Experience of using GATE for NLP R&D.

Hamish Cunningham, Diana Maynard,
Kalina Bontcheva, Valentin Tablan and **Yorick Wilks**
Department of Computer Science and
Institute for LAnguage, Speech and Hearing,
University of Sheffield, UK
{hamish,diana,kalina,valyt,yorick}@dcs.shef.ac.uk

Abstract

GATE, a General Architecture for Text Engineering, aims to provide a software infrastructure for researchers and developers working in NLP. GATE has now been widely available for four years. In this paper we review the objectives which motivated the creation of GATE and the functionality and design of the current system. We discuss the strengths and weaknesses of the current system, identify areas for improvement.

1 Introduction

This paper relates experiences in projects that have used GATE (General Architecture for Text Engineering) over the four years since its initial release in 1996.

We begin in section 2 with some of the motivation behind this type of system, and go on to give a definition of *architecture* in this context (section 3). Section 4 briefly describes GATE; section 5 covers a range of projects that have used the system. These experiences form the input to section 6 which discusses the system's strengths and weaknesses.

2 Motivation

If you're researching human language processing you should probably not be writing code to:

- store data on disk;
- display data;
- load processor modules and data stores into processes;
- initiate and administer processes;
- divide computation between client and server;
- pass data between processes and machines.

A *Software Architecture* for language processing should do all this for you. You will have to parameterise it, and sometimes deployment of your work into applications software will require some low-level fiddling for optimisation purposes, but in the main these activities should be carried out by infrastructure for the language sciences, not by each researcher in the field.

We can go further and say that you shouldn't have to reinvent components and resources outside of your specialism if there is already something that could do the job. A statistician doesn't need to know the details of the IEEE Floating Point computation standard; a discourse processing specialist doesn't need to understand all the ins and outs of part-of-speech tagging (or worse still how to install a particular POS tagger on a particular machine).

If you're a professional mathematician, you probably regard a tool like SPSS or Mathematica as necessary infrastructure for your work. If you're a computational linguist or a language engineer, the chances are that large parts of your work have no such infrastructural support. Where there is infrastructure, it tends to be specific to restricted areas. GATE, a General Architecture for Text Engineering (Cunningham et al., 1997), represents an attempt to fill this gap, and is a software architecture for language processing R&D.

We now have four years of experience with GATE, work on which began in 1995, with a first widespread release late in 1996. The system is currently at a pivotal point in its development, with a new version in development.

3 Infrastructure for Language Processing R&D

What does infrastructure mean for Natural Language Processing (NLP)? What sorts of tasks should be delegated to a general tool, and which should be left to individual projects? The position we took in designing GATE is to focus on the *common elements of NLP systems*.

There are many useful tools around for performing specific tasks such as developing feature structure grammars for evaluation under unification, or collecting statistical measures across corpora. To varying extents, they entail the adoption of particular theories. The only common factor of NLP systems, alas, seems to be that they very often create information about text. Developers of such systems create modules and data resources that handle text, and they store this data, exchange it between various modules, compare results of test runs, and generally spend inordinate amounts of time pouring over samples of it when they really should be enjoying a slurp of something relaxing instead.

The types of data structure typically involved are large and complex, and without good tools to manage and allow succinct viewing of the data we work below our potential. At this stage in the progress of our field, no one should really have to write a tree viewing program for the output of a syntax analyser, for example, or even have to do significant work to get an existing viewing tool to process their data.

In addition, many common language processing tasks have been solved to an acceptable degree by previous work and should be reused. Instead of writing a new part of speech tagger, or sentence splitter, or list of common nominal compounds, we should have available a store of reusable tools and data that can be plugged into our new systems with minimal effort. Such reuse is much less common than it should be, often because of installation and integration problems that have to be solved afresh in each case (Cunningham et al., 1994).

In sum, we defined our infrastructure as an architecture, framework and development environment, where an architecture is a macro-level organisational pattern for the components and data resources that make up a language processing system; a framework is a class library implementing the architecture; a development environment adds graphical tools to access the services provided by the architecture.

4 GATE

GATE version 1.n does three things:

- manages textual data storage and exchange;
- supports visual assembly and execution of modular NLP systems plus visualisation of data structures associated with text;
- provides plug-in modularity of text processing components.

The architecture does this using three subsystems:

- GDM, the GATE Document Manager;
- GGI, the GATE Graphical Interface;
- CREOLE, a Collection of REusable Objects for Language Engineering.

GDM manages the information about texts produced and consumed by NLP processes; GGI provides visual access to this data and manages control flow; CREOLE is the set of resources so far integrated. Developers working with GATE begin with a subset of CREOLE that does some basic tasks, perhaps tokenisation, sentence and paragraph identification and part-of-speech tagging. They then add or modify modules for their specific tasks. They use a single API for accessing the data and for storing their data back into the central database. With a few lines of configuration information they allow the system to display their data in friendly graphical form, including tree diagrams where appropriate. The system takes care of data storage and module loading, and can be used to deliver embeddable subsystems by stripping the graphical interface. It supports modules in any language including Prolog, Lisp, Perl, Java, C++ and Tcl.

5 Projects that used GATE

5.1 ECRAN

Goal: ECRAN (Extraction of Content: Research at Near-market) (Basili et al., 1997) was a 3-year EU funded research

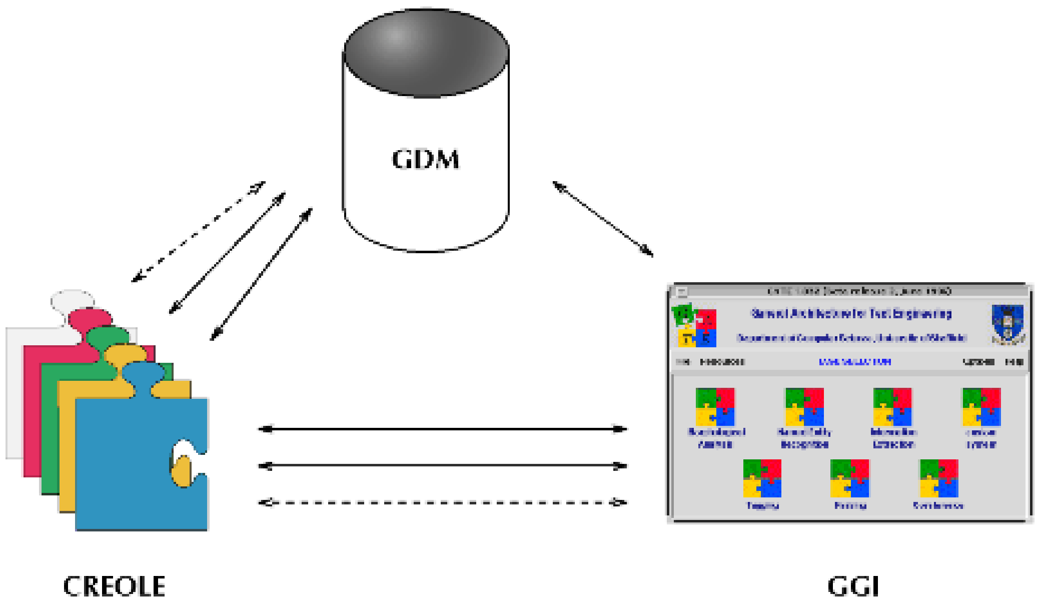

Figure 1: Gate Architecture

project with the main aim of carrying out information extraction using adapted lexicons.
Participants: Thomson-CSF (Paris) (project co-ordinators), SIS (Smart Information Systems, Germany), University of Sheffield, University of Rome La Sapienza, University of Geneva, NCSR "Demokritos" (Athens)
Description: GATE was mainly used in this project to implement a general word sense disambiguation engine based on a combination of classifiers.
Benefits: The modular architecture of GATE allowed this to be carried out very rapidly.
Drawbacks: Two main disadvantages were found with GATE. (1) The architecture was under development at the same time as the word sense disambiguation engine. (2) The speed of database access for the Tipster database was found to be slow for large amounts of lexical data. The solution used was to store large amounts of lexical data separately from GATE as gdbm hash tables.

5.2 Cass-SWE

Goal: The aim of the Cass-SWE project (A Cascaded Finite-State Parser for Syntactic Analysis of Swedish) (Kokkinakis and Johansson-Kokkinakis, 1999) was to create a parsing system for fast and accurate analysis of large volumes of written Swedish.
Participants: Språkdata/Göteborg University, Sweden.
Description: Cass-SWE implements the grammar as a modular set of 6 small grammars. GATE is used to integrate all the required software components into one system prior to parsing, and to enable the results to be visualised in a user-friendly environment.
Benefits: GATE allows the tagging process to be carried out sequentially, and enables modification of individual elements without disruption to others. Using GATE as a visualisation environment also enables the results of Cass-SWE to be further used in applications such as information extraction tasks and additional semantic processing.
Drawbacks: There were a few initial difficulties understanding the workings of the GATE system, but problems originally thought to be caused by GATE were later traced to the CASS parser.

5.3 GIE

Goal: The aim of the GIE (Greek Information Extraction) project (Petasis et al., 1999) was to develop a prototype named entity recognition model for Greek.
Participants: NCSR "Demokritos" (Athens), University of Sheffield
Description: The GIE system is based on the VIE system provided with GATE, but requires different language-specific resources such as gazetteers and grammars. Using GATE enables non-language specific resources to be reused from the English version, thereby saving time and effort.
Benefits: GATE facilitated significantly the integration of existing and new modules in GIE, as well as the validation of the final demonstrator. It was generally found to be fast, easy to use and powerful.
Drawbacks: GATE's demand for system resources as document size increases can become a serious limitation. Complex compilation processes made the embedding of static modules difficult. GATE also has some difficulties supporting non-Latin languages. mostly relating to the GUI. Many minor possible improvements to the GUI and to GATE in general (such as the addition of new features) were identified during this project.

5.4 LaSIE

Goal: LaSIE (Wilks and Gaizauskas, 1999)is an advanced large-scale IE system, performing named entity recognition, coreference resolution, template element filling and scenario template filling.
Participants: University of Sheffield
Description: LaSIE was designed specifically to work within the GATE architecture, and led to the free distribution of its counterpart, VIE, a base-line IE system. LaSIE modules within GATE have also formed part of other customised projects within the EC Fourth Framework (AVENTINUS and ECRAN).

5.5 EMPathIE

Goal: EMPathIE (Enzyme and Metabolic Path Information Extraction) was an 18-month research project aimed at applying Information Extraction technology to bioinformatics tasks.
Participants: Dept. of Information Studies & Dept. of Computer Science (University of Sheffield), Glaxo Wellcome plc., Elsevier Science.
Description: EMPathIE aims to extract details of enzyme reactions from articles in biomedical journals. The IE system is derived from LaSIE and was developed within the GATE architecture.
Benefits: The embedding of EMPathIE within the GATE environment means that many modules can be reused. EMPathIE thus makes use of many of the LaSIE modules, and itself produces modules which have been used for other related projects. Using GATE therefore enables much of the low-level work in moving IE systems to new domains to be carried out effortlessly.

5.6 SVENSK

Goal: SVENSK (Olsson, 1997; Olsson et al., 1998; Gambäck and Olsson, 2000) was a 4-year project aimed at developing an integrated toolbox of language processing components and resources for Swedish.
Participants: SICS (Swedish Institute of Computer Science), NUTEK, Uppsala University, Göteborg University, PipeBeach AB., Telia Research AB, IBM Svenska AB
Description: The toolbox is based on the GATE language engineering platform and incorporates language processing tools developed at SICS or contributed by external sources.
Benefits: Each component has a standardised interface, so users have the choice of working within GATE or selecting and combining supplied components for integration into a user application. GATE is useful in that it is not committed to any particular type of data or task. The emphasis on modularity was also found to be particularly appealing.

Drawbacks: GATE was at the time still in its early phases and had some problems with very large-scale resources. Specification of byte offset and I/O requirements for different modules was also difficult.

5.7 LOTTIE

Goal: LOTTIE (Low Overhead Triage from Text using Information Extraction) was a demonstrator project for the GATE infrastructure. It aimed to provide proof-of-concept by implementing demonstration software dealing with the major technological problems involved in computer-assisted triage.
Participants: University of Sheffield
Description: LOTTIE did not itself use GATE, but formed a basis on which to build it. Parts of it were real, based on a project in a different domain, and parts of it served as a test case for GATE development and as a demonstration of future possibilities.

5.8 AVENTINUS

Goal: AVENTINUS (Advanced Information System for Multinational Drug Enforcement) is an EU funded research and development programme set up to build an information system for multinational drug enforcement.
Participants: SIETEC (Germany), ADB (France), Amt für Auslandsfragen (Germany), Bundeskriminalamt (Germany), Sprakdata Gothenburg (Sweden), Institute for Language and Speech Processing (Greece), INCYTA (Spain), University of Sheffield.
Description: AVENTINUS aims to collect information from distributed international sources, using advanced linguistic techniques to improve IE, involving multimedia resources and supporting multilinguality.

5.9 TRESTLE

Goal: TRESTLE (Text Reuse, Extraction and Summarisation for Large Enterprises) (TRESTLE, 2000) is a 2-year project involving IE from electronic alerting bulletins distributed daily throughout the pharmaceutical industry.
Participants: Glaxo-Wellcome plc, University of Sheffield Dept. of Computer Science and Dept. of Information Studies.
Description: TRESTLE is based on the LaSIE IE system, but requires different domain-specific resources, such as gazetteers and ontology, and substantial modification of the discourse interpreter and template writer.
Benefits: GATE provides domain independent linguistic components for TRESTLE, the most important of which is the semantic parser. Named Entity recognition requires only the installation of domain specific gazetteers.
Drawbacks: It is very difficult to make even minor modifications to existing components of GATE. Current documentation is inadequate, and very strong computing skills are necessary in order to make the most of it.

5.10 PASTA

Goal: PASTA (Protein Active Site Template Acquisition) (K. Humphreys and Gaizauskas, 2000) extracts information about protein structures directly from scientific journal papers, and stores them in a template.
Participants: Depts. of Computer Science, Molecular Biology & Biotechnology, and Information Studies (University of Sheffield).
Description:The system has been adapted to the molecular biology domain from pre-existing IE technology such as LaSIE. The progress so far demonstrates the feasibility of developing intelligent systems for IE from text-based sources in the pursuit of knowledge in the biological domain.
Benefits: The use of a common database for storing intermediate results offers several advantages. GATE allows simple integration of heterogeneous system components and algorithms. The user interface is also attractive.
Drawbacks: GATE is slow and memory hungry, even for medium-sized documents, and is not very robust, particularly when upgrading is carried out.

5.11 EUDICO

Goal: The aim of Eudico was a distributed multimedia infrastructure supporting annotation of speech and video corpora (Brughman et al., 1998).
Participants: Max Planck Institute for Psycholinguistics (Nijmegen, Netherlands), University of Sheffield
Description: Eudico enables transcriptions of utterances to be time-aligned with speech and video data, so that dynamic and simultaneous viewing and editing is possible. Integration with GATE was carried out in order to benefit from GATE's ability to represent, store and visualise linguistic data.
Benefits: The flexibility of GATE's data model enabled the seamless integration between EUDICO's time-based data and GATE's offset-based annotations. This enabled the representation, manipulation and display of time-aligned transcriptions into GATE's viewers, allowing the user to manipulate the different types of data simultaneously in a uniform environment.
Drawbacks: There is a certain lack of support for distributed/remote access to the document manager. Therefore in a client-server environment, the entire data has to be sent over the network instead of just the parts that are needed.

5.12 German Named Entity Recognition

Goal: German Named Entity Recognition (Mitchell, 1997) was an MSc project to adapt part of the LaSIE system to deal with German, and to test whether the architecture was suitable for processing a language other than English.
Participants: Dept. of Computer Science, Sheffield University
Description: The system followed the same general architecture as LaSIE, but with modifications to various modules such as the grammar and tokeniser.
Benefits: Using the GATE architecture meant that only fairly minor modifications to individual modules were necessary, and rule adaptation was easy. The evaluation of LaSIE as a tool for processing other languages was very positive (as borne out by the later development of M-LaSIE (a multilingual IE system).
Drawbacks: The GATE API was large, complex and difficult to understand and modify, The ability to group modules into blocks for processing would be a useful addition, as would an easier method of inserting new modules in the correct place.

6 Strengths and Weaknesses

GATE has proved successful in a number of contexts, with users reporting a variety of work with the system, for example:

- Teaching undergraduates and postgraduates. Our colleagues at UMIST and the Universities of Edinburgh, and Sussex have reported using the system for teaching, as have the Universities of Stuttgart and Saarburcken.
- Information Extraction in English, Swedish, French, Spanish and Greek. Our colleagues in Fribourg University collaborated with us on a French IE system; both ILSP and NKSR Demokritus in Athens are developing a Greek IE system; the University of Gothenburg has a Swedish system; the University of Catalonia in Barcelona are working on Spanish.
- Integrating information extraction with Information Retrieval. The Naval Office of R&D (NRaD) in San Diego is using GATE for research on text summarisation and IE/IR integration.
- Integrating a national collection of NLP tools for Swedish. See http://www.sics.se/humle/projects/svensk/
- ESTEAM Inc., of Gothenburg and Athens are using the system for adding name recognition to their MT systems (for 26 language pairs) to improve performance on unknowns.
- The Speech and Hearing group at Sheffield are modelling out-of-

vocabulary language using VIE and GATE (Gotoh et al., 1998).

- Numerous postgraduates in locations as diverse as Israel, Copenhagen and Surrey are using the system to avoid having to write simple things like sentence splitters from scratch, and to enable visualisation and management of data.

Abstracting from their experiences and that of users at Sheffield, GATE's strengths can be summarised as:

- facilitating reuse of NLP components by reducing the overheads of integration, documentation and data visualisation;
- facilitating multi-site collaboration on IE research by providing a modular base-line system (VIE) with which others can experiment;
- facilitating comparative evaluation of different methods by making it easy to interchange modules;
- facilitating task-based evaluation, both of "internal" components such as taggers and parsers, and of whole systems, e.g. by using materials from the ARPA MUC programme (Grishman and Sundheim, 1996) (whose scoring software is available in GATE, as is the Parseval tree scoring tool (Harrison, 1991), and a generic annotation scoring tool);
- contributing to the portability of NLP systems across problem domains by providing a markup tool for generating training data for example-based learning (it can also take input from the Alembic tool (Day et al., 1997) for this purpose, using Edinburgh's SGML processing library (McKelvie et al., 1997)).

There several weaknesses in the system, and some areas that are underdeveloped or lacking polish. In rough order of severity:

1. Version 1 is biased towards algorithmic components for language processing, and neglects resource components.
2. Version 1 is biased towards text analysis components, and neglects text generation components.
3. The visual interface is complex and somewhat non-standard.
4. Installing and supporting the system is a skilled job, and it runs better on some platforms than on others (UNIX vs. Windows).
5. Sharing of modules depends on sharing of annotation definitions (but isomorphic transformations are relatively easy to implement).
6. It only caters for textual documents, not for multi-media documents.
7. It only supports 8-bit character sets.

Points 1 and 2 compromise the generality of the system, and have limited take-up, as well as the number of CREOLE modules integrated with the system. For modules like taggers, parsers, discourse analysers (i.e. just about anything that performs an analysis task) the GATE integration model provides a convenient and powerful abstraction layer based on storing information in association with the text under analysis. For resources like lexicons or corpora, no such layer exists. Similarly, for modules that do generation-side tasks, since there is no text under analysis, the utility of a text-based model is limited.

For details of the means by which we intend to combat these problems and extend the range of the system, see (Cunningham, 2000). More details of requirements for this type of system, and how to evaluate them, are available in (Cunningham et al., 2000).

7 Conclusion

Based on the collective experiences of a sizeable user base across the EU and elsewhere, the system can claim to be a viable infrastructure for certain sections of the field. Given further development, we hope that it can take on this role for a wider variety of tasks.

8 Acknowledgements

This work was supported by EPSRC grants GR/K25267 and GR/M31699.

References

R. Basili, M. Pazienza, P. Velardi, R. Xatizone, R. COllier, M. Stevenson, Y. Wilks, O. Amsaldi, A. Luk, B. Vauthey, and J. Grandchamp. 1997. Extracting case relations from corpora. ECRAN Deliverable 2.4 version 1.

H. Brughman, A. Russel, P. Wittenburg, and R. Piepenbrock. 1998. Corpus-based research using the Internet. In *First International Conference on Language Resources and Evaluation (LREC) Workshop on Distributing and Accessing Linguistic Reseources*, Granada, Spain.

H. Cunningham, M. Freeman, and W.J. Black. 1994. Software Reuse, Object-Oriented Frameworks and Natural Language Processing. In *New Methods in Language Processing (NeMLaP-1), September 1994*, Manchester. (Re-published in book form 1997 by UCL Press).

H. Cunningham, K. Humphreys, R. Gaizauskas, and Y. Wilks. 1997. Software Infrastructure for Natural Language Processing. In *Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP-97)*, March. `http://xxx.lanl.gov/abs/cs.CL/9702005`.

H. Cunningham, K. Bontcheva, V. Tablan, and Y. Wilks. 2000. Software Infrastructure for Language Resources: a Taxonomy of Previous Work and a Requirements Analysis. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC-2)*, Athens. `http://gate.ac.uk/`.

Hamish Cunningham. 2000. Software Architecture for Language Engineering. Forthcoming.

D. Day, J. Aberdeen, L. Hirschman, R. Kozierok, P. Robinson, and M. Vilain. 1997. Mixed-Initiative Development of Language Processing Systems. In *Proceedings of the 5th Conference on Applied NLP Systems (ANLP-97)*.

B. Gambäck and F. Olsson. 2000. Experiences of Language Engineering Algorithm Reuse. In *Second International Conference on Language Resources and Evaluation (LREC)*, pages 155–160, Athens, Greece.

Y. Gotoh, S. Renals, R. Gaizauskas, G. Williams, and H. Cunningham. 1998. Named entity tagged language models for lvcsr. Technical Report CS-98-05, Department of Computer Science, University of Sheffield.

R. Grishman and B. Sundheim. 1996. Message understanding conference - 6: A brief history. In *Proceedings of the 16th International Conference on Computational Linguistics*, Copenhagen, June.

P. Harrison. 1991. Evaluating Syntax Performance of Parsers/Grammars of English. In *Proceedings of the Workshop on Evaluating Natural Language Processing Systems, ACL*.

G. Demetriou K. Humphreys and R. Gaizauskas. 2000. Two applications of information extraction to biological science journal articles: Enzyme interactions and protein structures. In *Proc. of Pacific Symposium on Biocomputing (PSB-2000)*, Honolulu, Hawaii.

D. Kokkinakis and S. Johansson-Kokkinakis. 1999. Cascaded finite-state parser for syntactic analysis of swedish. Technical Report GU-ISS-99-2, Dept. of Swedish, Göteborg University. `http://svenska.gu.se/ svedk/-publications.html`.

D. McKelvie, C. Brew, and H. Thompson. 1997. Using SGML as a Basis for Data-Intensive NLP. In *Proceedings of the fifth Conference on Applied Natural Language Processing (ANLP-97)*, Washington, DC.

B. Mitchell. 1997. Named Entity Recognition in German: the identification and classification of certain proper names. Master's thesis, Dept. of Computer Science, University of Sheffield. `http://www.dcs.shef.ac.uk/-campus/dcscd/projects/bm.pdf`.

F. Olsson, B. Gambäck, and M. Eriksson. 1998. Reusing Swedish Language Processing Resources in SVENSK. In *Workshop on Minimising the Efforts for LR Acquisition*, Granada.

F. Olsson. 1997. Tagging and morphological processing in the svensk system. Master's thesis, University of Uppsala. `http://http://stp.ling.uu.se/ fredriko/-exjobb.ps`.

G. Petasis, G. Paliouras, V. Karkaletsis, C.D. Spyropoulos, and I. Androutsopoulos. 1999. Resolving part-of-speech ambiguity in the greek language using learning techniques. In *Proc. of the ECCAI Advanced Course on Artificial Intelligence (ACAI)*, Chania, Greece.

TRESTLE. 2000. The TRESTLE project. `http:/www.dcs.shef.ac.uk/research/-groups/nlp/trestle`.

Y. Wilks and R. Gaizauskas. 1999. Report on epsrc research grant on the large scale information extraction research project. Technical Report GR/K25267, University of Sheffield.