# Outilex, a Linguistic Platform for Text Processing

**Olivier Blanc**
IGM, University of Marne-la-Vallée
5, bd Descartes - Champs/Marne
77454 Marne-la-Vallée, France
oblanc@univ-mlv.fr

**Matthieu Constant**
IGM, University of Marne-la-Vallée
5, bd Descartes - Champs/Marne
77 454 Marne-la-Vallée, france
mconstan@univ-mlv.fr

## Abstract

We present Outilex, a generalist linguistic platform for text processing. The platform includes several modules implementing the main operations for text processing and is designed to use large-coverage Language Resources. These resources (dictionaries, grammars, annotated texts) are formatted into XML, in accordance with current standards. Evaluations on efficiency are given.

## 1 Credits

This project has been supported by the French Ministry of Industry and the CNRS. Thanks to Sky and Francesca Sigal for their linguistic expertise.

## 2 Introduction

The Outilex Project (Blanc et al., 2006) aims to develop an open-linguistic platform, including tools, electronic dictionaries and grammars, dedicated to text processing. It is the result of the collaboration of ten French partners, composed of 4 universities and 6 industrial organizations. The project started in 2002 and will end in 2006. The platform which will be made freely available to research, development and industry in April 2007, comprises software components implementing all the fundamental operations of written text processing: text segmentation, morphosyntactic tagging, parsing with grammars and language resource management.

All Language Resources are structured in XML formats, as well as binary formats more adequate to efficient processing; the required format converters are included in the platform. The grammar formalism allows for the combination of statistical approaches with resource-based approaches. Manually constructed lexicons of substantial coverage for French and English, originating from the former LADL[1], will be distributed with the platform under LGPL-LR[2] license.

The platform aims to be a generalist base for diverse processings on text corpora. Furthermore, it uses portable formats and format converters that would allow for combining several software components. There exist a lot of platforms dedicated to NLP, but none are fully satisfactory for various reasons. Intex (Silberztein, 1993), FSM (Mohri et al., 1998) and Xelda[3] are closed source. Unitex (Paumier, 2003), inspired by Intex has its source code under LGPL license[4] but it does not support standard formats for Language Resources (LR). Systems like NLTK (Loper and Bird, 2002) and Gate (Cunningham, 2002) do not offer functionality for Lexical Resource Management.

All the operations described below are implemented in C++ independent modules which interact with each others through XML streams. Each functionality is accessible by programmers through a specified API and by end users through binary programs. Programs can be invoked by a Graphical User Interface implemented in Java. This interface allows the user to define his own processing flow as well as to work on several projects with specific texts, dictionaries and grammars.

[1] French Laboratory for Linguistics and Information Retrieval

[2] Lesser General Public License for Language Resources, http://infolingu.univ-mlv.fr/lgpllr.html.

[3] http://www.dcs.shef.ac.uk/ hamish/dalr/baslow/xelda.pdf.

[4] Lesser General Public License, http://www.gnu.org/copyleft/lesser.html.

## 3 Text segmentation

The segmentation module takes raw texts or HTML documents as input. It outputs a text segmented into paragraphs, sentences and tokens in an XML format. The HTML tags are kept enclosed in XML elements, which distinguishes them from actual textual data. It is therefore possible to rebuild at any point the original document or a modified version with its original layout. Rules of segmentation in tokens and sentences are based on the categorization of characters defined by the Unicode norm. Each token is associated with information such as its type (word, number, punctuation, ...), its alphabet (Latin, Greek), its case (lowercase word, capitalized word, ...), and other information for the other symbols (opening or closing punctuation symbol, ...). When applied to a corpus of journalistic telegrams of 352,464 tokens, our tokenizer processes 22,185 words per second[5].

## 4 Morphosyntactic tagging

By using lexicons and grammars, our platform includes the notion of multiword units, and allows for the handling of several types of morphosyntactic ambiguities. Usually, stochastic morphosyntactic taggers (Schmid, 1994; Brill, 1995) do not handle well such notions. However, the use of lexicons by companies working in the domain has much developed over the past few years. That is why Outilex provides a complete set of software components handling operations on lexicons. IGM also contributed to this project by freely distributing a large amount of the LADL lexicons[6] with fine-grained tagsets[7]: for French, 109,912 simple lemmas and 86,337 compound lemmas; for English, 166,150 simple lemmas and 13,361 compound lemmas. These resources are available under LGPL-LR license. Outilex programs are compatible with all European languages using inflection by suffix. Extensions will be necessary for the other types of languages.

Our morphosyntactic tagger takes a segmented text as an input ; each form (simple or compound) is assigned a set of possible tags, extracted from indexed lexicons (cf. section 6). Several lexicons can be applied at the same time. A system of priority allows for the blocking of analyses extracted from lexicons with low priority if the considered form is also present in a lexicon with a higher priority. Therefore, we provide by default a general lexicon proposing a large set of analyses for standard language. The user can, for a specific application, enrich it by means of complementary lexicons and/or filter it with a specialized lexicon for his/her domain. The dictionary look-up can be parameterized to ignore case and diacritics, which can assist the tagger to adapt to the type of processed text (academic papers, web pages, emails, ...). Applied to a corpus of AFP journalistic telegrams with the above mentioned dictionaries, Outilex tags about 6,650 words per second[8].

The result of this operation is an acyclic automaton (sometimes, called word lattice in this context), that represents segmentation and tagging ambiguities. This tagged text can be serialized in an XML format, compatible with the draft model MAF (Morphosyntactic Annotation Framework)(Clément and de la Clergerie, 2005).

All further processing described in the next section will be run on this automaton, possibly modifying it.

## 5 Text Parsing

Grammatical formalisms are very numerous in NLP. Outilex uses a minimal formalism: Recursive Transition Network (RTN)(Woods, 1970) that are represented in the form of recursive automata (automata that call other automata). The terminal symbols are lexical masks (Blanc and Dister, 2004), which are underspecified word tags i.e. that represent a set of tagged words matching with the specified features (e.g. noun in the plural). Transductions can be put in our RTNs. This can be used, for instance, to insert tags in texts and therefore formalize relations between identified segments.

This formalism allows for the construction of local grammars in the sense of (Gross, 1993). It has been successfully used in different types of applications: information extraction (Poibeau,

[5]This test and further tests have been carried out on a PC with a 2.8 GHz Intel Pentium Processor and a 512 Mb RAM.

[6]http://infolingu.univ-mlv.fr/english/, follow links Linguistic data then Dictionnaries.

[7]For instance, for French, the tagset combines 13 part-of-speech tags, 18 morphological features and several syntactic and semantic features.

[8]4.7 % of the token occurrences were not found in the dictionary; This value falls to 0.4 % if we remove the capitalized occurrences.

The processing time could appear rather slow; but, this task involves not so trivial computations such as conversion between different charsets or approximated look-up using Unicode character properties.

2001; Nakamura, 2005), named entity localization (Krstev et al., 2005), grammatical structure identification (Mason, 2004; Danlos, 2005)). All of these experiments resulted in recall and precision rates equaling the state-of-the-art.

This formalism has been enhanced with weights that are assigned to the automata transitions. Thus, grammars can be integrated into hybrid systems using both statistical methods and methods based on linguistic resources. We call the obtained formalism Weighted Recursive Transition Network (WRTN). These grammars are constructed in the form of graphs with an editor and are saved in an XML format (Sastre, 2005).

Each graph (or automaton) is optimized with epsilon transition removal, determinization and minimization operations. It is also possible to transform a grammar in an equivalent or approximate finite state transducer, by copying the subgraphs into the main automaton. The result generally requires more memory space but can highly accelerate processing.

Our parser is based on Earley algorithm (Earley, 1970) that has been adapted to deal with WRTN (instead of context-free grammar) and a text in the form of an acyclic finite state automaton (instead of a word sequence). The result of the parsing consists of a shared forest of weighted syntactic trees for each sentence. The nodes of the trees are decorated by the possible outputs of the grammar. This shared forest can be processed to get different types of results, such as a list of concordances, an annotated text or a modified text automaton. By applying a noun phrase grammar (Paumier, 2003) on a corpus of AFP journalistic telegrams, our parser processed 12,466 words per second and found 39,468 occurrences.

The platform includes a concordancer that allows for listing in their occurring context different occurrences of the patterns described in the grammar. Concordances can be sorted according to the text order or lexicographic order. The concordancer is a valuable tool for linguists who are interested in finding the different uses of linguistic forms in corpora. It is also of great interest to improve grammars during their construction.

Also included is a module to apply a transducer on a text. It produces a text with the outputs of the grammar inserted in the text or with recognized segments replaced by the outputs. In the case of a weighted grammar, weights are criteria to select between several concurrent analyses. A criterion on the length of the recognized sequences can also be used.

For more complex processes, a variant of this functionality produces an automaton corresponding to the original text automaton with new transitions tagged with the grammar outputs. This process is easily iterable and can then be used for incremental recognition and annotation of longer and longer segments. It can also complete the morphosyntactic tagging for the recognition of semi-frozen lexical units, whose variations are too complex to be enumerated in dictionaries, but can be easily described in local grammars.

Also included is a deep syntactic parser based on unification grammars in the decorated WRTN formalism (Blanc and Constant, 2005). This formalism combines WRTN formalism with functional equations on feature structures. Therefore, complex syntactic phenomena, such as the extraction of a grammatical element or the resolution of some co-references, can be formalized. In addition, the result of the parsing is also a shared forest of syntactic trees. Each tree is associated with a feature structure where are represented grammatical relations between syntactical constituents that have been identified during parsing.

## 6 Linguistic Resource Management

The reuse of LRs requires flexibility: a lexicon or a grammar is not a static resource. The management of lexicons and grammars implies manual construction and maintenance of resources in a readable format, and compilation of these resources in an operational format. These techniques require strong collaborations between computer scientists and linguists; few systems provide such functionality (Xelda, Intex, Unitex). The Outilex platform provides a complete set of management tools for LRs. For instance, the platform offers an inflection module. This module takes a lexicon of lemmas with syntactic tags as input associated with inflection rules. It produces a lexicon of inflected words associated with morphosyntactic features. In order to accelerate word tagging, these lexicons are then indexed on their inflected forms by using a minimal finite state automaton representation (Revuz, 1991) that allows for both fast look-up procedure and dictionary compression.

## 7 Conclusion

The Outilex platform in its current version provides all fundamental operations for text processing: processing without lexicon, lexicon and grammar exploitation and LR management. Data are structured both in standard XML formats and in more compact ones. Format converters are included in the platform. The WRTN formalism allows for combining statistical methods with methods based on LRs. The development of the platform required expertise both in computer science and in linguistics. It took into account both needs in fundamental research and applications. In the future, we hope the platform will be extended to other languages and will be enriched with new functionality.

## References


Olivier Blanc and Matthieu Constant. 2005. Lexicalization of grammars with parameterized graphs. In *Proc. of RANLP 2005*, pages 117–121, Borovets, Bulgarie, September. INCOMA Ltd.

Olivier Blanc and Anne Dister. 2004. Automates lexicaux avec structure de traits. In *Actes de RECITAL*, pages 23–32.

Olivier Blanc, Matthieu Constant, and Éric Laporte. 2006. Outilex, plate-forme logicielle de traitements de textes écrits. In Cédrick Fairon and Piet Mertens, editors, *Actes de TALN 2006 (Traitement automatique des langues naturelles)*, page to appear, Leuven. ATALA.

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565.

Lionel Clément and Éric de la Clergerie. 2005. MAF: a morphosyntactic annotation framework. In *Proc. of the Language and Technology Conference, Poznan, Poland*, pages 90–94.

Hamish Cunningham. 2002. GATE, a general architecture for text engineering. *Computers and the Humanities*, 36:223–254.

Laurence Danlos. 2005. Automatic recognition of French expletive pronoun occurrences. In *Companion Volume of the International Joint Conference on Natural Language Processing, Jeju, Korea*, page 2013.

Jay Earley. 1970. An efficient context-free parsing algorithm. *Comm. ACM*, 13(2):94–102.

Maurice Gross. 1993. Local grammars and their representation by finite automata. In M. Hoey, editor, *Data, Description, Discourse, Papers on the English Language in honour of John McH Sinclair*, pages 26–38. Harper-Collins, London.

Cvetana Krstev, Duško Vitas, Denis Maurel, and Mickaël Tran. 2005. Multilingual ontology of proper names. In *Proc. of the Language and Technology Conference, Poznan, Poland*, pages 116–119.

Edward Loper and Steven Bird. 2002. NLTK: the natural language toolkit. In *Proc. of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics, Philadelphia*.

Oliver Mason. 2004. Automatic processing of local grammar patterns. In *Proc. of the 7th Annual CLUK (the UK special-interest group for computational linguistics) Research Colloquium*.

Mehryar Mohri, Fernando Pereira, and Michael Riley. 1998. A rational design for a weighted finite-state transducer library. *Lecture Notes in Computer Science*, 1436.

Takuya Nakamura. 2005. Analysing texts in a specific domain with local grammars: The case of stock exchange market reports. In *Linguistic Informatics - State of the Art and the Future*, pages 76–98. Benjamins, Amsterdam/Philadelphia.

Sébastien Paumier. 2003. *De la reconnaissance de formes linguistiques à l'analyse syntaxique. Volume 2, Manuel d'Unitex*. Ph.D. thesis, IGM, Université de Marne-la-Vallée.

Thierry Poibeau. 2001. Extraction d'information dans les bases de données textuelles en génomique au moyen de transducteurs à états finis. In Denis Maurel, editor, *Actes de TALN 2001 (Traitement automatique des langues naturelles)*, pages 295–304, Tours, July. ATALA, Université de Tours.

Dominique Revuz. 1991. *Dictionnaires et lexiques: méthodes et algorithmes*. Ph.D. thesis, Université Paris 7.

Javier M. Sastre. 2005. XML-based representation formats of local grammars for NLP. In *Proc. of the Language and Technology Conference, Poznan, Poland*, pages 314–317.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. *Proceedings of the International Conference on New Methods in Language Processing*.

Max Silberztein. 1993. *Dictionnaires électroniques et analyse automatique de textes. Le système INTEX*. Masson, Paris. 234 p.

William A. Woods. 1970. Transition network grammars for natural language analysis. *Communications of the ACM*, 13(10):591–606.