[*First International Conference on Language Resources & Evaluation*, Granada, Spain, 28-30 May 1998]

Issues in Natural Language Systems Evaluation

Margaret King
ETI/TIM (ex-ISSCO)
University of Geneva
54 rte des Acacias, CH-1227, Carouge, Geneva.
[Margaret.King@issco.unige.ch]

Bente Maegaard
Center for Sprogteknologi
Njalsgade 80, 2300 Copenhagen S, DENMARK
[bente@cst.ku.dk]

Abstract

The first step in developing a thorough evaluation methodology consists in description, structuring and standardisation of the elements of an evaluation. These elements are the system to be evaluated, the user profile of the intended user, and the metrics to be applied. The next step is to develop as far as possible computer tools to support the testing and evaluation. In this paper we describe the methodology developed by the EAGLES and TEMAA projects which ran under the European Commission's LRE and LE programmes. The methodology is sufficiently general to be used for any type of evaluation of language technology systems or products. In the oral presentation, we shall also comment on the outcome of the LREC workshop on evaluation infrastructure.

Background

Although a number of evaluations had been previously carried out, mainly of machine translation systems and mainly on behalf of individual potential customers of the system(s) being evaluated, it was the ARPA/DARPA series of evaluation guided research programmes which in the late 1980's and 1990's brought the importance of evaluation and evaluation techniques forcibly to the attention of the scientific community on both sides of the Atlantic. Substantial funding was made available for the development of evaluation methods and materials and for central organisation of comparative evaluation of systems performing in areas such as document retrieval Harman 1995a, 1995b), fact extraction (Cowie and Lehnert, 1996) and machine translation (White et al, 1994).

The main mission of the programmes was, to quote the co-ordinator of the machine translation programme 'to advance the core technology' rather than to take into account the needs of any specific customer or set of customers. The evaluation exercises therefore took the form of what were meant to be friendly competitions between participating systems and concentrated on evaluation of the functionalities of the systems in question, divorced from any context of use.

Interestingly, although several of these programmes were very successful, leading to a set of commonly accepted evaluation measures and to genuinely friendly sharing of data and even software. (Sparck Jones, 1998), others encountered a great deal of difficulty in defining acceptable metrics (White et al, 1994). (King, 1996, 1997) discuss some of the reasons why this should be so. In general, the successful programmes continue, whilst those where no consensus on appropriate metrics could be found have been dropped.

The EAGLES initiative, launched by the European Commission in 1993, included a working group on evaluation. In order to avoid duplication of effort and benefiting from the American experience, the remit of the EAGLES evaluation work was to aim at developing a general methodology for the design of evaluations which would be applicable to all evaluations in the field of language engineering, whether they be of systems, of products or even of projects. Taking users into consideration was considered to be a central part of the task. There is an obvious tension between defining a general methodology and taking into account user needs. This led us to work in terms of classes of typical users, where each class was considered to have its own typical needs.

EAGLES is the acronym for Expert Advisory Groups in Language Engineering *Standards*: the methodology to be developed was seen as eventually leading to the development of a standard for evaluation.

A General Methodology for Evaluation Design

A natural place to start in looking for a standard for evaluation methodology is with the work of ISO. In their 9000 series on software production, ISO provides a standard for the definition of quality and guidelines for an evaluation procedure (ISO, 1991). This proved a very good starting point for elaborating an evaluation methodology. An early EAGLES decision was to distinguish between two basic types of evaluation: 1) *adequacy evaluation,* where the performance of a product is seen from the point of view of the user/purchaser, and 2) *progress evaluation,* where development progress is measured against plans and previous versions. The ISO documents are mostly concerned with progress evaluation, in that they are aimed at software production environments, although a claim is made that the standard and guidelines should be more generally applicable. In contradistinction, it was decided within EAGLES to focus on adequacy evaluation, thereby testing the applicability of the ISO standard to a different kind of evaluation and simultaneously complementing the ARPA/DARPA emphasis on evaluating functionality divorced from context of use.

This paper reports both on work undertaken within the EAGLES initiative and in the associated TEMAA project. A practical test bed was needed to validate the theoretical framework being constructed within EAGLES. Thus, the emphasis of the first round of EAGLES work, between 1993 and 1995, was on the adequacy evaluation of commercial or near to market language engineering products, although it is important to bear in mind that the intention was not to carry out specific evaluations but to produce a general methodology from which specific evaluations could be derived. In TEMAA, the EAGLES ideas were further developed and formalised, and some implementation took place as well.

The ISO standard (ISO 9126) concerning the evaluation of software products was, as mentioned above, obviously potentially relevant to the work. Attempting to apply that standard to the particular domain of language engineering software and to make the standard more concrete raised a number of important issues, which are discussed in detail below.

The link with ISO work has been reinforced during the second round of EAGLES which is currently underway. A new draft of ISO 9126 came to the notice of the group after the first round of work had finished. The new draft was presented at the EAGLES Evaluation workshop of November 1997 by its technical editor (Bevan, 1998a) who has subsequently agreed to become a member of the EAGLES Evaluation scientific committee. The new draft differs from the 1991 standard in being much more detailed, in placing increased emphasis on the notion of 'quality in use' and in according a significantly larger importance to the development of metrics. We are happy that although ISO and EAGLES work were developing independently until recently, a strong convergence of ideas can be observed. The new draft is mentioned below as ISO 98 where relevant.

Elements of an evaluation methodology

The following elements of an evaluation methodology can be isolated:

- Descriptions of
    - Users (including the context of use)
    - Systems
    - Measures
    - Methods
- Test materials

and, if possible

- Computer tools for automatic testing and report generation.

In EAGLES descriptions of systems and users are made in terms of features and values associated with those features. The ISO 9126 quality characteristics are defined in the standard as sets of features, and also correspond to features in EAGLES work.

Quality Characteristics

The ISO 9126 standard sets out six quality characteristics, which between them are meant to capture the major components that make up the quality of a piece of software: functionality, reliability, usability efficiency, maintainability and portability.

The limited number of characteristics as well as their intended application to any kind of software necessarily means that the characteristics are very general in their nature. The EAGLES group eventually added a seventh characteristic, customisability, on the grounds that it was extremely unlikely that any language engineering product bought off the shelf would satisfy any user's needs without some modification: even a humble spelling checker will typically require words to be added to its dictionary.

More important, though, was the realisation that whether exactly the right names or the right definitions had been chosen for the quality characteristics was relatively unimportant. Their primary purpose is to serve as a skeleton framework, which, for any specific evaluation has to be filled out with sub-characteristics and sub-sub-characteristics in a hierarchy whose depth is not predetermined. The choice and organisation of sub-characteristics constitutes a definition of quality specific to a product or class of products.

This view of the quality characteristics is reinforced by the new version of ISO 9126, which suggests sub-qualities for each of the quality characteristics, and, interestingly, includes "changeability" as a sub-characteristic of maintainability, adding the note "If the software is to be modified by the end user, changeability may be a pre-requisite for operability" - thus neatly solving the EAGLES original customisability problem.

To make all this a little more concrete, let us take as an example the working out of quality characteristics for spelling checkers in the TEMAA project. First, functionality, which was assumed to have three sub-characteristics

- recall (the degree to which the checker accepts all the valid words of the language)
- precision (the degree to which the checker rejects all valid words)
- suggestion adequacy (in case of invalid words, does the checker provide correct suggestions)

Space will not allow us to give all the details of how each of these is in its turn broken down. As an example, precision was decomposed into

- typographical errors
- spelling errors
- medium related errors

each of which was in turn broken down again. For example, typographical errors was broken down into

- insertion
- deletion
- substitution
- transposition

which were again broken down into, for example,

- doubling of a letter
- deletion of a letter

• interchanging of two letters and so on.

It can readily be seen that the hierarchy of features associated with any given quality characteristic is in principle infinite. It is also easy to appreciate that the construction of an appropriate feature structure for a particular class of systems is a lengthy and fastidious task which also requires a great deal of thought. Once done, however, the feature structure can be re-used for any other systems that fall into the same general class.

Description of Users

Users are described basically in the same way as systems, in terms of features and their values. Indeed, it is basic to the methodology that descriptions of systems and of users make use of the same set of features, in order that systems and users can be directly compared (preferably automatically) in order to find out whether a system will fit a user's needs. A user has a certain background, works in a certain context and most importantly, a user has a *task* to perform within that context. The description of a particular user also gives desired values or thresholds for the relevant (sub)-features and weightings for each (sub)-feature. This description is called a *user profile,* and directly reflects the specific requirements of a particular user or class of users. For example, let us imagine a user whose texts never make use of acronyms. For him, the sub-feature of the recall feature which covers the spelling checker's treatment of acronyms is irrelevant. A weighting given so that sub-feature will reflect this and will ensure that acronym treatment is ignored when evaluating spelling checkers for this user. It is an important feature of the EAGLES/TEMAA framework to place the user and his task in such a central role.

It was mentioned above that the description of a user uses the same features as the system description in order to allow a comparison of systems and users. However, should it turn out that a certain aspect of a user profile cannot be matched by a feature of the system description, this will be a clear indication that the system itself or the system description lacks this feature, i.e. the user needs a quality characteristic which the system does not possess.

Formalisation

Perhaps the most important extension of the ISO standard was to formalise these descriptions of products and users in terms of the sort of feature structures familiar to computational linguists since the early work on unification grammars (Shieber 1984), i.e. in terms of organised structures of attribute value pairs where every value may itself be a feature structure.

There are two reasons for attempting formalisation. First of all, formalised descriptions are more easily standardised and it is easier to check conformity. Secondly, formalisation makes automation possible and automation makes for more reliable metrics. In particular automation allows for much larger text samples to be tested and therefore provides better and more reliable results. Of course, not all characteristics of a system and its behaviour, nor all characteristics of a user role, can be formalised, but experience shows that with ingenuity quite good results may be obtained.

The description of user, system, method and measure should therefore be formalised as far as possible. Methods for measurement (see below) should be both formalised and where possible automated.

Can Quality be Measured?

The feature structure created for a class of products (for example spelling checkers, grammar checkers or translation memory systems) can be seen as a model of quality for that class of systems.

This raises another important issue: quality is not a single, unchanging characteristic of software: it means different things to different people at different times of a product's life cycle. The EAGLES work was concerned with evaluation of whether a product would meet the needs of a particular class of users, and the formal descriptions therefore took into account the purpose of the evaluation. If the purpose of the evaluation was to discover whether a product satisfied a set of requirement specifications, the quality model would be different and different attributes would be chosen to play a part in the formalisation. However, the basic modelling tool would remain unchanged.

Remember too that a particular user's requirements can be modelled using the same tools. Thus, his particular needs are reflected by the relative importance given to specific attributes within the quality model, or by threshold values specified for certain attribute values. So, quality can be measured only with respect to a certain user and a certain task - quality as an abstract general notion does not exist in this model. In terms of the new ISO draft, we are aiming to predict quality in use for a specific user or class of users.

Computability

Once a model has been formalised it is possible to create computer programs making use of that model. As mentioned above, the TEMAA project took this idea much further and created a prototype *Evaluator's Workbench.* Essential elements of the workbench were descriptions of specific products, of classes of products and of typical users, all in terms of the feature structures described briefly above. For any specific product, values of the attributes in the quality model could be determined and compared to the user's own needs.

If the system being evaluated is a spelling checker - as was the case for the TEMAA project - the user tasks or roles can be described as the *writer,* the *end user,* the *reader,* the *customer.* A writer in turn may have characteristics such as *technical writer, native speaker, foreign language speaker.* The reason that we need to distinguish these different classes of users is their different language competence which leads to different importance of, for example, the spelling checker's suggestion adequacy (how often is the first suggestion for a wrongly spelled word correct? how often is the correct word among those suggested at all?). A native speaker may, in the limit, be satisfied to have an error brought to his attention; a foreign language speaker probably needs the correct solution to be made available to him. And it may even be that his knowledge of the language is so weak that it is highly desirable that the correct solution be the first one offered.

Formalising the description of the class of spelling checkers requires construction of a feature structure which embraces all the possible needs of all users. Description of a particular system is limited to that subset of features present in that system. For any particular system, the feature description bottoms out in values, which must be inserted either manually (in the case, for example, of the price of the system) or calculated automatically (for example, through automatic measurement of a spelling checker's lexical coverage). Measurement often relies on the availability of test material, whose nature depends on the definition of the metric associated with obtaining the value and which can again be expensive to construct. To give just a few examples of test materials produced in the TEMAA project:

1. A spelling checker is supposed to flag only errors, so at least all common words must be known to the system, in order for it not to wrongly flag correct words. Therefore, a list of commonly used words was constructed.
2. A spelling checker should find all spelling mistakes in a text. In order to automate the testing of this feature, lists of correct words were corrupted following rules of misspelling and the spelling checker was run on the list of misspelled words, automatically comparing the result with the original correct word.

Testing of spelling checkers can be to a large extent automated by defining metrics which carry out automatic testing based on test materials of the sort described above. A session using the spelling checker to check the test materials is simulated automatically and the results made available to the workbench. The Evaluator's Workbench then compares the result of the testing with the user profile and computes a report which gives the result of the evaluation of a specific spelling checker for a given user profile, i.e. when different spelling checkers have been tested, this report gives the input for the user to choose the most adequate system for a certain task/class of tasks.

Metrics and validity

A metric is associated with each attribute value pair which determines how the value is to be determined in a specific case. For example, one metric associated with the lexical coverage attribute of spelling checkers measured what percentage of a pre-determined core vocabulary for the language in question was covered by the spelling checker's dictionary.

Attributes in the TEMAA/EAGLES model are typed by the type of value they can take. Some values are simply facts, and can be determined, for example, by looking in the relevant literature. Other values involve human judgement. Yet others are determined by carrying out some kind of test on the product being evaluated. With this latter type, it is sometimes possible to automate the testing, as was done for a number of attributes of spelling checkers in the TEMAA Evaluator's Workbench.

Metrics should be both valid and reliable, that is, they should test what they are supposed to test and repeated testing should produce consistent results. EAGLES distinguished two types of validity, internal and external validity. Internal validity is inherent to the metric itself A classic example is that of reading tests based on use of specifically chosen vocabulary. The validity of the test relies only on whether the vocabulary has been correctly chosen. External validity relies on correlation with some external criterion. A classic example here are the criteria used by insurance companies to determine the size of life insurance premiums. Factors such as the existence of hereditary disease in the family, whether the applicant has undergone certain types of surgery etc. are held to have a strong correlation with life expectancy. Unfortunately, the new ISO standard uses the same terms to distinguish internal and external metrics. Internal metrics are characteristics of the software, such as, for example, the number of lines of code or the number of sub-processes. External metrics are characteristics associated with the software in use, for example whether it delivers the correct results. As can be seen, there is a resemblance between the EAGLES use of the terms and the ISO use, but one task to be done is to clarify the terminology issue.

Determining validity and reliability is a far more delicate issue than it may at first seem. The EAGLES work drew heavily on work in the social sciences on these issues, and the new ISO draft has considered metrics of sufficient importance to deserve separate documents. Choice and validation of metrics merits considerable further work.

Application to other areas

Above, spelling checkers have been used as the example of a language technology tool. Spelling checkers are probably the simplest possible language technology So, the interesting question is: Can this methodology be used also for more complicated systems and user roles, e.g. for grammar checkers, for information retrieval message understanding, translation tools, machine translation?

Translation is different from spelling in that whereas only one correct spelling of a word exists (or in some cases there are two alternative approved spellings), there will normally be more than one correct translation of a sentence. Also, the borderline between correct and incorrect translation is not so clear-cut, which means that it will be more difficult to set up automatic testing as described above. With this caveat though, much can be achieved by formalising the descriptions of systems and users, and there is no obvious reason why the general methodology should not be applicable to other applications. Some preliminary work on validating this has already been done.

The EAGLES subgroup on translation tools studied translation memories in particular. Feature check list examples were drawn up like the following, which is part of the top level feature description relevant to translation memory update and maintenance:

- Carrying out alignment
- Importing an aligned source language (SL)- and target language (TL)-segment into the translation memory database
- Adding an SL-segment and its translation to a translation memory (TM) while translating in TM mode,

• Modifying existing contents of TMs (apart from adding/importing).

As always, each of these questions is broken down into a number of more detailed sub-features, cf. (EAGLES, 1996).

In the same context of applying EAGLES methodology to the evaluation of translation memory systems, user profiles are described first by the *dimensions of translation*, i.e. quantity of translated text, text type, text domain, languages involved, translation quality, language policy of the organisation etc., then by *typical user profiles* such as freelance translator, small translation company, large translation company, translation department in small or large organisation, bilingual organisation etc. Typical user profiles in combination with dimensions of translation then provides input for the user requirements for the translation tool to be evaluated. As before, features will be of different types depending on the type of value they take. In the case of translation memory systems a large part of the evaluation will be factual, i.e. values are obtained by answering yes or no to questions like 'can the memory be edited ?' In this case no testing involving large data sets will be involved. Other metrics involve the building of test data sets and consequently, especially given the nature of translation memory systems, may turn out to imply a rather demanding job.

This preliminary work tends to confirm that the EAGLES methodology can be applied to a wide range of language engineering products. Together with the prototyping work in TEMAA and other prototyping work done by the LRC in Dublin for the localisation industry, it has also led to the definition of a new project, TransRouter, which intends to develop a prototype translation router, a decision aid which will support a translation manager in deciding on whether a translation project should best be handled by human translation, translation using a translation memory system, by machine translation and so on.

Experimental design

The first round of EAGLES and TEMAA work concentrated very heavily on the functionality characteristic, on the formalisation of the quality model and on the construction of tests which could be automatically administered. There was very little work on attributes of type judgement. In consequence, it was possible to avoid or at least to play down important issues of experiment design, such as sampling techniques, avoidance of bias and so on.

The second round of EAGLES evaluation work started early in 1997, and whilst primarily intended as a dissemination and consolidation effort, nonetheless hopes to go further with some issues that time and funding constraints prevented the first round of EAGLES work from picking up. One of these areas is to pick up on standard work on experiment design and make it available to the evaluation community, at least in the form of providing appropriate bibliographical references The same issue was discussed at length in the November workshop.

Usability and Quality in Use

Concentrating on functionality in the first round seemed justified both because providing appropriate functionalities is a necessary condition for a product to be acceptable and because it is in functionality that language engineering products are likely to be most different from other kinds of software. Furthermore, it might be expected that general work on software evaluation would come up with attributes and metrics for the reliability, efficiency, portability and maintainability characteristics of the ISO model which would be directly applicable to language engineering products. This leaves the usability characteristic as one deserving more attention in the second round of EAGLES work. A major part of the EAGLES workshop held in November of 1997 was devoted to this question, bringing us into contact with work in the INUSE project, with ISO work related to user centred design and to the notion of "quality in use".

Quality in use is defined in the new ISO 9126 standard as "the user's view of the quality of a system containing software, and is measured in terms of the result of using the software, rather than the properties of the software itself. A further note adds that "quality in use is an external measure of the combination of functionality, usability and efficiency".

Measuring usability is of importance to system manufacturers as well as to system users. For the former, usability ensures market share, for the latter it helps to improve both efficiency and job satisfaction, both being reflected directly in cost trimming. Well known techniques for measuring usability are essentially based on observing representative users in the process of using the system. This procedure clearly again raises issues of experimental design. Techniques and related questions were a major topic of discussion at the November workshop (Bevan, 1998b).

The Future

The future, then, holds many promising areas to be explored. Those who would like to join us in the exploration can do so by visiting our web site, at http://www.cst.ku.dk/projects/eagles2.html, where we intend not only to collect together basic reference material and resources, but to host an on-going discussion forum on evaluation and issues related to it. We also plan to hold a second workshop in the summer of 1998. Information about that too will soon be available at the web site.

Acknowledgements

All of the work in EAGLES and TEMAA is a collective effort. We are particularly indebted to the other contributors to the final TEMAA report: Louis des Tombe, Shona Douglas, Steven Krauwer, Sandra Manzi, Patrizia Paggio, Gurli Rohde, Merle Tenney, Nancy Underwood, and to all those, too numerous to mention here, who have contributed so generously to EAGLES work.

Bibliographical References

Bevan, N.(1998a). ISO 9126. Presentation at the EAGLES Evaluation workshop, Brussels, November 1997. Report available at http://www.cst.ku.dk/projects/eagles2.html

Bevan, N.(1998b). User centred design : INUSE and RESPECT. Presentation at the EAGLES Evaluation workshop, Brussels, November 1997. Report available at http://www.cst.ku.dk/projects/eagles2.html

Cowie, J. and Lehnert, W. (1996).Information Extraction. In Wilks, Y., (Ed.) Special Edition of CACM on Natural Language Processing, vol. 39, 1, 73-80.

EAGLES (1996) EAGLES Evaluation of Natural Language Processing Systems, Center for Sprogteknologi, Copenhagen. Also available at http ://issco-www.unige.ch/projects/ewg96/ewg96.html

Harmann, D.K. (1995a). Overview of the Third Text Retrieval Conference. National Institute of Standards and Technology Special Publication 500-225, Gaithersburg, MD.

Harmann, D.K. (1995b). Special Issue on the Second Text Retrieval Conference. In Information. Processing and Management, 31, 3 1995, 269-448.

Hutchins, J. (1997): Evaluation of Machine Translation and Translation Tools. In: Varile, N., A. Zampolli (Eds.): *Survey of the State-of-the-Art in Human Language Technology,* Cambridge University Press.

King, M. (1996). Evaluating Natural Language Processing Systems. In: Wilks, Y., (Ed.) Special Edition of CACM on Natural Language Processing, vol. 39, 1, 73-80.

King, M. (1997) Evaluating Translation. In: Hauenschild, C. and S. Heizmann (Eds.) *Machine Translation and Translation Theory.* Mouton de Gruyter.

Lehrberger, J. and L. Bourbeau (1988): *Machine Translation: linguistic characteristics of MT systems and general methodology of evaluation.* John Benjamins, Amsterdam, Philadelphia.

ISO (1991) ISO 9126 *Quality characteristics and guidelines for their use,* ISO, Geneva.

Maegaard, B. (1997): Evaluation of Language Tools. In: *Translating and the Computer 19,* Aslib, London.

Nomura, H. and H. Isahara: JEIDA's criteria on machine translation evaluation. In: *Proceedings of the International Symposium on Natural Language Understanding and AI,* Kyushu Institute of Technology, 1992.

Shieber, S.M. (1984). The design of a computer language for linguistic information. In Proceedings of Coling84. ACL, 362-366.

Sparck Jones(1998) User centred design : Coherent Approaches to Evaluation. Presentation at the EAGLES Evaluation workshop, Brussels, November 1997. Report available at http://www.cst.ku.dk/projects/eagles2.html

TEMAA Final report, Center for Sprogteknologi, Copenhagen, 1996.

White, J.S., O'Connell, T and O'Mara, F.E. (1994). The ARPA MT Evaluation Methodologies: Evolution, Lessons and Further Approaches. In *Translation technology for Crossing the Language Barrier.* Proceedings of the First Conference of the Association for Machine Translation in the Americas., Columbia, MD.