Computational Environments for Grammar Development and Linguistic Engineering

Proceedings of a Workshop
Sponsored by the
Association for Computational Linguistics

Edited by
Dominique Estival, Alberto Lavelli, Klaus Netter
and
Fabio Pianesi

12 July 1997
Universidad Nacional de Educación a Distancia
Madrid, Spain

Computational Environments for Grammar Development and Linguistic Engineering

Proceedings of a Workshop
Sponsored by the
Association for Computational Linguistics

Edited by
Dominique Estival, Alberto Lavelli, Klaus Netter
and
Fabio Pianesi

12 July 1997
Universidad Nacional de Educación a Distancia
Madrid, Spain

Order additional copies from:

ACL
P.O. Box 6090
Somerset, NJ, 08875 USA
+1-908-873-3898
acl@bellcore.com

Preface

With a growing number of NLP applications going beyond the status of simple research systems, there is also a more evident need for better methods, tools and environments to support the development and reuse of large scale linguistic resources and efficient processors. This new area of research, often referred to as Linguistic Engineering, is rapidly gaining interest besides the more traditional ones concerned with formalisms or algorithm studies and development.

Aspects of linguistic engineering range from grammar development environments, through the construction and maintenance of large scale linguistic resources, to methodologies for quality assurance and evaluation. Some of the most prominent examples of sophisticated development platforms comprising tracer, debugger and all kinds of highly important visualization tools are ALEP (funded by the European Union), GATE (common infrastructure for building LE architectures using pre-existing components), GWB (LFG-workbench developed at Xerox Parc) PAGE (a grammar development environment developed at DFKI), and others, many of which are documented also in this volume. There have been a number of projects on the development of large-scale computational lexicons (e.g. Acquilex), as well as projects concerned with the development of standards and reference data for diagnostics and evaluation (e.g. TSNLP).

However, while these platforms and components typically provide fairly clean formalisms, processing components and data, it is not yet clear to which extent current results and approaches fit the requirements for scale development and deployment of real NLP applications.

In this connection, a number of pending issues need be addressed, the relevance of which becomes particularly clear when the focus is shifted from linguistic formalism to usability and user/application requirements. The purpose of the workshop documented in this volume was to shed some light on some of these questions, and represent the state of the art in grammar development from some quite different perspectives. The issues that were sketched in the call for papers comprised the following questions:

- What is the state of the art in Grammar Development Environments (GDEs)?

  There are a number of systems on the market already. Given the enormous cost of developing such environments, it is unlikely that many others will be developed from scratch. Up to what point do the existing systems meet actual user requirements? What experiences are there in tailoring such systems to specific applications?

  The many medium to large scale GDEs which are illustrated in this volume provide an incomplete but nevertheless highly representative overview on the current state of the art. Most of them are GDEs based on major linguistic theories, such as the LFG-oriented Xerox Linguistic Environment (Kaplan and Newman) or an object-oriented GDE from LIMSI/CNRS (Vapillon et al.), the XTAG system from IRCS at UPenn (Doran et al.) or the HPSG-oriented ConTroll System from Tübingen (Götz and Meurers). Another group of GDEs are designed more or less as theory independent platforms, such as the aforementioned ALEP system (Theofilides et al.), the GEPPETTO Development Environment (Ciravegna et al.), the GTU environment which serves mainly as a tutorial system (Volk and Richarz), or the Hdrug system which consists of a broad range of visualization tools and processing components (van Noord and Bouma).

  The paper by Strömbäck describes EFLUF, an environment in which it is possible to experiment with unification formalisms. Bateman focuses on the different requirements imposed by NL analysis and generation on the GDEs. Also dealing with the parser/generator dichotomy is the paper by Penn and Popescu, which presents a new method for efficiently compiling grammars for head-driven generation in ALE.

- What is the appropriate division of labour in a large scale development environment?

  Sophisticated applications may require a whole range of expertise, comprising computational morphology, syntax, semantics, lexicography, corpus analysis, parsing and generation to name but a few. The development of linguistic knowledge bases in a manner which is completely detached from the development of processing techniques may form the basis of a clean division of labour, but often also leads to inefficient systems. What approaches and methods can be devised and which tools and facilities should be employed to facilitate and support the integration of different levels of linguistic abstraction on the one hand, and the cooperation between grammar writing and processing on the other.

  Some of these aspects are described in the paper by Ciravegna et al. on the GEPPETTO Development Environment and the methodology underlying the approach.

- How can we meet the demands arising from distributed and multi-lingual grammar development?

  Even if in the past the biggest systems have been based on the work of one individual, it is unwise and un-practical to have one large grammar be developed by single writers. Thus, the development and maintenance of large grammars tends to be more and more a joint effort involving many computational linguists. What specific requirements and prerequisites have to be met in a development environment to ensure a smooth cooperation between different authors leading to the necessary modularity, consistency and integrability of grammar fragments?

  For many applications (even outside machine translation proper) multi-linguality is becoming an indispensable standard feature. The parallel development of several grammars in different languages will require some synchronization of linguistic knowledge bases and sharing of processing components. Can different language specific grammars share a common core grammar?

  Both of these questions were addressed by the paper on ALEP-based distributed grammar development, which describes a methodology which also goes in the direction of sharing grammatical descriptions and representations across different languages.

- How can we facilitate the shift from re-usability to usability?

  Grammar development in academic and research oriented environments has often concentrated on the maximum generality and re-usability of the linguistic resources developed. However, for the purpose of building actual applications and applying systems to specific domains, this generality can turn out to be a drawback rather than an asset. Thus, the question is: how is it possible to support the specialization and customization to more constrained domains without sacrificing the advantages of a general and reusable design.

  One solution to this problem is the adaptation of broad coverage grammar to a specific domain, which is described for example in the paper by Henschel and Bateman on automatic subgrammar extraction. In a similar vein, more abstract grammars can be refined on the basis of contextual information, as it is suggested by Theeramunkong et al.

- What are the necessary ingredients for quality assurance in grammar development?

  The incremental construction of large grammars, in particular in a distributed environment, makes it necessary to maintain sufficient control over different versions. Coverage and speed are expected to increase over the development cycles. Quality assurance, testing and diagnostics cannot be carried out properly, if they are based on a random collection of test items or on arbitrarily chosen corpus fragments. Evaluation of a system, which goes even further, will require a minimum degree of standardization of reference material. What are then the appropriate methods and data to be applied for these purposes? How can they be constructed, collected and customized to specific applications and domains?

From the wide range of activities which can be observed in this area, two aspects are illustrated in this volume. One concerns the issue of documentation in grammar development, which is nicely solved by a tool supporting hyper-textual annotations to grammars, specifically those based on typed feature logics (Dini and Mazzini). Another aspect is the annotation of corpora along the lines of Treebanks, which also allows to train parsers on the basis of disambiguated trees (Carter).

The original call for papers resulted in 24 submissions, from which 15 papers were selected on the basis of a thorough reviewing process, in which each paper was reviewed by three independent experts. Due to the high quality of submissions we decided to retain such a large number of submissions also at full length, even if some of them have only to be presented in a poster session at the actual workshop.

We would like to thank the two other members of the organizing committee, Dominique Estival and Alberto Lavelli, for their support in preparing this workshop.

Our gratitude also goes to Kordula de Kuthy and Anne Marie Mineur (DFKI) for their help in the editing and reviewing process.

We also thank the invited speaker, Hans Uszkoreit (DFKI and University of Saarbrücken), who presented a talk on the relationship and relevance of reference data and corpora for grammar engineering.

Finally, we would like to express our deep gratitude to the members of the program committee (Harry Bunt, Bob Carpenter, Jochen Dörre, Dominique Estival, Dan Flickinger, Steven Pulman, Antonio Sanfilippo), who did a great job in reviewing a large number of papers each.

Klaus Netter and Fabio Pianesi
Co-Chairs, Program Committee

Contents