be selected according to explicit 'external' criteria, that is to say «derived from the examination of the communicative function» of the texts (Sinclair 2004, 5). Conversely, no 'internal' criteria referring to the specific linguistic properties of the texts should be adopted. This prevents the corpus from being created exactly for the aims of the linguistic research that makes use of corpus data; (c) a corpus has to be representative of a language (or a variety of it) to provide empirical evidence to linguistic research.

[1] Three Latin morphological analysers are nowadays available: LEMLAT (Passarotti 2007a), Whitaker's Words, and Morpheus (Crane 1991), which was first developed in the Perseus Digital Library for Ancient Greek (in 1985) and extended to support Latin in 1996. Some attempts of syntactic parsing of Latin in a rule-based fashion are in Koch (1993) and in Koster (2005).

2.1 *Approaches to corpus data*

Tognini-Bonelli (2001) has recognised and established two different ways of looking at corpus data in linguistic research: these are the so-called ‘corpus-based’ and ‘corpus-driven’ approaches.

In the corpus-based approach, corpus data are used in order to confirm, refine, improve and/or modify pre-corpus linguistic theories which have been designed on the basis of the linguist’s knowledge of language. There are several different ways of considering empirical data in such an approach, especially if they do not fit the theory. In particular, Tognini-Bonelli (2001, 68-77) distinguishes three of them: (a) ‘insulation’: theoretical hypothesis are made in a fully intuition-based fashion and expressed in a formal grammar. Only afterwards, data are taken into account as a testbed for the formal grammar which is extended by confrontation; (b) ‘standardisation’ (or ‘enrichment’): it is a more empirical approach, which tests the accuracy of a grammar through data annotation and revises it according to data evidence; (c) ‘instantiation’: it is a paradigmatic approach to grammar, where data are considered as abstract possibilities in a system of choices. There is a strict connection between the probability of the possible choices in the paradigmatic grammar and their frequency of occurrence provided by corpus evidence.

On the other side, the corpus-driven approach develops linguistic theories which completely reflect the empirical evidence provided by data: grammars are produced by induction on corpus data, where the knowledge and experience of the linguist applies. This approach is followed, for instance, in the Cobuild project by Sinclair (1987), whose claim is that linguistic annotation is a sort of loss of information on corpus data: «Sinclair points out that the replacement of a text by a string of tags is a reduction of information: for example, words which are different but are allocated the same word class lose this distinctiveness in favour of the recognition that they belong to the same word class» (Tognini-Bonelli 2001, 73). Moreover, no annotation guidelines are objective and independent from a pre-corpus theory: this means that annotating a corpus implies imposing on the corpus itself a particular view of language that is always subjective and, therefore, not reproducible.

The gap between corpus-based and corpus-driven approaches generally reflects two different views of linguistic research, respectively focused more on ‘competence’ or on ‘performance’ (Chomsky 1965). Nowadays, corpus development requires these two attitudes to work in a more collaborative way than before: as a matter of fact, no grammar based on competence can cover all the possible cases represented in the performance, and no grammar based on performance can reflect all the possible well-formed structures of a language.

Since the annotation of the Brown and Lancaster-Oslo/Bergen (LOB) corpora in the 1970s, applications of computational linguistics in building annotated corpora have mainly followed the ‘standardisation’ approach. Over the years, this has been performed in order to add meta-linguistic information on corpora at an always growing level of granularity and specificity, ranging from simple morphological annotation to syntactic, semantic, rhetorical, pragmatic, stylistic and, ultimately, multi-modal ones.

Given the central role played by syntax in linguistics during the last century, it seems that the task of developing syntactically annotated corpora (known as ‘treebanks’) has been and still is a privileged field where to apply the standardisation approach. This is due to the wide availability of syntactic theories created in a pre-corpus era that can now be evaluated on real empirical data. Starting from sets of hand-crafted guidelines built on the general basis of a grammar framework, these are then refined in the process of annotation of the corpus. As Sampson (1995) claims, this process is similar to the creation of a legal system by the tradition of the common law: the way the law evolves according to the precedent of earlier cases, so the annotation evolves taking into account what has been previously annotated.

2.2 *Treebanks. PSG and DG*

Treebanks are syntactically parsed corpora: the name ‘treebank’ refers to the usual representation of the syntactic structure of sentences as trees. The trees are usually designed according to two main grammar frameworks: Phrase Structure Grammars (PSG) and Dependency Grammars (DG).

The main logical operation in PSG trees is categorisation: the syntactic description of a sentence moves from specific information in the leaves of the tree (the words of the sentence) to the generic start symbol ‘S’ (‘Sentence’) as the root, passing through hierarchically embedded intermediate nodes, such as Parts-of-Speech (PoS) and phrases.

On the other side, DG trees represent relations among words, and not among PoS and/or phrases. In such trees only lexical nodes are allowed, and branches describe relations (‘dependencies’) between head and dependent nodes.

Both PSG and DG do not allow cyclic or linear relations: this means that no horizontal branches are permitted in the trees.

PSG have been mainly developed in the context of transformational-generative grammar, based on relations among constituents as described in derivation trees. Constraint-based feature-structure grammars (Kaplan 2003, 86-88) like Lexical Functional Grammar (Bresnan 2001) and Head-driven Phrase Structure Grammar (Pollard and Sag 1994) have been applied to the syntactic annotation of treebanks such as the Penn Treebank for English (Cahill et al. 2002) and the BulTreebank for Bulgarian (Osenova and Simov 2003).

On the other hand, DG are predicate-focused grammars based on the notions of 'dependency' (the head-dependent relations among words) and 'valency' (the number of obligatory arguments for verbs, nouns and adjectives); usually, in DG there is no annotation of the surface word-order of the sentence. Although there are many different current DG flavours[2], they are all developed on a common theoretical background as defined in Tesnière (1959).

While since the 1970s the first treebanks were annotated on PSG-based schemata (as in IBM, Lancaster and, later on, Penn treebanks), in the last decade many projects of dependency treebanks development have been started, such as the ALPINO treebank for Dutch, the Turin University Treebank for Italian, or the Danish Dependency Treebank. This is due to a number of reasons. Firstly, PSG have been much more successful than DG in theoretical linguistics: this means that many more PSG-based formalisms were already available when the first treebanks were built. Secondly, those treebanks were mainly English language corpora. Since English is a poorly inflected language with a fixed word-order and few discontinuous constituents, PSG were a well suitable framework; later on, the syntactic annotation of moderately free word-order languages data required the adoption of the DG framework, which is more appropriate than PSG for such a task. Finally, Carroll et al. (1998) showed that inter-annotator agreement was significantly better for dependency treebanks, indicating that phrase structure annotation was requiring too many irrelevant decisions.

3. *Latin*

Latin is a richly inflected language, showing:
- discontinuous constituents ('non-projectivity'): this means that it can be the case that phrasal constituents are not continuous, but broken up by words of other constituents. An example is the following sentence by Ovid (*Met.* I.1-2): «In nova fert animus mutatas dicere formas corpora» («My mind leads me to tell of forms changed into new bodies»). In this sentence, both the nominal phrases 'nova corpora' and 'mutatas formas' are discontinuous;
- a moderately free word-order: for instance, the order of the words in a sentence like «audentes fortuna iuvat» («fortune favours the bold»; Verg. *Aen.* 10.284) could

[2] See for instance the following: Dependency Unification Grammar (Hellwig 1986), Functional Generative Description (Sgall, Hajičová and Panevová 1986), Meaning Text Theory (Mel'čuk 1988), Word Grammar (Hudson 1990).

be changed into «fortuna audentes iuvat», or «fortuna iuvat audentes», without affecting the meaning of the sentence.
As the next section points out, these features of Latin affect the choice as to the most suitable grammar framework in building Latin annotated corpora.

3.1 *Latin Treebanks*

The first two projects for the development of Latin treebanks started only recently: namely, they are the Latin Dependency Treebank (LDT) at the Tufts University in Boston (within the Perseus Digital Library) on texts of the Classical era (Bamman 2006), and the *Index Thomisticus* Treebank (IT-TB) at the Catholic University of the Sacred Heart in Milan on the opera omnia by Thomas Aquinas (Passarotti 2007b).

Taking into account the features of Latin as described in the previous section, both the treebanks independently chose the DG framework as the most suitable one for data annotation. The same approach was later on followed also by a third Latin treebank now available, which is developed at the University of Oslo in the context of the PROIEL project (Pragmatic Resources in Old Indo-European Languages): the aim of PROIEL is the syntactic annotation of the New Testament oldest extant versions in Indo-European languages such as Greek, Latin, Gothic, Armenian and Church Slavonic (Haug and Jøhndal 2008).

3.2 *Annotation Guidelines*

Since LDT and IT-TB were the first projects of their kind for Latin, no prior established guidelines were available to rely on for syntactic annotation.

So, the so-called ‘analytical layer’ that is used for annotation in the Prague Dependency Treebank (PDT) for Czech (Hajič et alii 1999) was chosen as a general model for the style of representation and was then adapted for the treatment of specific or idiosyncratic constructions of Latin, which could be syntactically annotated in several different ways. These constructions (such as the ablative absolute or the passive periphrastic) are common to Latin of all eras: rather than have each treebank project decide upon and record each decision for annotating them, LDT and IT-TB decided to pool their resources and create a single annotation manual that would govern both treebanks (Bamman et al. 2007a; Bamman et al.i 2007b).

Dealing with Latin dialects separated by 13 centuries, sharing a single annotation manual is very useful for comparison purposes, such as checking annotation consistency or diachronically studying specific syntactic constructions. In addition, the task of data annotation through these common guidelines allows annotators to base the decisions on a variety of examples from a wider range of texts and combine the two datasets in order to train probabilistic dependency parsers.

Table 1 lists all of the syntactic tags currently in use in IT-TB and LDT (Bamman et al. 2008).

Pred	predicate
Sb	subject
Obj	object
Atr	attributive
Adv	adverbial
Atv/AtvV	complement
PNom	predicate nominal
OComp	object complement
Coord	coordinator
Apos	apposing element
AuxP	preposition
AuxC	conjunction
AuxR	reflexive passive
AuxV	auxiliary verb
AuxX	commas
AuxG	bracketing punctuation
AuxK	terminal punctuation
AuxY	sentence adverbials
AuxZ	emphasizing particles
AuxS	root of the tree
ExD	ellipsis

Table 1. Complete Latin tagset

Like in PDT, all of the tags can be appended with a suffix in the event that the given node is member of a coordinated construction (_Co), an apposition (_Ap) or a parenthetical statement (_Pa).

The tag Pred is given to the predicate of the main clause (or clauses, in case of coordination or apposition) of a sentence; the head verbs of the subordinate clauses are annotated according to the clause role in the sentence (for instance, a declarative clause acting as subject is annotated with the tag Sb).

An Atr is a sentence member that further specifies a noun in some respect; typical attributives are adjectives (‘bonus puer’: ‘good boy’) and nouns in the genitive case (‘domus patris’: ‘the father’s house’).

The difference between Obj and Adv roughly corresponds to the one between arguments (inner participants) and adjuncts of verbs or adjectives, i.e., between those called ‘actants’ and ‘circonstants’ in the terms of Tesnière (1959). A special kind of Obj is the determining complement of the object, which is tagged with OComp, such as ‘senatorem’ in a sentence like «aliquem senatorem facere» («to nominate someone senator»). The determining complement of the subject is, conversely, tagged using PNom; this mainly occurs in case of constructions like «aliquis senator fit» («someone becomes senator»). The tag OComp covers some of the functions of the Atv/AtvV tag (Verbal Attribute) as used by the PDT: departing from PDT style, we assign a different tag to object complements (OComp) and to complements that are not direct arguments of the verb (Atv/AtvV). These are usually noun phrases and adjectives that agree with their head noun morphologically, but differ from typical attributes in that they also qualify the function of the verb: the use of Atv/AtvV is largely similar to the account of ‘praedicativa’ given in Pinkster (1990, 142-162).

Although PROIEL annotation guidelines are grounded on the same grammar framework as LDT and IT-TB, they differ in a number of details, some of which are described below.

PROIEL makes use of some more specific tags, such as NARG for the arguments of nouns (for instance, ‘in sanctitatem’ in «ingressio in sanctitatem Dei», «entrance in God’s santity»[3]), AG(ent), covering agents in passive constructions (‘ab his’ in «quae tibi obiciuntur ab his», «[those things] that are objected to you by these») and OBL(ique), assigned to those verbal arguments that are not subject or object to the clausal node (‘mihi’ in «et dixit mihi angelus», «and the angel told me»). OBL includes also non-accusative objects (such as the ablative object of ‘utor’, ‘to use’) as well as prepositional arguments (‘eum’ in «et introibo ad eum», «and I will enter him»).
Another difference is that AuxC and AuxP tags are not adopted in PROIEL: conjunctions and prepositions are tagged according to the sentence role of the phrase, or the subordinate clause they introduce. Instead, in LDT and IT-TB style this annotation is given to the main predicate of the clause introduced by the conjunction, or to the prepositional argument(s): like in PDT, conjunctions and prepositions are considered as “bridge” auxiliary structures (respectively tagged with AuxC and AuxP). For instance, in the tree of the sentence «cenabo cum illo» («I will have dinner with him»), ‘illo’ depends on ‘cum’: in such a tree, PROIEL assignes Adv to ‘cum’ and OBL to ‘illo’, while in LDT and IT-TB ‘cum’ is an AuxP and ‘illo’ is an Adv.

[3] The examples are excerpted from PROIEL guidelines, which are available on-line at http://www.hf.uio.no/ifikk/proiel/publications/guidelines.pdf.

4. *The Index Thomisticus Treebank*

The *Index Thomisticus* (IT) by Roberto Busa SJ (1974-1980) was started in 1949 and is considered as a pathfinder in computational linguistics. It retains the opera omnia by Thomas Aquinas (118 texts) and 61 texts by other authors related to Thomas, for a total of around 11 million tokens. The corpus is morphologically tagged and lemmatised.

Early in the 1970s Busa started to plan a project aimed both at the morphosyntactic disambiguation of the IT lemmatisation and the syntactic annotation of its sentences: nowadays, these tasks are performed by the IT-TB project, which is part of a wider one, named 'Lessico Tomistico Biculturale', whose target is to develop a lexicon from the IT texts.

Presently (September 2008) the size of IT-TB is 40.062 tokens, for a total of 1.813 parsed sentences excerpted from the *Scriptum super Sententiis Magistri Petri Lombardi*.

4.1 *Annotation Procedures*

Up to now, annotation of IT-TB data has been performed manually, using the tree editor TrEd, developed by Petr Pajas for PDT requirements[4].

The annotation of a sentence requires the three following steps:

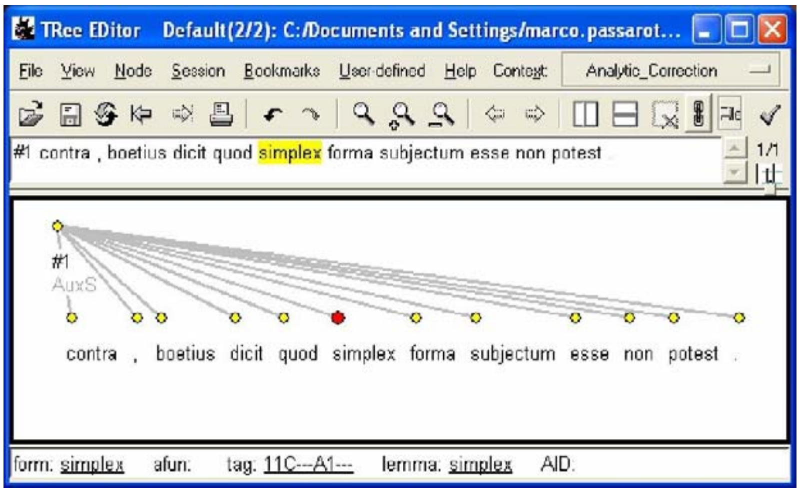
(a) checking and (possibly) correcting the IT morphological analysis. In fact, IT texts are tagged in a way that among the possible morphological analyses of each word only the first in the grammars is assigned: this means that a word like 'puella' is always tagged as a singular nominative and never as a singular vocative, or ablative;

(b) assigning to each word a syntactic tag;

(c) defining and designing the relations between the words in the tree.

Figure 1 reports the image of a sentence tree before manual annotation is performed.

[4] TrEd is freely available at http://ufal.mff.cuni.cz/~pajas/tred/.


TRee EDitor Default(2/2): C:/Documents and Settings/marco.passarot...
File View Node Session Bookmarks User-defined Help Context: Analytic_Correction
1/1
#1
AuxS

#1 contra , boetius dicit quod simplex forma subjectum esse non potest

contra , boetius dicit quod simplex forma subjectum esse non potest .

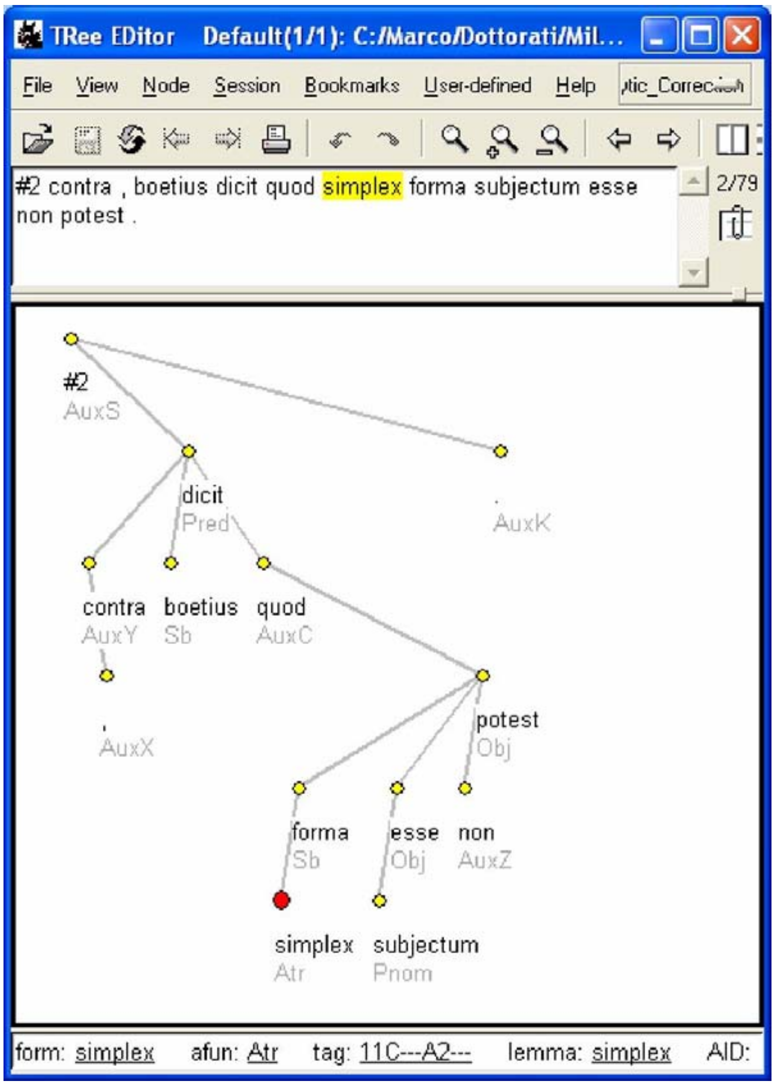
form: simplex afun: tag: 11C--A1--- lemma: simplex AID:

Figure1. Pre-annotation sentence

The sentence is reported in the higher part of the screen: «contra, Boetius dicit quod simplex forma subjectum esse non potest» («on the other hand, Boece says that a simple form cannot be subject»).

The tree is shown below. Each node in the tree corresponds to a word in the sentence (and viceversa), except for the root, which reports the number of the sentence in the treebank (in this case, it is the first one); as figure 1 displays, before annotation is performed all the nodes are linked to the root.

In the lower part, the morphological tagging of the selected word in the sentence (in figure 1, 'simplex') is reported. The word 'simplex' is morphologically tagged as a form of the lemma 'simplex', with the following morphological tags: nominal-adjectival inflection (1), non-comparative degree (1), third declension noun - second class adjective (C), singular nominative (A), masculine (1).
Figure 2 reports the tree of the sentence after manual annotation.


TRee EDitor Default(1/1): C:/Marco/Dottorati/Mil...
File View Node Session Bookmarks User-defined Help ...tic_Correc...
2/79
#2
AuxS
dicit
Pred
.
AuxK
contra
AuxY
boetius
Sb
quod
AuxC
,
AuxX
potest
Obj
forma
Sb
esse
Obj
non
AuxZ
simplex
Atr
subjectum
Pnom

#2 contra , boetius dicit quod simplex forma subjectum esse non potest .

form: simplex afun: Atr tag: 11C---A2--- lemma: simplex AID:

Figure 2. Post-annotation sentence

In figure 2, each node of the tree is annotated with a syntactic tag (for instance, the node of the word ‘forma’ is tagged with ‘Sb’ - ‘Subject’); the syntactic relations are represented by the branches of the tree, and the correction of the morphological tagging is performed: ‘simplex’ is now correctly annotated with the feminine gender (tag 2 in eighth position), instead of the masculine (1).

A semi-automatic annotation procedure is presently in its first phase of application: annotators do not have to face a virgin tree anymore, but a tree produced by a probabilistic dependency parser trained on annotated data. Therefore, annotators just need to correct the wrong analyses produced by the parser and not to design the entire tree from scratch. So far, only MaltParser (Nivre et al. 2006) has been trained[5]. Table 2 reports the global results of the parser in terms of Labelled Attachment Score (LAS), Unlabelled Attachment Score (UAS) and Label Accuracy (LA), according to the evaluation metrics used in the CoNLL Shared Task 2006 (Buchholz and Marsi 2006)[6].

LAS	0.639
UAS	0.713
LA	0.739

Table 2. MaltParser accuracy

Table 3 shows the parser's performance by individual tag[7].

Precision	Recall	F-score	Tag
1	1	1	AuxK
0.952	0.978	0.965	AuxX
0.996	0.927	0.961	AuxP
0.946	0.835	0.887	AuxC
0.965	0.762	0.852	AuxZ
0.888	0.761	0.82	Coord
0.667	1	0.8	AuxR
0.793	0.781	0.787	Atr
0.783	0.766	0.774	Pred
0.715	0.744	0.729	Adv
0.698	0.733	0.715	ExD
0.68	0.752	0.714	Pnom
0.79	0.646	0.711	AuxY
0.72	0.668	0.693	Sb
0.778	0.594	0.674	Pred_Co
0.667	0.667	0.667	Apos
0.636	0.556	0.593	Obj
0.5	0.5	0.5	Coord_Ap
0.5	0.429	0.462	AuxV
0.535	0.377	0.442	Atr_Co

[5] The parser was trained on 9/10 of the treebank (36.056 words) and tested on the remaining one-tenth with disambiguated morphological tags (gold standard).

[6] LAS is the percentage of tokens with correct head and relation label; UAS is the percentage of tokens with correct head; LA is the percentage of tokens with correct relation label.

[7] Precision is defined here as the percentage of times a tag X is correctly assigned to the correct head with respect to the number of occurrences of that tag in the automatically parsed data; recall is the percentage of times a tag X is correctly assigned to the correct head with respect to the number of occurrences of that tag in the gold standard. F-score (or F-measure) is the weighted harmonic mean of precision and recall, calculated as follows: F = 2*(precision*recall) / (precision+recall).

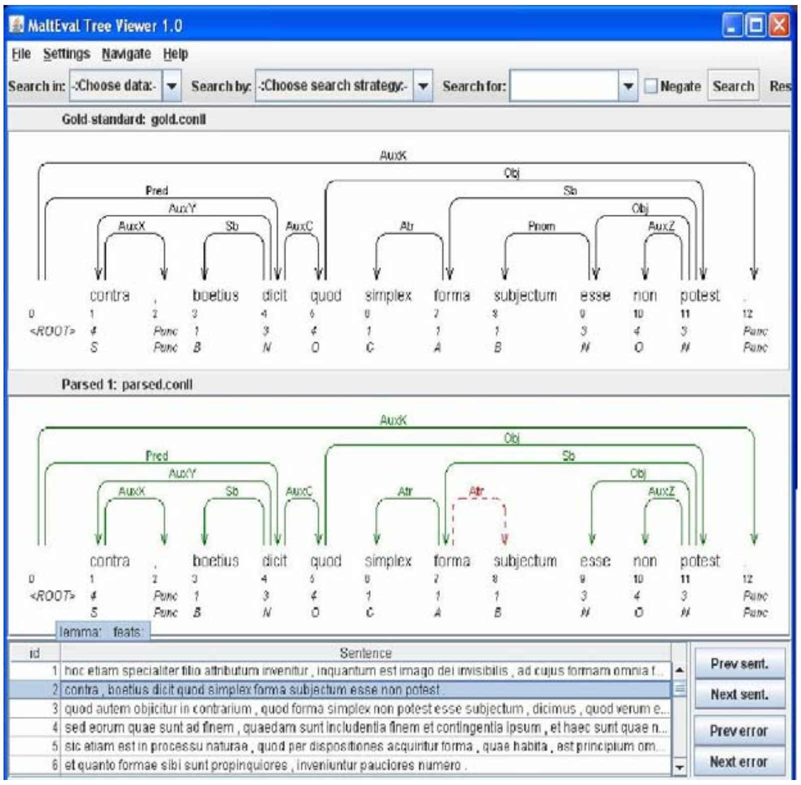
0.5	0.333	0.4	Apos_Co
0.413	0.351	0.38	Adv_Co
0.37	0.312	0.339	Pnom_Co
0.342	0.197	0.25	ExD_Co
0.364	0.16	0.222	Sb_Co
0.154	0.125	0.138	Obj_Co

Table 3. Labelled Precision/Recall and F-Score by syntactic tag

Except terminal punctuations (AuxK) and commas (AuxX), the best F-score is for prepositions (AuxP), subordinating conjunctions (AuxC), emphasizing particles such as ‘etiam’ and ‘non’ (AuxZ) and coordinators (Coord). The F-score for attributives (Atr), predicates (Pred), adverbials (Adv) and subjects ranges from 0.78 to 0.69, while objects (Obj) show a lower score (0.593). The lowest scores are for relations involved in coordination (_Co), or apposition (_Ap)[8].

Figure 3 shows a capture from MaltEval (Nilsson and Nivre 2008), an evaluator with a graphical user interface for comparing gold standard data with automatically tagged, parsed or chunked natural language texts. In the higher part of the screen the gold standard is reported, while the sentence as parsed by MaltParser is displayed underneath: here, the parts corresponding to the gold standard are represented by continuos line, while the differences are represented by a dashed line and the branches are dotted. In this case, the parser made only two mistakes: the word ‘subjectum’ is assigned the tag Atr (instead of PNom) and ‘subjectum’ is made dependend on ‘forma’ (instead of ‘esse’).

[8] Table 3 does not report the null F-scores, which were found for the following tags: Adv_Ap, Atr_Ap, Atv, AtvV, Atv_Co, Coord_Pa, Obj_Ap, OComp, Sb_Ap.


MaltEval Tree Viewer 1.0
File Settings Navigate Help
Search in: -:Choose data:- Search by: -:Choose search strategy:- Search for: Negate Search
Gold-standard: gold.conll
Parsed 1: parsed.conll
AuxK
Obj
Sb
Pred
AuxY
AuxX
AuxC
Atr
Pnom
AuxZ
contra , boetius dicit quod simplex forma subjectum esse non potest .
<ROOT>
lemma: feats:
Prev sent.
Next sent.
Prev error
Next error

id	Sentence
1	hoc etiam specialiter filio attributum invenitur , inquantum est imago dei invisibilis , ad cujus formam omnia f...
2	contra , boetius dicit quod simplex forma subjectum esse non potest
3	quod autem objicitur in contrarium , quod forma simplex non potest esse subjectum , dicimus , quod verum e...
4	sed eorum quae sunt ad finem , quaedam sunt includentia finem et contingentia ipsum , et haec sunt quae n...
5	sic etiam est in processu naturae , quod per dispositiones acquiritur forma , quae habita , est principium om...
6	et quanto formae sibi sunt propinquiores , inveniuntur pauciores numero .

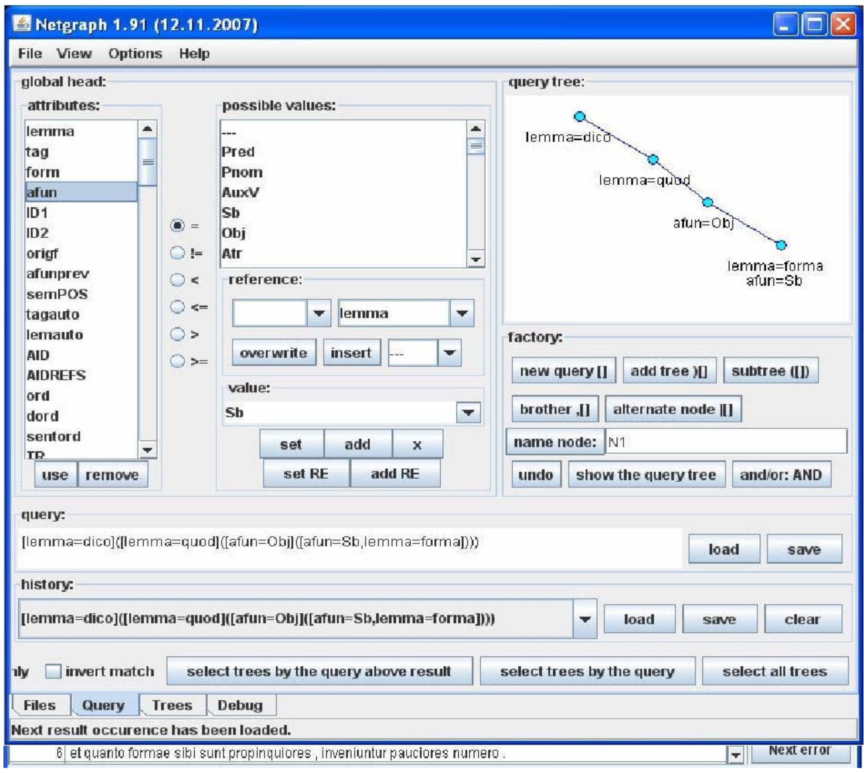
Figure 3. MaltEvalt

4.2 *Browsing the IT-TB*

IT-TB data can be browsed on-line at http://itreebank.marginalia.it through the searcher and viewer Netgraph (Mírovský 2006).

In the main window of Netgraph it is possible to draw a subtree as a query tree: the tool searches for all the trees where the query tree occurs at least once[9]. The query tree in figure 4 describes a structure where a form of the lemma ‘dico’ governs an object subordinate clause headed by the conjunction ‘quod’ («dico quod...», «I say that...»); in addition, the subject of the verb in the subordinate clause is a form of the lemma ‘forma’.

[9] Another way to create a query is to write it manually in the “Query” box. The syntax of the query language is described in the user manual of Netgraph, which is available on-line. Instead, when a query tree is drawn, the corresponding query as a string of characters is written automatically.



value: Sb

name node: N1

query: [lemma=dico]([lemma=quod]([afun=Obj]([afun=Sb,lemma=forma])))

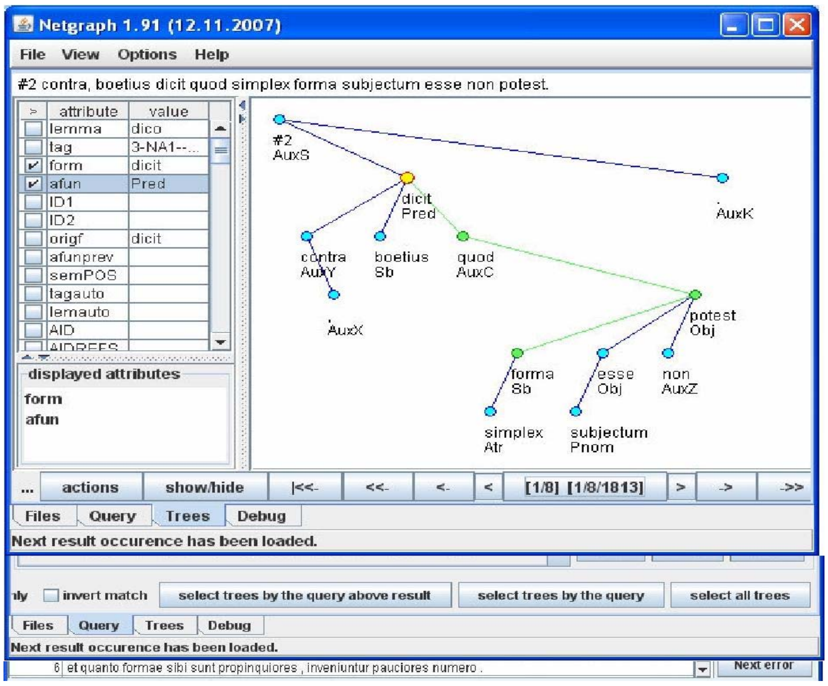
history: [lemma=dico]([lemma=quod]([afun=Obj]([afun=Sb,lemma=forma])))

Next result occurence has been loaded.

6 et quanto formae sibi sunt propinquiores , inveniuntur pauciores numero .

Figure 4. A query in Netgraph

Figure 5 reports the first of the output trees.



#2 contra, boetius dicit quod simplex forma subjectum esse non potest.

attribute	value
lemma	dico
tag	3-NA1--...
✓ form	dicit
✓ afun	Pred
ID1	
ID2	
origf	dicit
afunprev	
semPOS	
tagauto	
lemauto	
AID	
AIDREFS	

Next result occurence has been loaded.

6 et quanto formae sibi sunt propinquiores , inveniuntur pauciores numero .

Figure 5. An output in Netgraph

The subtree matching the query is represented by a light grey line. The sentence corresponding to the tree is reported on top of the screen; on the left, information about the lemmatisation of the selected word in the tree is provided (in this case, the word 'dicit'); on the bottom, the numbers "1/8/1813" mean that 1.813 sentences have been queried and 8 of them correspond to the query, the current being the first one. The buttons on the left and the right of these numbers allow to move within the query results, respectively showing the tree of the previous and of the next output sentence.

5. *Conclusion*

IT-TB annotation follows a standardisation corpus-based approach: grounding on PDT analytical layer guidelines adapted to Latin, the annotation tests and, possibly, revises them on real data evidence. Along with LDT and PROIEL, this is the first project in syntactic annotation of Latin texts. Such a large-scale annotation of textual data can act as a testbed where established syntactic theories and properties specific of Latin can be empirically tested.

In the near future, annotation will be completely performed in a semi-automatic way. This will require:

- to apply to the IT texts the Latin morphological analyser LEMLAT, in order to produce their full morphological annotation;
- to make use of PoS taggers, as for instance TreeTagger (Schmid 1994), to disambiguate the full morphological annotation;
- in addition to MaltParser, to train other probabilistic dependency parsers, such as MST (McDonald et alii 2005), DeSR (Attardi 2006) and ISBN (Titov and Henderson 2007). The tagger and the parser with the best accuracy rate will be used to annotate the data. Since data from IT-TB and LDT are annotated under the same general guidelines, the two datasets can be combined to increase the size of the training set for such parsers, as they perform best with larger amount of data. PROIEL dataset will be used too, thanks to an ongoing conversion procedure from its annotation style to LDT and IT-TB. Since the texts of the three treebanks are separated by many centuries and diverge in style, their syntax can be very different: this can affect the training of parsers and must be taken into account while combining datasets.

The development of a dynamic valency lexicon from IT-TB data is in progress, too. The entries of such a lexicon will correspond to the verbs of the IT-TB and record their valency as induced from data evidence. The lexicon will be realised so that it will be dynamically updated as annotation increases[10].

Milano, Università Cattolica del Sacro Cuore Marco Passarotti

REFERENCES

G. Attardi, *Experiments with a Multilanguage non-projective dependency parser*, in *Proceedings of the 10th Conference on Computational Natural Language Learning* (CoNLL-X), 2006, 166-170.

D. Bamman, *The Design and Use of Latin Dependency Treebank.,* in J. Hajič and J. Nivre (eds.), *TLT 2006. Proceedings of the Fifth Workshop on Treebanks and Linguistic Theories. December 1-2, 2006, Prague, Czech Republic*, Institute of Formal and Applied Linguistics, Prague 2006, 67-78.

D. Bamman and G. Crane, *Building a Dynamic Lexicon from a Digital Library*, in *Proceedings of the 8th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL 2008)*, Pittsburgh 2008.

D. Bamman, M. Passarotti, G. Crane and S. Raynaud, *Guidelines for the Syntactic Annotation of Latin Treebanks*, «Tufts University Digital Library», 2007a. Available at: http://dl.tufts.edu/view pdf.jsp?urn=tufts:facpubs:dbamma01-2007.00002.

D. Bamman, M. Passarotti, G. Crane and S. Raynaud, *A Collaborative Model of Treebank Development*, in K. De Smedt, J. Hajič and S. Kübler (eds.), *Proceedings of the Sixth International Workshop on Treebanks and Linguistic Theories. December 7-8, 2007, Bergen, Norway,* Northern European Association for Language Technology (NEALT) Proceedings Series, Vol. 1, Bergen 2007b, 1-6.

D. Bamman, M. Passarotti, R. Busa and G. Crane, *The annotation guidelines of the Latin Dependency Treebank and Index Thomisticus Treebank. The treatment of some specific syntactic constructions in Latin*, in *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC 2008),* Marrakech, Morocco 2008.

J. Bresnan, *Lexical-Functional Syntax*, Blackwell Publishers Ltd, Oxford 2001.

S. Buchholz and E. Marsi, *CoNLL-X shared task on multilingual dependency parsing*, in *Proceedings of the X CoNLL Shared Task*, SIGNLL, 2006.

[10] The development of the dynamic valency lexicon will refer to the work done in existing similar lexica such as PDT-Vallex (Hajič et alii 2003), PropBank (Kingsbury and Palmer 2002) and VALEX (Korhonen et alii 2006). In addition, a dynamic lexicon for Latin and Greek is now in progress from the Classical texts in the *Perseus Digital Library* (Bamman and Crane 2008).

R. Busa, *Index Thomisticus: sancti Thomae Aquinatis operum omnium indices et concordantiae, in quibus verborum omnium et singulorum formae et lemmata cum suis frequentiis et contextibus variis modis referuntur* quaeque / consociata plurium opera atque electronico IBM automatousus digessit Robertus Busa SJ, Frommann-Holzboog, Stuttgart-Bad Cannstatt 1974–1980.

A. Cahill, M. McCarthy, J. van Genabith and A. Way, *Automatic Annotation of the Penn-Treebank with LFG F-Structure Information*, in A. Lenci, S. Montemagni and V. Pirrelli (eds.), *LREC 2002 workshop on Linguistic Knowledge Acquisition and Representation - Bootstrapping Annotated Language Data, Proceedings of LREC 2002, Third International Conference on Language Resources and Evaluation*, ELRA - European Language Resources Association, Paris 2002, 8-15.

J. Carroll, T. Briscoe and A. Sanfilippo, *Parser Evaluation: a Survey and a New Proposal*, in *Proceedings of the First International Conference on Language Resources and Evaluation (LREC 1998). May 28-30, 1998, Granada, Spain*, 1998, 447-454.

N. Chomsky, *Aspects of the Theory of Syntax*, The MIT Press, Cambridge, Massachusetts 1965.

G. Crane, *Generating and Parsing Classical Greek*, Literary and Linguistic Computing, 6(4), 1991, 243-245.

J. Hajič, J. Panevová, E. Buránová, Z. Urešová and A. Bémová, *Annotations at Analytical Level. Instructions for annotators*, Institute of Formal and Applied Linguistics, Prague 1999. Available at: http://ufal.mff. cuni. cz/pdt2.0/doc/manuals/en/a-layer/pdf/a-man-en.pdf.

J. Hajič, J. Panevová, Z. Urešová, A. Bémová, V. Koláfová-Rezníccková and P. Pajas, *PDT-VALLEX: Creating a Largecoverage Valency Lexicon for Treebank Annotation*, in J. Nivre and E. Hinrichs (eds.), *TLT 2003 – Proceedings of the Second Workshop on Treebanks and Linguistic Theories*, volume 9 of *Mathematical Modelling in Physics, Engineering and Cognitive Sciences*, Växjö University Press, Växjö, Sweden 2003, 57-68.

D. Haug and M. Jøhndal, *Creating a Parallel Treebank of the Old Indo-European Bible Translations*, in *Proceedings of the Language Technology for Cultural Heritage Data* Workshop (LaTeCH 2008), Marrakech, Morocco 2008, 27-34.

P. Hellwig, *Dependency Unification Grammar*, in *Proceedings of the 11th International Conference on Computational Linguistics*, Universität Bonn, Bonn 1986, 195-198.

R. Hudson, *English Word Grammar*, Blackwell Publishers Ltd, Oxford 1990.

R.M. Kaplan, *Syntax,* in R. Mitkov (ed.), *The Oxford Handbook of Computational Linguistics*, Clarendon Press, Oxford 2003, 70-90.

P. Kingsbury and M. Palmer, *From Treebank to Propbank,* in *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002).* Las Palmas, Canary Islands, Spain 2002.

U. Koch, *The Enhancement of a Dependency Parser for Latin. Research Report AI-1993-03*, University of Georgia Athens, Georgia, USA 1993. Available at: http://www.ai.uga.edu/ftplib/aireports/ai199303.pdf.

A. Korhonen, Y. Krymolowski and T. Briscoe, *A Large Subcategorization Lexicon for Natural Language Processing Applications,* in *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, Genoa, Italy 2006.

C.H.A. Koster, *Constructing a Parser for Latin*, in *Proceedings CICLing-2005*, Springer LNCS 3406, 2005, 48-59.

R. McDonald, F. Pereira, K. Ribarov and J. Hajič, *Non-projective dependency parsing using spanning tree algorithms*, in *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, 2005, 523-530.

I. Mel'cuk, *Dependency Syntax: Theory and Practice*, State University Press of New York, Albany/NY 1988.

J. Mírovský, *Netgraph: a Tool for Searching in Prague Dependency Treebank 2.0*, in J. Hajič and J. Nivre (eds.), *TLT 2006. Proceedings of the Fifth Workshop on Treebanks and Linguistic Theories. December 1-2, 2006, Prague, Czech Republic*, Institute of Formal and Applied Linguistics, Prague 2006, 211-222.

J. Nilsson and J. Nivre, *MaltEval: An Evaluation and Visualization Tool for Dependency Parsing,* in *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco 2008.

J. Nivre, J. Hall and J. Nilsson, *MaltParser: A data-driven parser-generator for dependency parsing,* in *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC 2006)*, Genoa, Italy 2006, 2216–2219.

P. Osenova and K. Simov, *The Bulgarian HPSG Treebank: Specialization of the Annotation Scheme*, in J. Nivre and E. Hinrichs (eds.), *TLT 2003 – Proceedings of the Second Workshop on Treebanks and Linguistic Theories*, volume 9 of *Mathematical Modelling in Physics, Engineering and Cognitive Sciences*, Växjö University Press, Växjö, Sweden 2003.

M. Passarotti, *LEMLAT. Uno strumento per la lemmatizzazione morfologica automatica del latino*, in F. Citti and T. Del Vecchio (eds.), *From Manuscript to Digital Text. Problems of Interpretation and Markup. Proceedings of the Colloquium (Bologna, June 12th 2003)*, Papers on Grammar, IX-3, Herder, Roma 2007a, 107-128.

M. Passarotti, *Verso il Lessico Tomistico Biculturale. La treebank dell'Index Thomisticus*, in R. Petrilli and D. Femia (eds.), *Il filo del discorso. Intrecci testuali, articolazioni linguistiche, composizioni logiche. Atti del XIII Congresso Nazionale della Società di Filosofia del Linguaggio, Viterbo, 14-16 Settembre 2006,* Aracne Editrice, Pubblicazioni della Società di Filosofia del Linguaggio, 04, Roma 2007b, 187-205.

H. Pinkster, *Latin Syntax and Semantics*, Routledge, London 1990.

C. Pollard and I.A. Sag, *Head-Driven Phrase Structure Grammar*, University of Chicago Press, Chicago 1994.

G. Sampson, *English for the Computer: the SUSANNE Corpus and Analytic Scheme*, Clarendon Press, Oxford 1995.

H. Schmid, *Probabilistic part-of-speech tagging using decision trees*, in *International Conference on New Methods in Language Processing*, Manchester, UK 1994.

P. Sgall, E. Hajičová and J. Panevová, *The Meaning of the Sentence in its Semantic and Pragmatic Aspects*, D. Reidel, Dordrecht, NL 1986.

J. Sinclair (ed.), *Looking up: an Account of the COBUILD Project in Lexical Computing*, London 1987.

J. Sinclair, *Corpus and Text: Basic Principles*, in M. Wynne (ed.), *Developing Linguistic Corpora: a Guide to Good Practice*, AHDS, 2004, 5-21. Available at: http://www.ahds.ac.uk/creating/guides/linguistic-corpora/index.htm.

L. Tesnière, *Éléments de syntaxe structurale*, Editions Klincksieck, Paris 1959.

I. Titov, J. Henderson, *A Latent Variable Model for Generative Dependency Parsing*, in *Proceedings of the International Conference on Parsing Technologies (IWPT-07)*, Prague 2007.

E. Tognini-Bonelli, *Corpus Linguistics at Work*, J. Benjamins, Amsterdam Philadelphia 2001.

Abstract. This paper concerns Latin annotated corpora, referring to three now available Latin treebanks, and describing in particular the *Index Thomisticus* Treebank project features. Some general premises about corpus linguistics are given, focusing on different grammar frameworks for annotation and empirical approaches in linguistic research.

Latin, corpus linguistics, treebanks, Index Thomisticus

AN EXPERT SYSTEM FOR THE CLASSICAL LANGUAGES: METRICAL ANALYSIS COMPONENTS

1. *Introduction*

Almost 40 years ago, when the first computer applications to the humanities were just beginning to appear, A.W. Bulloch could already write that "our understanding of the Greek hexameter [...] would certainly be fundamentally reestablished, if not revolutionized, if all known examples were to be analysed on a computer according to their most important characteristics"[1]. Such a view is probably too emphasized, anyway it would be easy to show that at least two points still today can be dramatically relevant in the study of a wide range of prosodical, metrical and linguistic phenomena of the Classical texts: the availability of massive and detailed data collected with a uniform method, and the requirement of a well-defined theory formulated in a strongly formalized way to collect them.

As for the first point, at least since the XIX century philologists have been providing many studies devoted to a huge number of aspects of Classical versification in very different contexts, from general or monographical studies to manuals, or even occasional journeys in the realm of metrics only to support a specific hypothesis about any other field of Classics. As a result, even today most of the numerical figures collected by such studies are heavily influenced by their origin and purpose: they refer to portions of selected works ranging from a few tens to some thousands lines, collected by scholars spanning more than two centuries, and often lead (and sometimes mislead) by very different theorical beliefs[2]. Still, even the manuals refer to some of these studies to present what should be the plain description of the phenomena: it's a matter of fact that even today the various specific studies produced in the latest centuries are the only source to get at least approximate figures about them. Of course, the scholarly studies of past centuries are the big foundation of any modern work, but we are facing lots of limitations due to the simple fact that humans cannot spend their lives collecting data from thousands and thousands of lines, syllable by syllable. That's more a machine work.

For instance, the fundamental Metrik by P. Maas[3] reports violations of Hermann's law or spondaic hexameters every 1000 and 50 lines respectively, but in the English edition of the same work[4] these figures are clearly undersized to "about" every 390 and 18 lines (i.e. from 0.1% and 2% to 0.26% and 5.56%). Nonetheless,

[1] Bulloch 1970.
[2] A clear example of theorical assumptions leading to completely unreliable data and conclusions is the well-known study by O'Neill (1942), which is still quoted as the source for many numerical figures.
[3] Maas 1923.
[4] Maas 1972.

many studies and manuals still repeat the first figures, and there is no way to verify them other than simply repeating the analysis. The fundamental problem here is that all such data come from analysis necessarily limited to very small samples (furthermore coming from different and sometimes outdated editions of the sample text) by many different scholars, who rarely attempt to explicitly state all the implicit assumptions in their data collection method. Yet these are the best sources of data, as in too many other cases scholars have limited themselves to verbal approximations which apart from being subjective are simply unusable for any other purposes (nobody could try to base a new hypothesis on indications like “often”, “rarely”, “about”, “generally”, etc.). Furthermore, the numbers themselves can be misleading if we don’t test their significance with adequate statistical tools (what happened almost always in past studies, while nowadays many recent studies have realized and applied this principle). Lastly, such tools are invaluable for determining the significance of a phenomenon but are completely useless if the data they are applied to have been collected with non-uniform methods (a non-systematic error in the measurement tool would make any statistical processing unreliable).

This leads us to the second point: method. Here things can become even more uncertain, not only because of different theorical beliefs, but also because almost no theory is without “obscure” regions which are left to scholars judgment or common-sense: everyone reading a grammar or a metrics manual is familiar with approximations like “often”, “usually”, “etc.”, “and the like”, but unfortunately a machine cannot operate with such vagueness. Of course many of these problems cannot be fully defined from a theorical point of view, but whoever wants to apply computer analysis must at least try to find a practical solution for them. This is at the same time the most difficult and intriguing character of computer-related solutions applied to humanities: every single theorical aspect must be fully defined and formalized with no room for uncomplete or vague statements. To make a trivial sample, one may think about the detection of word-ends in a line, implying an (at least practical) definition of what can be considered a “word” in metrical and linguistic terms, and the much debated notion of appositive words. From such a definition derive fundamental assumptions for word-ends, metrical laws and inner metre structure: speaking of wordends and bridges, already Maas clearly pointed out the relevance of what he called the *Wortbild* for their correct evaluation, as e.g. a sequence of (graphical) words like *ho patèr gàr* should be treated as a unique word for the purpose of metrics, and it’s easy to see that this syntethic definition implies a number of criteria like metrics (*cesurae* and bridges), semantics (words somewhat considered ‘meaningless’ like conjunctions or particles), syntax (cfr. the word-order constraints and the grammatical classes of proclitics and enclitics), prosodies (mainly accent: clitics vs. ‘full’words). Because of such complexity, the definition of “word” still largely remains confined in the area of the pre-theorical linguistical notions any native speaker is aware of, whence the well-known ‘definition’ by Lyons («a Word is what you think is a Word»[5]). This is mainly due to the fact that there are several definitions of word according to the linguistical level taken into account: phonology (the word as an accentual unit, where the accent is truly the *anima vocis*, and ideally one accent corresponds to one word), morphology (the word as the biggest morphological constituent, composed by morphemes, like in Greek *e-philé-sa-men*: augment, theme, suffix, ending), syntax (where instead the word is the minimal constituent, the smallest element for building up phrases and sentences), and semantics (especially our metrics handbooks or grammars are full of sometimes impressionistic distinctions between ‘fully meaningful’ and ‘meaningless’ words, the latter being elements like particles, conjunctions, prepositions, etc.; a more recent terminology today distinguishes between ‘lexical’ and ‘non-lexical’ words according to their level of referentiality). The last level is one of the most widely used (and somewhat abused especially in metrics), and dates already to Aristotle’s (mainly logical) notion of *sýndesmos*, which is asemantic like the *stoikheion* or the *syllabe*, but it’s used not for building semantic words like these, but rather to connect the terms of a proposition[6]. Of course, in more modern terms no sign as such could be considered as meaningless, and words with purely grammatical semantic traits have a meaning which simply varies according to their context, either linguistic or extra-linguistic (the so-called *shifters*, like anaphorics or demonstratives). The syntactical level in turn, much more important in modern theories, is one of the most widely used tools for defining the ‘parts of speech’, by considering words for their role in the sentence rather than in the abstract collection of a dictionary: their mobility in the sentence and their property of not being divided by other linguistical material are strong criteria for a syntactical definition, and as such were widely used by fundamental studies like Dover 1960; yet, this is prone to a some kind level of abuse in a metrical context, where several references to a ‘syntactical connection’ often stem from purely aesthetical considerations. Anyway, as pointed by Jakobson, in metrics the appositives question is purely rhytmical, and its unique direct connection is with phonology. For ‘dependent’ words we must distinguish (Klavans 1985) between a phonological and a syntactical host. Also, it’s easy to remember that phonological and syntactical structures are anisomorphic, as

[5] Cfr. Fruyt 1980, 114.
[6] Cfr. e.g. the traditional *mére lógou* scheme in Lallot 1980, 129.

we can clearly learn from ambiguous sentences, where the same words can be grouped differently (e.g. *polỳ pleîstoi kaì áristoi*). Thus, several scholars even in recent times have completely abandoned any attempt of dealing with such a complex notion, and provide metrical studies which either treat each sequence of characters separated by spaces in our printed text as a ‘word’, or often rely to impressionistic considerations in treating the same word both as dependent and independent. This of course provides no ground for a serious quantitative analysis, which should rather rely onto a theorical framework defined according to several combined criteria, like Greek graphical system, ancient grammatical traditions, phonology (from both a synchronic and diachronic perspective), morphology, syntax and a number of affecting factors (like e.g. marked word order, phonological extent as related to frequency of use, etc.). If we try to collect evidence from each of these areas combining it with our knowledge of metrics we can afford a well-founded enough and at least working definition for Greek ‘lexical’ and ‘appositive’ words, thus enabling us to build a computer tool capable of producing a full prosodical and metrical analysis of any Classical verse, which can prove very useful in providing at least the massive and yet fully detailed data for the *observatio* which should be the foundation of any hypothesis, thus contributing to better define some of the most debated questions in the metrical field. I have already tried to show some examples of such results in papers about even well-known and long-studied phenomena like hexameter’s laws and their evolution in time from Homer to Nonnus (e.g. the paradigmatic and syntagmatic nature of Hermann’s and Lehers’ laws[7]), or about the linguistical implications of peculiar metrical usages in late-Latin poets like Luxorius[8].

Finally, when analyzing metrics we must never forget that we are dealing with very complex organisms where linguistic material is structured according to several combining factors. When we want to study a single phenomenon we should be able to isolate at least the most important bias factors affecting it, and this requires an extremely detailed and huge amount of data, even beyond the scope of our (or others) specific study. Let’s make a trivial sample: consider the dactylic hexameter. Here the combinations of spondaic and dactylic feet produce 32 variants. Of course dactylic and spondaic feet are not evenly distributed along the line, but they vary, with preference for dactyls especially towards line end. For instance, in the very

[7] Fusi 2002.
[8] Fusi 2004.

short hexametric text I'm using here as a sample (Aratus *Phaenomena*, 1153 lines) the spondaic feet are distributed as in the following chart[9].

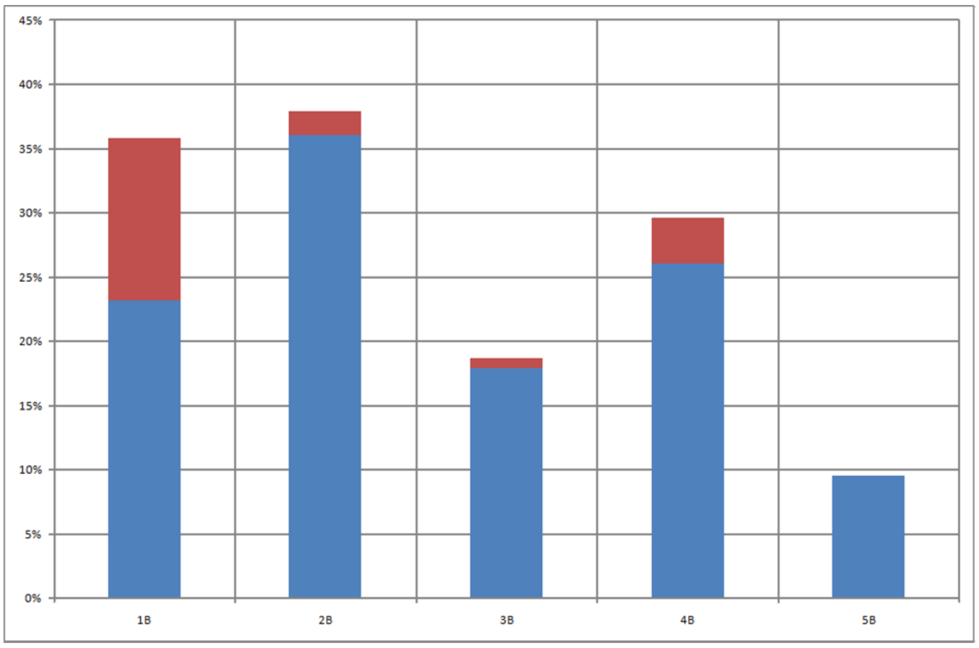


Chart 1: distribution of spondaic feet in positions 1-5

The chart shows the percentages of spondaic feet at each position (1-5 from left to right); the topmost part of each bar with a different color represents the portion of these spondaic feet with a "true" wordend[10] after them[11].

Now, consider the distribution of wordends in the most sensitive positions we call *caesurae*: among them the most important is the feminine, which cuts the two short syllables in the third dactylic foot: this of course implies that we must have a dactyl in the third foot. Thus, here we have two combining factors at play: the frequency of wordends at each position in the line, and the distribution of dactylic and spondaic

9 This chart as all the data discussed in this paper come from the computer analysis illustrated here, but are limited to a very small sample to avoid more complex discussions on genres, chronology, etc. as my aim here is just to present an overview of this expert system. For the same reason in this paper the charts appear very small as they are meant to just provide a sketch of the phenomena (refer to the accompanying slides for bigger versions).

10 For the meaning of this distinction see section 4 (Syntax) below.

11 This specific distinction is due to well-known tendencies to avoid a wordend after spondaic foot especially in certain positions, as expressed by hexameter "laws" like Wernicke, Hilberg and Naeke.

feet. If we want to study wordends in isolation we should remove the bias of the latter on the former, e.g. “weight” the wordends on the feet types.

Chart 2 shows how our results may vary: it represents the frequency of wordends in the hexameters of Aratus from line beginning (=left) to end (=right). As you can see, the peaks correspond to *caesurae* and the valleys to bridges (e.g. Lehrs and Hermann, to quote the most known). Here the lines represent the absolute percentage of wordends, while the areas represent the percentage calculated no more on the total lines count but on the lines showing either dactylic or spondaic foot at each sensible position. As you can see, things vary noticeably, and the “weighted” percentages are much higher. You can also see in the bottom part of the chart (f-line in the slides) the distribution of false wordends, i.e. wordends involving appositives, which too were kept isolated from true wordends.

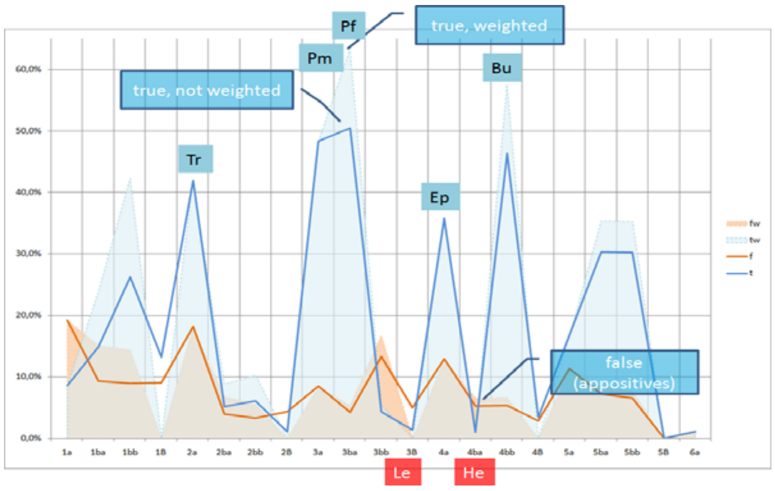


Chart 2: distribution of wordends along the whole line. Labels mark *caesurae* and some bridges. Areas refer to “false” wordends.

Another very trivial sample can illustrate the interaction among language and metrics: elision has a complex profile involving the types of words it affects, but first of all it implies the obvious condition that there must be a wordend to allow for it. This brings into play the distribution of wordends in the line, on which we should “weight” the distribution of elisions themselves. Let us consider chart 3: it shows the distribution of various types of elided words in the line.

Notice the peaks at the positions corresponding to the *caesurae* in the third and fourth feet: they look so high that one might be tempted to infer that there is some sort of correlation between wordend in these positions and elision. Yet this is not the case, as we can see if we “weight” our percentages by calculating them no more on the total lines count, but rather on the frequency of wordend at each position.

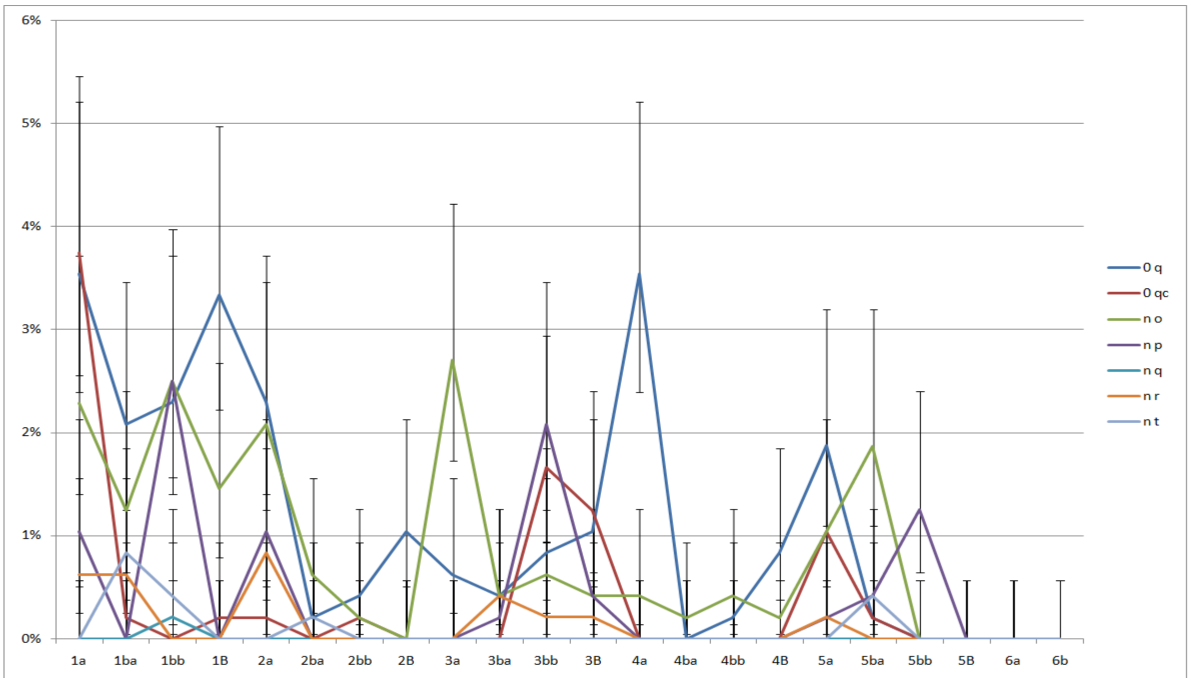


Chart 3: percentages of elisions along the whole line, calculated on the total count of lines examined. Different lines refer to different word classes.

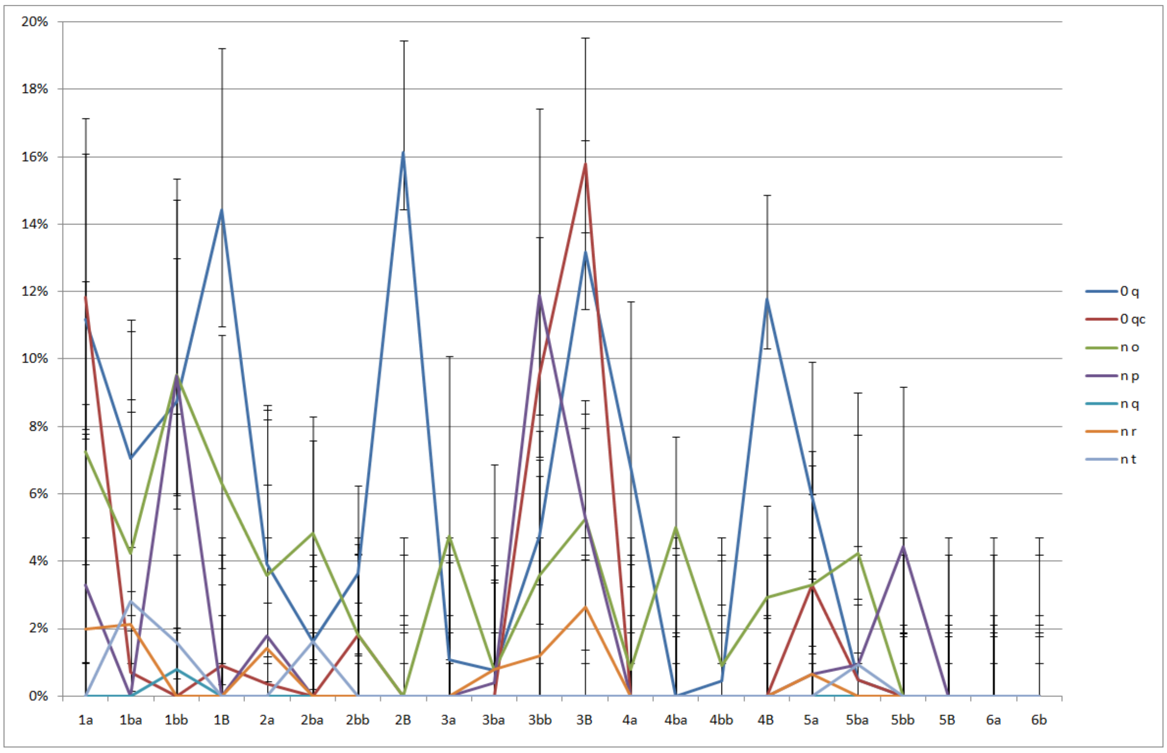


Chart 4: percentages of elisions along the whole line, calculated on the frequency of wordends at each position.

As you can see, things change completely: now at the fourth foot we have a valley instead of a peak. The trivial reason for that is of course that wordend is the necessary condition for elision, and thus wherever wordends are frequent (*in caesura*), we can expect elisions to be frequent, too. Without taking this simple consideration into account we might be fooled by apparently correct data. Of course, this requires us to provide detailed data about elisions and wordends at each single position in the line for each line in our sample: it is precisely here that our machine analysis can prove useful.

These are just trivial samples I chose for their simplicity, but it is easy to understand that in analyzing any aspect of a metrical text we should keep into adequate account all the factors which concur in producing a surface phenomenon. This often requires us to provide much more data than what we could expect at a first glance, even if we just need them to properly view our specific subject of study. In contrast with all the limitations we have summarized here, machine analysis provides the possibility to examine huge amount of texts with extreme levels of detail, collecting data for thousands of lines down to the single phoneme. Even more important, it implies the definition of a highly formalized method, and it grants its rigorously uniform application throughout the whole text sample. This in turn allows us to properly use statistical methods to test the significance of our data and thus confirm or reject the validity of working hypotheses. Finally, a machine-driven analysis can be repeated indefinitely on new texts building on top of existing data, either to update them by using new editions or to add new texts or new types of obervations.

Of course, this comes at a price: the machine knows nothing of metrics and language and its analysis is purely formal (and strictly speaking this might not be so bad for prosodies and metrics), so every single methodological detail must be fully defined without ambiguities or approximation. This requires a big effort for both scholars and programmers, but I think it can be rewarding not only for metrics but also in the context of the more general expert system some parts of which I'll try to show in the following overview.

2. *System Overview*

The metrical analysis system I shortly present here fits into a much bigger picture where highly specialized components (metrical analysis, automatic inflection of Latin and Greek language, lemmatization, etc.) operate side by side with more generic digital editions frameworks. Any account of this system would be outside the limited scope of this short presentation, but this is one of the reasons for the componentized structure and abstraction level of several aspects of this metrical subsystem. For instance, all the components which provide the full phonological and prosodical analysis of a line are completely shared among other subsystems, first of all the morphological one, whose aim is to generate all the inflected forms for a given word in a historical perspective. As any student of language learns since his first grammar, morphology of course requires phonology: knowing where syllabic boundaries fall, whether a vowel or a syllable is short or long, where the accent is located, etc. is often a condition for the formulation of several grammar rules. At the same time, metrics shapes most of the same phonological phenomena into more or less complex patterns, and this explains why the phonological analysis components are effectively shared among these different subsystems. Furthermore, many theorical aspects at their foundation can be generalized at such a level of abstraction that they can be easily applied to both Greek and Latin, or even to any other language. Thus, in this context not only components are shared among subsystems (e.g. phonological analysis serving both metrics and morphology), but also parts of their implementations (e.g. the detection of theorical syllabic boundaries according to phonematic openings used by any languagespecific syllabification function: Greek, Latin and even Italian). Finally, all these components work together in even bigger frameworks, for instance when a digital corpus provides sample texts for metrical analysis and this in turn outputs its results in a form which can be easily integrated into the original corpus, thus creating a specialized edition built on top of an existing one.

In the following sections I'll present a very short account of the most important components used by the metrical subsystem, leading the reader through the essential stages traversed by my software in its analysis.

3. *(a) Prosodies*

The first stage for metrical analysis consists in importing the text itself. Typically the texts to analyze are extracted from large digital corpora like Packard Humanities Institute (PHI) cd-roms using another software component I created for this purpose: it can read any PHI cd-rom, extract the desired portion of text and fully recode it from Betacode into Unicode or into any encoding, either standard or not, using a heuristic approach. In the context of the bigger picture illustrated above this component is shared among several other subsystems, like e.g. digital editions, to generate output in any textual encoding and format ((X)HTML, RTF, XAML, XPS, etc.). The same component is also used wherever the metrical subsystems needs to get text input from the user, or conversely to output some formatted text. Again the same component is used also in the form of a Word addin to ease the input of ancient Greek Unicode text, thus providing popular word processors like Word with the powerful editing and conversion environment specialized for Greek and Latin texts used in my own software (including digital epigraphical editions).

Thus, whatever the input text format and encoding may be (either coming from digital corpora or directly typed by user), the software finally gets a plain Unicode text to analyze. Of course, this text is almost always somewhat complicated by additional data like e.g. line numbers, layout features (line indent and spacing), special characters (e.g. the diplai in the TLG cd-rom text of Homer), etc. The first stage of the analysis proper thus consists in a smart parsing of the input text as we need to remove all the irrelevant "noise" characters: some (like line numbers) are used to collect metadata, others (like non-textual characters as diplai, parentheses, etc.) are just discarded, but in both cases they will be restored at their place when generating an output for the end user. The text filtered in this way is then normalized (for e.g. extra spaces, letters casing, glyph variants, etc.) so that the phonemic analysis can get an input where all the confusing or irrelevant variants have been removed.

A process we may call "phonemization" then occurs after parsing: at this stage, the program uses a set of external phonological parameters (each specific for a given language) to deduce a sequence of phonemes (or allophones) from the input sequence of graphemes. All the required contextual analysis (like e.g. the detection of diphthongs) happens at this stage too, as of course deducing sounds from letters is never a one-to-one mapping (cf. e.g. Latin «x» = /ks/ and «qu» = /kw/), but often complex algorithms are involved.

All the language-specific data at this stage are kept in a set of XML files separate from the program itself, which tell how to interpret each letter phonetically (e.g. providing point and mode of articulation, highness, phonological traits like voicing, rounding, length, etc.). This allows the same program to work for different languages like Greek and Latin[12].

The program uses this information to build a structure which will be the basis for every successive analysis: the text is represented as a chain of segments, each linked to any number of data stored in different "layers" : such layers tell whether a given segment is long or short, where syllabic boundaries fall, how words are connected, how syllable weights relate to metrical schemes, etc.: all the data calculated by the system in the next stages are stored in these layers, which can grow indefinitely so that we are free to link specific information to each phonological segment without complicating the analysis with the insertion of non-textual material among them.

[12] At the time of writing the metrical components are ready only for Greek, but the phonological analysis of Latin is the basis for automatic inflection of this language.

As for the analysis of Greek graphemes (and even more for Latin, where all the letters are dichronae), further difficulty arises from the so-called *dichronae* letters[13] A I Y, which can represent either short or long vowels. The software is able to tolerate such lack of information, but it strives to find out every possible bit of data: it uses word accentuation to infer lengths, but it also recurs to a sort of “prosodical lexicon” which can be queried during analysis. This lexicon is generated and mantained by the software itself using contextual analysis (see below under section 5). A second issue comes from a great deal of prosodical variants which can severely affect the verse: for instance, the alternative syllabic divisions of some consonant groups (e.g. the so-called *muta cum liquida* group and more rarely other sequences), the potential redoubling of some consonants in specific word positions (e.g. initial nasal or liquid), and finally hiatus, which may shorten a long vowel or diphthong (*correptio epica*). All these parameters may optionally affect the line, and again are stored in separate XML files, specific for each language (or for each variant of a language, which is the case for the different Greek literary dialects used in various genres). Such files tell the software that some consonantic sequences or single consonants, eventually in specific word positions, might or not trigger a different syllabification by virtue of tautosyllabic measurement or reduplication, with different levels of probability. At this stage the software takes no specific action in such cases and usually applies the standard treatment, but it keeps this information in a layer linked to the related segments, so that the metrical component will be able to use it later.

Finally, the prosodical subsystem detects syllabic boundaries by analyzing the opening of each phoneme in the segmental sequence defined earlier. To do this it relies on Saussure classical model of the so-called phonological syllable, which is a universal model defining a syllable as the distance between the two segments with the lowest opening value. Since the program has already collected data about each phoneme, including its opening, it can easily infer the theorical syllabification, which can be output to the user for diagnostic purposes as shown in chart 4.

[13] For this definition cf. Rossi 1963.

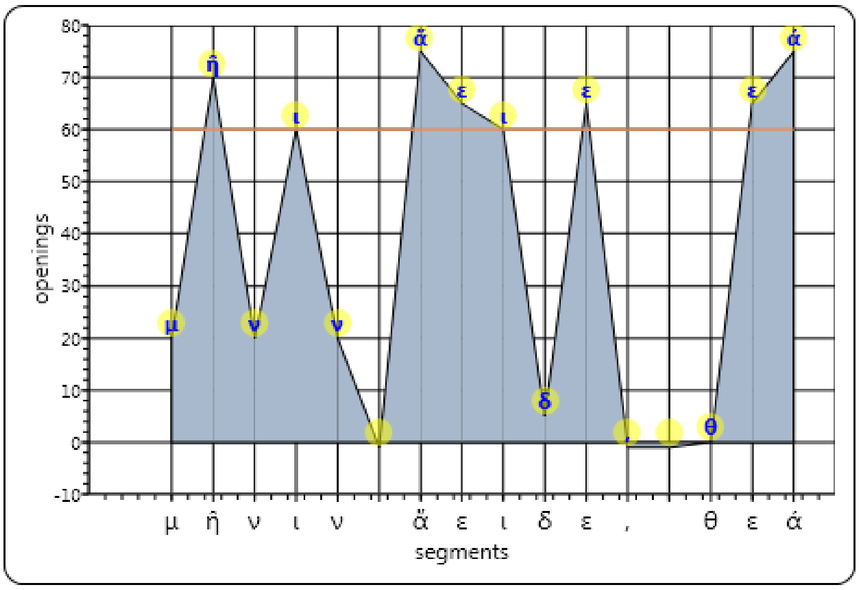


Chart 5: computer analysis of phonematic openings in the first words of the *Iliad*

This chart shows the syllabification of the first words of Iliad, 1. As can be seen, the openings drive the detection of syllabic boundaries, with their subsequent adjustments for Greek language. Of course, this model is just an approximation and each language requires its own specific adjustments[14], so after a first detection syllabic boundaries are adjusted by specialized functions.
Once syllabic boundaries are in place, it's relatively easy to detect (wherever possible) syllable weights, as of course any closed syllable is heavy, while in open syllables the weight depends on vowel length.

This defines a very efficient model for phonological analysis: the same software components can be shared among several specialized analysis subsystems, e.g. for Greek, Latin or Italian. All the components which parse a text, phonemize it, add special markings for optional prosodical treatments, and provide a generic syllabification are fully shared. What is specific to each single language is relegated to external XML parameter files. Only the last stage, which adjusts syllable boundaries, is necessarily specific for each language.

[14] Think e.g. of a sequence like Latin *stare*: here the first "phonological" syllable would be just *s*-, as it's followed by a lower opening segment (*t*, a voiceless plosive). Even if this has some support from the evolution of the language itself (cf. the prothetic vowel in Italian *istare*), the "phonological" model here must be corrected with the "phonetic" one, whence the two syllables *sta.re*.

4. *(b) Syntax*

After prosodies, the second big subsystem is the syntactic one. It deals with words classification by distinguishing among the so-called “lexical” words and the crucial class of appositives, which typically have higher textual frequency, lower lexical frequency and a very small size. They in turn include words with and without accent (clitics), which finally part into enclitics and proclitics according to their connection to the left or right.

Any serious metrical analysis cannot be done without this words classification, as it literally shapes the verse for its “inner” metric (wordends, bridges, etc.: as already Maas pointed out clearly, a sequence of graphical “words” like *kaì tòn patêrá mou* is just one “linguistical” word). This is not the place for even a short discussion of this complex subject, but one can immediately grasp the big difference between analyses which take into account this problem and those which don’t by just having a look at charts 5-6.

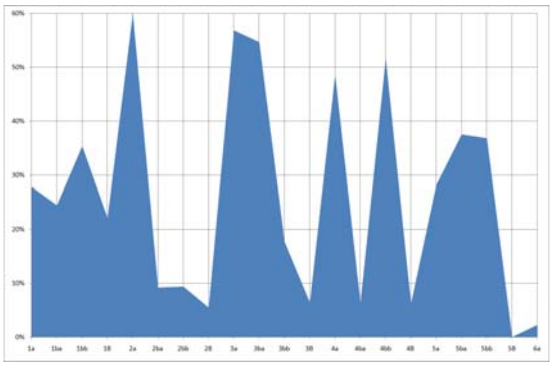


Chart 5: distribution of graphical wordends

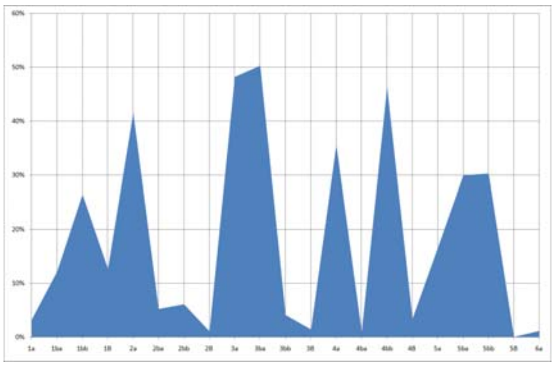


Chart 6: distribution of ‘true’ wordends
(same sample and scale of Chart 5)

Chart 5 shows the distribution of purely graphic (“false”) wordends in our usual sample text, while Chart 6 shows the (unweighted[15]) distribution of “true” wordends. As you can see, peaks in Chart 5 roughly correspond to the well-known caesural places, but they shape a picture where secondary *caesurae* like the trithemimeres would appear higher than primary *caesurae* like the penthemimeres (cf. the leftmost peak), and even the most severe bridges (like Hermann or Lehrs) would allow for an unexpectedly high percentage of violations (cf. the valleys at about the middle of the horizontal axis). Of

[15] For this concept see the above discussion about Chart 2.

course this picture is completely distorted by the superficial treatment of each graphical space as a true wordend boundary, as if in English we were considering as full “words” the articles “the” and “a” or the preposition “in” in a sentence.

If instead we apply a more realistic linguistical analysis to our sample text, by taking into adequate account word types and their *liaison* direction and accentuation, what we get is Chart 6: here you can easily see that the curve visually defines the expected shape for the (Hellenistic) hexameter: for instance, peaks to the left and right have been lowered and the prominence of the penthemimeres caesura clearly emerges; also, the feminine caesura looks more frequent than the masculine (cf. the top-right angle in the central peak against the topleft one in Chart 5); and finally, bridge positions show clearly deeper valleys corresponding to a much lower percentage of violations. This time the chart truly provides a visual representation of the inner structure of the metre with its well-balanced distribution of wordends peaks.

This trivial sample should be enough to show the crucial relevance of a linguistically serious treatment of wordends. Of course, an adequate and at least pragmatically usable definition of appositives requires a lot of effort and the combination of phonological, syntactical, semantic, lexical and rhythmical factors: but the complexity of such a definition should not be an excuse for avoiding it at all and simply sticking with graphical words, as the same common sense which does not trouble the native speakers of a language in defining a “word” shows us that we could never place at the same level “words” like te or perí and “words” like *polýtropos*. Of course, it’s not just a matter of mechanical classifications: it would be wrong to rely on a single aspect like monosyllabic or bisyllabic body, presence or absence of accent, lexical character etc… The status of a word comes from the combination of several factors, and we must also face with a heterogeneous graphical system (which of course is the only source of data for the machine) which often hides further complexities (e.g. think of the purely graphical accentuation of proclitics).

The analysis is further complicated by the fact that words must be analyzed in their context, which may severely alter their surface shape: think for instance of sequences of clitics with eventual development of enclisis accents, or of the phenomenon of barytonesis, and of even more complex syntagmatic phenomena like the so-called “continuatives”. To make a trivial sample, one makes just think of this three “words” sequence: *perì d’ouranón*. From left to right we have a proclitic word (no accent and connection to the right; the accent we note here is merely graphical, and the software must know it), followed by an elided enclitic (which too has no accent, even if it bears a graphic one in its unelided form) and finally by an orthotonic lexical word. Now, in this sequence the short body of the elided *d’* isn’t enough to stop the preceding preposition *perí* to connect not only with this particle, but also to *ouranón*: in this sense we say that *d'* is a continuative enclitic, as it allows the connection to the right of the preceding preposition to continue even after itself to make a bigger group[16].

Thus, the syntactic subsystem uses very complex syntagmatic algorithms to take into account all the surface changes of these words and detect their nature. During analysis, it takes all the data about appositives (paradigmatic form, "true" and graphical accentuation, direction of connection, continuative potential, etc.) from a relational database, and it enables the metrical subsystem to deal with some 16 types of wordends, as defined by the combination of four factors: true or false – i.e. merely graphical – wordend; presence of hiatus between words; presence of aspiration in hiatus; presence of elision in hiatus. All this information[17] is stored as usual in the data layers linked to the text segments.

5. *(c) Metrics*

The third subsystem after prosodies and syntax is finally metrics. Until now, the system has taken a text, parsed and converted it into Unicode, defined its phonological values and syllabification, and classified its words: all this has been done by accessing several parameters from XML files and relational databases.

The metrical subsystem too takes its parameters from an external XML file: it contains the definition of any verse design we want to use (either stichic or strophic: hexameters, elegiacs, trimeters, etc.). Its task is to generate all the possible implementations of each verse design, and match the sequence of syllabic weights coming from earlier analysis to them. This is not a straightforward process, as apart from potential ambiguities (wherever a sequence of unknown weight syllables is long enough to cause them), the matching process itself is designed so to trigger relevant changes in prosodies, which in turn imply a new scansion. For instance, as required by context it could move syllabic boundaries in a group of *muta cum liquida*, or redouble a consonant, or shorten a long vowel in hiatus, etc. In some cases, because of the potential lack of data, the program can prompt the user to

[16] For this concept see especially Cantilena 1995.

[17] This subsystem also copes with some language-specific issues like e.g. the contextual disambiguation between **to*- derived forms in Greek: in this language, forms derived from IE **to*- may be anaphoric pronouns (orthotonic), articles (proclitic) or relative pronouns (prepositive). Six rules are provided to achieve a semiautomatic distinction of anaphoric and relative / article values. For these rules I draw data mainly from P. Monteil 1963.

resolve ambiguities whenever more than one verse design implementation might theorically fit the given syllabic sequence.

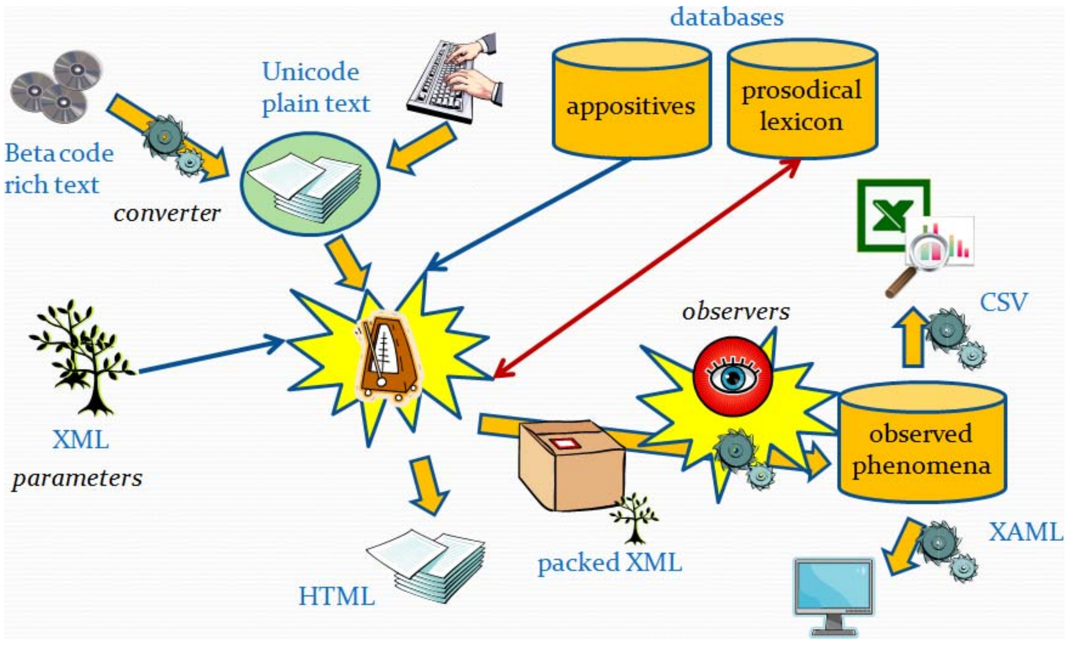
At this stage the software also makes its vowel lengths deductions according to the metrical context, thus feeding the prosodical dictionary cited above. This dictionary is stored in a relational database, which (whenever the user activates this option) is continually enriched by the analysis process itself. Initially empty, the database is filled by the software whenever its metrical scansions give any clue for lengths deductions. After any deduction the corresponding word with its deduced lengths are stored in this dictionary, so that the next time the program finds the same word it will be able to use the deduced lengths and thus reduce the lack of information. This allows the program to “learn” from experience: the more lines it scans, the more words are added to the dictionary[18].

6. *Output, Observation and Evaluation*

This three-tiers system generates several outputs: some are designed to be immediately displayed to the user (e.g. for diagnostic purposes) and are based onto XAML, an XML dialect; other produce richly formatted and detailed (X)HTML files for reporting; and finally all the data coming from analysis (from prosodies up to metrical scansion) are stored in a standard packed-XML format[19] ready to be read by the next subsystem.

[18] Of course, the software is able to generate the paradigmatic form of a word by removing all the phonosyntactic or even graphic modifications that may affect it in a given context (e.g. barytonesis, enclisis accents, letter casing, etc.). It may also happen that a word with more than one unknown length vowel gets only some of them specified by metrical context. The software can handle such cases and also retrieve later the same word to further specify its still unknown lengths as soon as they eventually get specified by other contexts. Furthermore, we must also take into account the possibility of more measurements for the same word form (e.g. *kalós* with either short – like in Attic – or long *a* – like in Ionic – , as a consequence of the different syllabification of the original group -*lw*- and the subsequent compensatory lengthening), or of homographs (e.g. *eruthrá* which could be either feminine singular – long *a* – or neuter plural – short *a*). As for several other cases, fully describing the analysis process would be outside the scope of this paper, but here it’s enough to remark that such problems have been taken into proper account so that we can rely on the most accurate analysis as possible. Obviously even the most accurate machine analysis might not be perfect, but the availability of huge amount of data coming from thousands of lines analyzed with a rigorous and uniform method together with their proper statistic evaluation should allow us to discord such objections as generally irrelevant.

[19] The choice of packed XML is first of all dictated by the desire of reducing the output size, but it also offers a powerful standard to store more structured XML data files.



The report files generated by the metrical analysis subsystem are mainly used for diagnostic purposes, but they can also constitute the foundation of a full “metrical edition” of any given text: we may think of a poetical text where each line has the full detail of prosodical, syntactical and metrical analysis summarized in a synthetic table and available to user interactivity. For instance, in XHTML or XAML format this output contains a full text where the user can read each line with its syllable weights and boundaries and metrical scansion; also, by just hovering the mouse on the text he can get details about each single syllable.

Anyway, the most important output is represented at this stage by the packed XML data which include all the machine-readable information produced by the previous analysis. This information is the input for the fourth big subsystem, which reads the data packed in XML and literally observes them to detect any kind of phenomena we may be interested into, at any level: prosodies (e.g. muta cum liquida, redoubled consonants, accents distribution, etc.), syntax (word types distribution and connections, etc.), metrics (verse instances, “laws”, *caesurae*, bridges, etc.). This architecture is modular, and new specialized “observers” can be added at any time. This allows us to reuse the same analysis data for observing new kinds of phenomena, as it often happens that the study of data makes it necessary to observe new things, or to view existing data under another perspective (cf. our discussion of the bias factors affecting the phenomena under study). At this time, I have implemented 23 observers which collect some 200 data for each single line.

This means that for a *corpus* like the one I analyzed using a previous generation of this system (about 90,000 lines) we could get more than 1 million data.

All these data in their full detail are stored by such observers in another relational database. Once this database has been filled with observations, another software component is used to allow users to query it in whatever form they prefer. Users can ask the program to provide the details (with text line by line) for each combination of any phenomena, or generate synthetic reports by aggregating and filtering data as requested. The program offers a user interface where users can visually build their query by combining any of the phenomena observed with logical operators (AND, OR, NOT), get a subset of them or group them in any desired way. This allows a tremendous flexibility as users have at their disposal a truly digital metrical "edition" of texts which they can interactively query at each time for any given data they are interested in. Instead of a model where data (whatever their detail may be) are output once for all uses, this system implements a model where they are selectively published at each single user requests. This of course is the only way of making such a huge amount of detailed data usable to end-users, who may want to use them for different purposes each them they are studying a specific phenomenon or combination of phenomena. This also allows us to fully appreciate the bias that several factors may exert on surface phenomena, as sampled at the beginning of this paper.

The program outputs here range from formatted (X)HTML with fully highlighted text to detailed data reports in standard formats like XML or CSV. Finally, these files are imported by third-party software specialized for statistical analysis and charting, for instance spreadsheet applications like Excel. Here we can test data for their significance, and emit hypotheses about the explanation of phenomena. Armed with this knowledge we can return any time to the previous program and ask for new data which can further enlighten obscure points and newly arising questions. The system thus grants a fully interactive analysis process, a true laboratory for metrical and linguistical analysis.

7. *Technical Overview*

Any technical detail would be outside the scope of this paper, but given the context of this conference it will be useful to sum up the main aspects of the implementation of the system illustrated above.

As a programmer and philologist I have personally conceived and implemented the whole system, which as shown above has several other applications, ranging from full-featured truly digital editions with any sort of specialized content to many more commercial-oriented applications (multiple language dictionaries, literary *corpora*, search engines, thematic dictionaries, complex redactional applications, etc.).

The software is fully written in C# in the context of Microsoft DotNet Framework. Its most recent portions and all the components which would get essential benefits from the migration have been upgraded to the latest available version of the framework (3.5 SP1, which comes especially handy when dealing with XML data using LINQ and with complex user interfaces using WPF). All the components which build up the system are implemented as several separate modules (assemblies hosted in DLL's), so that the engine is completely independent from the user interface.

User interfaces are almost completely built with WPF, apart from older components still using WinForms. External data are stored in XML files or in relational databases implemented with SQL Server and accessed via ADO.NET or LINQ. Text encoding is always Unicode, but the output can be generated with any even non-standard encoding.

Finally, as pointed above the output is variously represented by SQL Server databases, XHTML + CSS, XAML, XPS, XML. XML transformations are done via XSLT and eventually C# extensions.

8. *Perspectives*

Like every other machine analysis, the system shortly sketched in this paper should be simply regarded as a raw research tool, providing first of all a large and solid quantitative foundation for the observation of metrical phenomena, with special attention to their interferences and to their relation with language and its evolution. As already pointed out by some sample applications of this tool (Fusi 2002 – late-Latin poetry practice in relation with language change – and 2004 – evolution of a couple of Greek hexameter laws, where some originally conditioning linguistic and metrical factors are later removed, but their effect is continued as what has largely become a traditional poetical device), since its inception its main purpose has been targeting non-lyrical poetry, where the métrique verbale prevails, thus minimizing the risk of other interfering factors like music, dance and even strongly conflicting metrical interpretations, falling well beyond the capabilities of a mostly formal automated analysis based on the sole text. Even if the system prosodical capabilities are generalized and efficient enough to be applied to a number of different problems, also outside poetry (e.g. the study of rhytmic *clausolae* in prose), or might be useful in collecting some raw data about the lyric structures themselves, it has rather been developed to target very large corpora spanning centuries of poetical production with non-lyrical metres recurring in simple structures or katà stíkhon; here the availability of such detailed data can prove useful in analyzing the surfacing phenomena and the interaction of their producing factors, especially in different chronological stages of the evolution of the Classical poetry, or between different poetical or linguistical (Greek and Latin) traditions. Also, developing a software system for this purpose not only forces us to give a very formalized account of our analysis method, but also provides a new set of interactive research tools and true metrical databases, whose scope may well fall beyond their original purposes (cf. e.g. the usage of the same software components by automatic inflection systems or specialized digital editions).

Roma, Università La Sapienza Daniele Fusi

REFERENCES

(The references are strictly limited to the works expressely quoted in this paper)

Bulloch A.W., *A Callimachean Refinement to the Greek Hexameter*, CQ n.s. 20, 1970, 258-268.

Cantilena M., *Il ponte di Nicanore*, in Fantuzzi M., Pretagostini R. (cur.), *Struttura e storia dell'esametro greco*, 9-67, Roma 1995.

Dover, K.J., *Greek Word Order*, Cambridge 1960.

Fruyt M., *Le mot: aperçu théorique et terminologique. Peut-on donner une définition théorique du mot? La polysémie de mot et ses consequences*, Lalies 10
(1989) 113-124, 1989.

Fusi D., *Appunti sulla prosodia del Lussorio di Shackleton-Bailey: alcune questioni di metodo*, in Bertini F. (cur.), *Luxoriana*, Genova 2002, 193-313.

Fusi D., *Fra metrica e linguistica: per la contestualizzazione di alcune leggi esametriche*, in Di Lorenzo E. (cur.), *L'esametro greco e latino: analisi, problemi e prospettive, Atti del convegno di Fisciano* 28-29 maggio 2002, 33-63, Napoli 2004.

Fusi D., *Edizione epigrafica digitale di testi greci e latini: dal testo marcato alla banca dati*, in Ciula A., Stella F. (cur.), *Digital Philology and Medieval Texts*, Pisa 2007, 121-163.

Klavans A., *The Independence of Syntax and Phonology in Cliticization*, Language 61 (1985) 95-120, 1985.

Lallot J., *Le mot, dans la tradition prégrammaticale et grammaticale en Grèce*, Lalies 10 (1989) 125-134, 1989.

Maas P., *Griechische Metrik*, Leipzig/Berlin 1923; Id., *Greek Metre*, translated by Hugh Lloyd-Jones, Oxford 1972.

Monteil P., *La phrase relative en grec ancien. Sa formation, son développement, sa structure, des origines à la fin du Ve siècle A. C.*, Paris 1963.

O'Neil E.G., *The localization of metrical word types in the Greek hexameter*, YCIS 8 (1942) 105-178.

Rossi L.E., *Anceps: vocale, sillaba, elemento*, RFIC 91 (1963), 52-71.

Abstract. This paper presents some aspects of a computer expert system created to perform linguistical and metrical analysis of Greek and Latin texts keeping into account such theorical problems, and also acting as a subsystem for another project offering a transformational model for automatic inflection of Greek and Latin languages through all the reconstructed historical stages. The metrical system can generate truly interactive metrical "editions" which can be queried by scholars for any combination of observed prosodical, syntactical and metrical phenomena, and several outputs for different uses, with special attention to its integration in other projects like digital editions.

Classical texts, Metrical Analysis, digital editions

MANUZIO: AN OBJECT LANGUAGE FOR ANNOTATED TEXT COLLECTIONS

1. *Introduction*

The traditional way of representing textual information for automatic processing is through some kind of enrichment of the base text with other, distinguished, text carrying some information, like metadata, formatting instructions, etc., or by exposing the structure of the text by marking its components, like chapters, verses, etc. This approach, in which the distinguished text is called the *markup*, has been widely diffused also by the availability of standard markup languages, like SGML and, in the recent years, XML, which made possible the definition of standards specific for literary texts, like the TEI system based on the representation of text through abstract structures.

The great advantages of a marked text is that it can be read and written with relative ease by a human being, as well as efficiently processed with a computer program. Moreover, when the marking of the text follows some widely accepted standard, it can be exchanged among different systems, processed by different applications, and, in general, used in a robust, interoperable way. Finally, the use of an extendible markup language, like XML, allows any kind of information to be added to the text in a string-encoded format.

These advantages are, however, balanced by several, noteworthy, shortcomings, both on the power and expressiveness of the representation and on the way in which computation can be carried over it. A first severe limitation is that marking can be applied only to contiguous segments of text that cannot overlap. Moreover, a text can be structured only in a strictly hierarchical fashion. Solutions exist to overcome some of these limitations, like the ones surveyed in (DeRose, 2004), but they tend to be cumbersome, to produce complex unreadable texts, and to notably increase the complexity of programs dealing with such texts. We could summarize these critics by saying the traditional markup approach is not *scalable:* it is a simple and elegant solution for simple text annotations, but it is not adequate to deal with very complex situations, where annotations are grams for processing marked text are not easily written, requiring the mastering of complex query languages, not specialized for the particular domain, like those typical of XML (for instance, XPath, XQuery, XSLT). In particular, they are not easily grasped by scholars and researchers in the humanities, which, on the contrary, should have the possibility to write queries or even programs over such kind of data. This problem becomes particularly serious when one has the objective of developing complex text analysis applications, like for instance those in the field of text mining, or applications which perform sophisticated syntactic or semantic analysis.

For all the above reasons, in this paper we propose a radically departure from the traditional markup approach. Such approach, already successfully applied to represent, in a computer, knowledge and information in fields different from that of text processing, is known in the software engineering area as *object-oriented modeling.*

The objectives that we are trying to achieve through this approach are the following:

- to represent collections of texts with any kind of structure, including different overlapping structures for the same text;
- to represent any kind of annotations, even with complex information, on any part of the text, taking into account whatever text structure we are interested in;
- to provide a simple way to make queries, even sophisticated ones, on text and annotations;
- to provide tools to simplify the construction of complex, efficient, textual analysis programs;
- to lay out the theoretical and practical foundation of a general system to deal with multi-user annotated text corpora, or digital library.

Solutions which are not markup-oriented have been already presented in the literature. For instance (Coombs et al., 1987; DeRose et al., 1997) present a model where text is seen as one or more hierarchies of objects that is the foundation of more complex systems like those presented in (Carletta et al., 2003; Petersen, 2002; Deerwester et al., 1992).

The approach that we propose presents a few similarities with those described in these papers, but it aims to provide a more complete solution. On the one hand the Manuzio model is easily scalable, as the structure of each textual collection can be defined ad hoc. On the other hand Manuzio provides a full programming and query language along with the model; such a language has been built to be expressive and easy to use in its specific domain of application. Finally, the Manuzio system is aimed to allow to store the data in a persistent database, to annotate it in a multi-user way, and to share results effortlessly.

The rest of the paper is organized as follows: in section 2 the foundations of the Manuzio data model are presented. In section 3 we have a look at the major features of the Manuzio language and finally, in section 4 an overview of the full system is given.

2. *The Manuzio Model*

We consider the textual information in a dual way: as a formatted sequence of characters, as well as a composition of logical structures called textual objects. This latter structural aspect has many similarities with other computer science models called *object-oriented data models*, which are based on abstraction mechanisms to represent a certain reality of interest (Nierstrasz, 1989).

The Manuzio model is characterized by the notions of *textual object, composition* and *repetition* of textual objects, *attributes* of textual objects, textual objects and attribute *types*, *inheritance definition* and *specialization* among types, *underlying normalized text.* To make the presentation simpler to understand, we introduce these concepts through a graphical notation, while the language constructs for the complete model specification will be presented in the next section.

2.1 *Textual Objects*

Definition 2.1 A *textual object* is a software entity with an identity, a state and a behavior. The identity defines the precise portion of the text underlying the object. The state is constituted by a set of *properties* which are either component textual objects or attributes that can assume values of arbitrary complexity. The behavior is constituted by a collection of local procedures, eventually with parameters, called *methods*, which define computed properties or perform operations on the object.

For instance Fig. 1 shows the structural aspects of a small set of textual objects. Each box represents a textual object and encloses its underlying text. If a box A is contained in another box B, then the textual object corresponding to A is a component of the object corresponding to B.

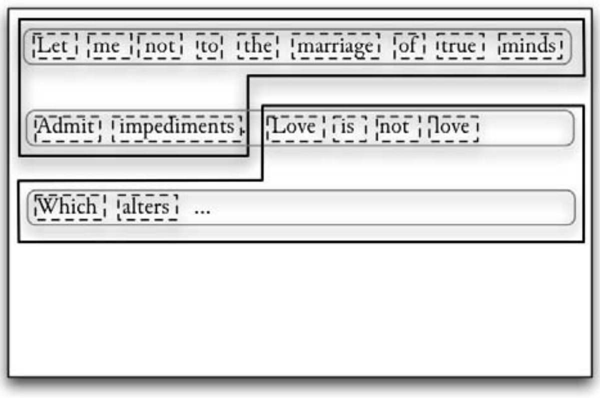


Figure 1: Example of Textual Object

Attributes

As previously stated, the properties of a textual object are either other objects, called components, or attributes. The intended use of attributes is to complement textual objects with annotations, metadata, variants, and in general any other type of information of interest.

Definition 2.2 An attribute is a value of any complexity which is a property of a textual object.

An attribute has a type which can be one of the common data types present in many programming and database languages, like integers, strings, booleans, arrays, records, etc. In Fig. 2 the first line is associated with an attribute that represents its meter, while the word “marriage” has another attribute which contains a comment about it.

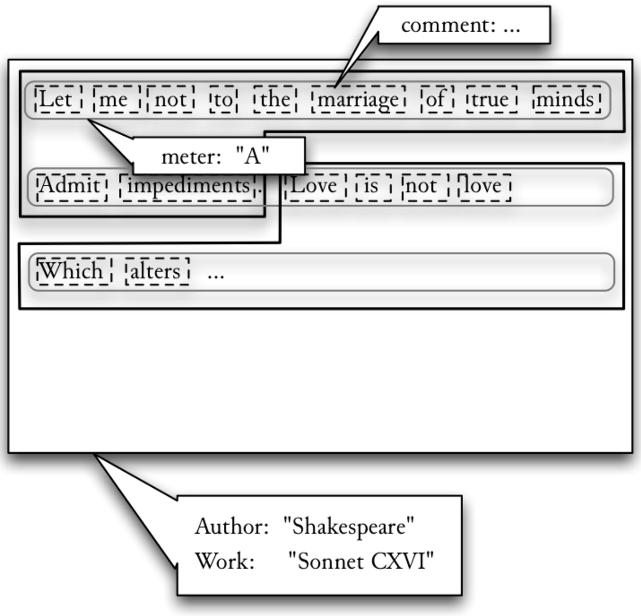


Figure 2 : Example of Textual Object with Attributes

Types

In computer science a *type* defines a set of possible values and the operations which can be applied to them. In the construction of our model, the types arise from the abstraction process of grouping different parts of text that share similar characteristics. Then the differences among the elements of those groups are ignored in order to put in evidence their similarities, i.e. their structure. For instance, the first two verses of the above example are considered (classified) as textual objects of type Line in order to stress the fact that they all have the same kind of properties (words, meter, etc.) and share the same behavior.

Every textual object is an instance of a *textual object type*. Each type has an associated interface that defines the ways that type's instances can be used to access their properties and methods.

Definition 2.3 A *textual object type interface* specifies the names and types of the properties and the names and the parameter and result types of the methods.

In Fig. 3 we show the types corresponding to the different kinds of boxes.

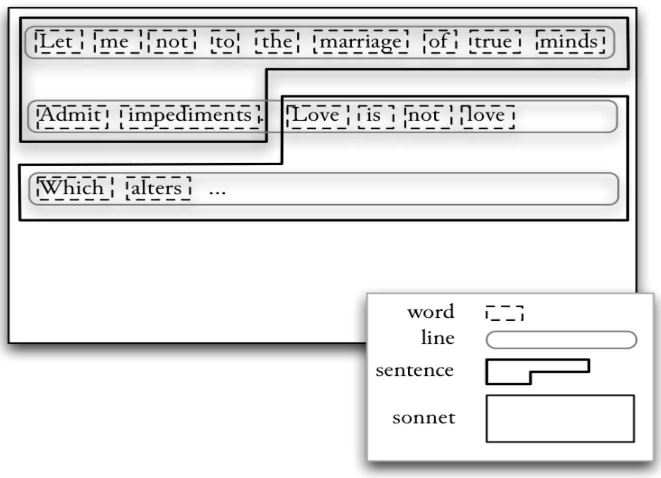


Figure 3: Example of Textual Object Types

Graphical Representation

As previously mentioned, we will use a graphical notation to put in evidence the structure of a textual model in terms of its textual objects types, their attributes and the component relations among them. In our notation an object type and its attributes are represented by a rectangle split in two parts. The upper part contains the name of the type, while the lower one, if present, contains the name and the data type of its attributes.

To put in evidence the fact that if a textual object is component of another then there is a *component relation* between their types, this relation is graphically represented through an arrow which connects the two types, labelled with the component name. And, since this relation can be one to one (for instance each poem has only a title which is a sentence) or one to many (for instance each poem has many lines) we distinguish this fact with a different graphical notation. The former case is represented by a single-pointed arrow, the latter by a double pointed arrow. For instance, in Fig. 4, we represent a very simple model about poems which arises from the previous examples.

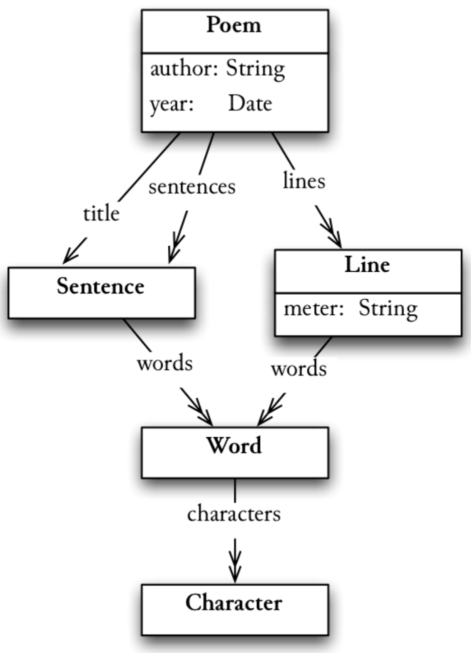


Figure 4: A simple model about Poems

In Manuzio, the textual object type which has no component is called Unit. A Unit type is always present, must be unique, and defines the minimal part of text which can be manipulated. For instance, in Fig. 4, the Unit type is Character. Different textual models can have different Unit types, depending on the granularity of the textual analysis in which the user is interested to. For instance, one could be interested in lemmas, or in syllables, instead of characters for different kind of analysis, or in written representation of phonemes to develop phonetic analysis programs, etc.

2.2 *Type Inheritance*

Another important information that can be modelled in Manuzio is that textual objects types are not always independent, but can exist a particular relation among them, called *specialization*, through which we can model objects at different levels of detail. If a type A is defined as specialization of type B, then the instances of A inherits all the characteristics of the instances of B, in addition to having other, proper ones. For example, an hendecasyllable has all the characteristics of a line (it *is* in effect a line), but it has also the property of having exactly eleven syllables. The presence of the specialization between two textual object types A (which will be called the *subtype)* and B (the *supertype)* has the effect that every instance of the subtype (for example every hendecasyllable), can be treated both as a generic line (for instance when performing a textual search), and as a line with specific number of syllables (for instance to define specific methods which are significant only for hendecasyllables).

Definition 2.4 A type A is *subtype* of a type B if it is defined as such; in this case A inherits all the properties and the behavior of B. A can also have new properties and methods, and can redefine the type of its components with a more specialized type.

Graphical Representation

A subtype is graphically connected to its supertype through an arrow with a hollow arrowhead, and shows only the new information (with respect to its supertype). For this reason the lower part of the rectangle contains only the new attributes, while only the arrows representing the new components are drawn. In Fig. 5 an example about simple works is shown.

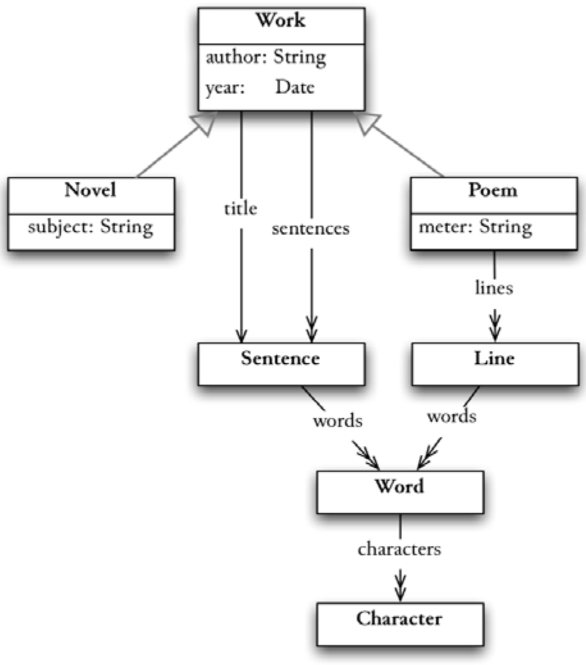


Figure 5: A model about poems and novels

In fig. 5, both Novel and Poem are subtypes of Work so that they inherit the components title and sentences, as well as the attributes year and author. Moreover, the Novel type has the new attribute subject, while the Poem type has the new attribute meter and a component lines which allows to model lines of a poem.

2.3 *Underlying Text*

Each textual object has a direct correspondence to a part of the text to be represented in our model (for instance a single work, a corpora, a library, etc.). We assume that such texts exist, as sequences of Unicode characters, in a format which is chosen by some expert. The following definitions specified the exact relation between texts and objects of the model.

Definition 2.5 The *full text* is a sequence of Unicode characters that represents all the text described by a specific Manuzio model.

Each textual object has an underlying text, defined as:

Definition 2.6 The *underlying text* of a textual object is a subsequence of the characters of the model's *full text* associated to it.

As previously mentioned, the identity of a textual object is determined by the underlying text: two objects are identical if they have the same underlying text. Moreover, this concept is fundamental also in defining an important semantics property of the Manuzio model:

Definition 2.7 A textual model is *well-formed* if each component of that textual object has an underlying text which is a proper subsequence of its underlying text.

Finally, the underlying text will be useful in the system to provide the user a concrete representation of a textual object.

While the Manuzio model has been presented so far through a graphical notation, a formal language is necessary in order to implement a computer system for representing textual models according to our approach and to operate on them. In the following section the main features of such a language are introduced.

3. *The Manuzio Language*

The Manuzio language is a full programming language with construct to define textual models and write complex query expressions or sophisticated textual analysis applications. Such a language is intended to be used by a multi-user system to store persistently a digital collection of texts over which these programs are evaluated.

The full syntax and semantics of the language will be available in the forthcoming manual and are beyond the scope of this article. In particular, the language features aimed to describe the textual object types of a model will be only shown through the following example (Fig. 6):

```
Schema PlaySchema

type Play
has
title : Sentence
acts : Acts
attribute
author : String
year   : Integer
end

type Act
has
scenes: Scenes
attribute
directions : String
end

type Scene
has
speeches : Speeches
end

type Prologue is Scene end

type Epilogue is Scene

(continue to the next column)

has
salutations : Speech
end

type Speech

has
sentences: Sentences
lines : Lines
attribute
speaker : String
end

type Line
has words : Words
end

type Sentence
has words : Words
end

type Word
has characters : Characters
end

End
```

Figure 6: Schema definition example.

The model described in fig. 6 concerns simple plays, where a Play[1] is composed by a title and some acts, and has attributes author and publication year. The Speech type has two components, lines and sentences, which are independent ways of considering a speech. The types Prologue and Epilogue inherits their properties from the supertype Scene. While Prologue does not have any additional property, the Epilogue type adds a salutations component. The type Word has components of the basic predefined type Character, which represents a unicode character.

3.1 *Repeated Textual Objects*

In Fig. 6, types like Words or Lines, which are not explicitly defined, are used. In fact, when we define a textual object type, there is an implicit definition of its "plural form", which is a type whose instances, called *repeated textual objects*, contains repetitions of objects of the "singular form" type. For example, an object of type Words is composed by a repetition of instances of type Word.
The existence of these types in the language has the following consequences:

- A repeated textual object is in effect a textual object, can have properties or methods, and in general can be treated as any other textual object.
- Specific operators exist which take into account the multiplicity of the elements of a repeated textual object. For instance, we can count how many words are present in an object of type Words, we can select the first word, and so on.
- The query language operators on single textual objects can be applied also to their repetitions, with the meaning that the operator is applied to all the repetition's elements and returns the collection of the results. For instance, the operation that returns the title of a single poem, when applied to a collection of poems, returns all their titles.

We are ready to present now the operational features of the language, but the reader must keep in mind that, while Manuzio is a full programming language, like java or prolog, which allows experienced users to write programs of arbitrary complexity, in the rest of this section we will discuss only its query-like operators. These operators allow non-programmer to work with the system in an easy way to fetch, refine,

[1] By convention, a type name is capitalized.

display and annotate query results, in a manner similar to other data query languages, like, for instance, the relational databases language SQL.

3.2 *Basic Textual Objects Access*

A uniform notation is used to select both properties and methods of a textual object. Such a selection is performed through the access operator *of*. For instance, if P is a poem then:

*title **of** P*

returns the textual object of type Sentence which is the component title of the poem P, while

author of P

returns instead the value of the attribute author. When the result of the 'of' operator is a textual object, the operation can be repeated:

words of title of P

returns the textual object of type Words which contains the words of the title of P. As previously mentioned, when a component access is applied to a repeated textual object, the result is again a repeated textual object, as in:

words of lines of P

which returns a textual object containing *all* the words of *all* the lines of P.

As stated in the definition 2.4 in the Section 2, for the inheritance property of the subtyping mechanism, the *of* operator for a certain type can be applied also the instances of its subtypes. For instance, since Epilogue is a subtype of Scene, we can select the speeches of a epilogue E (since it is also a scene), in the same manner as its salutations:

speeches of E

Analogously, if there were methods defined on Scene, they could be applied also to instances of Epilogue.

3.3 *Repeated Objects Operators*

There exist operators specific to repeated textual objects which take into account the elements of those objects by performing some operations on all, or a subset, of them.

A first group of operators can be used to get a part of a repetition by specifying the elements in which we are interested either by using ordinal adjectives or numeric ranges.

first *line* ***of*** *P*
words(1..3) ***of second*** *line* ***of*** *P*
last *sentence* ***of*** *P*

It is also possible to count the elements of a repetition, like in:

number ***of*** *lines* ***of*** *P*

as well as use other operations on sequences, like concatenation of sequences, test for a condition holding on some or all the elements, test for inclusion of an element, etc., which are not described here since their are typical for data structures like sequences or arrays present in other languages.
The most important operator of this category, which is the foundation of the query-like part of the Manuzio language, is the 'select' operator. As suggested by the name, it has a syntax similar to SQL selection, and can be used to retrieve objects through conditional expressions. Here, it will be described through examples, in which we assume C a collection of poems.
The first example shows the simple form "select E1 from Id in E2", where E2 is an expression returning a repeated textual object whose elements are bound, in order, to the identifier Id, used in the evaluation of the expression E1. The result is the collection of such values.

select *title* ***of*** *p*
from *p* **in** *poems* ***of*** *C*

The first example returns a repeated textual object composed by the sentences formed by all titles of the poems of the collection C. The type of the resulting object is Sentences, and, in this simple form, it is equivalent to the expression:

title ***of*** *poems* ***of*** *C*

A select expression can have a ‘where’ clause, which can be used to give a condition to filter the elements over which the construct iterates. For instance, the following example returns only the titles of Shakespeare’s poems.

select *title* ***of*** *p*
from *p* ***in*** *poems* ***of*** *C*
where *author* ***of*** *p = “Shakespeare”*

The last example shows how to use this construct to build complex queries. For instance, to find all the lines of Shakespeare’s poems with exactly five words, we could write:

select *l*
from *l* ***in (select*** *lines* ***of*** *p*
from *p* ***in*** *poems* ***of*** *C*
where *author* ***of*** *p = “Shakespeare”)*
where *number* ***of*** *words* ***of*** *l = 5*

The internal ‘select’ returns a repeated textual object, over which the external one iterates.

3.4 *Access to the underlying text*

Given a textual object, we can select its underlying text with ‘text’. For instance, given a word W, the following expression:

text ***of*** *W*

returns a string that contains the unicode characters of the word W.
It is also possible to access the starting position of the underlying text of an object:

text_position ***of*** *W*

which returns a number which specify the offset position of the underlying text of W from the start of the full text.
Note that, since the language has a complete set of operators on regular data types, once we get the text of a textual object we can apply to it all the operators available on character strings (which includes, among others, pattern matching through regular expressions).

3.5 *Comparisons and test operators*

In the language, the usual comparison operators on strings and other simple values (=, >, <, etc.) are available.

On the other hand, two textual objects can be tested for identity (i.e. if they are in effect the same object) with the identity operator '==':

o1 == o2

When applied to textual objects, the string equality operator is an abbreviation for testing the equality of their underlying text, like in:

o1 = o2

which is equivalent to:

text ***of*** *o1 = text* ***of*** *o2*

Also the other comparison operators, when applied to textual objects, takes into account their underlying text. For instance:

o1 < o2

is an abbreviation for:

text_position ***of*** *o1 < text_position* ***of*** *o2*

that returns true when the object o1 precedes o2 in the full text (analogously for <, >=, <=).
In addition, the operator '><' returns true when an object overlaps another, so that, for instance, we can test if a sentence and a line overlap.

Finally, the language has a few other operators to test about the relative positions of two textual objects, known as the Allen relations (Allen, 1983). They allow, for instance, to know if an object is fully contained in another one, if one partially precedes another, and so on.

4. *An overview of the Manuzio System*

The purpose of Manuzio is to be a system, based on the presented language, with the capabilities of storing in a persistent way complex annotated text collections and allowing their manipulation by different users in a coherent and cooperative way. To reach such a goal the Manuzio language has other features, in addition to those shown in section 3, to deal with users permissions, dynamic annotations, and management of the model's persistency.

While the full system architecture is the subject of a forthcoming paper, here we will present an overview of its capabilities.

1. The system provides an efficient way of storing, in a persistent way, and querying very large quantities of textual material, together with annotations. To achieve this objective we are investigating different solutions including those based on relational database technology.
2. The system has as its main programming and administration language the Manuzio language. For this reason, the language has also constructs to extend the model's schema with new types, to extend a type with new attributes and methods, to make persistent textual objects retrieved by a query, and to add and modify annotations on them.
3. The system has tools that allow the access to concurrent users, through an appropriate set of permissions. For instance, different groups of users can work with different sets of annotations, that can later be compared and finally merged.
4. Users can interact with the system either through the Manuzio language or through a friendly graphical interface to perform assisted queries whose results are visualized with a choice of different graphical formats and mediums.
5. To exchange texts and annotations with other systems the XML standard format can be used through a set of tools which facilitates the mapping between it and the Manuzio internal format. In particular, XML is the privileged way of loading the data into the textual database, an operation which is done by a parsing process that can be automatic, semiautomatic or manual, depending on the complexity of the source data.

The system is currently under development, but the Manuzio model capabilities and a subset of the language's features have been tested using a simple prototype built with the Ruby programming language. The prototype has a fixed scheme of medium complexity concerning epic Latin poems, and has been used to successfully performs clause-related analysis on a medium-sized corpora.

5. *Conclusions and future work*

This paper is an introduction to a novel approach for dealing with annotated collection of texts. We have presented the Manuzio model, which has some similarities with other object-oriented models, although it is specialized for the specific domain. Our proposal includes also a full programming language, the Manuzio language, of which only the query features have been discussed. Finally, a sketch of a complete solution, consisting of a system based on that language, has been presented. While Manuzio is still a work in progress, our first experiments have shown the feasibility of the approach in dealing with collection of literary texts.

The next step in Manuzio development will be the complete language specification and implementation, along with a full functional architecture of the system.

Acknowledgment
This work has been supported in part by grants of the Italian Ministero dell'Istruzione, Università e Ricerca Scientifica, under the PRIN Project *Musisque deoque II. Un archivio digitale dinamico di poesia latina, dalle origini al Rinascimento italiano* about a digital archive of Latin poetry. We are indebted to Paolo Mastandrea and Luigi Tessarolo, who have stimulated our interest in this subject and provided useful suggestions and challenging examples through discussions about their previous work.

Università Ca' Foscari, Venezia Marek Maurizio e Renzo Orsini

REFERENCES

C. Sperberg-McQueen, L. Burnard, *Guidelines for electronic text encoding and interchange*, Oxford 2002.

S.J. DeRose, *Markup overlap: A review and a horse*, in *Extreme Markup Languages. A Conference of IdeAlliance,* Montreal-Quebec 2004.

J.H. Coombs, A.H. Renear, S.J. DeRose, *Markup systems and the future of scholarly text processing.*, Communications of ACM 30(11)**,** 1987, 933-947.

S. DeRose, D. Durand, E. Mylonas, A. Renear, *What is text, really?* ACM SIGDOC Asterisk Journal of Computer Documentation 21(3), 1997, 1-24.

J. Carletta, S. Evert, U. Heid, J. Kilgour, J. Robertson, H. Voormann, *The NITE XML Toolkit: flexible annotation for multi-modal language data*, Behavior Research Methods, Instruments, and Computers 35 (3), 2003, 353-363.

U. Petersen, *The standard MDF model,* 2002, unpublished article; obtainable from URL: http://emdros.org.

S.C. Deerwester, K. Waclena, M. LaMar*, A textual object management system*, in N.J. Belkin, P. Ingwersen, A.M. Pejtersen eds. SIGIR, ACM, 1992, 126-139.

P. Mastandrea e L. Tessarolo, *De fine versus. Repertorio di clausole ricorrenti nella poesia dattilica latina dalle origini a Sidonio Apollinare,* Hidesheim 1992.

J.F. Allen, *Maintaining knowledge about temporal intervals*, Communications of the ACM 26 (11)**,** 1983, 3-22.

Abstract. Traditionally, textual collections are digitally represented as a set of files containing the text along with some kind of markup to define extra information, like metadata, annotations, etc. We propose a different approach which exploits the natural structure of a text to build specialized abstractions, called textual objects, over literary text’s collections. These objects can be used to make non-hierarchically nested multi-level annotations, to create complex metadata, and to perform complex queries and analysis on the collection. Manuzio, the result of this approach, consists of a model, a language and a system to manage persistent text’s collection and write complex applications over them. In this paper we introduce the main features of the Manuzio model and language, as well as a sketch of the system.

Object-oriented language, document database system, text-analysis