AUTOMATICALLY GENERATING NATURAL LANGUAGE STATUS REPORTS

Jugal Kalita and Sunil Shende
Department of Computer and Information Sciences
University of Pennsylvania, Philadelphia, PA 19104

ABSTRACT

In this paper, we describe a system which generates compact natural language status reports for a set of inter-related processes at various stages of progress. The system has three modules — a rule-based domain knowledge representation module, an elaborate text planning module, and a surface generation module. The knowledge representation module models a set of processes that are encountered in a typical office environment, using a body of explicitly sequenced production rules implemented by an augmented Petri net mechanism. The system employs an interval-based temporal network for storing historical information. A text planning module traverses this network to search for events which need to be mentioned in a coherent report describing the current status of the system. The planner combines similar information for succinct presentation whenever applicable. It also takes into consideration various issues such as relevance and redundancy, simple mechanisms for viewing events from multiple perspectives and the application of discourse focus techniques for the generation of good quality text. Finally, an available surface generation module which has been suitably augmented is used to produce well-structured textual reports for our chosen domain.

1. INTRODUCTION

In recent years, several natural language generation systems have attempted to address the issue of generating coherent multisentential text. These include TEXT [18], KDS [14], ANA [12], and NAOS [20], among others. TEXT which answers questions about database structure, deals with a static knowledge base whereas ANA and NAOS concern themselves with situations which are dynamic in nature. The KDS system is provided with a full-fledged proposition-type representation of the information to be conveyed whereas the others start from more elementary knowledge representations such as a hierarchical conceptual network (TEXT), or a set of numbers denoting fluctuating values of certain salient domain variables over a specified period of time (ANA), or a graphics system depicting animation of objects such as various types of cars (NAOS). Other systems which have attempted to generate multi-sentential connected text include story generation systems: TALE-SPIN [19], SAM [23], and the tutoring system SCHOLAR [6], etc. In spirit, the system discussed in this paper is more akin to NAOS which attempts to generate a textual description of a dynamically varying scene.

This paper describes the design and implementation of a text generating system which produces natural language reports on the status of a system of inter-related processes at various stages of progress. In many AI applications, monitoring the state of a system is deemed essential. The ability of a system in such an environment to describe its actions in natural language will be certainly very useful.

Section 2 of this paper gives a brief description of the domain being modeled along with an overview of the system components. Section 3 describes the representation of the processes in terms of a network of rules. Section 4 elaborates the nature of the historical information store. Section 5 deals with text planning which is crucial for selecting information to be presented in a coherent manner. Section 6 details the actual process of text generation. Finally, we conclude with a discussion about enhancements to the existing system which can contribute towards a more sophisticated implementation.

2. OVERVIEW OF THE SYSTEM

The System Architecture: The system consists of three main modules as shown in figure 1. They are an augmented production system for domain knowledge representation, a text planner module (strategic component), and a text generation module (tactical component). The production system is used for simulating the events and processes that take place in the domain. When requests are made for textual report generation, the strategic component determines the content and the order of the text in consultation with the knowledge base. Its output is an ordered message in an internal representation which is passed to the tactical component that determines how to verbalize the message in English by deciding on phrasal and sentential structure, and choosing the actual words of the text.

This separation of conceptual and linguistic decisions provides a simple, clear-cut and easy-to-handle control structure in the decisional processes involved. This approach which is similar to the one taken by McKeown [18] and Kukich [12] for text generation, differs from the approaches taken by Hovy [10], Danlos [7], Ritchie [21], Mann [15], or Appelt [2] which emphasize much closer coordination among the different modules. Mann allows the systemic linguistic component to query other parts of the system such as the knowledge base and the text planner when required. Appelt, and Danlos have developed integrated models of language generation with no clear separation between the phases. Appelt characterizes all levels of language generation as a planning problem which can interact with one another. Danlos claims that extensive and unpredictable interactions are required between conceptual and linguistic decisions; no general principles ordering these decisions can be developed. Hovy and Ritchie also allow complicated interactions among the modules at various choice points.

As our sample domain, we chose a scenario where the system is used to assist the secretary for an academic journal by keeping track of a paper submitted for publication. The process being modeled is that of paper-submission with the usual participating agents being the author of the paper, the journal editor, the reviewers assigned to evaluate the paper and the journal secretary.

We model the system of processes in terms of a production system with an explicit control structure for rule sequencing; this representation is due to [25]. External as well as system generated events trigger execution of rules. The execution of these rules builds a history
network of events and processes. This network is modeled loosely on the notion of intervals proposed in [3].

The creation of a report in natural language entails that the actual description be easy to understand. The report must omit inessential information, be cogent, and give the user a sense of temporal structure. Towards this end, planning at the representational level is required of the system. The text generator used is MUMBLE [17], which is a collection of morphology and grammar modules driven by user created objects representing pieces of text and containing information about the high-level specifics of the text.

3. REPRESENTATION OF DOMAIN KNOWLEDGE

The domain described above has been modeled using a production system whose rules are sequenced by imposing a Petri net structure on them. The decision to employ a production system has been influenced by the event-driven nature of the processes in the domain allowing declarative encoding of relevant knowledge.

The Processes: In the domain under consideration, there are several concurrent, independent processes. All activities in the domain can be viewed as constituting a single process — the overall process of journal editing. The overall process comprises a collection of time-ordered atomic actions and/or subprocesses. Communication and coordination is required among these processes, and the atomic actions constituting these processes for achieving proper temporal ordering.

In our implementation, the system of actions and processes is represented as a set of production rules. Since in a pure production system it is difficult to achieve substantial interrule communication, a control structure where the current state of the system in conjunction with the history of prior rule execution determines subsequent decisions, first proposed in the context of an augmented Petri net model in [25], has been adopted. A state in the Petri net represents a state of the journal secretary (since our representation is from the viewpoint of the secretary). A transition between two states denotes an interaction between an external agent (such as the author, the editor or one of the referees) and the secretary. Each interaction is represented by a production rule which resides on a transition in the augmented Petri net.

The Domain Representation: In the augmented Petri net representation of our domain, the system (representing the editor's secretary) can exist in several states such as, waiting for the paper to arrive, waiting for the editor to designate referees, waiting for a referee to respond, waiting for a referee to submit his report, etc. Several interactions can take place between the secretary and the various external agents. Two sets of interactions are illustrated in figure 2. Some of the rules in the production system are shown in figure 3.

4. THE NETWORK OF INTERVALS: HISTORICAL KNOWLEDGE

An Interval-based Model: In order to generate a report about a particular paper, the production system discussed in the previ-
ous section builds an historical record of event instances as they take place over time. The representation used must adequately capture temporal relationships among events, and also minimize the amount of computation necessary in order to achieve reasonable levels of efficiency. The representation we adopt is based on the model of time based on intervals and proposed in [3]. The model allows for representing the relationships between temporal intervals in a hierarchical manner using a constraint propagation technique with simple representation of concurrency, facilitates structuring knowledge in a way convenient for typical reasoning tasks, and also is logically sound.

Following [3], intervals in our system are of two types — simple intervals (designated henceforth as intervals) and reference intervals. Intuitively, an interval corresponds to the time between two successive interactions between agents in the system. In contrast, a reference interval corresponds to the time during which a whole series of interactions take place (i.e., the temporal duration of a process or sub-process). Reference intervals, allow us to conveniently group together clusters of simple intervals. Each interval is identified further by a description of some of the events which occurred during the two interactions whereas each reference interval is identified by the intervals which comprise it. The relationships among intervals are maintained in the network where nodes represent individual intervals, and each arc's label indicates the possible relationships between the two intervals represented by its end nodes. Since the network built by our production system is historic in nature, there is no temporal uncertainty, and therefore, each arc has only one label which indicates strict temporal precedence among the intervals.

An interval (or reference interval) is said to be instantiated if it becomes part of the history network. In particular, every instantiated interval has slots which contain its starting event, its ending event and a link to its predecessor interval in the history network. In addition to the starting and ending events, an interval may also contain side events that neither start nor end the interval but which occur in the

- Rule-1: If a paper is received,
  - acknowledge the author of its receipt, and
  - request the editor to designate names of referees.
- Rule-5: If the author withdraws the paper, instantiate termination procedure.
- Rule-6: If the editor does not respond within two weeks, send him a reminder letter.

Figure 3: Examples of production rules

domain process as a matter of course between the starting and ending events and do not have any further consequences.

An interval is, designated open if it has been started by some event but which is yet to be completed by another event. A completed interval is called a closed interval. In our representation, an instantiated open interval usually has as its starting event, an action performed by the secretary, whereas input from the external agents determines the ending event of some hitherto open interval.

Monitors: Although our system is basically input-driven at the user-system interface, there are many situations where the script of the domain process demands some temporal deadlines or monitors. For instance, in the event that the editor fails to respond (by way of input) to the request for names of reviewers, the system has to generate a perforce, another request as a reminder. To enable the system to faithfully retrieve such timeout activity from the network, we employ demons to explicitly monitor open intervals pending response. On timeout, the system simply creates another open interval with the same starting event as the previous one timed out by its associated demon.

Repeated Interval Instantiation: It is often the case that processes (sub-processes) are generic in nature; i.e., they can be repeatedly instantiated for different agents. For example, our domain calls for a certain sequence of interactions between the secretary and a referee. This sequence defines a generic process and our system needs to keep track of three such processes, one for each referee assigned to review a paper. The second type of interval in our system, the reference interval which corresponds to processes and sub-processes, helps the text planner combine multiple instances of events into succinct specifications without having to repeat descriptions for each individual instance. It is also desirable in an application like ours to annotate the text describing a process with summaries or paraphrases of key sub-processes that occur naturally in the domain. Since it is clear that an open reference interval captures our intuitions about an ongoing process, the planner can generate appropriate temporal cues from the interval representation to determine text parameters like tense and aspect for a paraphrase.

An instance of the network configuration built up by the production system is shown in figure 4. However, not all details are shown in the figure.

5. TEXT PLANNING

Our computational goal, as stated earlier, was to describe or report on a set of processes in a compact manner, without disturbing the temporal nature of the process description. The text planning module thus needs to ensure that the text plan is ordered in time and that multiple event instances in processes be encapsulated, if possible, into concise descriptions. The former requirement entails that the events we set out to describe be ordered in time. From the description of the system network in the previous section, we notice that most of the sequencing is avoided by simply working one's way backward along the predecessor links in an interval data structure. However, the conciseness requirement forces the planner, at some points, to combine information which is common to more than one interaction sequence (sub-process) without ruining the nature of the then current plan. Our representation of reference intervals associated with ordinary intervals gives the extra information necessary to guide the planner in such cases.

Steps in the Planning Process: Briefly, the planner goes through several steps: it determines the content of the overall plan by way of a controlled search over the network of intervals and then produces a sequence of specifications for the actual text generator in the order

SPIE Vol. 937 Applications of Artificial Intelligence VI (1988) / 535
in which they are to be realized into text along with information regarding discourse focus. Since our intention is to provide as informed a description (report) of the system's activity as possible, we have chosen to restrict the content of a report to the interactions occurring from the time the previous request was issued to the time of the current request. In this respect, the interval data structure provides us with some useful information pertaining to what happened *just before* the previous report was requested and what is *expected* to occur *just after* the current one. This information is contained in the open intervals at the time of the previous request, and at the present moment respectively.

Combining Similar Events: First, the planner combines a search over the required part of the interval network starting from the intervals that are currently open with an ordering over the events which occurred in the intervals in the search space. Since reports can be generated at any point in time during the life of a paper, and one can ask for repeated reports at later times, *only events that have not been reported in an earlier report* need to be included in it. However, to ensure that the current report establishes a relationship with the immediately preceding report, one or more events that have been reported earlier may be included. This will be clear from the example presented later. The ordering ranks events tagged with their associated intervals (called "situations") temporally, while simultaneously ensuring that *similar* events corresponding to different instances of the same generic process are clubbed together. It should be noted that during this phase when similar events with one or more different participants are combined together for the purposes of generation, the system's decision *need not be totally based on temporal sequentiality*. Combinations may be performed for similar events although they occurred widely separated on a linear time scale. For example, suppose a referee sends his review quite early (say, at a time $t_1$), whereas a second referee does so with considerable amount of delay (at a time $t_2$), and several other things happen during the interval $(t_1, t_2)$. While reporting, the system will combine the two reviewing events into a single sentence such as *The two referees A and B sent their reviews by date $t_1$*, and report the events that took place in the interval $(t_1, t_2)$ either before this sentence or after it depending on other considerations.

Further Annotations: Next, the planner processes this stream of ranked situations to create the appropriate specifications for the text generator. This includes information about the sentence content (derived directly from the event in the situation) along with various features of the sentence like tense and indications to conjoin noun phrases etc. For the present, our tense assignment is rather naive and only relies on the network to provide temporal indicators based on the relationship of the time at which an event occurred in the context of the overall time interval covered by the plan. Thus we only assign present, present perfect, past and past perfect tenses to sentences. A more sophisticated tense assignment, perhaps along the lines of [16], would be necessary for future extensions of this project.

Thus, temporal relationships and event similarities are used to order the events into an appropriate sequence for presentation. However, in order to determine how exactly each sentence is going to be constructed, subsequently, the planner annotates the text specifications with information regarding discourse focus. Generating the actual text involves several decisions such as choice of sentence structure, and sentential subject, etc. Assuming, each sentence is of the seq¹ form, focus considerations can be used to choose between active and passive voices, and to decide which case slot's (e.g., direct-object, indirect-object, source, destination, etc.) filler to be used to fill the position of the sentential subject.

**Discourse Focus Considerations:** *Choice of subjects* for generated sentences can be done considering the movement of focus of discourse from sentence to sentence as following the analysis of Sidner (1983). It should be noted that McKeown, in the TEXT system (McKeown 1985), uses focus to mainly guide what to say whereas we use it to solely decide the syntactic form of the sentence. Focus information also facilitates pronominalization decisions.

There are two types of foci: *global* and *immediate*. Global focus is constant, but the immediate focus may vary from sentence to sentence. We can use the *current immediate focus* of the proposition to *produce the surface subject* of the corresponding sentence. To track immediate focus, we keep track of three variables: $CF$ — current focus, $PFL$ — potential focus list, and $FS$ — past immediate focus or focus stack. $CF$ is the focus of the current sentence whose initial value is the global focus; it might become the $CF$ for the next sentence to be generated. $PFL$ consists of the items referred to in the current sentence. $FS$ is updated each time the focus changes. The focus rules used to choose sentential subject are as follows (in order):

1. Shift focus to a member of previous $PFL$, i.e. $CF$ (new sentence) $\in$ $PFL$ (last sentence). In other words, if the current proposition² refers to any of the entities mentioned in the previous sentence in a non-subject position (e.g., direct-object, indirect-object, source, destination, etc.), choose the item thus referred to as the the surface subject of the current sentence. If there are more than one

---

¹That is, it has a subject, a verb and an object phrase with optional following prepositional phrases

²We assume each proposition corresponds to a sentence to be generated.
candidate, choose the filler of a default slot depending on the verb under consideration as the surface subject.

2. Maintain the same focus, i.e., CF(new sentence) = CF(last sentence). That is, the surface subject of the sentence is chosen to be the same as the surface subject of the immediately preceding sentence.

3. Return to a previous focus, i.e., CF(new sentence) \in FS. In other words, if the current proposition refers (in any case role) to a member of the focus stack, choose that item as the subject of the current sentence.

4. If none of the above three cases can be used, then
   - If the global focus is mentioned in the current proposition, choose it as the subject of the current sentence.
   - Otherwise, choose the filler of a default case slot (this default depends on the verb under consideration) as the subject of the current sentence.

It should be noted that these are preference rules only. Other preference rules may work as well, but these rules seem to satisfy our requirements. These rules give us a principled way to decide on the surface subject of the sentences in the text. We would like to note at this point that no other generation system has discussed how decisions are made regarding the placement of the fillers of various case roles (or, for that matter, components of any internal representation) in the surface sentence when multiple choices are possible. As far as we understand, these systems use the filler of a default case slot as the surface subject.

Change of focus from sentence to sentence, sometimes, may require the system to perform a view transformation on an event corresponding to a sentence in order to get the proper sentential subject. It also might dictate the use of passive voice in sentences in order to force an entity to the surface subject position of the resultant sentence.

**View Transformation on Chosen Events:** In order to generate a sentence with a particular participating agent as the sentential subject, it is necessary to perform a view transformation on the event. We have only one internal representation of the events — this is with respect to the secretary. In order to view the event from the perspective of another agent, we must generate another event which is a faithful representation of the same event from the other viewpoint. For example, the events corresponding to the sentences *John sold a book to Mary* and *Mary bought a book from John* are essentially identical — one should be derivable from the other as and when required.

Transformational mappings are performed among the case structures that represent events. A view transformation is thus a function from the set of case frame representations to itself. A view transformation takes an event representation corresponding to a base event and produces another event representation corresponding to a derived event. This transformation is guided by a specification of a case role whose filler is the agent from whose viewpoint the event is looked at. As a result of a transformation, the action slot of the event changes, and the relation of the various slot fillers to the central event needs to be modified.

Consider the simple illustrative example given below. The event is an instance of a send event. The base event *event-n1* is described internally as shown in the left hand side of figure 5. This representation is from the point of view of the secretary and can be translated to *The secretary received the paper from the author on January 10, 1986*, or to *I received the paper from the author on January 10, 1986* if the secretary is the designated speaker.

If we want to represent the same event from the author’s point of view, the transformation will modify it to a pseudo-event *event-n2* shown on the right hand side of figure 5 where T is a date which is the same as or is earlier than January 10, 1986. This can be translated to *The author sent the paper to the secretary*. It should be noted that the event is transformed from an instantiation of a receive event to an instantiation of a send event. The relations of the role fillers with the event are also redefined. Since the time slot is not fully specified in *event-n2*, it is not mentioned in the sentence.

The approach taken to view transformations here is simplistic. Receiving an object by an agent at a given time always implies that the object was sent to the agent by another agent at some prior time. However, the reverse inference need not be always true. It is only with some high probability that we can assume that the sending was followed by receiving. In other words, it is the default inference in the lack of any information to the contrary. We do not view receive/send as the same event looked at from two different perspectives. They are distinct events, and it is common-sense knowledge about the world which allows us to do the perspective transformation that is called for. That is, the transformations are not symmetric. The transformation from receive to send can always be done whereas in the other direction it is not always permissible. This is due to real world knowledge as well as the fact that that receive is a verb that places empathy on the subject [13], whereas send does not require that the speaker’s empathy be placed on the subject.

The transformation that needs to be performed can be declared in advance and stored in a table. There is an entry in this table for each event specifying how the roles need to be relabeled when the event is viewed by various participants. It should be noted that before translating into English the contents of the slots have to be ordered to obtain the appropriate sequence of sentential components in the final English text. This ordering depends on the filler of the action slot in the case structure. This order is declared *a priori* for each verb and stored in another table. For example, for the verb receive, the order is *(recipient action object source [date])*.

---

The case representation used and the fillers are simplified forms of the ones actually used in our system.

---

SPIE Vol. 937 Applications of Artificial Intelligence VI (1988) / 537
**Conciseness Considerations:** We have also investigated the processing required for generating concise reports, although implementation is not complete. The problem of what need not be mentioned in the text has been faced by others. All relevant events need not be mentioned in a report. This is because we know that the hearer is an intelligent agent and can make inferences. For example, Badler [4] had so much of information from the frames of the computer-generated movie, that he had to apply several stages of reduction in the amount of information to be conveyed in the generated text. This fact has also been observed by Mann and Moore [14].

In order to generate concise reports we have to understand the nature of the events in our domain. One theory we postulate, and which is in early stages of refinement, groups the events in our domain into a number of interacting event clusters. Each cluster contains a salient event, and optionally a set of events preceding it which may be called preparatory events and another set of events following it which we call confirmatory events. The salient event is the nuclear event in the cluster and the preparatory and confirmatory events are of secondary importance. These two categories of events can be understood only in terms of the salient event in the cluster. Preparatory events take place before the salient event and they prepare the environment for the salient event to take place. They may involve events which satisfy some immediate preconditions of the salient event, or they may be other events which have to precede the salient event but are of secondary importance from the point of view of reportability. Confirmatory events in our domain take place after the successful completion of a salient event. They serve the purpose of confirming the fact that the salient event has taken place, or they inform either the participating agents (agents in the salient event) or other agents in the domain the completion of the salient event. Examples of such event associations are shown in figure 6.

Each salient event can be reported as a complete sentence. If an event is a salient event, we need not include the corresponding preparatory or confirmatory events in the report. However, to establish the context for reporting the salient event, it may be necessary to include one or more of the preparatory events. When this is done, instead of using a whole sentence for the preparatory event, we can use a phrase in the sentence reporting the salient event. These details need further study. Furthermore, it is not clear how one can automatically determine which events are salient and which ones are accompanying preparatory or confirmatory events. This classification appears to be domain dependent although it may be possible to devise heuristics which may facilitate determining such associations among events.

![Figure 6: Examples of event association in our domain](image)

**Extensions for Generating Reports from Various Perspectives:** We have also investigated the issues involved in using the same historical knowledge to generate reports from the viewpoint of various participants in a consistent manner. Psychological experiments performed by Black, Turner & Bower [5], and Abelson [1] among others clearly demonstrate the influence of point of view on the conciseness of natural language text.

In order to be able to use the same historical knowledge base to produce textual reports from the viewpoint of various participants, we have to consider several issues, such as reportability criteria for choosing events, appropriate transformations on the relevant events stored in the history network in order to generate description of the events from the correct perspective, and generation of appropriate sentences based on the transformed events. Among the criteria that may be used to determine whether an event is relevant or reportable are direct or indirect involvement of the speaker and the hearer, their goals and responsibilities, limitations regarding what the hearer is allowed to know, and considerations regarding how the speaker thinks the event affects the hearer's goal. The speaker uses his/her knowledge of his/her and the hearer's goals and his/her knowledge of the objects and the actions performed by the various agents to determine if an action is relevant. Currently we are exploring these issues and plan to incorporate our findings to extend the existing system.

6. **THE TACTICAL COMPONENT**

The tactical component used by the system to generate actual text is MUMBLE [17]. The decision to employ MUMBLE was primarily influenced by its success in generating a large variety of grammatical constructs in several text generation applications as reported in [11,17,22], the flexibility in the design of its input as well as its ready availability. Its design is based on the assumption that division between planning the content of speech and its verbalization is unambiguously defined.

The input to MUMBLE consists of realization specifications (r-specs) representing the system's communicative goal, and produced by the planner (or the strategic component). Given the input r-spec, MUMBLE assigns (or, attaches) it to an appropriate position in an incipient surface structure tree. A depth-first traversal of the tree accompanied by recursive phrasal realization of unprocessed embedded elements results in the production of well-defined English text.
The tactical component improves the coherence in the generated text by appropriate lexical substitution for referring expressions allowing previously generated sentences to exert influence on the realization of the current sentence. The system keeps track of the objects which have been referred to so far and how this reference has been made. This along with focus information enables the system to refer to objects by incomplete descriptions and to introduce pronominalization. This avoids unnecessary repetition leading to succinctness in the text. More importantly, it enables the hearer to distinguish new information from old so that comprehension is not hampered. Otherwise, it may lead to misunderstanding on the part of the hearer; examples of this phenomenon can be seen in [9].

Referring Noun Phrases: Incomplete description of referring noun phrases in our system include usage of phrases such as “the author” or “the paper” in subsequent references instead of the complete phrases “the author D.D. McDonald” or “the paper entitled “Generation Using MUMBLE” which are used for introduction. Additionally, when a person introduced earlier by name is referred to subsequently at a point where pronominalization is deemed inappropriate, the person is referred to by his/her name. Similarly a group of people may be introduced by a phrase such as “the three referees, viz., B.L. Webber, A.K. Joshi and T. Finin”. Subsequently, the first time one of these persons is referred to alone, we refer to him/her by name. If the same person is referred to again, only then pronominalization is resorted to provided it does not lead to ambiguity. Incomplete description also enables us to use phrases such as “on the same day”, “yesterday”, and “today” instead of always producing the complete phrase such as “on January 18, 1986” at all times. Also note that, the first time the system specifies a date, it specifies the year. However, in subsequent specifications, the year is not mentioned unless it is different from that of the immediately preceding date mentioned.

Adapting MUMBLE to our system required substantial additional extensions for handling of cardinal numbers, proper names, various tenses, etc. A number of new structures had to be added to support the desired features in the final text. Some of these are simple, others complex. Some are general while others are domain specific. At this point, we feel that it is pertinent to corroborate the observations in [11] that it is difficult to create new structures that capture language generalizations due to the total absence of constraints on their nature.

Finally, we conclude this section by presenting examples of text produced by the system (assuming that the secretary is producing the report). The whole system has been designed such that we can perform simulation of events such as arrival of the paper, arrival of the reviews by the secretary. At any point during this simulation, the system may be asked to generate a report. Below, we reproduce two such reports — the first one was produced at an arbitrary point during the life of the paper. The second one was produced for the same paper after the paper was processed completely. In the reports presented here, conciseness issues have not been considered.

I received the paper entitled “Generation Using MUMBLE” from the author D.M. McDonald on January 1, 1986. He was informed of the receipt of the paper on the same day. I requested the names of three referees for the paper from the editor A.K. Joshi on the same day. He sent the names, viz., T. Finin, N. Badler and B. Webber a week later. I have sent a postcard inquiring availability to review the paper to each of them today.

Assuming we continue the simulation performed to the end and ask the system to produce a report again, the system generates the following text.

I had sent a postcard inquiring availability to review the paper entitled “Generation Using MUMBLE” to each of the referees, T. Finin, N. Badler, and B. Webber on January 15, 1986. Each sent a letter expressing ability to review the paper to me by January 20. I sent a copy of the paper to each of them. Each sent a review of the paper to me by February 15. I requested the editor to make a decision regarding publication of the paper on the same day. He sent a positive decision regarding publication of the paper to me yesterday. I informed the author of the decision yesterday.

7. CONCLUSIONS

In this paper, we have described a system which has attempted to apply artificial intelligence techniques to handle the requirements of a real-life situation. We have successfully modeled the various processes involved, simulated the process model, created an historical knowledge base, and produced suitably structured status reports in English based on the stored history. A prototype system has been implemented as discussed in the preceding sections on a SYMBOLICS LISP machine with a VAX-11/785 VMS host for file storage. However, the system reported here can be improved in many different areas in order to obtain a much more powerful and sophisticated system for report generation. In the rest of this section, we discuss avenues for such system enhancements. Issues which are currently under investigation include taking a more sophisticated approach to providing information about the past (when multiple reports are requested) will involve reasoning about saliency of events in order to select events from past history for reporting, the formulation of the actual text for summarization, drawing inferences about expected future events based on representation of current status and the system’s history, incorporation of explicit textual links between the contents of a sentence and that of its predecessor(s), etc. Furthermore, the quality of text produced can be improved substantially by incorporating a selection of commonly used temporal expressions in English.

ACKNOWLEDGEMENTS

The second author was partially supported by NSF grants MCS-8219116-CER, MCS-82-07294 and DCR-84-10413 during the course of his work on this project. We would like to thank Dr. Bonnie Webber for her painstaking reviewing of several drafts of the technical report on which this paper is based.

References


SPIE Vol. 937 Applications of Artificial Intelligence VI (1988) / 539
Appendix

The following is the input to the surface generator (or, output of the text planner) corresponding to the first report quoted in section 6. Please note that the speaker for the text generated is the secretary, and the hearer is presumed to be anyone else. Each sentence in the text corresponds to two sub-lists below. The first list sets up the current focus; the second list produces the actual text.

```
(CURRENT-FOCUS-IS (NAMED-OBJECT PAPER-1 ("Generation Using MUMBLE'")))
(RECEIVE (NAMED-PERSON SECRETARY-1 IB. Norman) (NAMED-PERSON AUTHOR-1 ID.M. McDonald)
(NAMED-OBJECT PAPER-1 ("Generation Using MUMBLE'")) (JANUARY 1 1986)) $SIMPLE-PAST$

(CURRENT-FOCUS-IS (NAMED-PERSON AUTHOR-1 ID.M. McDonald))
(ACKNOWLEDGE (NAMED-PERSON SECRETARY-1 IB. Norman) (NAMED-PERSON AUTHOR-1 ID.M. McDonald)
(RECEIPT-OF (NAMED-OBJECT PAPER-1 ("Generation Using MUMBLE'"))) (JANUARY 1 1986)) $SIMPLE-PAST$

(CURRENT-FOCUS-IS (NAMED-PERSON SECRETARY-1 IB. Norman))
(REQUEST (NAMED-PERSON SECRETARY-1 IB. Norman) (NAMED-PERSON EDITOR-1 IA.K. Joshi)
(REFERENCE-NAMEs (NAMED-OBJECT PAPER-1 ("Generation Using MUMBLE'"))) 3) (JANUARY 1 1986)) $SIMPLE-PAST$

(CURRENT-FOCUS-IS (NAMED-PERSON SECRETARY-1 IB. Norman))
(RECEIVE (NAMED-PERSON SECRETARY-1 IB. Norman) (NAMED-PERSON EDITOR-1 IA.K. Joshi)
(PERSON-NAMES-LIST (NAMED-OBJECT PAPER-1 ("Generation Using MUMBLE'")))
((NAMED-PERSON REFEREE-1 IT. Finin) (NAMED-PERSON REFEREE-2 IN. Badler)
(NAMED-PERSON REFEREE-3 IB. Webber)))
(JANUARY 8 1986) $SIMPLE-PAST$

(CURRENT-FOCUS-IS (NAMED-PERSON SECRETARY-1 IB. Norman))
(SEND (NAMED-PERSON SECRETARY-1 IB. Norman)
(PERSON-LIST (NAMED-PERSON REFEREE-1 IT. Finin)
(NAMED-PERSON REFEREE-2 IN. Badler) (NAMED-PERSON REFEREE-3 IB. Webber))
(POSTCARD-INQUIRY (NAMED-OBJECT PAPER-1 ("Generation Using MUMBLE")))
(JANUARY 8 1986) $SIMPLE-PAST$
```

540 / SPIE Vol. 937 Applications of Artificial Intelligence VI (1988)