N.b.: A graphical user interface for annotating spoken dialogue

Giovanni Flammia and Victor Zue
Spoken Language Systems Group
Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139 USA

Abstract

Corpora of transcribed and annotated dialogues are very useful for developing and evaluating the coverage of algorithms for discourse generation and interpretation and dialogue modelling. On the other hand, there is no agreement on the choice of units and conventions for annotating discourse constituents, and the annotation process can be difficult and prone to inconsistencies. This paper presents *N.b.*, a graphical user interface for annotating the discourse structure of spoken dialogue. Different annotation instructions and different theories about discourse interpretation and generation can be easily incorporated in the annotation process without the need of changing the graphical user interface. The instructions and the annotated text are displayed in a clear-cut way, and typing is reduced to a minimum. We describe how to use *N.b.* for annotating embedded discourse segments and the system's end-to-end performance in a transcribed dialogue.

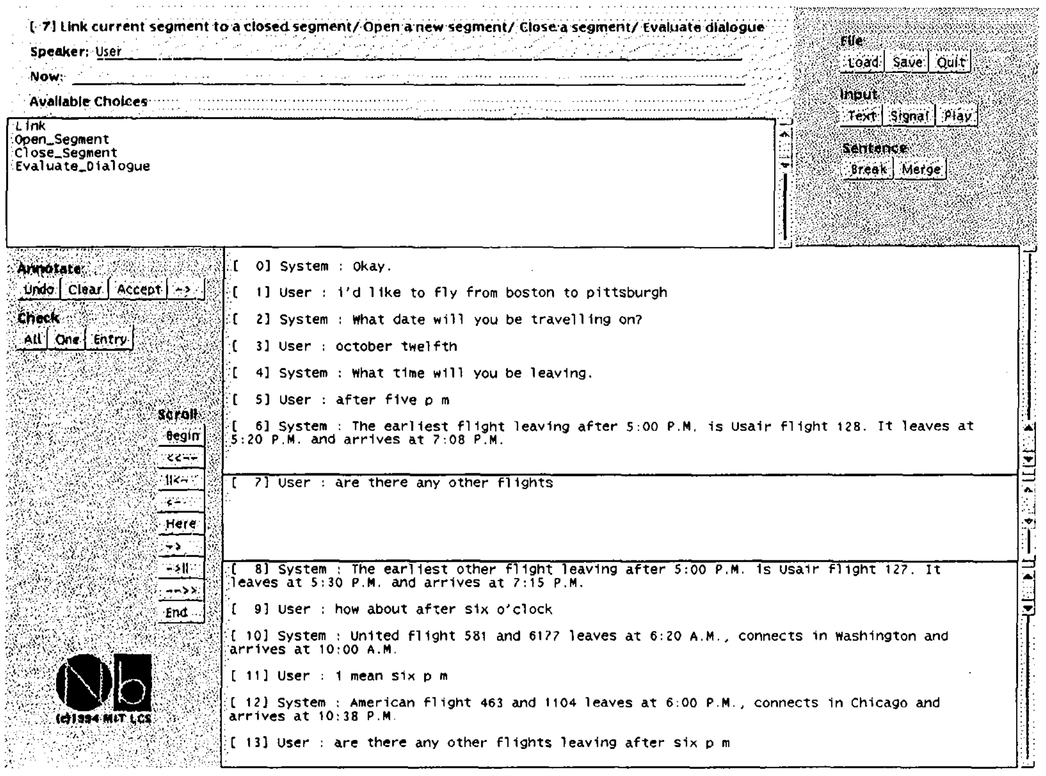
Introduction

Research and development of human language technology typically relies on the availability of suitably collected corpora. Such corpora can help researchers understand the regularity/variability of the linguistic phenomena under investigation, propose computational models to mimic their behavior, estimate the parameters of the models, and evaluate the effectiveness of either the models or systems that embed these models(Goodine *et al.* 92, Silverman *et al.* 92, Hirshman *et al.* 93). To develop phonetic recognition algorithms, for example, researchers in the US have relied on the TIMIT corpus (Lamel *et al.* 86) to understand the acoustic realizations of phonemes under varying phonetic environments, thereby developing phonetic models to capture such contextual variations (Lee and Hon 89, Goldenthal and Glass 94).

For a corpus to be truly useful, it must be properly annotated. Corpus annotation involves defining the inventory of constituent units (phonemes, syntactic categories, semantic categories, etc.), together with a set of annotation conventions. For example, at the syntactic level, pronouns might be annotated along with the noun phrases they refer to (Walker 89) and at the semantic level, phrases might be annotated with their meaning according to formal models like logical forms or semantic frames (Allen 94).

Annotation of low-level linguistic phenomena, such as phonetic variants and disfluencies, are relatively straightforward, since agreement on the choices of units and conventions can often be reached (Lee and Hon 89, Silverman *et al.* 92). As a result, the task of annotation can often be shared across site, and the aggregate corpora are larger and more useful to a wider community. As we move up the linguistic chain, however, the picture can rapidly deteriorate. In most cases, the controversy stems from the fact that the choice of units and conventions are often tied to linguistic theories that are not universally subscribed. Therefore, corpora annotated by one site may not be useful to researchers from other sites, leading to duplication of effort and inhibiting cross-system comparisons. One approach to dealing with this problem is to provide a set of minimal, theory-neutral annotations. The Penn Treebank (Marcus *et al.* 93) is an excellent example of linguistic data annotated using this approach. Syntactic structure of sentences is implicitly described by *bracketing* major constituents without actually attaching labels to them. While there were some initial doubts regarding the ultimate utility of such an annotation scheme, they were largely put to rest once researchers had a chance to make use of the corpus. The Penn Treebank has been instrumental in facilitating the comparison of several general English parsers (Black 93).

Over the past few years, our group has been involved with research and development of interactive conversational systems, i.e., systems that can understand and respond to verbal input. We have become increasingly aware of the importance of discourse and dialogue modelling, and the need for properly annotated data to aid research in this area. To uncover the discourse structure of a dialogue, utterances might be classified into speech acts. Sequences of utterances might be bracketed into a multi-level structure of embedded seg-



```
[ 7] Link current segment to a closed segment/ Open a new segment/ Close a segment/ Evaluate dialogue
Speaker: User
Now:
Available Choices
Link
Open_Segment
Close_Segment
Evaluate_Dialogue
```

```
[  0] System : Okay.
[  1] User : i'd like to fly from boston to pittsburgh
[  2] System : What date will you be travelling on?
[  3] User : october twelfth
[  4] System : What time will you be leaving.
[  5] User : after five p m
[  6] System : The earliest flight leaving after 5:00 P.M. is Usair flight 128. It leaves at
5:20 P.M. and arrives at 7:08 P.M.
[  7] User : are there any other flights
[  8] System : The earliest other flight leaving after 5:00 P.M. is Usair flight 127. It
leaves at 5:30 P.M. and arrives at 7:15 P.M.
[  9] User : how about after six o'clock
[ 10] System : United flight 581 and 6177 leaves at 6:20 A.M., connects in Washington and
arrives at 10:00 A.M.
[ 11] User : i mean six p m
[ 12] System : American flight 463 and 1104 leaves at 6:00 P.M., connects in Chicago and
arrives at 10:38 P.M.
[ 13] User : are there any other flights leaving after six p m
```

Figure 1: *N.b.* main window at the beginning of a discourse annotation session. The top left area displays annotation instructions and choices for the current sentence (sentence 7) and the main text panel (bottom right) displays a window of the text to be annotated. The current sentence is displayed in a different color in the center of the text panel.

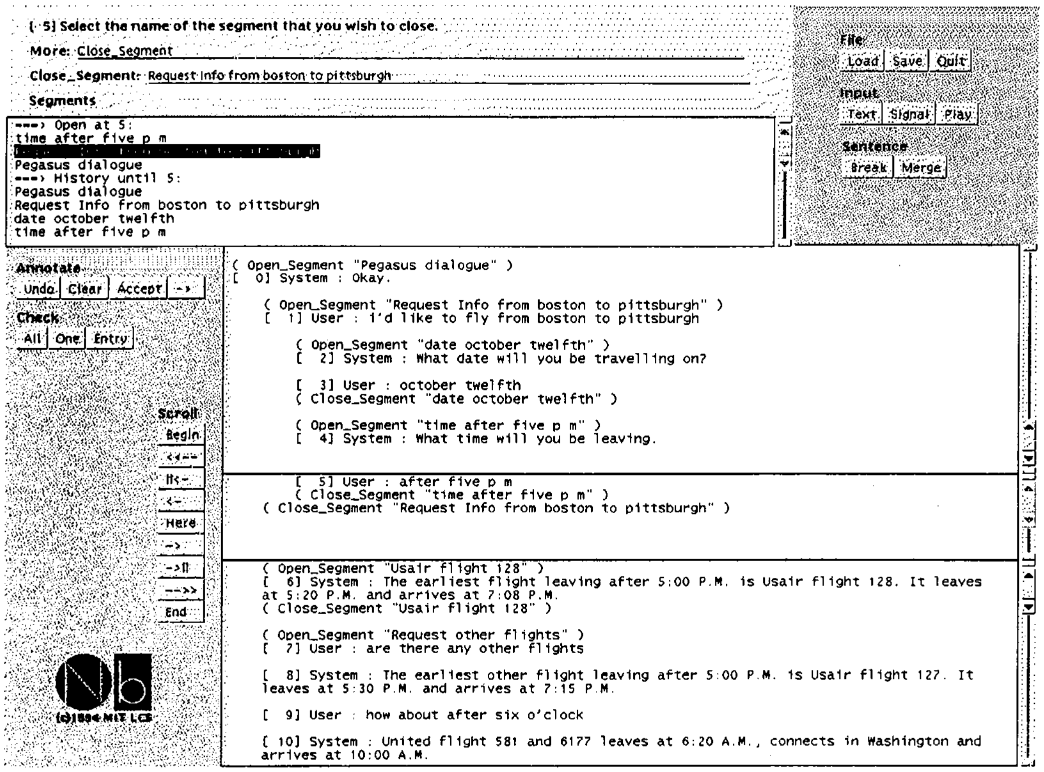
ments, and each segment might be annotated with its purpose with respect to the entire discourse (Grosz and Sidner 86, Mann and Thompson 88, Cohen *et al.* 90, Moore and Pollack 92, Hovy 93, Allen 94). Uncovering the discourse structure is a difficult task to accomplish and to evaluate (Rotondo 84). Recent work on human discourse annotation reliability (Grosz and Hirshberg 92, Passoneau and Litman 93) and discourse segmentation algorithms (Morris and Hirst 91, Hearst 94) has inspired us to investigate general discourse annotation methods and evaluation tools.

Following the example set forth by the Penn Treebank project, we have decided to develop a minimal and theory neutral annotation method. We believe that the development of efficient annotation tools constitutes an important aspect of this research. In a typical annotation session, a subject is presented with some instructions and with the text to be annotated, and is expected to produce an annotated text according to a specified format. This process can be difficult, time consuming and prone to inconsistencies, unless the subject is provided with an appropriate interface for entering the annotation. A good set of tools can greatly facilitate the annotation process, both in throughput, accuracy, and consistency, thereby leading to useful data that can serve the needs of the research community.

We think that the ideal interface should be ergonomical and portable. By ergonomical we mean that it should present the annotation instructions and the annotated text in a clear-cut way, so that subjects will be able to produce a consistent annotation without difficulty. By portable we mean that the interface should be general enough to accomodate a large variety of annotating instructions and linguistic theories, so that changes in the instruction formats and different experimental conditions can be incorporated easily, possibly without modifying the internal structure of the interface software.

In the following sections, we describe *N.b.* (*Nota Bene*[1]), a dialogue annotation tool that we are developing with the goal of meeting the two requirements of ergonomicity and portability. We will describe its functions, provide some examples of how it can be used, and summarize future development plans.

[1] *N.b.* stands for the Latin *Nota Bene* which means *to annotate well.*



```
[ 5] Select the name of the segment that you wish to close.
More: Close_Segment
Close_Segment: Request Info from boston to pittsburgh
Segments
```

```
===> Open at 5:
time after five p m
[illegible]
Pegasus dialogue
===> History until 5:
Pegasus dialogue
Request Info from boston to pittsburgh
date october twelfth
time after five p m
```

```
( Open_Segment "Pegasus dialogue" )
[  0] System : Okay.

   ( Open_Segment "Request Info from boston to pittsburgh" )
   [  1] User : i'd like to fly from boston to pittsburgh

      ( Open_Segment "date october twelfth" )
      [  2] System : What date will you be travelling on?

      [  3] User : october twelfth
      ( Close_Segment "date october twelfth" )

      ( Open_Segment "time after five p m" )
      [  4] System : What time will you be leaving.

      [  5] User : after five p m
      ( Close_Segment "time after five p m" )
   ( Close_Segment "Request Info from boston to pittsburgh" )

   ( Open_Segment "Usair flight 128" )
   [  6] System : The earliest flight leaving after 5:00 P.M. is Usair flight 128. It leaves
   at 5:20 P.M. and arrives at 7:08 P.M.
   ( Close_Segment "Usair flight 128" )

   ( Open_Segment "Request other flights" )
   [  7] User : are there any other flights

   [  8] System : The earliest other flight leaving after 5:00 P.M. is Usair flight 127. It
   leaves at 5:30 P.M. and arrives at 7:15 P.M.

   [  9] User : how about after six o'clock

   [ 10] System : United flight 581 and 6177 leaves at 6:20 A.M., connects in Washington and
   arrives at 10:00 A.M.
```

Figure 2: Using *N.b.* to annotate embedded discourse segment boundaries. The annotator has chosen to close the segment named *Request Info from boston to pittsburgh* at sentence 5. The text is indented by the program according to the proposed segmentation.

Overview of *N.b.*

N.b. is a graphical user interface for editing annotations of text and transcribed speech according to instruction sequences specified in program arguments. In this section we give an overview of the user interface and in the following sections we describe how the program works and how to specify annotation instructions by way of two simple examples.

Figure 1 shows the main window in the current implementation after loading a log file with the transcription of an actual human-computer travel planning dialogue using the PEGASUS spoken language system (Zue *et al.* 94). The window presents to the user the information necessary for annotating the discourse segments of the dialogue. The main window is divided into two different areas. The annotation area for each sentence is located at the top left and the main text panel displaying the annotated text is located in the bottom right.

Annotation Area

The top left rectangular area displays the annotation instructions and annotation choices available for each sentence. A message displays brief instructions for annotating the current sentence. A text panel displays the available choices for annotating the current sentence. The annotator can grab text from this text panel to build annotations. In the figure, the current sentence is number 7, the current field is named **Now**, and the annotator has the options of opening a new segment, closing a segment, or linking two segments. Annotation instructions, fields, and messages are all program arguments specified in a configuration file.

Annotate Buttons

After the annotator has entered a value, she or he has the options of deleting the last annotation step (Undo), clearing the annotation field (Clear), accepting the annotation and proceed according to instructions (Accept), or scroll to the next sentence (− >).

Check Buttons

When clicked, these buttons call functions that help in evaluating the consistency of the annotation process. The functions being computed depend on the annotation instructions and are specified as external programs. In the case of discourse segmentation, the Check All button shows a popup window with the annotated text *pretty-printed* with each segment assigned to a different color and an appropriate level of indentation. The Check One button highlights one discourse segment at the time, and the Check Entry button displays all the annotations entered for the current sentence and the list of segments.

Scroll Buttons

Nine scroll buttons can center the main text panel to the first sentence (Begin), to the last one (End), to the currently annotated sentence (Here), to the preceding (< −) or next sentence (− >), to the preceding page (< −−) or the next page (−− >) or to specific annotated end-points (|| < − and − > ||), The annotator can scroll back and forth through the entire text and does not need to annotate the sentences in any particular order.

Main Text Panel

The main text panel displays a window of the annotated text. The currently annotated sentence (sentence 7 in Figure 1) is highlighted in a different color in the center of the text panel. The annotator can grab words and phrases from this text panel with the mouse to build annotation strings.

File Buttons

These buttons are used for loading text files and annotated text files, saving annotated text files, and quitting the program.

Input Buttons

Input Text shows a large popup window with the entire input text file. Input Signal runs an external program for viewing the speech waveform corresponding to the current sentence, and Input Play plays back the speech waveform corresponding to the current sentence. Playback and editing commands are specified as external programs in a configuration file.

Sentence Buttons

These buttons can be used for editing the text. Sentence Break is used for breaking the sentence into two or more separate units for annotating them separately, and Sentence Merge is used for merging two sentences into one single unit to be annotated.

Using *N.b.*

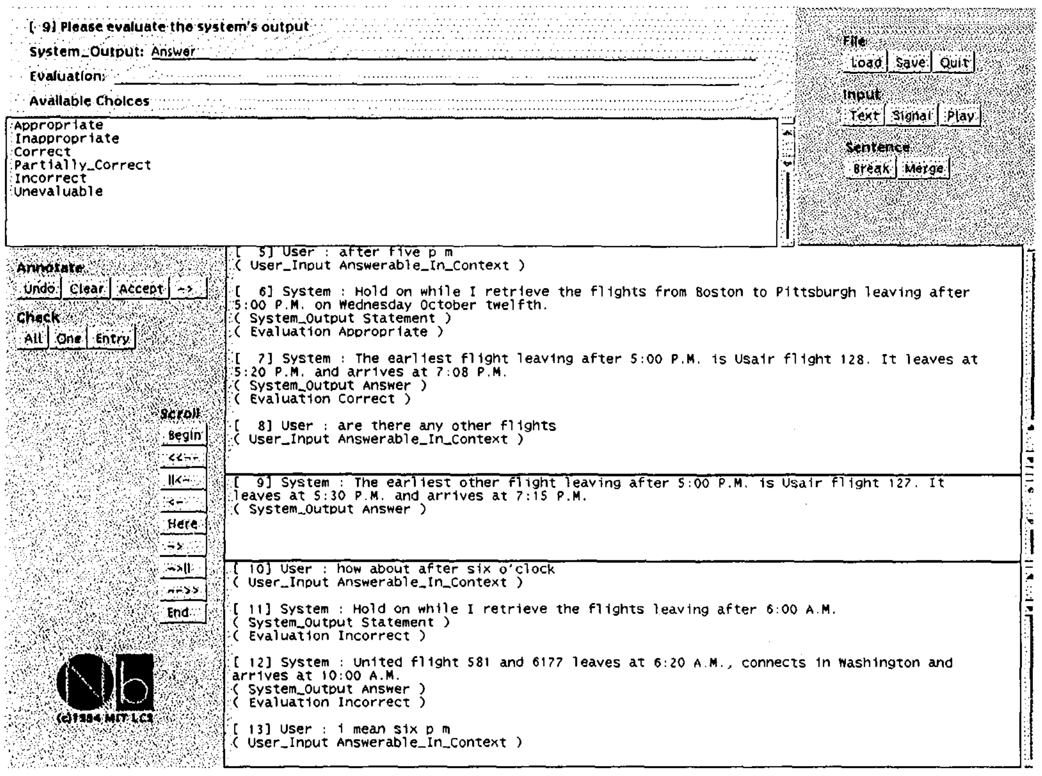
In the following, we illustrate *N.b.* works and how to write annotating instruction sequences by way of two simple examples. We annotate an actual human-machine dialogue using PEGASUS. This is a mixed initiative system in which the machine may prompt the user for specific missing information in order to specify the user's request, and misunderstandings may occur because of speech recognition errors or semantic analysis failures. In the first example, the task is to annotate the discourse segments. In the second example, the task is to evaluate the system's performance in the dialogue, proceeding utterance by utterance.

Annotating Discourse Segments

For each sentence, the initial choices are either `Link` , `Open_Segment`, `Close_Segment`, or `Evaluate Dialogue`. This last field is used for answering some questions about global properties of the dialogue, such as what are the customer and the agent goals and whether or not the dialogue has been successful. The annotator can bracket the text with embedded segments (see Figure 2). Segment boundaries are annotated in the text with the words `(Open_Segment Name)` and `(Close_Segment Name)`. Names can specify discourse segment purposes or intentions and provide easily memorized identifiers for each se gment. They can either correspond to specific keywords displayed in the text panels, or they may be typed in. The entire text is indented automatically according to the proposed segmentation. Warning messages are issued when crossing brackets violate the balanced structure of the segment sequence. If the annotator wishes to model the segments with a graph structure rather than with a tree structure, she or he can link two or more non-overlapping segments using the command `Link`, at which point *N.b.* prompts for selecting the names of the segments to be linked. The commands `Open_Segment` and `Close_Segment` can be used to move existing segment end-points in the annotated text. The Undo button can be used at each sentence to delete annotations and backtrack to a previous annotation step. For example, undoing `(Open_Segment Name)` effectively deletes a segment from the annotated text.

In Figure 2, the annotator has chosen `Close_Segment` for sentence 5. Then, the program displays a list of existing segments in the Available Choices text panel. When the annotator grabs the words *Request Info from boston to pittsburgh* with the mouse, the choice is displayed in the annotation text field, the segment is closed, and the annotated text is automatically indented and displayed again.

The annotation instructions for opening and closing segments are specified in a configuration text file. The instructions are illustrated at the top of Figure 4 at the end of this paper. The instructions specify the name and the possible values for each field to be annotated and in which order the fields should be filled out. For each field, we specify its name, what to display in the Available Choices text panel, what field has to be annotated after each one of the choices has been made, a default next field, and the instruction message to be displayed. The statement `Display Field` specify to display all the values that have been entered for `Field` up to the current sentence. The statement `Display Open_Segment` specifies to display all the segments annotated between the first sentence and the current sentence.



[9] Please evaluate the system's output

System_Output: Answer

Evaluation:

Available Choices

```
Appropriate
Inappropriate
Correct
Partially_Correct
Incorrect
Unevaluable
```

```
[  5] User : after five p m
( User_Input Answerable_In_Context )

[  6] System : Hold on while I retrieve the flights from Boston to Pittsburgh leaving after
5:00 P.M. on Wednesday October twelfth.
( System_Output Statement )
( Evaluation Appropriate )

[  7] System : The earliest flight leaving after 5:00 P.M. is Usair flight 128. It leaves at
5:20 P.M. and arrives at 7:08 P.M.
( System_Output Answer )
( Evaluation Correct )

[  8] User : are there any other flights
( User_Input Answerable_In_Context )

[  9] System : The earliest other flight leaving after 5:00 P.M. is Usair flight 127. It
leaves at 5:30 P.M. and arrives at 7:15 P.M.
( System_Output Answer )

[ 10] User : how about after six o'clock
( User_Input Answerable_In_Context )

[ 11] System : Hold on while I retrieve the flights leaving after 6:00 A.M.
( System_Output Statement )
( Evaluation Incorrect )

[ 12] System : United flight 581 and 6177 leaves at 6:20 A.M., connects in Washington and
arrives at 10:00 A.M.
( System_Output Answer )
( Evaluation Incorrect )

[ 13] User : i mean six p m
( User_Input Answerable_In_Context )
```

Figure 3: Using *N.b.* to evaluate the behavior of a spoken language system. The annotator can classify the user's input and evaluate the correctness of the system's output using key words proposed by the interface.

Evaluating A Spoken Language System

In this example, we use *N.b.* for annotating each sentence separately with specific sequences of key words. The task is to evaluate the end-to-end performance of the PEGASUS system for the transcribed dialogue. The main window at the end of the annotating session is displayed in Figure 3, and the bottom of figure 4 displays the annotation instructions for evaluating the system's performance.

When evaluating the system's utterance, first the annotator has to classify it according to some types such as **Question** or **Answer** and then she or he has to evaluate the sentence by grabbing one of the available choices, at which point instead of annotating another field, **Go_To_Next_Entry** specifies to scroll the main text panel to the next sentence. Different evaluation protocols can be easily integrated in the annotation process by changing the specifications in the configuration file.

Conclusions

In this paper we have presented *N.b.*, a graphical user interface for annotating spoken dialogue in a consistent fashion, with typing reduced to the minimum, according to instruction sequences that can be easily specified and modified.

Currently, we are testing and improving the various parts of the program in order to ensure ergonomicity and portability with respect to different annotation instructions, and to allow for maximum flexibility for input and output file formats. We are using this interface for developing a minimal and theory neutral annotation method of corpora of naturally occuring task-oriented human-human telephone conversations. We succesfully completed an experiment for evaluating the inter-subject annotation agreement with a corpus of 12 recorded telephone conversations between customer and agent, each conversation being annotated by 5 different subjects. We are also investigating the definition of evaluation criteria such as precision, recall, and accuracy for the problem of comparing embedded non-linear bracketings of text. The evaluation of the annotated data will serve as benchmark study for assessing the human performance in this annotation task and the basis for developing and evaluating an algorithm for the discovery of the discourse constituents of spoken dialogue.

We hope that the annotating instruction specifications of *N.b.* are general enough to accomodate a large variety of linguistic theories, and even different annotation purposes than the ones we are currently interested

```
-------------------------------------------
FIELD
     Now
CHOICES
     Link                -> Refer_To
     Open_Segment        -> Open_Segment
     Close_Segment       -> Close_Segment
     Evaluate_Dialogue -> Client_Goal
     Default             -> More
HELP
     Link current segment to a closed
     segment/ Open a new segment/
     Close a segment/ Evaluate dialogue
END
FIELD
     Close_Segment
CHOICES
     Display Open_Segment
     Default             -> More
HELP
     Select the name of the segment
     that you wish to close.
END
-------------------------------------------
FIELD
   System_Output
CHOICES
   Statement
   Answer
   Question
   Answer_With_Question
   Statement_With_Question
   Failure_To_Understand -> Go_To_Next_Entry
   Unclear               -> Go_To_Next_Entry
   Default               -> Evaluation
HELP
   Classify the type of the system's output
END
FIELD
   Evaluation
CHOICES
   Appropriate
   Inappropriate
   Correct
   Partially_Correct
   Incorrect
   Unevaluable
   Default          -> Go_To_Next_Entry
HELP
   Please evaluate the system's output
END
-------------------------------------------
```

Figure 4: Two examples of annotation instructions specifications. Top: Instructions for annotating discourse segments. Bottom: Instructions for evaluating a spoken language system. The instructions specify the name and the possible values for each field to be annotated and in which order the fields should be filled out.

in.

While *N.b.* is intended for annotating discourse in spoken dialogue, we found that it can be used for other tasks, such as evaluating spoken language system performance. We think it is also possible to use *N.b.* for annotating words and phrases with their syntactic or semantic categories.

Currently, the program has been implemented in C on a Sun workstation under the X windows graphic environment. When the program will reach a deliverable state, *N.b.* will be available to all interested researchers.

Acknowledgments

We wish to thank Joe Polifroni, Christine Pao, Michael McCandless, Rob Kassel and Christine Nakatani for evaluating early versions of the program and providing very useful suggestions for improving the user interface.

This research was supported by ARPA under Contract N6601-94-C-6040, monitored through the Office of Naval Command, Control and Ocean Surveillance Center.

The corpora of telephone conversations used for the discourse annotation experiments is being contributed by American Airlines SABRE Travel Information Network and by Bell South Intelliventures.

World Wide Web Reference

Up-to-date information about *N.b.* can be found in the World Wide Web resource:
http://sls-www.lcs.mit.edu/~flammia/Nb.html

References

Allen J., *Natural Language Understanding*, Second Edition, Benjamin/Cummings, Redwood City, CA, 1994.

Black E. "Parsing English by computer: the state of the art", *Proc. ISSD-93 International symposium on spoken dialogue*, Tokyo, 1993. pp. 77-81.

Cohen P.R., Morgan J., and Pollack M.E., editors. *Intentions in Communication*, MIT Press, Cambridge, MA. 1990.

Hearst M.A. *Context and Structure in Automated Full-Text Information Access*, PhD Thesis report no. UCP/CSD-94/836, University of California, Berkeley, CA. 1994.

Hirshman L. *et al.* "Multi-site data collection and evaluation in spoken language understanding" *Proc. Human Language Technology Workshop*, Bates M., editor, Princeton, March 1993. pp. 19-24.

Hovy E.H., "Automated discourse generation using discourse structure relations", *Artificial Intelligence*

Vol. 63. pp. 341-385.

Goodine D., Hirschman L., Polifroni J., Seneff S., and Zue V. "Evaluating interactive spoken language systems.", *Proc. Int. Conf. on Spoken Language Processing*, Banff, Canada, 1992. pp. 201-204.

Goldenthal W.D. and Glass J.R. "Statistical trajectory models for phonetic recognition", *Proc. Int. Conf. on Spoken Language Processing*, Yokohama, Japan, 1994. pp. 1871-1874.

Grosz B., Sidner C., "Attentions, Intentions and the Structure Of Discourse.", *Computational Linguistics*, Vol. 12. No. 3, 1986. pp. 175-204.

Grosz B., and Hirshberg J.. "Some intonational characteristics of discourse structure". *Proc. Int. Conf. on Spoken Language Processing*, Banff, Canada, 1992. pp. 429-432.

Lamel L., Kassel R., and Seneff S. "Speech database development: design and analysis of the acoustic-phonetic corpus", *Proc. DARPA Speech Recognition Workshop*, Report No. SAIC-86/1546, February, 1986. pp. 100-109.

Lee K.F., and Hon H.W. "Speaker-independent phone recognition using hidden Markov models", *IEEE Trans. ASSP*, Vol. 37, No. 11, November 1989. pp. 1641-1648.

Mann, W.C. and Thompson S.A. "Rethorical Structure Theory: A theory of text organization.", *Text*, Vol. 8. No. 3. 1988. pp. 243-281.

Marcus M., Santorini S., and M. Marcinkiewicz, "Building a large annotated corpus of English: the Penn Treebank", *Computational Linguistics*, Vol. 19, No. 2, 1993. pp. 313-330.

Moore J.D. and Pollack M. "A problem for RST: The need for multi-level discourse analysis", *Computational Linguistics*, Vol. 18. No. 4. 1992. pp. 537-544.

Morris J. and Hirst G. "Lexical cohesion computed by thesaural relations as an indicator of the structure of text". *Computational Linguistics.*, Vol. 17, No. 1, 1991. pp.21-48.

Passoneau R.J. and Litman D.J. "Intention-based segmentation: Human reliability and correlation with linguistic cues" *Proc. 31st Annual Meeting of the Association for Computational Linguistics*, 1993. pp. 148-155.

Rotondo J.A. 1984. "Clustering analysis of subject partitions of text", *Discourse Processes*, Vol. 7. pp. 69-88.

Silverman K. *et al.* "TOBI: A standard for labeling English prosody", *Proc. Int. Conf. on Spoken Language Processing*, Banff, Canada, 1992. pp. 867-870.

Walker M.A. "Evaluating discourse processing algorithms". *Proc. 27th Annual Meeting of the Association for Computational Linguistics.* 1989. pp. 251-260.

Zue V., Seneff S., Polifroni J., Phillips M., Pao C., Goddeau D., Glass J., and Brill E. "PEGASUS: A spoken language interface for on-line air-travel planning", *Proc. ARPA Human Language Technology Workshop*, Princeton, New Jersey, 1994. pp. 201-206.