The MATE Workbench – A Tool in Support of Spoken Dialogue Annotation and Information Extraction

Laila Dybkjær and Niels Ole Bernsen

Natural Interactive Systems Laboratory
Science Park 10, 5230 Odense M, Denmark
Email: laila@nis.sdu.dk, nob@nis.sdu.dk
Tel: +45 65 50 35 53, +45 65 50 35 44, Fax: +45 63 15 72 24

ABSTRACT

The increasing variety and sophistication of spoken language dialogue systems (SLDSs) emphasises the need for tools in support of their development and evaluation as well as for appropriate evaluation criteria. In this paper we describe how the MATE workbench can be used during SLDSs development to efficiently produce corpus-based information on SLDSs and their components. The information retrieved from the annotated corpora can be used for evaluation purposes and provide important directions for further development. Examples are drawn from dialogue management and human factors of SLDSs.

1. INTRODUCTION

The increasing variety and sophistication of SLDSs emphasises the need for development support tools. Such tools include not only development platforms, developers' kits, and plug-and-play components, such as speech recognisers and synthesisers. Tools are also needed for handling data on vocabulary, grammar and other linguistic aspects, on users' dialogue behaviour, etc. The more innovative and sophisticated the SLDSs are that we want to build, the more detailed knowledge is required on, e.g., the dialogue acts which occur or users' misunderstandings of system utterances. The collection of such information is time consuming, requiring the recording of spoken dialogues between people and a, possibly simulated, system, transcription of the dialogues, markup of the phenomena of interest in the corpus, and, finally, systematic extraction and analysis of information on those phenomena. We need tools which can facilitate this entire process, making it faster, more cost-effective and able to produce re-usable data.

The MATE project (http://mate.nis.sdu.dk) was launched in early 1998 in response to the need for standards and tools in support of annotation and exploitation of spoken language resources. This paper first summarises the results of MATE, including a brief description of the MATE workbench. We then discuss and illustrate in detail how the MATE workbench can be used during SLDSs development and evaluation to produce corpus-based information on important issues, such as dialogue design adequacy and users' communication strategies. Examples are drawn from work done elsewhere on dialogue management and human factors of SLDSs.

2. MATE RESULTS

2.1 Theoretical Issues

The approach adopted in MATE was to begin by reviewing existing coding schemes for selected annotation levels. The annotation levels are prosody, (morpho-)syntax, co-reference, dialogue acts, and communication problems, as well as cross-level issues concerning interactions among the levels. The selected levels are very different and thus present a broad range of problems for the coding tool builder. Our assumption was that if a markup framework and a tool set could be created which would work for those levels in a consistent and homogeneous manner, then the framework and the tool set are likely to generalise to other annotation levels as well.

The state-of-the-art review analyses more than 60 coding schemes from projects world-wide and belonging to the MATE annotation levels. For each scheme the following information items were requested: coding book, if any, the number of annotators who had worked with it, the number of annotated dialogues/segments/utterances, evaluation results, if any, the underlying task, a list of annotated phenomena, and the markup language used. Annotation examples are provided as well. The results are presented in [9].

Based on the state-of-the-art review, MATE has developed a markup framework proposal for a standard for the definition and representation of markup for spoken dialogue corpora at multiple levels [5]. The framework is a conceptual model which basically describes how files are structured, i.a. to allow for multi-level annotation, how tag sets are represented in terms of elements and attributes, and how to provide essential information on markup, semantics, coding purpose, relations to other codings, etc. by using coding modules. The coding module is the core concept of the framework. It extends and formalises the concept of a coding scheme. Roughly speaking, a coding module includes or describes everything that is needed in order to perform a certain kind of markup of spoken dialogue corpora.

For each of the annotation levels addressed, MATE has selected one or several of the most commonly used state-of-the-art coding schemes and turned these into MATE best-practice coding schemes described according to the markup framework [10]. This worked out nicely, ensuring a common and user-friendly approach across annotation levels. It is easy for the annotator to work on multiple coding schemes and/or levels, because use of the same set of software tools is facilitated and the same interface look-and-feel provided independently of the annotation level in question.

2.2 The MATE Workbench

Along with the theoretical work, a workbench has been developed which supports the MATE markup framework and all coding schemes expressed in terms of the framework. The MATE workbench is easy to use, enables annotation, information extraction including statistics, and import from, and export to, different file formats. The workbench is implemented in Java to make it platform-independent. The workbench has a modular architecture which facilitates updates and addition of new tools and annotation schemes by its users. XML is used for the internal coding file representation. The source code of the workbench is available under an open source license, see http://mate.nis.sdu.dk where also a discussion forum has been started with the aim of addressing identified problems and pointing to additions made by individual workbench users and which may be of benefit to other users. The MATE developers continue to add and improve functionality as reflected in the currently most recent version available at the MATE web site.

To provide input to the workbench specification, a number of existing annotation tools were analysed, such as the Alembic workbench and Nb (for both, see [8]). Moreover, the MATE markup framework and the analysis, adaptation and inclusion in the workbench of existing state-of-the-art coding schemes have served as sources of input to workbench functionality and usability design and development. At the time of writing, the following functionalities are available in the MATE workbench:

The MATE best practice coding modules are included as examples which means that there is immediate support for coding at five annotation levels.

There is no transcription module in the workbench. However, a converter from Transcriber format (http://www.etca.fr/CTA/gip/Projets/Transcriber/) to MATE format enables transcriptions made using Transcriber to be annotated using the MATE workbench.

A coding module editor enables users to add new coding modules for already existing levels as well as for new levels (including the transcription level).

Style sheets are used for the visual presentation of corpora which are being annotated using a particular coding module. Phenomena of interest in the corpus may be shown in, e.g., a certain colour or in boxes. Style sheets can be edited and new ones created using the MATE workbench.

During annotation, an audio tool enables users to listen to speech files and have them displayed as a waveform.

Import of files from XLabels and BAS Partitur to XML format is supported. Other converters can easily be added. Export to file formats other than XML can be achieved by using style sheets. For example, information extracted by the query tool may be exported to HTML to serve as input to a browser.

The workbench enables information extraction of any kind from annotated corpora. Query results are shown as sets of references to the queried corpus. Extraction of statistical information from corpora, such as the number of marked-up nouns, is also supported. Computation of important reliability measures, such as kappa values, is enabled.

On the usability side it has been a priority to achieve ease of use. This is why, for instance, the coding module editor has been added. It helps the user to specify the markup declaration for a new coding module almost without requiring any knowledge of the underlying XML representation. The coding module editor automatically generates a DTD which is then used internally by the workbench. Unfortunately, the MATE markup framework has not yet been fully exploited as an intermediate layer between user interface and internal representation. Thus, the underlying XML format has not been entirely hidden from the coding module editor’s interface. Peculiarities of, and lack of flexibility in, XML have been allowed to unduly influence the way the user must specify the markup declaration. It is on our action list to solve this problem.

In spite of the above, the coding module editor actually works quite well from a usability point of view. The real usability problem in the workbench is the creation of new coding visualisations. Writing the style sheets actually requires programming skills because no editor is provided. The user must edit the raw style sheet code (or write new code). It is high on our wishlist to enable users to easily define new visualisations.

3. DEVELOPMENT AND EVALUATION USING THE MATE WORKBENCH

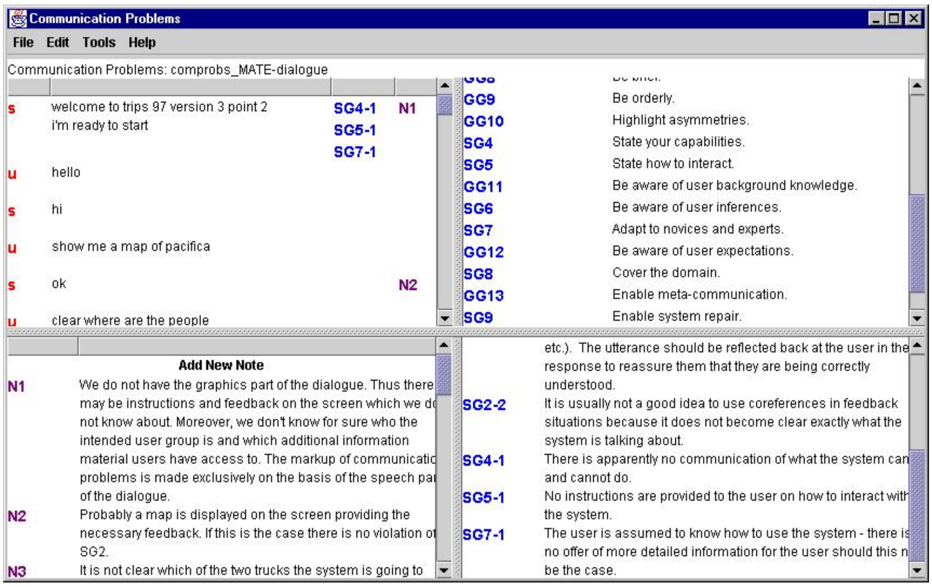
During SLDSs development, even of simple systems, many different issues must be considered and evaluated in some way. Quite often, corpora are collected to help the development of the system proceed in the right direction. This happens throughout the development process. At the very beginning, transcribed human-human dialogues may provide useful input to the system design. Later on, e.g., Wizard-of-Oz corpus data and field test corpus data may be used to evaluate and guide development, or data from controlled user tests may be used to decide if the system conforms to the specifications. At any stage during development, corpus data are being used for a wide variety of development and evaluation purposes, depending on, i.a., the particular SLDSs aspect or module that is being addressed. For instance, corpus data may be used for training and testing of the speech recogniser, or corpus data may be used to decide which vocabulary and grammars to include in the system and to test their coverage, etc.

In recent years, increasing attention has been paid to SLDSs aspects such as dialogue management and human factors. It has become clear that corpus data can provide much valuable information on the adequacy of the interaction design. In the EU DISC project on best practice in the development and evaluation of SLDSs (http://www.disc2.dk), we have proposed that the overall design goals for creating appropriate dialogue managers and usable interactive walk-up-and-use SLDSs may be systematically pursued by focusing on two comprehensive sets of best practice dialogue manager issues and usability issues, respectively. When developing a dialogue manager, for instance, the developer should decide whether or not the dialogue manager should provide top-down support for input language processing. If the answer is ‘yes’, then dialogue manager optimisation must take that issue into account. Thus, each relevant best practice issue provides a focal point for optimising the dialogue manager during development and evaluation. To do a complete evaluation of the dialogue manager, the developers must apply evaluation criteria which correspond to all the relevant issues [2, 3]. A similarly comprehensive set of issues and evaluation criteria for human factors in SLDSs are presented in [4, 6].

Many of the evaluation criteria which we have identified for dialogue management and human factors are to be applied to data extracted from annotated corpora. One of the best practice coding modules in the MATE workbench is a coding module for the markup of human-machine spoken communication problems. This coding module enables markup of data that are relevant to the evaluation of a range of human factors and dialogue management issues. It was developed to support the design of co-operative system utterances in spoken human-machine dialogue. It is based on 24 guidelines for co-operative human-machine spoken dialogue which were developed from a set of simulated human-machine dialogues and which include and extend Grice’s co-operativity maxims [7]. Eleven of the 24 guidelines are generic ones which express what to do or take into account when communicating co-operatively. The 13 specific guidelines are each subsumed by one of the generic guidelines, explain how to do something expressed by the generic guideline, and are specifically aimed at system design ([1], cf. Figure 1). The coding scheme based on those guidelines has later been successfully tested on other spoken human-machine corpora. However, as this was all done manually and without the help of any tools for annotation and information extraction, the coding and extraction process was very time-consuming and yielded annotated data which were ill-suited for re-use by others. The MATE workbench is changing all that.

We have not yet had the opportunity to use the MATE workbench for marking up communication problems during "real" SLDSs development and evaluation. Thus far, we have only coded communication problems in test dialogues followed by information extraction. The workbench has performed well in these tests. Figure 1 shows a screen shot of the workbench during markup of communication problems. The dialogue is shown in the upper left-hand pane. The guidelines for cooperative dialogue are shown in abbreviated form in the upper right-hand pane. Types of violations of the guidelines are incrementally added in the lower right-hand pane. This pane is empty when annotation starts. The blue markup in the dialogue refers to the types of violations described in this pane and the violations themselves refer to the guidelines. The lower left-hand pane contains annotator's notes. Again, this pane is empty when annotation starts. Notes can be added whenever the annotator needs to add some kind of explanation of, e.g., why something went wrong in a dialogue so as to cause a communication problem.

An interesting next step is to build on the evaluation criteria that were systematically developed in DISC, in order to create new coding modules which can be included in the MATE workbench and distributed for general use. This would help promote best practice in SLDSs evaluation and facilitate the exploitation of collected human-machine spoken dialogue corpora. Coding modules can be added or modified via the coding module editor shown in Figure 2.



Communication Problems: comprobs_MATE-dialogue

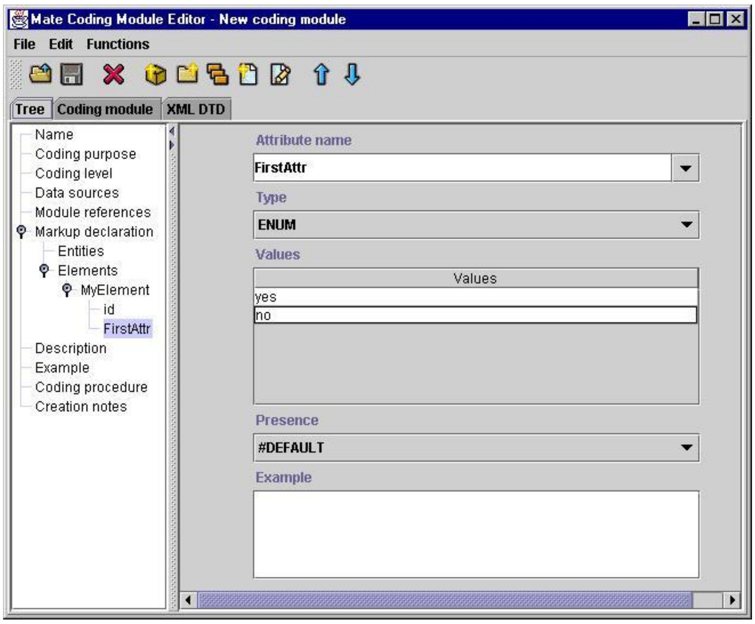
s	welcome to trips 97 version 3 point 2 i'm ready to start	SG4-1 SG5-1 SG7-1	N1
u	hello		
s	hi		
u	show me a map of pacifica		
s	ok		N2
u	clear where are the people		

GG9	Be orderly.
GG10	Highlight asymmetries.
SG4	State your capabilities.
SG5	State how to interact.
GG11	Be aware of user background knowledge.
SG6	Be aware of user inferences.
SG7	Adapt to novices and experts.
GG12	Be aware of user expectations.
SG8	Cover the domain.
GG13	Enable meta-communication.
SG9	Enable system repair.

N1	We do not have the graphics part of the dialogue. Thus there may be instructions and feedback on the screen which we d not know about. Moreover, we don't know for sure who the intended user group is and which additional information material users have access to. The markup of communicatio problems is made exclusively on the basis of the speech pa of the dialogue.
N2	Probably a map is displayed on the screen providing the necessary feedback. If this is the case there is no violation o SG2.
N3	It is not clear which of the two trucks the system is going to

	etc.). The utterance should be reflected back at the user in the response to reassure them that they are being correctly understood.
SG2-2	It is usually not a good idea to use coreferences in feedback situations because it does not become clear exactly what the system is talking about.
SG4-1	There is apparently no communication of what the system can and cannot do.
SG5-1	No instructions are provided to the user on how to interact wit the system.
SG7-1	The user is assumed to know how to use the system - there is no offer of more detailed information for the user should this n be the case.

Figure 1. Markup of communication problems using the MATE workbench. s is system, u is user, N is note, GG is generic guideline, SG is specific guideline.



Attribute name: FirstAttr

Type: ENUM

Values: yes, no

Presence: #DEFAULT

Example:

Figure 2. Adding a new coding module to the MATE workbench.

4. CONCLUSION

MATE has taken a major step towards standardisation and universal support of spoken dialogue data annotation and exploitation by proposing a framework for the annotation of spoken dialogue corpora at multiple levels and by building a workbench in support of this framework. In parallel, work on best practice in the development and evaluation of SLDSs and components has been carried out in the DISC project. We have illustrated how SLDS development and evaluation can be supported and made more cost-effective by using the MATE workbench. This was done by describing the coding scheme for communication problems which is included as a coding module in the MATE workbench and which is based on work done in DISC. We believe that it will be possible to further combine MATE and DISC results to help engineer a more efficient SLDSs development and evaluation process. The obvious next step is to develop and test MATE coding modules for the most important among the evaluation criteria that were generated in DISC.

5. REFERENCES

1. Bernsen, N.O., Dybkjær, H. and Dybkjær, L.: Designing Interactive Speech Systems. From First Ideas to User Testing. Springer Verlag 1998.

2. Bernsen, N.O. and Dybkjær, L.: Draft Proposal on Best Practice Methods and Procedures in Dialogue Management. DISC Deliverable D3.5, 1999.

3. Bernsen, N. O. and Dybkjær, L.: A Methodology for Evaluating Spoken Language Dialogue Systems and Their Components. Proceedings of the Second International Conference on Language Resources and Evaluation (LREC 2000), Athens, 2000, 183-188.

4. Dybkjær, L. and Bernsen, N.O.: Usability Issues in Spoken Language Dialogue Systems. To appear in Natural Language Engineering, 2000.

5. Dybkjær, L., Bernsen, N.O., Dybkjær, H., McKelvie, D. and Mengel, A.: The MATE Markup Framework. MATE Deliverable D1.2, 1998.

6. Failenschmid, K., Williams, D., Dybkjær. L. and Bernsen, N.O.: Draft Proposal on Best Practice Methods and Procedures in Human Factors. DISC Deliverable D3.6, April 1999.

7. Grice, P.: Logic and conversation. In P. Cole and J. L. Morgan (Eds.), Syntax and Semantics Vol. 3: Speech Acts. New York: Academic Press 1975, 41-58. Reprinted in Paul Grice: Studies in the Way of Words. Cambridge, MA, Harvard University Press, 1989.

8. Isard, A., McKelvie, D., Cappelli, B., Dybkjær, L., Evert, S., Fitschen, A., Heid, U., Kipp, M., Klein, M., Mengel, A., Møller, M.B. and Reithinger, N.: Specification of Workbench Architecture. MATE Deliverable D3.1, 1998.

9. Klein, M., Bernsen, N.O., Davies, S., Dybkjær, L., Garrido, J., Kasch, H., Mengel, A., Pirrelli, V., Poesio, M., Quazza, S. and Soria, S.: Supported Coding Schemes. MATE Deliverable D1.1, 1998.

10. Mengel, A., Dybkjær, L., Garrido, J., Heid, U., Klein, M., Pirrelli, V., Poesio, M., Quazza, S., Schiffrin, A. and Soria, C.: MATE Dialogue Annotation Guidelines. MATE Deliverable D2.1, 2000.