Natural Language Engineering **5** (1): 1–16. Printed in the United Kingdom

© 1999 Cambridge University Press

A definition and short history of Language Engineering

HAMISH CUNNINGHAM

Institute for Language, Speech and Hearing (ILASH), and
Department of Computer Science, University of Sheffield, Sheffield, UK
e-mail: hamish@dcs.shef.ac.uk

(*Received 9 June 1998, Revised November 1998*)

Abstract

This paper discusses the nature, history and current characteristics of Language Engineering, which is contrasted with Natural Language Processing and Computational Linguistics, and which is shown to have attained its own distinct identity in recent years. Major trends in the field are examined, including its focus on large-scale practical tasks and on quantitative evaluation of progress, and its willingness to embrace a diverse range of techniques. The importance of software engineering in this context is noted, as are some sociological aspects of the practitioner group.

1 Introduction

What is *Language Engineering* (LE)? Is it new? Is it any different from applied Computational Linguistics (CL) or Natural Language Processing (NLP)? Where did it come from and where is it going? This paper situates LE in the context of its parent and sister disciplines, and its recent history. One aim of this analysis is to provide a basis for constructing and evaluating infrastructural systems, such as software architectures and development environments, that facilitate LE research and development.

Attempting to define something as dynamic and multi-faceted as a research field concerned with human language is a difficult task; given ten researchers there will likely be ten definitions, all using similar terminology. The temptation in such circumstances is to invent a new vocabulary of terms to inject a new precision, but the risk is that the Gestalt will then be lost. At the opposite pole, a phenomenological account might avoid definition altogether and simply compile anecdotes. We'll try to steer a course between these extremes: we'll use existing terminology and let a number of researchers speak for themselves in quotation, but still arrive at definitions that try to capture the essentials.

Section 2 presents definitions of CL, NLP, and LE. Section 3 looks at the history of the field, and at some of the factors influencing its genesis. Finally, section 4 examines six salient characteristics of LE as it stands in the late 1990s.

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press

2 Definitions

We begin with the general terms *language* and *engineering*, then move on to CL, NLP and LE.

The *Collins English Dictionary* defines language as "a system for the expression of thoughts, feelings, etc., by the use of spoken sounds or conventional symbols" (Makins 1991). Language is a communication mechanism whose medium is text or speech, and LE is concerned with computer processing of text and speech. We will define *engineering* in contrast to *science* (and, later, LE in contrast to CL and NLP).

Herb Simon contends that "...far from striving to separate science from engineering, we need not distinguish them at all. But if we insist upon a distinction, we can think of engineering as science for people who are impatient". This point is undermined a little by a criterion which immediately follows and by which the two fields may be separated: "While the scientist is interested specifically in creating new knowledge, the engineer is interested also in creating systems that achieve desired goals" (Simon 1995). We'll preserve the distinction here and use a definition of science as "the systematic study of the nature and behaviour of the material and physical universe, based on observation, experiment, and measurement, and the formulation of laws to describe these facts in general terms" (Makins 1991).

Engineering connotes "creating cost-effective solutions to practical problems by applying scientific knowledge" (Shaw and Garlan 1996), or "applying scientific principles to the design, construction and maintenance of engines, cars, machines etc." (Makins 1991), or a "rigorous set of development methods" (Brown 1989). The "...basic difference between science and engineering is that science is concerned with finding out how the world works, while engineering is concerned with using that knowledge to build artifacts in such a way that one can expect them to perform as required" (Cowling 1996).

Buried in the phrases "achieve desired goals" and "perform as required" is the implication that engineered systems must conform to externally-specified criteria of adequacy, for example criteria relating to the space/time profile of an executable processing a certain type of data. This in turn has implications for the engineering process: unless our external criteria are floating free in time, a system will not perform as required or achieve its goals if the process of its development is unbounded with respect to resource requirements. In other words, engineers have to operate within finite time and with finite personnel resources and equipment. As well as implying that its outputs perform predictably, engineering implies that the process of constructing those outputs is also predictable.

If the *act* of engineering is a particular kind of construction process, the *field* of engineering is both the body of scientific knowledge relevant to a particular engineering task and also what we may call craft, art, lore, or heuristics: a body of known practice offering solutions to problems. For example, in electrical engineering, books with titles like "The Art of Electronics" (Horowitz and Hill 1989); in software engineering books like "Numerical Recipes in C – the Art of Scientific Computing" (Press, Teukolsky, Vetterling and Flannery 1995), or "Expert C Programming – Deep C Secrets" (van der Linden 1994), or "Design Patterns – Elements of Reusable

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press

Object-Oriented Software" (Gamma, Helm, Johnson and Vlissides 1995). This craft knowledge takes many forms, e.g. a description of what scientific results are relevant to particular problems; example solutions to problems; notes of common problems or pitfalls in particular implementation paradigms; anecdotal evidence suggesting that certain solutions may be better than others in some circumstances. For example, a book on database design may refer to relational algebra; a book on algorithms may provide a proven, tested method for sorting with a known complexity and space requirement; a book on expert programming may contain hints like "versions of the `indent` code formatting program up to the mid-80's silently transform `x=!y` into `x!=y`"[1]; a book on design patterns might suggest that implementing a monitor pattern under a particular thread regime is inefficient.

To summarise, we define engineering as a construction process that is directed both by intended conformance of the resultant artifact to well-specified criteria of fitness and by constraints operating on the nature of the process itself; both the construction process itself and its outputs should be measurable and predictable; the activity is informed by relevant scientific knowledge and practical experience.

Turning to *computational linguistics*, we refer to our definitions of science and language: CL is that part of the science of human language that uses computers to aid observation of, or experiment with, language. If "Theoretical linguists... attempt to characterise the nature of... either a language or Language" or "a grammar or Grammar", then "...theoretical Computational Linguistics proper consists in attempting such a characterisation *computationally*" (Thompson 1985). In other words, CL concentrates "on studying natural languages, just as traditional Linguistics does, but using computers as a tool to model (and, sometimes, verify or falsify) fragments of linguistic theories deemed of particular interest" (Boguraev, Garigliano and Tait 1995).

Natural language processing is a term used in a variety of ways in different contexts. Much work that goes under the heading of NLP could well fit under our definition of CL, and some could also fit the definition of LE that follows. Here we'll use a narrow definition that makes CL and NLP disjoint. Whereas CL is "a branch of linguistics in which computational techniques and concepts are applied to the elucidation of linguistic and phonetic problems" (Crystal 1991), NLP is a branch of computer science that studies computer

> ...systems for processing natural languages. It includes the development of algorithms for parsing, generation, and acquisition of linguistic knowledge; the investigation of the time and space complexity of such algorithms; the design of computationally useful formal languages (such as grammar and lexicon formalisms) for encoding linguistic knowledge; the investigation of appropriate software architectures for various NLP tasks; and consideration of the types of non-linguistic knowledge that impinge on NLP. It is a fairly abstract area of study and it is not one that makes particular commitments to the study of the human mind, nor indeed does it make particular commitments to producing useful artifacts. (Gazdar 1996)[2]

[1] Bizarre but true – see van der Linden (1994, p. 11.)

[2] Though note that Gazdar was referring here to 'theory of linguistic computation', which he takes to be a component of NLP, not NLP *per se*.

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press

There are elements of both CL and NLP in Winograd's early description of the work as

> ...part of a newly developing paradigm for looking at human behaviour, which has grown up from working with computers. ... Computers and computer languages give us a formal metaphor, within which we can model the processes and test the implications of our theories. (Winograd 1972)

To summarise: CL is a part of the science of language that uses computers as investigative tools; NLP is part of the science of computation whose subject matter is computer systems that process human language[3]. There is crossover and blurring of these definitions in practice, but they capture some important generalisations.

We have by implication conflated spoken language and textual language, but this hides the fact that there is a distinct field concerned with computer processing of spoken language which is

> ...known as **speech processing** or just **Speech**. Surprisingly, perhaps, Speech and NLP are really rather separate disciplines: until recently there has been little overlap in the personnel involved, the journals and conferences are largely disjoint, and the theoretical approaches and methods used have had little in common. Speech research studies the problems that are peculiar to processing the spoken form of language whereas NLP research, in principle at least, studies the problems that are common to the processing of both spoken and written forms of natural languages. (Gazdar 1996)

We'll have some more to say about speech below, but now we turn to our main subject, language engineering.

From our perspective, Gazdar's definition of "Applied NLP" is close to that of LE, a subject which

> ...involves the construction of intelligent computational artifacts that process natural languages in ways that are useful to people other than computational linguists. The test of utility here is essentially that of the market. Examples include machine translation packages, programs that convert numerical data or sequences of error codes into coherent text or speech, systems that map text messages into symbolic or numeric data, and natural language interfaces to databases. (Gazdar 1996)

LE is the application of NLP to the construction of computer systems that process language for some task usually other than modelling language itself, or "...the instrumental use of language processing, typically as part of a larger system with some practical goal, such as accessing a database" (Thompson 1985)[4]. Probably as good a short definition as any of the *products* of LE is given by Jacobs: LE "systems are meant to be fast, effective and helpful" (Jacobs 1992). A longer definition of the field appears in the editorial of the first issue of this journal, which also contrasts the practices of NLP and CL with those of LE[5]:

> The principal defining characteristic of NLE work is its objective: to engineer products which deal with natural language and which satisfy the constraints in which they have

[3] So if NLP is a part of AI, then AI is a branch of computer science whose subject matter is programs that display capabilities similar to facets of human intelligence.
[4] Note that Thompson was referring to 'Applied CL' here.
[5] LE and Natural Language Engineering (NLE) are synonymous.

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press

to operate. This definition may seem tautologous or a statement of the obvious to an engineer practising in another, well established area (e.g. mechanical or civil engineering), but is still a useful reminder to practitioners of software engineering, and it becomes near-revolutionary when applied to natural language processing. This is partly because of what, in our opinion, has been the ethos of most Computational Linguistics research. Such research has concentrated on studying natural languages, just as traditional linguistics does, but using computers as a tool to model (and, sometimes, verify or falsify) fragments of linguistic theories deemed of particular interest. This is of course a perfectly respectable and useful scientific endeavour, but does not necessarily (or even often) lead to working systems for the general public. (Boguraev, Garigliano and Tait 1995)

They go on to define the "constraints to be satisfied", including the existence of a target user group with a defined need that can be solved by the LE system, favourable cost-benefit profile and defined criteria for testing and evaluation of the system.

The final component of our definition of LE is that it is an art, or a body of rules of thumb. This is little written of at present, but has been noted at various workshops. For example, at panel discussions quoted in Mitkov (1995, 1996): Tsujii called LE heuristic and empirical (and not normally interested in language *per se*); Matusmoto said it is art, craft and technique. Not strictly relevant, but included here for flavouring are Yusoff's comments that language engineers make things work without knowing why, whereas computational linguists know why their systems don't work, and that art is 'doing it', engineering is 'doing right things' and science is 'doing things right'. As with any engineering discipline, LE rests on an ever-expanding body of craft, lore, art, heuristics and practice.

To summarise, our gloss on these various definitions is this:

Language Engineering is the discipline or act of engineering software systems that perform tasks involving processing human language. Both the construction process and its outputs are measurable and predictable. The literature of the field relates to both application of relevant scientific results and a body of practice.

Having arrived at an answer to 'What is LE?', we now turn to the question of whether it constitutes a new field or just a new label.

3 History

What is the difference between LE and applied CL, or applied NLP, and hasn't this sort of work been going on since at least the early days of MT? This section addresses these points.

The development of applications software that makes use of NLP techniques is not a new thing: Systran (the first major commercial MT system), for example, was programmed in the 1960s, and there has been a steady stream of technology transfer to products in the subsequent decades (from the use of formal language theory in compiler construction to morphology or dictionary lookup or spell checking in word processors – see Church and Raul (1995) for a review of commercial applications of NLP). In the sense of the production of language processing systems, LE-type activities have been around for several decades; in the sense of an engineering discipline with an established body of practice, however, LE is much more recent.

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press

Two points illustrate this distinction: (a) engineering is not the same as production; (b) dissemination of the experience, art and craft of the field by explicit publication is a relatively recent phenomenon.

In support of point (a), the existence of products does not in itself imply that the systems were engineered in the sense defined above. Although the current organisation of production dictates that the ultimate test of the usefulness of artifacts is the market, just because a system is marketed doesn't mean that is has been successfully engineered – it may have come in way over budget, only run on machines far bigger than those available to the target user base, or be much less accurate than originally planned, and the methods used in its production may have been completely *ad hoc*.

In support of point (b), the idea that mere experience of the field deserved discussion, definition and record is a relatively new one. It is in the 1990s that LE work has become more prevalent and better defined, and in the process acquired its own name, journal and expanding literature.

The origin of the term is traced by Mitkov (1995) to a panel proposed by Nagao at COLING-88; the phrase entered common currency in Europe in the early 1990s with the eponymous funding programme of the European Commission (EC) (Boguraev, Garigliano and Tait 1995; Mitkov 1996). Referring to this programme, the EC wrote in 1996 that "Language Engineering has successfully promoted a shift from long-term research activities to more immediately feasible and industrially relevant RTD themes" by "...supporting projects aiming at market opportunities in the short term and the medium term..." (EC 1996). An EC requirement for projects in this programme is that they must be industrially led.

On the other side of the Atlantic in the recent years [D]ARPA[6] has moved towards requiring research software to be built within efficient delivery vehicles that use a standardised API for data management and storage (Grishman 1997; Cunningham *et al.* 1997). Part of the idea behind this is to encourage a competitive market in modules by enabling plug-and-play swapping of components from different suppliers.

In parallel with these changes the body of knowledge that a practising language engineer can draw on has also begun to expand. This journal has already been mentioned; we can also note the Applied Natural Language Processing (ANLP) conference series (ACL 1997), and although none currently exist it is becoming possible to imagine cookbooks for building certain types of LE systems. For example, Named Entity recognition[7] is now sufficiently well-understood that a set of recipes could be collected for systems based on

1. Markov models (Bikel, Miller, Schwartz and Weischedel 1997; Miller, Crystal, Fox, Ramshaw, Schwartz, Stone and Weischedel 1998), or

[6] The [Defense] Advanced Research Projects Agency (the 'D' seems to come and go every few years).

[7] Named Entity recognition was one of the tasks of the MUC-6 and MUC-7 Message Understanding Conferences, a DARPA-sponsored bi-annual information extraction competition series of which the seventh, in 1998, was the last (Grishman and Sundheim 1996; Hobbs 1993; Cunningham 1997).

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press

2. Brill-style transducers (Vilain, and Day 1996; Day, Robinson, Vilain and Yeh 1998), or
3. chart parsing (Gaizauskas, Wakao, Humphreys, Cunningham and Wilks 1995; Humphreys, Gaizauskas, Azzam, Huyck, Mitchell, Cunningham and Wilks 1998).

In each case the relevant algorithms and data structures could be described, along with the expected performance levels and run-time resource requirements relative to certain types of text (based on current practice, probably only news wire texts), and the profile of the production process required (in (1.), parameter selection, text markup and training; in (2.), the same as 1. followed by manual tweaking; manual rule collection in (3.)).

To summarise, in important respects LE is a creation of the last decade, during which time the term and a body of literature describing practice, relevant science and experience have arrived.

There are a number of commonly cited reasons for the establishment of LE including:

- computer hardware advances which have increased processor speeds and memory capacity, while reducing prices, and therefore made computationally expensive processes that have typified language processing work more viable;
- the increasing availability of large-scale, language-related, on-line resources, such as dictionaries, thesauri, and 'designer' corpora – corpora selected for representativeness and perhaps annotated with descriptive information (Wilks, Guthrie and Slator 1996);
- the demand for applications in a world where electronic text has grown exponentially in volume and availability, and where electronic communications and mobility have increased the importance of multi-lingual communication;
- maturing NLP technology which is now able, for some tasks, to achieve high levels of accuracy repeatedly.

The next section looks at the current state of the field.

4 Characteristics

Rumour has it that the word "hermeneutics" is now regularly being heard in the corridors of Palo Alto research centers and that may be a sure sign of desperation among some of the more theoretically oriented.
(Wilks 1996).

The 1990s have seen a "...wholesale invasion of the NLP community and culture of the more engineering-oriented approaches and behaviour of speech engineers" (Wilks 1996), a change dramatic enough to constitute a "...revolution that has taken place in natural language processing research over the last five years" (Gazdar 1996). As noted earlier, speech research has in the past been largely separate from NLP and CL, a fact reflected in the ethos and methods of the respective groups. Speech research has for a number of years focused on large-scale or 'real-world' tasks, i.e. tasks driven by the possibilities for marketable applications software

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press

based on automatic speech recognition and synthesis. In parallel with this focus has been an emphasis on quantitative evaluation as the arbiter of the worth of different approaches, and a willingness to consider any technique irrespective of psychological plausibility. (We should note here that perhaps the most significant reason for the growth of empiricism in speech research has been DARPA's insistence on quantitative evaluation in their speech and language programmes. As we'll see below, this has also had an effect on NLP and LE.)

Church and Mercer (Church and Mercer 1993) note a correlation between use of empirical methods and large-scale tasks, and knowledge-based methods and small-scale tasks, and the most salient characteristic of recent approaches to speech recognition has been their reliance on statistical modelling of language using hand-annotated corpora.

The growth of LE has seen similar characteristics emerge in textual language processing. Other significant trends involve social factors such as the increasing pressure for research groups to collaborate, and the nature and composition of these research groups themselves. The rest of this section discusses these trends in turn.

4.1 Large-scale tasks

It has been a common observation that early AI approaches to language tended towards a 'toy problem syndrome': AI has often chosen to investigate artificial, small-scale applications of the technology under development. These 'toy' problems are intended to be representative of the work involved in building applications of the technology for end-user or 'real-world' tasks, but scaling up problem domains from the toy to the useful has often shown the technology developed for the toy to be unsuitable for the real job. This has led to "...a growing skepticism about the importance of small-scale, research systems and whether many of them are genuinely original, as opposed to being notational variants in a field not very aware of its own history" (Wilks 1996), and to an increasing focus on larger-scale tasks.

For example, Wilks writes in Cunningham, Gaizauskas and Wilks (1995) that he began a

> ...Prolog grammar project in 1985 (Farwell and Wilks 1989): by 1987 it was perhaps the largest DCG (Definite Clause Grammar) grammar anywhere, designed to cover a linguistically well-motivated test set of sentences in English. Interpreted by a standard parser it was able to parse completely and uniquely virtually no sentence chosen randomly from a newspaper. We suspect most large grammars of that type and era did no better, though reports are seldom written making this point.
>
> The mystery for linguists is how that can be: the grammar appeared to inspection to be virtually complete – it *had* to cover English, if thirty years of linguistic intuition and methodology had any value. It is a measure of the total lack of evaluation of parsing projects up to that time that such conflicts of intuition and result were possible, a situation virtually unchanged since Kuno's large-scale Harvard parser of the 1960's (Kuno and Oettinger 1962) whose similar failure to produce a single, preferred, spanning parse gave rise to the AI semantics and knowledge-based movement. The situation was effectively unchanged in 1985 but the response this time around has been quite different.

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press

In the late 1980s and the 1990s,

> ...the combination of pressure from the U.S. government funders... and the Zeitgeist itself have pushed NLP towards specific applications, systems evaluation and above all, large-scale language processing systems. ...The move to the large scale, to applications and to evaluation in NLP is worldwide at the moment. (Wilks 1996)

This shift to applications has had several knock-on effects. As with speech research, it has led to a belief that what matters about systems is performance, not the interesting things about language or mind that might be illustrated. In many cases, the technologies now being developed are assistive, rather than fully automatic, aiming to enhance or supplement a human's expertise rather than attempting to mimic it (Kay 1997*b*; Kay 1997*a*; Gaizauskas, Cunningham, Wilks, Rodgers and Humphreys 1996). It has also meant that measures of performance have become much more important – see the next section – and a renewal of interest in performance-based models of language, with a corresponding renewal and extension of statistical techniques – see section 4.3. Lastly, software skills themselves have become more important. As an engineering discipline whose product is software, successful LE is predicated on successful software engineering: "...whoever builds a large software system inevitably runs up against software engineering concerns. Failing to handle them well harms the overall effort accordingly. ... System development issues should be as important as the dictionary definitions" (Cowie and Lehnert 1996).

4.2 Evaluation

It may seem obvious that language processing systems should be subject to empirical criteria of effectiveness. The big problem, of course, is determining precisely what the criteria of success should be. There is now a substantial recent literature on this question (Crouch, Gaizauskas and Netter 1995; Sparck-Jones 1994; Sparck-Jones and Galliers 1996; Gaizauskas 1997), and practical solutions to the evaluation problem have emerged in a number of areas.

Participants in the MUC (Message Understanding Conference, an information extraction competition) and TREC (Text Retrieval Conference, a competition for information retrieval, or "document detection") competitions (ARPA 1996), for example, build systems to do precisely-defined tasks on selected bodies of news articles. Human analysts are employed to produce correct answers for some set of previously unseen texts, and the systems produce machine output for those texts. The performance of the systems relative to human annotators is then measurable quantitatively. Quantitative evaluation metrics bring numerically well-defined concepts like precision and recall, long used to evaluate information retrieval systems, to other areas of language engineering.

4.3 Performance vs. competence

Following Chomsky, a distinction can be made between human linguistic capabilities, or their language *competence*, and the observable usage of language in everyday communication, or its *performance*. "A linguistic *competence* model aims to characterise the set of grammatical sentences together with the possible analyses that can be

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press

assigned to these sentences; a linguistic *performance* model aims to describe the actual production and perception of natural language sentences in specific situations" (Bod 1995). It has been noted that competence models associate strongly with rationalism and that performance models associate strongly with empiricism (Church and Mercer 1993). The performance of language is our experience of language; our capacity for language is a more abstract object, the subtler aspects of which can be reasoned about at length without stumbling over inconvenient realities. With the shift to large-scale tasks and quantitative evaluation, language research since the early 1980s has seen increasing reliance on large text collections (corpora) intended as representative samples of some aspect of language performance. These corpora are often annotated with the intended output of some task (e.g. part-of-speech tagging, or named-entity recognition and classification, or alignment of translations) and used to train statistical or connectionist models of the language input / task output relationship embodied in the data. Interestingly, this focus on corpora has more in common with past practice in lexicography than in linguistics, a fact which informed some of the upsurge in Machine Tractable Dictionary research in the 1980s (Wilks, Guthrie and Slator 1996).

The shiny new empiricism eventually triumphed over the evil old rationalism and everyone lived happily ever after. Well, no, not really. Although statistical methods climbed the accuracy curve rapidly to begin with, it became apparent that two factors undermined the simplistic separation of the new and the old sketched out above. First, the addition of markup to corpora has often been informed by linguistic intuition, for example about the nature of syntax (Marcus, Santorini and Marcinkiewicz 1993). It is not clear, then, that the statistical model built from this data is in fact independent of a rationalist explanation of language competence. Secondly, purely statistical systems soon encountered problems dealing with non-local reference, and with the sparse phenomena that Zipf showed in the 1930s was an intrinsic property of language (Zipf 1935). Although initial results were promising, statistical MT, for example, appeared to be hitting a hard ceiling at fairly low performance levels (Wilks 1994). Responses to these factors were also twofold. Firstly, many systems now incorporate both statistical and symbolic linguistic knowledge; this trend led Gazdar to write of 'paradigm merger' in Gazdar (1996). Secondly, research into statistical NLP has become more interested in models "...that yield good results with relatively small samples" (Dunning 1993; Karov and Edelman 1996).

Hybrid systems are indeed increasingly common, but it may be too early to talk of consensus breaking out into full paradigm merger. Charniak wrote in 1993 that "...it is fair to say that few, if any, consider the traditional study of language from an artificial-intelligence point of view a "hot" area of research. A great deal of work is still done on specific NLP problems... but for me at least it is increasingly hard to believe that it will shed light on broader problems, since it has steadfastly refused to do so in the past" (Charniak1993). There are also still plenty of purely statistical systems out there, and for some tasks their results are comparable with the best available, for example the Named Entity recogniser developed at BBN that uses Markov models (Bikel *et al.* 1997).

In summary, NLP is still a science encompassing paradigmatic conflicts arising

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press

from substantially different views on almost everything. This theoretical diversity (with a parallel diversity of technical approaches in LE) means that the field is dynamic and rapidly changing, and, given the distance between our current capabilities and those of a fully language-capable computer, this state of affairs can be expected to last well into the next millennium. The field may not be hot, but it is simmering.

4.4 Data reuse vs. algorithmic reuse

The idea of reusing previously developed components in new software systems has been part of the Holy Grail for software engineering for some time now (Yourdon 1996), and has also become an issue for LE (Cunningham, Freeman and Black 1994) in the context of the large-scale systems discussed earlier. If we distinguish the data used by LE systems for training or as internal knowledge resources (e.g. lexicons) from the algorithms used in processing, we get two quite different pictures of the current level of reuse in LE.

In line with the trend to empiricism the last ten years have seen substantial interest in large-scale processing of Machine-Readable Dictionaries (MRDs) and a number of MRDs have become available to the research public, each stimulating a flurry of research into their possible uses. For example: the *Merriam-Webster New Pocket Dictionary* (Amsler and White 1979; Amsler 1980; Amsler 1981); *Webster's Seventh New Collegiate Dictionary* (Chodorow, Byrd and Heidorn 1985; Markowitz, Ahlswede and Evens 1986; Binot and Jensen 1987); *The Longman Dictionary of Contemporary English* (Michiels, Mullenders and Noel 1980; Michiels and Noel 1982; Walker and Amsler 1986; Boguraev, Briscoe, Carroll, Carter and Grover 1987; Boguraev and Briscoe 1987; Wilks, Fass, Guo, McDonald, Plate and Slator 1987; Guthrie, Slator, Wilks and Bruce 1990; Bruce and Guthrie 1992); the Wordnet thesaurus (Miller 1990; Resnik 1993); and the *EDR* dictionary (Yokoi 1995; Miller, Kado, Hirakwa and Ichikawa 1995; Lenat, Miller and Yokoi 1995). Similarly, there are many reports of work with large corpora of texts (Ejerhed and Dagan 1996) in general, and the Penn Tree Bank (Marcus, Santorini and Marcinkiewicz 1993) in particular, e.g. for extracting grammars (Gaizauskas 1995) or for building taggers and parsers and measuring their performance (Bod 1996; Magerman 1994; Magerman 1995; Brill 1992; Brill 1995; Miller *et al.* 1998).

In contrast to this wealth of reuse of data resources, algorithmic reuse remains low, one key reason being that the integration and reuse of different components can be a major task. Referring to the need for algorithmic reuse, the EAGLES tools group noted that

> Unfortunately, the linguistic software that exists at present (see for example the lists at the Natural Language Software Registry) only begins to cover the growing needs. Industrial software is often expensive or unavailable, and usually hard to adapt or extend. On the other hand, the substantial body of natural language processing academic software is often experimental and hard to get, hard to install and under-documented, and sometimes unreliable. In both cases tools are typically embedded in large, non-adaptable systems which are fundamentally incompatible. Although efforts to develop standards for data representation are underway, little effort has been made to develop standards for linguistic software, and **software reusability** is virtually non-existent.

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press

As a result, there is a serious lack of generally usable tools to manipulate and analyze the text and speech corpora and collections that are now becoming widely available. Worse, there is enormous duplication of effort: it is not at all uncommon for researchers to develop tailor-made systems that replicate much of the functionality of other systems and in turn create programs that cannot be reused by others, and so on in an endless **software waste cycle**. The reusability of data is a much-discussed topic these days; similarly, software reusability is needed, to avoid the re-inventing of the wheel characteristic of much language-analytic research in the past three decades. (Veronis and Ide 1996) (Emphasis in the original.)

On a smaller scale, it is not unusual for the life-cycle of doctoral research in AI/NLP to be similarly wasteful, with software created during the research becoming unused fairly soon after completion (particularly at the original site when the student has moved on).

A framework which enabled relatively easy reuse of past work could significantly increase research productivity, and this was the rationale for the development of GATE, a General Architecture for Text Engineering, a system that promotes reuse via standardised language data storage, management and visualisation (Cunningham *et al.* 1997).

4.5 Collaborative research

If we think of NLP's theoretical diversity as a centrifugal force pushing researchers outwards from consensus, the EC and DARPA funding programme requirements that sites must collaborate is a a centripetal tendency pushing them back together again[8]. These requirements show no signs of changing: collaborative development of LE systems is here to stay for at least the immediate future.

4.6 Small groups, big systems

NLP and LE research groups are typically small: between 5 and 25 seems common, usually covering a large range of different subfields. This means that these groups are unable to bear large overheads in their development methods. Additionally, most personnel are not practising software engineers, or at least have little practical experience of engineering large systems. Together these social factors militate against the adoption of rigorous development methods in language processing labs. This is a significant problem, as Cowie and Lehnert point out (with reference to developing Information Extraction (IE) components):

Automation and speed-up of these components are critical for rapidly porting systems to new domains. The speed of the system is another critical factor in iterating to achieve high levels of accuracy through a fast development cycle. ... The idea that good software development always requires good software engineering is worth repeating in the IE context, because NLP researchers do not always think of software engineering as a critical NLP research weapon. (Cowie and Lehnert 1996)

More knowledge and use of software engineering methods would be valuable.

[8] This distinction was drawn in Gaizauskas *et al.* (1996).

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press

5 Summary

We began by defining LE as the discipline or act of engineering software systems that perform tasks involving processing human language. Both the construction process and its outputs should be measurable and predictable. The literature of the field relates to both application of relevant scientific results and a body of practice.

We then discussed the history and current characteristics of LE, concluding that it is indeed a new and distinct field, with its own priorities and concerns. Chief among these are:

- large-scale tasks of practical utility;
- measuring progress quantitatively relative to performance on examples of large-scale tasks;
- willingness to consider any technique, linguistically plausible or otherwise, in service of advances in performance;
- a growing realisation of the importance of software engineering in general, and reusability, robustness, efficiency and productivity in particular;
- a need to do all this in small groups, working collaboratively.

Acknowledgements

Thanks to Yorick Wilks, Kalina Bontcheva, Robert Dale and the anonymous reviewers for many useful comments on earlier drafts.

References

ACL (1997) *Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP-97)*. Association for Computational Linguistics.

Advanced Research Projects Agency (ARPA) (1996) *Proceedings of the TIPSTER Text Program (Phase II)*. San Mateo, CA: Morgan Kaufmann.

Amsler, R. A. (1980) Structure of the Merriam-Webster Pocket Dictionary. *Technical Report TR-164*, University of Texas at Austin.

Amsler, R. A. (1981) A taxonomy of English nouns and verbs. In *Proceedings of the 19th Annual Meeting of the Association for Computational Linguistics*, pp. 133–138. Stanford, CA.

Amsler, R. A. and White, J. S. (1979) Development of a Computational Methodology for Deriving Natural Language Semantic Structures via Analysis of Machine-readable Dictionaries. *Technical Report MCS77-01315*, NSF.

Bikel, D., Miller, S., Schwartz, R. and Weischedel, R. (1997) Nymble: a High-Performance Learning Name-finder. *Proceedings of the Fifth Conference on Applied Natural Language Processing*.

Binot, J. and Jensen, K. (1987) A semantic expert using an online standard dictionary. *Proceedings of the 10th International Joint Conference on Artificial Intelligence (IJCAI-87)*, pp. 709–714. Milan, Italy.

Bod, R. (1995) *Enriching Linguistics with Statistics: Performance Models of Natural Language*. Institute for Logic, Language and Computation, University of Amsterdam.

Bod, R. (1996) Two Questions about Data-Oriented Parsing. *Proceedings of the Fourth Workshop on Very Large Corpora*.

Boguraev, B. and Briscoe, T. (1987) Large Lexicons for Natural Language Processing: Exploring the Grammar Coding System of LDOCE. *Computational Linguistics*, **13**.

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press

Boguraev, B., Garigliano, R. and Tait, J. (1995) Editorial. *Natural Language Engineering*. **1**(1).
Boguraev, B. K., Briscoe, T., Carroll, J., Carter, D. and Grover, C. (1987) The derivation of a grammatically indexed lexicon from the Longman Dictionary of Contemporary English. *Proceedings of the 25th Annual Meeting of the ACL*, pp. 193–200. Stanford, CA.
Brill, E. (1992) A simple rule-based part of speech tagger. *Proceedings of the DARPA Speech and Natural Language Workshop*. Harriman, NY.
Brill, E. (1995) Transformation-Based Error-Driven Learning and Natural Language. *Computational Linguistics* **21**(4).
Brown, A. W. (1989) *Database Support for Software Engineering*. Kogan Page.
Bruce, R. and Guthrie, L. (1992) An Automatically Generated Semantic Hierarchy. *Proc. COLING-92*. Nantes, France.
Charniak, E. 1993. *Statistical Language Learning*. MIT Press.
Chodorow, M. S., Byrd, R. J. and Heidorn, G. E. (1985) Extracting Semantic Hierarchies from a Large On-Line Dictionary. *Proceedings of the 23rd Annual Meeting of the ACL*. pp. 299–304. Chicago, IL.
Church, K. and Mercer, R. L. (1993) Introduction. *Special issue on Computational Linguistics Using Large Corpora, Computational Linguistics*, **19**(1).
Church, K. and Rau, L. (1995) Commercial applications of natural language processing. *Commun. ACM* **38**(11).
Cowie, J. and Lehnert, W. (1996) Information Extraction. *Commun. ACM* **39**(1): 80–91.
Cowling, T. (1996) Personal Communication. Department Computer Science, University of Sheffield.
Crouch, R., Gaizauskas, R. J. and Netter, K. (1995) Interim Report of the Study Group on Assessment and Evaluation. *Technical report*, EAGLES project, Language Engineering Programme, European Commission.
Crystal, D. (1991) *A Dictionary of Linguistics and Phonetics. 3rd ed*. Oxford: Blackwell.
Cunningham, H. (1997) Information Extraction – a User Guide. *Research Memorandum CS-97-02*, Department of Computer Science, University of Sheffield, January. Available: `http://xxx.lanl.gov/ps/cmp-lg/9702006`.
Cunningham, H., Freeman, M. and Black, W. J. (1994) Software reuse, object-oriented frameworks and natural language processing. *New Methods in Language Processing (NeMLaP-1)*. Manchester.
Cunningham, H., Gaizauskas, R. G. and Wilks, Y. (1995) A General Architecture for Text Engineering (GATE) – a new approach to Language Engineering R&D. *Technical Report CS-95-21*, Department of Computer Science, University of Sheffield. Available: `http://xxx.lanl.gov/ps/cmp-lg/9601009`.
Cunningham, H., Humphreys, K., Gaizauskas, R. and Wilks, Y. (1997) Software Infrastructure for natural language processing. *Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP-97)*. Available: `http://xxx.lanl.gov/ps/9702005`.
Day, D., Robinson, P., Vilain, M. and Yeh, A. (1998) MITRE: Description of the *Alembic* system used for MUC-7. *Proceedings of the 7th Message Understanding Conference (MUC-7)*. San Mateo, CA: Morgan Kaufmann.
Dunning, T. (1993) Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics* **19**(1).
EC. (1996) *Telematics Applucations Programme, Language Engineering Project Directory*. European Commission Directorate General XIII, Luxembourg.
Ejerhed, E. and Dagan, I. (eds) (1996) *Proceedings of the 4th Workshop on Very Large Corpora*. Association for Computational Linguistics.
Farwell, D. and Wilks, Y. (1989) ULTRA – a machine translation system. Memoranda in Computer and Cognitive Science NM1989, Computing Research Lab, New Mexico State University.
Gaizauskas, R. (1995) Investigations into the grammar underlying the Penn Treebank II. *Research Memorandum CS-95-25*, Department of Computer Science, Univeristy of Sheffield.

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press

Gaizauskas, R., Wakao, T., Humphreys, K., Cunningham, H. and Wilks, Y. (1995) Description of the LaSIE system as used for MUC-6. *Proceedings of the 6th Message Understanding Conference (MUC-6)*. San Mateo, CA: Morgan Kaufmann.

Gaizauskas, R. G. (1997) A review of *Evaluating Natural Language Processing Systems*, Sparck-Jones and Galliers, 1996. *J. Natural Language Engineering* **3**.

Gaizauskas, R. G., Cunningham, H., Wilks, Y., Rodgers, P. and Humphreys, K. (1996) GATE – an Environment to Support Research and Development in Natural Language Engineering. *Proceedings 8th IEEE International Conference on Tools with Artificial Intelligence (ICTAI-96)*, Toulouse, France.

Gamma, E., Helm, R., Johnson, R. and Vlissides, J. (1995) *Design Patterns – Elements of Reusable Object-Oriented Software*. Addison-Wesley.

Gazdar, G. (1996) Paradigm merger in natural language processing. In: R. Milner and I. Wand (eds), *Computing Tomorrow: Future Research Directions in Computer Science*, pp. 88–109. Cambridge: Cambridge University Press.

Grishman, R. (1997) TIPSTER Architecture Design Document Version 2.3. *Technical report*, DARPA. Available: `http://www.tipster.org/`.

Grishman, R. and Sundheim, B. (1996) Message understanding conference – 6: A brief history. *Proceedings 16th International Conference on Computational Linguistics*, Copenhagen, Denmark.

Guthrie, L., Slator, B., Wilks, Y. and Bruce, R. (1990) Is there content in empty heads? *Proceedings of the 13th International Conference on Computational Linguistics (COLING-90), number 3*, pp. 138–143. Helsinki, Finland.

Hobbs, J. R. (1993) The generic information extraction system. *Proceedings 5th Message Understanding Conference (MUC-5)*. San Mateo, CA: Morgan Kaufmann.

Horowitz, P. and Hill, W. (1989) *The Art of Electronics*. Cambridge: Cambridge University Press.

Humphreys, K., Gaizauskas, R., Azzam, S., Huyck, C., Mitchell, B., Cunningham, H. and Wilks, Y. (1998) Description of the LaSIE system as used for MUC-7. *Proceedings 7th Message Understanding Conference (MUC-7)*. San Mateo, CA: Morgan Kaufmann.

Jacobs, P. S. (ed) (1992) *Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval*. Hillsdale, NJ: Lawrence Erlbaum.

Karov, Y. and Edelman, S. (1996) Learning similarity-based word sense disambiguation from sparse data. In: E. Ejerhed and I. Dagan (eds), *Proceedings 4th Workshop on Very Large Corpora*.

Kay, M. (1997*a*) It's Still the Proper Place. *Machine Translation* **12**: 3–23.

Kay, M. (1997*b*) The proper place of men and machines in language translation. *Machine Translation* **12**: 3–23.

Kuno, S. and Oettinger, A. (1962) Multiple path syntactic analyser. *Information Processing 1962, Proceedings IFIP*. Amsterdam: North-Holland.

Lenat, D., Miller, G. and Yokoi, T. (1995) CYC, WordNet, and EDR: Critiques and Responses. *Commun. ACM* **38**(11).

Magerman, D. (1994) *Natural Language Parsing as Statistical Pattern Recognition*. PhD thesis, Department of Computer Science, Stanford University.

Magerman, D. (1995) Statistcal decision tree models for parsing. *Proceedings of ACL*.

Makins, M. (ed) (1991) *Collins English Dictionary, 3rd ed*. Harper Collins.

Marcus, M.P., Santorini, B. and Marcinkiewicz, M. A. (1993) Building a large annotated corpus of english: The penn treebank. *Computational Linguistics* **19**(2): 313–330.

Markowitz, J., Ahlswede, T. and Evens, M. (1986) Semantically significant patterns in dictionary definitions. *Proceedings 24th Annual Meeting of the Association for Computational Linguistics*, pp. 112–119. New York.

Michiels, A., Mullenders, J. and Noel, J. (1980) Exploiting a Large data Base by Longman. *Proceedings 8th International Conference on Computational Linguistics (COLING-80)*, pp. 374–382. Tokyo, Japan.

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press

Michiels, A. and Noel, J. (1982) Approaches to Thesaurus Production. *Proceedings of the 9th International Conference on Computational Linguistics (COLING-82)*, pp. 227–232. Prague, Czechoslovakia.

Miller, E., Kado, M., Hirakwa, M. and Ichikawa, T. (1995) HI-VISUAL as a User-Customizable Visual Programming Environment. *Proceedings VL'95 11th International IEEE Symposium on Visual Languages, Darmstadt*. IEEE Press.

Miller, S., Crystal, M., Fox, H., Ramshaw, L., Schwartz, R., Stone, R. and Weischedel, R. (1998) Description of the BBN System Used for MUC-7. *Proceedings 7th Message Understanding Conference (MUC-7)*. San Mateo, CA: Morgan Kaufmann.

Miller, G. A. (ed) (1990) WordNet: An on-line lexical database. *Int. J. Lexicography*, **3**(4): 235–312.

Mitkov, R. (1995) Language Engineering on the Highway: New Perspectives for the Multilingual Society. *Proceedings of NLPRS*.

Mitkov, R. (1996) Language Engineering: towards a clearer picture. *Proceedings of the International Conference on Mathematical Linguistics (ICML '96)*.

Press, W., Teukolsky, S., Vetterling, W. and Flannery, B. (1995) *Numerical Recipes in C – the Art of Scientific Computing*. Cambridge: Cambridge University Press.

Resnik, P. (1993) *Selection and Information: A Class-Based Approach to Lexical Relationships*. PhD thesis, Institute for Research in Cognitive Science, University of Pennsylvania.

Shaw, M. and Garlan, D. (1996) *Software Architecture*. New York: Prentice Hall.

Simon, H. A. (1995) Artificial intelligence: and empirical science. *Aritificial Intelligence* **77**(1).

Sparck-Jones, K. (1994) Towards better NLP system evaluation. *Proceedings of the Second ARPA Workshop on Human Language Technology*. San Mateo, CA: Morgan Kaufmann.

Sparck-Jones, K. and Galliers, J. (1996) *Evaluating Natural Language Processing Systems*. Springer-Verlag.

Thompson, H. (1985) Natural language processing: a critical analysis of the structure of the field, with some implications for parsing. In: K. Sparck-Jones and Y. Wilks (eds), *Automatic Natural Language Parsing*. Chichester: Ellis Horwood.

van der Linden, P. (1994) *Expert C Programming – Deep C Secrets*. New York: Prentice Hall.

Veronis, J. and Ide, N. (1996) Considerations for the Reusability of Linguistic Software. *Technical report*, EAGLES, April. Available: `http://www.lpl.univ-aix.fr/projects/multext/LSD/LSD1.main.html/`.

Vilain, M. and Day, D. (1996) Finite-state phrase parsing by rule sequences. *Proceedings of COLING-96*.

Walker, D. E. and Amsler, R. A. (1986) The use of machine-readable dictionaries in sublanguage analysis. In: R. Grishman and R. Kittredge (eds), *Analyzing Language in Restricted Domains*. Hillsdale, NJ: Lawrence Erlbaum.

Wilks, Y. (1994) Developments in MT Resaerch in the US. *Proceedings of ASLIB* **46**(4).

Wilks, Y. (1996) Natural Language Processing, guest editorial. *Commun. ACM* **39**(1).

Wilks, Y., Fass, D. C., Guo, C., McDonald, J. E., Plate, T. and Slator, B. M. (1987) A tractable machine dictionary as a resource for computational semantics. In: B. Boguraev and T. Briscoe (eds), *Computational Lexicography for Natural Language Processing*. Harlow, UK: Longman.

Wilks, Y., Guthrie, L. and Slator, B. (1996) *Electric Words*. Cambridge, MA: MIT Press.

Winograd, T. (1972) *Understanding Natural Language*. New York: Academic Press.

Yokoi, T. (1995) The EDR Electronic Dictionary. *Commun. ACM* **38**(11).

Yourdon, E. (1996) *The Rise and Resurrection of the American Programmer*. New York: Prentice Hall.

Zipf, G. K. (1935) *The Psycho-Biology of Language*. Boston: Houghton Mifflin.

https://doi.org/10.1017/S1351324999002144 Published online by Cambridge University Press