A Model of the Time Course and Content of Reading*

ROBERT THIBADEAU
MARCEL ADAM JUST
PATRICIA A. CARPENTER

Department of Psychology
Carnegie-Mellon University
Pittsburgh, PA 15213

This paper describes a computer simulation of reading that is strongly driven by eye fixation data from human readers. The simulation, READER, is a natural language understanding system that reads a text word by word and whose processing cycles on each word have some correspondence with the human gaze duration on that word. READER operates within a newly developed information processing architecture, a Collaborative, Activation-based, Production System (CAPS) that permits the modeling of the temporal properties of human comprehension. CAPS allows for concurrent, collaborative execution of processes operating at different levels of analysis. As READER encounters each successive word, the word is operated on by processes at the levels of word encoding, lexical access, syntactic and semantic analysis, and referential and schema-level processes. Like human readers, READER uses a strategy of immediacy of comprehension, attempting to interpret each word as soon as it is encountered, rather than unnecessarily buffering information. A major contribution of this simulation is its use of human performance characteristics in constraining and determining the model’s mechanisms.

This paper describes a computer simulation of reading that is strongly driven by what we know about reading. The data consist of the readers’ gaze durations on each word of a text. People reading a text at the rate of 250 words per minute could spend a quarter of a second on each word. But they don’t. Instead, their time on different words shows considerable systematic vari-

*This research was supported in part by grants from the National Institute of Education (G-79-0119), the National Institute of Mental Health (MH-29617), the Sloan foundation, and the Office of Naval Research (N-00014-79-C-0661). We thank Allen Newell and Brian MacWhinney for helpful comments on earlier drafts of this paper. Requests for reprints may be sent to any of the authors. Copies of the CAPS system (for VAX/VMS or VAX/UNIX) on magnetic tape will be provided on arrangement.
ation, with gazes ranging from 150 msec to 1 sec, and with some words, mainly short function words, being skipped entirely. We have assumed that the gaze duration on a given word reflects the processing time consumed on that word. The gaze duration data were initially used to construct a theory of reading that indicated what types of processes occur, what variables affect the durations of the various processes, and how the different processes might interact with each other (Just & Carpenter, 1980).

The present report describes the next major increment in the theoretical development, a natural language understanding system that reads the text word by word, and whose processing time on each word corresponds to the human gaze duration on that word. The simulation, called READER, required the development of a system architecture that permitted the mapping between the human and machine processing times. This architecture is called CAPS because it is a Collaborative, Activation-based, Production System.

The paper has several sections. The first gives a brief overview of the simulation, as well as the theory and data from human readers that motivated the simulation. The second section describes the major features of CAPS, indicating how it differs from other production system architectures. The third section gives a detailed description of four major aspects of READER: the representational scheme, the word encoding and lexical access processes, the syntactic and semantic parsing processes, and the text integration processes.

AN OVERVIEW

Before describing the CAPS architecture and READER's properties, we present a general overview of what they accomplish together. Their final goal is to understand a scientific passage, one that is on the topic of *Fly-wheels* and is listed in Table 1. READER, of course, is more than a system to understand a single passage. The language-understanding mechanisms that READER uses are quite general and, except for lacking lexical information and a more complete parser, we believe that READER would be sufficient to comprehend general expository passages of this type. Moreover, we will show that the CAPS architecture is also general and provides a way to match human chronometric data with simulation models for a variety of complex processes.

At the highest level, READER uses a schema to organize the information contained in brief scientific passages that describe the mechanism underlying some man-made device or natural phenomenon. The MECHANISM schema is a general one that has slots for the various kinds of information to be expected in a passage of this type: the mechanism's name, its goal, its
physical properties, the operating principles that relate its physical properties to its goal, who made it, who uses it, some examples, and so on. After the passage has been read and understood, the schema, whose slots are now instantiated with the information from the passage, is used to retrieve the gist of the passage. READER produces a good, human-like, summary of the passage.

**TABLE I**
The Flywheel Passage that READER Processes

| Flywheels are one of the oldest mechanical devices known to man. Every internal-combustion engine contains a small flywheel that converts the jerky motion of the pistons into the smooth flow of energy that powers the drive shaft. The greater the mass of a flywheel and the faster it spins, the more energy can be stored in it. But its maximum spinning speed is limited by the strength of the material it is made from. If it spins too fast for its mass, any flywheel will fly apart. One type of flywheel consists of round sandwiches of fiberglas and rubber providing the maximum possible storage of energy when the wheel is confined in a small space as in an automobile. Another type, the "superflywheel," consists of a series of rimless spokes. This flywheel stores the maximum energy when space is limited. |

During comprehension, READER constructs a model of the referential world that is described in the passage, and uses it to interrelate the information presented in different sentences. Within-sentence relations are analyzed using a conceptual dependency analysis, similar to that proposed by Schank and his co-workers, that establishes the conceptual relations among the objects described in a sentence, relations like agent and object. The conceptual dependency analysis is accompanied by a grammatical analysis that searches for surface syntactic relations, such as subject, verb, object, prepositional phrase, and so on. READER also constructs and operates on lexical representations, since much of its knowledge is represented in terms of word meanings. These are accessed after an encoding operation that can identify subwords, prefixes and suffixes, and by default, novel words whose meaning must be inferred.

READER operates from left-to-right on the sentence, processing it a word at a time, and operating on several levels at once—the schema level, the model referent level, the conceptual-dependency analysis, the syntactic analysis, and the lexical level. Declarative knowledge is represented in propositional form. Each proposition has an associated level of activation (that roughly corresponds to the confidence in the proposition) that can be manipulated by the comprehension processes. The processes are realized as productions, condition-action rules that are executed whenever their enabling conditions arise in working memory. Several productions can execute at the same time. The conditions are always propositions and the actions always
create or delete propositions, or increase or decrease the confidence in an existing proposition by manipulating its activation level.

**Implementation.** As of November 1980, CAPS was a running system, written in MACLISP, on the PDP-10a in the Computer Science department at Carnegie-Mellon University. The psychological assumption that all CAPS cycles take the same amount of time is clearly untrue of the PDP-10, but it may possibly be true on other machines, such as the human brain. READER performs the functions described in the manuscript, although some changes have been made in READER since November 1980. READER has 225 productions, with about 20 of them designed for handling word recognition and lexical access, 30 for surface syntactic analysis, 15 for agreement and other forms of consistency checking, 85 for conceptualizations or case relations, and 75 for schema-level processes. This classification is somewhat arbitrary because some productions are multi-faceted.

**General Themes**

This paper is bound together by several themes that repeatedly appear, and which we have collected and present here as an organizational aid.

1. The *principle of immediacy*, a major backbone of human comprehension strategy, proposes that information is processed at several levels, lexical, syntactic and semantic, as soon as possible, rather than being routinely buffered. READER's use of the immediacy strategy distinguishes it from almost all other computer understanding systems.

2. Knowledge is represented in terms of propositions that have *continuous activation levels* denoting the extent to which each proposition is believed. Activation is directed from one known fact to another, as yet uncertain fact by the comprehension processes. Much of the computation operates on knowledge with subthreshold activation levels that may only gradually reach threshold as evidence accumulates for the proposition.

3. Skilled reading is performed by processes that are automatically evoked when their appropriate enabling conditions exist in the current state of knowledge. This *distributed control of processing* has been modeled by the formalism of production systems. There is *functional parallelism* among processes, so that operations of different types or of the same type can be executed concurrently, in temporally parallel streams of activity.

4. Comprehension is the result of the *collaboration* of various levels of processing, without relying on any single level. The various
"levels" are defined in terms of their function, not in terms of temporal sequence, although there are normative sequences between some pairs of levels. Also, the levels are not composed of isolated modules, but a collection of subprocesses that can communicate with subprocesses of the same or other levels.

**Previous Research on this Project**

Since the READER model and CAPS architecture were strongly driven by the human performance data, we will briefly describe the experimental procedure, data analysis, and results that motivated their development (Just & Carpenter, 1980).

**The Data.** The data consist of the measurements of time readers spent looking at each word while reading a scientific text. Fourteen college students read 15 passages, each approximately 130 words long, taken from the Science and Medicine sections of Time and Newsweek magazines. While the student read, his or her eye fixations were monitored so that it was possible to determine the location (within three character spaces) and duration (± 16.7 msec) of each fixation. At the end of each passage, the reader orally reported what he had read. The overall reading rate was approximately 225 words per minute. The nature of the data can be appreciated by examining Table 2, which shows the eye fixation protocol of one person reading the first two sentences of a passage about Flywheels. The protocol illustrates two important features of the eye fixations; first, readers fixated almost all of the content words, (83% on average) and not so many of the short function words like the, and, and by (only 38% on average). The second prominent feature is that the time on a word varied considerably from word to word. The total uninterrupted looking time on a word, called the gaze duration, was measured for each word, and then the average was computed over readers. The mean gaze duration on a word was 238 msec, with a standard deviation of 168 msec.

**The Data Analysis.** The goal of the data analysis was to determine which properties of the text affected the processing time on each word. For example, it is known that more frequent words generally take less time to recognize than less frequent words, and it is believed that frequency affects lexical access. By examining the function that relates frequency to gaze duration, we were able to infer some of the characteristics of the underlying lexical access process. We examined the effects of a number of variables through multiple regression techniques. One analysis attempted to account for the gaze duration on each of the 1,936 words in the 15 passages in terms of word-level processes affected by the perceptual, lexical, syntactic and
TABLE II
The eye fixations of a college student reading a scientific passage. The gazes within each sentence are sequentially numbered above the fixated words with the durations (in msec) indicated below the sequence number.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1566</td>
<td>267</td>
<td>400</td>
<td>83</td>
<td>267</td>
<td>617</td>
<td>767</td>
<td>450</td>
<td>450</td>
<td>400</td>
</tr>
</tbody>
</table>

Flywheels are one of the oldest mechanical devices known to man. Every

<table>
<thead>
<tr>
<th>2</th>
<th>3</th>
<th>5</th>
<th>4</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>616</td>
<td>517</td>
<td>684</td>
<td>250</td>
<td>317</td>
<td>617</td>
<td>1116</td>
<td>367</td>
<td>467</td>
</tr>
</tbody>
</table>

internal-combustion engine contains a small flywheel that converts the jerky

<table>
<thead>
<tr>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>483</td>
<td>450</td>
<td>383</td>
<td>284</td>
<td>383</td>
<td>317</td>
<td>283</td>
<td>533</td>
<td>50</td>
<td>366</td>
</tr>
</tbody>
</table>

motion of the pistons into the smooth flow of energy that powers the drive

<table>
<thead>
<tr>
<th>21</th>
</tr>
</thead>
<tbody>
<tr>
<td>566</td>
</tr>
</tbody>
</table>

semantic properties of each word. For example, the time to encode a word should be affected by some measure of its length, the time to access its meaning should be affected by its frequency of occurrence in the language, and the time to determine its syntactic role should be affected by its status vis-a-vis the preceding modifiers. The general regression equation for the duration on each word, indexed as (i), is:

\[
\text{Gaze Duration (i)} = \text{Intercept} + \sum_{m} (\text{(regression weight for variable m)} \\
\times \text{(value of independent variable m for word i)})
\]

where the independent variables code the properties of word i, such as: (a) length, (b) logarithm of its normative frequency (Kucera & Francis, 1967), (c) whether it is a novel word to the reader, and (d) whether it is the last word in a sentence.

Text integration processes may have effects that span more than a single word, so a second analysis tested for the presence of text-level effects, above and beyond those explained by word-level effects. It examined whether reading time was influenced by the nature of the information in a phrase or clause, such as whether it was a principle, a definition, an example, and so on. In this latter regression analysis, the dependent variable was the gaze duration on larger segments of text, and the independent variable was the text role of the segment, plus all of the variables for the word-by-word reading time for that segment (such as word length, frequency, and so on). The regression showed that readers spent more time on more important information, such as topics and definitions, and less time on details and examples.
To link the data to a model of comprehension, we made two major assumptions:

*The immediacy assumption:* The reader tries to interpret each word of a text as soon as his cognitive system encounters it, rather than routinely buffering information until more words have been read.

Immediate interpretation refers to processing at several levels, such as encoding the word, choosing one meaning of it, assigning a referent, and attempting to determine its status in the sentence and discourse. The evidence for the immediacy strategy is that when a reader encounters an infrequent word, a word whose interpretation requires a linking inference, a syntactically anomalous word, a semantic incongruity, or a word that introduces a new topic, there is an immediate increase in the processing time on that very word (see Carpenter & Just, in press, for a review). This means that the interpretation must have been occurring before any other words had been read. Sometimes, of course, no interpretation can be made before more words are read, and in such cases there is no recourse but to buffer the information. But the important contrast here is between relatively direct attempts to interpret information as it is encountered, as opposed to schemes that buffer information by default and do extensive processing only at the ends of phrases, clauses, or sentences, or schemes that involve multi-pass comprehension processes over units larger than a word.

The second assumption links gaze durations and processing times.

*The eye-mind assumption:* The eye remains fixated on a word as long as the word is being processed.

The assumption is that there is no appreciable lag between the time when the eye fixates a word and the time when the cognitive system encounters and processes the word. The eye-mind assumption, unlike immediacy, is specific to comprehension in the visual modality. The eye-mind assumption does not postulate that only the fixated word is being processed. Comprehension normally involves the use of information from previously read parts of the text, without necessarily making regressive fixations. Hence, the time on a word does not reflect the total processing time devoted to the corresponding concept. The assumption is that as successive words are fixated, the concomitant processing runs neither appreciably behind, nor ahead of the locus of fixation. There is some entailment between the two assumptions, such that eye-mind to some degree entails immediacy.

This section has provided an overview of the project. In the next section, we describe the CAPS architecture in more detail.

**THE CAPS ARCHITECTURE**

The reading model operates within a processing environment called CAPS. The motivation for designing CAPS was to create a medium that would
allow us to model both the time course and the content of a complex process. We will first list the main innovative features of CAPS, then discuss how this architecture fulfills the chronometric goals, and then return to a more detailed discussion of the CAPS features. The main features are:

1. CAPS is a production system whose procedural knowledge consists of productions that each specify enabling conditions and consequent actions.
2. The system's declarative knowledge base consists of propositional information in the form of concept-relation-concept triples, constituting a semantic network.
3. Every proposition has associated with it a numerical activation level or confidence value.
4. The conditions of all the productions are compared against the currently active knowledge in working memory and all the productions whose conditions are satisfied perform their action(s) concurrently. The condition elements of a production are specified not only in terms of presence or absence, but also in terms of a minimum activation level that a proposition must have before it satisfies the condition.
5. One processing cycle is defined as the matching process and consequent firing of all satisfied productions.
6. Once the enabling conditions for a production arise, the production will continue to fire on successive cycles until some event disables it.
7. Activation can be directed by a production from one proposition to another. The activation is directed to specific targeted propositions, rather than simply being spread by virtue of adjacency of propositions in a network. The amount of activation that is directed is some proportion of the source proposition's activation level. This proportion, called the activation weighting, is specified by the activating production.
8. The knowledge representations are fully accessible to all processes, so that different processes can collaborate by using each others' partial and final results. Lower-level processes can influence higher-level ones, and vice versa.

Some aspects of these features have appeared in other recently-proposed production system architectures (Anderson, Kline, & Lewis, 1977; Newell, 1980; Langley & Neches, 1981; Rosenbloom, 1980).

Modeling Chronometric Effects

CAPS provides a way to model chronometric effects by allowing a complex behavior of a computer model to be segmented into a sequence of episodes
and by providing a metric of the time expended on each episode. The features of CAPS that permit chronometric modeling are:

CAPS defines a psychologically meaningful machine cycle as the measure of processing time. Every CAPS cycle is assumed to take the same amount of time, and thus the count of CAPS cycles provides a measure of the duration of the processing activity.

CAPS permits the modeling of the time course of any mix of serial and parallel processes.

CAPS can represent not only discrete processing effects, but continuous effects as well by varying the activation level of a given piece of knowledge incrementally.

CAPS permits us to construct a model of natural language understanding whose "processing time" on successive words of a text is intended to correspond to relative human processing times on those words. The human processing duration on a word can simply be measured in terms of elapsed time. For a CAPS model, a processing cycle provides a primitive unit of time, and the duration spent on a given processing episode (such as the processing of one word) can be measured in terms of the number of CAPS cycles.

Three main factors can influence the number of CAPS cycles that READER spends on a word. First, sequential dependencies among productions set a lower bound on the number of cycles required to complete a given activity. If production B cannot fire until A has fired because one of B's conditions is one of A's actions, then it will take at least two cycles until B has fired. One example in the READER model is that the processes that derive the lexical interpretation of a word generally must wait until the encoding of the printed word has been completed.

The concurrent aspects of CAPS can also affect the time course. Since the processing on a given word continues until all the levels of analysis have been completed, the total number of cycles on a word will be determined by the longest process (i.e., the one requiring the most cycles to reach completion). As the number of processes increases, the probability increases that there will be a long one, that is, one involving many cycles. For example, if two words are processed identically, except that one of them initiates schema integration while the other one does not, then the one initiating schema integration may require more cycles.

The third major factor affecting the number of cycles on a processing episode is related to the activation levels of the knowledge structures. READER's knowledge often is accrued gradually, by means of one or more productions repeatedly incrementing the activation level of a given proposition over several CAPS cycles. This can be thought of as a growth process, with the extent and rate of growth varying with circumstances. The number of cycles it will take for the proposition's activation level to reach a thresh-
old depends on the proposition's base activation level, the threshold level to be reached, and the rate of increase per cycle (which will be explained in more detail in the description of the activation mechanism).

**Directed Activation.** Productions can increment or decrement the activation level of propositions, and they can also add or entirely delete propositions. This means that one piece of knowledge can evoke another, to a particular degree. This applies within levels of processing (e.g., one syntactic fact can evoke another) and between levels (e.g., lexical, syntactic, semantic, and schema-level knowledge can evoke each other). When a production fires, activation is propagated from a source proposition to one or more targeted propositions. The proposition that is the source of the directed activation is usually but not always one of the condition propositions. The amount of activation that is added is a proportion of the activation level of the source proposition. The proportion is called the *activation weight* associated with a production, distinct from the activation levels associated with propositions. The propositions that are the designated targets of the activation are defined by the productions that modify the activation levels, *not* by their position in a knowledge network. Thus, the rate of increase of activation of a proposition depends upon the activation level of the source proposition, because if one fact leads you to believe in another fact, your confidence in the new fact may depend on your confidence in the first one. It also depends on the activation weight, by which the production multiplies the source activation level before adding it to the target proposition. Table 3 shows a schematic representation of a CAPS model as it creates propositions and modifies activation levels over three cycles. For the two productions in this example, we have assumed that one of the condition propositions is the source of the directed activation.

Directed activation can be contrasted with spreading activation—a mechanism used in other models (Anderson, 1976; Collins & Loftus, 1975). Directed activation is propagated from a source proposition to other propositions, as specified by the procedural knowledge. By contrast, spreading activation is propagated from a source node to related nodes along all the links emanating from the source in the declarative knowledge base. Directed activation is propagated to only one other set of propositions; no second-generation propagation occurs unless explicitly specified. Spreading activation spreads in outward extending plies until some criterion is met. Finally, directed activation is manipulated by productions, just as are all the other processes. Spreading activation is often treated as an autonomous, qualitatively different process that is accomplished by some special mechanism.

**Self-Activation.** Propositions can also direct activation to themselves. Self-activation of a mental representation is an old concept in psychological theories, from Hartley's "vibrating" ideas to Hebb's reverberating cell
### TABLE III
A Representation of Productions and Their Activity Over Time in CAPS

<table>
<thead>
<tr>
<th>Production Name</th>
<th>Condition(s)</th>
<th>Activation Threshold</th>
<th>Activation Weight</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>X:</td>
<td>(Proposition A)</td>
<td>.5</td>
<td>.4</td>
<td>(Proposition B)</td>
</tr>
<tr>
<td>Y:</td>
<td>(Proposition B)</td>
<td>.8 &amp;</td>
<td>.5</td>
<td>(Proposition D)</td>
</tr>
<tr>
<td></td>
<td>(Proposition C)</td>
<td>.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>•</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>•</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>•</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

### WORKING MEMORY

<table>
<thead>
<tr>
<th>Time</th>
<th>Processes</th>
<th>Structures (with Activation Levels)</th>
</tr>
</thead>
</table>
| CYCLE 1 X: | (Proposition A) 1.0 → (Proposition B) | (Proposition A) 1.0  
                        | (Proposition C) 0.8                 | (Proposition B) 0.4  
                        | (Proposition C) 0.8                 | (Proposition C) 0.8 |
| CYCLE 2 X: | (Proposition A) 1.0 → (Proposition B) | (Proposition A) 1.0  
                        | (Proposition B) 0.8                 | (Proposition C) 0.8 |
| CYCLE 3 X: | (Proposition A) 1.0 → (Proposition B) | (Proposition A) 1.0  
                        | (Proposition B) 1.2                 | (Proposition C) 0.8  
                        | (Proposition B) 0.8                 | (Proposition D) 0.4  |
| Y:        | (Proposition C) 0.8 → (Proposition D) | (Proposition C) 0.8  
                        | (Proposition B) 0.8                 | (Proposition D) 0.4  |

*Long-term memory also contains propositions constituting the declarative knowledge base.*
assemblies, that has been used to explain the persistence of a mental representation when the external stimulus is no longer present. READER uses self-activation to explain the gradual emergence of a targeted proposition to some threshold level, until it is recognized as being present. Directing activation from a proposition back to itself with some weighting (if the weight is positive and fractional) will gradually increase its activation level to threshold, at which point feedback stops the self-activation process. In fact, READER's growth of knowledge proceeds at a rate that corresponds to certain human performance characteristics, a point we will return to in the discussion of lexical access. If the directed activation weighting is negative, a self "deactivation" process can gradually dampen a proposition that is no longer relevant.

**Concurrent Processes.** The CAPS architecture permits any number of productions to fire on the same cycle. In the case of the READER model, this includes productions that process lexical, syntactic, semantic, or schema-level information. For example, after encoding and accessing the word "engine" in the sentence "Every internal-combustion engine contains...", productions begin working on the syntactic analysis (determining that it is a noun that is the head of a noun phrase), and the referential analysis (determining that there is no previously mentioned engine and establishing a model-referent for this one). Each level of processing may involve a sequence of productions executed over successive cycles, but the streams of the various levels are concurrent. These concurrent streams of processing are schematically shown in Figure 1. Some recent language understanding systems such as the HEARSAY program (Reddy & Newell, 1974) have also explored some forms of concurrence, but among much larger modules (such as complete syntactic and semantic analysis) than individual productions.

**Collaboration.** The processes in READER collaborate to understand a piece of text by activating their intermediate hypotheses or confirmed knowledge so that it immediately becomes available to all the other processes. Since processes at all levels will act on any relevant knowledge regardless of its source, the potential for mutual influence is inherent in the production system flow of control. So one form of collaboration occurs when one process generates information that is useful to another process. Another form of collaboration occurs when two or more processes converge on establishing a given piece of knowledge by jointly incrementing its activation level to threshold, and when neither process alone may have effectively done so. For example, syntactic and semantic productions can collaborate on establishing the nature of a noun phrase.
Figure 1. A depiction of the major levels-of-processing in READER that are operating after the reader fixates the word "engine" in the text.
Control Mechanisms

There are several features of the system that maintain a disciplined flow of processing in CAPS.

*Subthreshold Condition Elements.* The condition element of a production can specify not only that a given proposition be present, but can also specify the degree to which the proposition must be present. Condition elements specify the minimum activation level a proposition must have before it will satisfy the condition. This feature allows for subthreshold computations, and permits some productions to fire when their enabling conditions are met with only partial confidence, while others’ conditions must be met with full confidence before the production will fire.

*Subthreshold Resolution of Conflict.* Often, one group of productions is bringing one proposition to threshold by repeatedly incrementing its activation level over cycles, while another group of productions is activating another, incompatible proposition. For example, two alternative subthreshold interpretations of a word-meaning may be vying for ascendancy. If one candidate reaches threshold before the other, then it becomes the accepted interpretation, and the other is deactivated. Much of READER’s decision making about alternative interpretations of the text is made by this process.

This method of resolving a conflict before it reaches threshold resembles a Perceptron device (Minsky & Papert, 1972), in that a linear integration of activations provides the basis for discriminating among competing choices. However, in Perceptron devices, the pathways between activation sources and the targets are fixed over time. In CAPS, by contrast, all numerical integration is controlled by productions that behave contingent on their conditions being satisfied in the current context. For example, the activation weights themselves (in addition to the act of incrementation) can be contingent on the context. Thus a CAPS model can alter its basis for discrimination according to the context, giving it more power than a standard Perceptron device.

*Conflict Resolution.* If two or more mutually inconsistent propositions rise above threshold, then both of them are deactivated, resulting in a lack of knowledge about the problem to which they were the multiple answers. If this ignorance is tolerable because no goal requires that the problem be solved, then READER simply proceeds to the next word in spite of its ignorance. This mimics the human inability to understand difficult or contradictory text in a brief reading. However, if a suprathreshold conflict arises on a part of the text that it is important to understand (because there is an explicit goal to be satisfied), then the text is reread, possibly with a global reduction in activation weights.
Global Changes in Activation Weights. There is an action that globally alters the weights used in directed activation. The weights are initially set to small, fractional, positive values so that it takes several CAPS cycles to bring a proposition’s activation level to threshold. The global change decrements or increments all the weights uniformly by executing the RE-WEIGHT action to slow down or speed up its processing, depending on whether it is reading what it considers to be an important or unimportant part of the text. Slowing down can be construed as processing with more care. In fact, the quantitative change in activation weights produced by RE-WEIGHT can evoke a qualitative change in processing, because RE-WEIGHT alters the relative time course of various processes. We will return to this issue in the discussion of READER’s tuning its processing rate to the part of the text it is reading.

Specifying Condition Elements in General Terms. A knowledge element that is an instance of a generalized condition will satisfy that condition. For example, READER has the declarative isa knowledge that a mechanical object is man-made. Any production whose condition involves a man-made device will automatically be satisfied by a mechanical device, because the isa knowledge is used implicitly by the production system interpreter that matches condition elements of productions to the elements of working memory. Thus, conditions can be specified in general terms and be satisfied in one of very many ways.

Feedback for Productions. Once the enabling conditions for a production arise, the production will continue to fire on successive cycles unless something disables it. There is no short-term refractory period, so a production does not become fatigued after firing for one or two cycles. One form of feedback control occurs if a production’s action removes the enabling conditions. Another form of feedback control is provided by a production’s goal-absence condition, that allows the production to fire on consecutive cycles until the goal is met. Such productions have as one of their conditions, the absence of their goal. Once the goal is met, the goal-absence condition is no longer satisfied and the production stops firing. For example in READER, the iterative self-activation of word-knowledge will continue only as long as READER does not know what the word-concept is. As soon as the activation level of the word-knowledge reaches threshold, the goal-absence condition will no longer be satisfied, and the production will stop firing.

Production Fatigue. A production will continue to fire on consecutive cycles as long as its conditions are met, but will become fatigued and stop firing if it does not reach fruition within a given number of cycles (currently set at 16). That is, a production (acting either alone or with some collabora-
tion) that has not succeeded in promoting its action to threshold within 16 cycles is temporarily disabled.

This section has described the major features of CAPS. The next four sections will describe how the READER model operates within the CAPS architecture, beginning with a discussion of the representational conventions in READER.

**READER: THE REPRESENTATION**

The representations in the working memory have an attribute-value structure, and use some global conventions that are common to all levels of processing. All propositions are constructed using only two fundamental relations, :HAS and :IS. The :HAS relation indicates that two symbols are joined by a dependency, such that one symbol has its significance elaborated by another. For example, to represent the syntactic relation between the subject and verb of a simple clause like "Engine contains," the dependency would be expressed as: (WORD2-CONTAINS :HAS WORD1-SUBJECT1-ENGINE). Often, dependencies between two concepts can be established in both directions, with respect to different relations, of course. The dependency relations can be extensively embedded, permitting arbitrarily large structures to be built, such as a representation of a sentence that has embedded within it the representations of words, word meanings, and syntactic relations.

The second fundamental relation, called :IS, denotes equivalence between two symbols. In the example above, the equivalence relations between WORD1 and ENGINE or between WORD2 and CONTAINS are not a part of the dependency structure per se, but reflect the use of different symbols to stand for the same concept. In the course of comprehension, two or more processes may derive information about a concept that has been denoted differently in two places, without initially having access to the fact that it is the same concept. Thus, more than one symbol may stand for the same concept. When this equivalence relation is later detected, the multiple symbols standing for the same concept are linked by :IS, denoting equivalence. These procedures somewhat resemble processes used by people who learn several facts about the same individual who is referred to by different labels (Anderson, 1977).

While all the levels of processing use the :IS and :HAS relations to construct dependency structures, the arguments in the structures are often level-specific. At the word level, these are symbols like WORD-PERCEPT that represent the visual (letter) features of a word percept, such as (WORD-PERCEPT1 :HAS FEATURE1) (FEATURE1 :IS C), and word-concepts that point to known words, such as (WORD5 :IS CONTAINS). At the syn-
tactic level, symbols like SUBJECT, OBJECT, HEAD (of prepositional phrase) organize the word concepts into syntactic units (e.g., (WORD2 :HAS SUBJECT2) (SUBJECT2 :IS WORD1)). At the semantic level, conceptual predicates organize entire conceptualizations, such as (*CONTAIN*1 :HAS CONTAINEE1) (CONTAINEE1 :IS FLYWHEEL2). (The conceptual predicates are denoted by asterisks.) Finally, at the text level, various conceptualizations fill the schema slots for the components of a discussion about mechanisms. For example, one of the principles of operation of a flywheel may be represented as (FLYWHEEL1 :HAS RULE1) (RULE1 :IS *CAUSE*3), where *CAUSE*3 is a pointer to an entire conceptualization. Some of the properties of the representational structure can be illustrated by considering a partial representation of the sentence “Flywheels are mechanical devices.”, shown in Table 4. The activation levels are not shown, but the example illustrates the various types of arguments for the different levels of processing and the common attribute-value structure.

### Table IV
A Partial Representation of the Sentence “Flywheels are Mechanical Devices.”

<table>
<thead>
<tr>
<th>Slot Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(WORD1 :IS FLYWHEELS)</td>
<td></td>
</tr>
<tr>
<td>(WORD1 :HAS MODEL-REFERENT1)</td>
<td></td>
</tr>
<tr>
<td>(MODEL-REFERENT1 :IS FLYWHEEL1)</td>
<td></td>
</tr>
<tr>
<td>(WORD2 :IS ARE)</td>
<td></td>
</tr>
<tr>
<td>(WORD2 :HAS SUBJECT2)</td>
<td></td>
</tr>
<tr>
<td>(SUBJECT2 :IS WORD1)</td>
<td></td>
</tr>
<tr>
<td>(WORD2 :HAS OBJECT2)</td>
<td></td>
</tr>
<tr>
<td>(OBJECT2 :IS WORD4)</td>
<td></td>
</tr>
<tr>
<td>(WORD2 :HAS CONCEPT2)</td>
<td></td>
</tr>
<tr>
<td>(CONCEPT2 :IS <em>EXAMPLE-OF</em>1)</td>
<td></td>
</tr>
<tr>
<td>(<em>EXAMPLE-OF</em>1 :HAS EX-OF-SUBJECT1)</td>
<td></td>
</tr>
<tr>
<td>(EX-OF-SUBJECT1 :IS FLYWHEEL1)</td>
<td></td>
</tr>
<tr>
<td>(<em>EXAMPLE-OF</em>1 :HAS EX-OF-OBJECT1)</td>
<td></td>
</tr>
<tr>
<td>(EX-OF-OBJECT1 :IS DEVICES1)</td>
<td></td>
</tr>
<tr>
<td>(WORD3 : IS MECHANICAL)</td>
<td></td>
</tr>
<tr>
<td>(WORD3 :HAS ADJECTIVE-MATE3)</td>
<td></td>
</tr>
<tr>
<td>(ADJECTIVE-MATE3 :IS WORD4)</td>
<td></td>
</tr>
<tr>
<td>(WORD4 :IS DEVICES)</td>
<td></td>
</tr>
<tr>
<td>(WORD4 :HAS MODEL-REFERENT4)</td>
<td></td>
</tr>
<tr>
<td>(MODEL-REFERENT4 :IS DEVICES1)</td>
<td></td>
</tr>
<tr>
<td>(WORD4 :HAS PUNCTUATION4)</td>
<td></td>
</tr>
<tr>
<td>(PUNCTUATION4 :IS <em>PERIOD</em>)</td>
<td></td>
</tr>
</tbody>
</table>

aEX-OF-SUBJECT1 symbolizes the conceptual relation that SUBJECT1 is an example of something.

bADJECTIVE-MATE denotes the slot for the nominal created by the adjective.
The attribute-value form of the representation is sometimes used to express a need for certain kinds of information that is then sought by appropriate productions. For example, one could indicate that "WENT1 needs an agent but does not yet have one," by representing (WENT1 :HAS AGENT1) without representing what AGENT1 is. This representation creates a "slot" or "attribute" without a value, which by convention acts as a request for a value. A similar mechanism is used by various "slot and frame" grammars (e.g., McCord, 1980). It is useful if decisions have to be made on not only how to fill slots, but whether slots need to be filled in the first place.

The semantic predicates that READER uses make certain high-level conceptual relations very accessible. For example, the *CONTAIN* predicate symbolizes the containment relation in all three of the following excerpts: "every internal-combustion engine contains a flywheel," "the more energy can be stored in it," and "when the wheel is confined in a small space." The single *CONTAIN* predicate highlights the commonalities of "contains," "stored," and "confined," at the cost of making the distinctions less accessible. Such predicates have been called compressionistic because they constitute a compression of lower-level meaning elements (Rieger, 1979). By contrast, atomistic predicates attempt to symbolize concepts directly in terms of lower-level meaning elements, typically a small number of semantic primitives, such as Schank's PTRANS (Schank, 1972). It is becoming clear from various natural language understanding systems that the choice between the two kinds of predicates is dictated by the type of processes that will most commonly use them. For READER's purposes, the compressionistic predicates are more convenient for relating the sentence-level information to a schema.

READER numbers concepts as they are encountered (e.g., WORD4), reflecting both a psychological claim and a computational argument. The psychological claim is that human readers encode at what location or at what relative time they encountered a particular word. At least for a brief time, and in some cases for a long time, readers know where they read a particular word or fact. The supporting evidence is that readers make accurate regressive fixations to previously read words that are sources of inconsistency or ambiguity (Carpenter & Daneman, 1981; Carpenter & Just, 1977), or to retrieve facts from a previously read text that is visually available (Christie & Just, 1976). Although the use of integers per se is not psychologically plausible, they provide a satisfactory first approximation of the time-space encoding.

The computational convenience of numbering concepts is that it is sometimes useful to keep different instances of the same proposition distinct. READER keeps two instances distinct by appending a number to a type, as in FLYWHEEL3 for a particular occurrence of FLYWHEEL, and
then attending to the numerical suffix, if necessary. Concept numbering facilitates collaboration among processes by providing an indexing scheme for computing coreference and other types of syntactic relations.

**Forgetting in Working Memory.** READER does not forever store every piece of the representation it constructs. While there is no general decay function, three main strategies and capacity limitations cause certain parts of the representation to be forgotten. First, an interpretation is forgotten (deactivated) if it does not reach threshold (because it loses the race to threshold to a competing interpretation). Second, two suprathreshold interpretations that are inconsistent with each other are both deactivated and remain so, unless the text is reread with the benefit of additional information or a slower processing rate. And third, the word-percepts associated with each sentence (i.e., the verbatim information) are forgotten at the end of each sentence because the word-percept encodings of the next sentence reuse the numbered word-percept slots. In contrast to the forgetting of low-level information and subthreshold information, the higher-level suprathreshold information is not deactivated or overwritten, although interference from other information or retrieval failure may make it inaccessible.

**WORD ENCODING AND LEXICAL ACCESS**

Reading a word involves translation from a printed symbol on a page to an abstract representation of the corresponding concept in the reader’s mind. It is commonly believed that in human readers, this transformation involves two main components: (1) the process of perceptually encoding the printed word; (2) the process of accessing its meaning in a mental lexicon (e.g. Glanzer & Ehrenreich, 1979). In READER, perceptual encoding productions take the visual features as conditions, and their action is to activate word-percepts. Lexical access productions use word-percepts as conditions, and they activate word-meanings.

One source of evidence for the distinction between encoding and access is that experimental variables associated with the two processes have additive effects on human performance. For example, the print quality of a word (degraded versus normal) is assumed to affect encoding, while a word's normative frequency of occurrence (frequent versus rare) is thought to affect lexical access. In fact, print degradation and normative frequency have additive effects on response time in lexical decision tasks, in which subjects must decide whether a stimulus item is a word or a non-word (Becker & Killion, 1977; Stanners, Jastrzembski, & Westbrook, 1975). Another source of evidence for the separability comes from the additive effects of word length and word frequency on the time readers spend looking at each word.
of a scientific text (Carpenter & Just, in press). These results suggest that word encoding and lexical access are performed by two different processes, the former sensitive to the physical properties of a word (such as print quality and length), and the latter sensitive to its frequency of usage. Furthermore, the additivity of the effects can be modeled in READER by making the lexical access productions contingent on the information that is usually provided by the encoding productions. This is an instance of two processes generally operating in sequence because of the nature of their functions, even though the architecture permits concurrence, or even precedence. Thus, if there is extremely strong evidence about the meaning of a word that has not yet been encoded, then the usual sequence can be bypassed, and the word’s meaning can be activated directly.

**Word Encoding Processes**

It is assumed that prior to the operation of the word encoding productions, the representations of the visual features of the word have been deposited in working memory. If we assume the visual features correspond to letters, then the featural representation would indicate the identity and sequence of the letters in a word, so the word “one” would be represented as (FEATURE1 :IS O), (FEATURE2 :IS N), (FEATURE 3 :IS E). Then the word encoding productions produce a word-percept: (WORD-PERCEPT3 :IS ONE). The presence of the word-percept in working memory indicates that the word, but not necessarily the word-meaning, has been identified.

**Word Length.** The time people take to encode a word (as measured by the gaze duration) increases with the word’s length. Length can be measured in several ways: the number of characters, the number of syllables, the number of sub-word units (prefixes, suffixes, subunits of a compound word), or combinations of these measures. Just and Carpenter (1980) argued for syllable length, both on theoretical and empirical grounds, but character length is a slightly better predictor of word-gaze durations than syllable length. The correlation among these various measures of length is sufficiently high that we will not attempt to discriminate the “correct” measure. READER can function equally well with characters or syllables as encoding units, but the current implementation uses characters. READER encodes the visual features of the characters sequentially (i.e., one character per cycle), producing the linear word-length effect found in human gaze durations. Table 5 shows a trace of the encoding process, cycle by cycle, for one word, in this case, “contains.”

The encoding processes operate on the visual features to produce percepts composed of orthographic or syllabic structures or, in most cases,
TABLE V
Description of READER’s encoding and lexical access of the word “contains” in the sentence “Every internal-combustion engine contains a small flywheel...”

Cycle 347
The processing of the word “engine” is completed since no further productions are satisfied. So the command to fixate the next word (“contains”) is added to working memory.

Cycle 348
The next word is fixated, resulting in a registration of the features in the word in a form that is available for further processing by READER. The command to fixate the next word is removed.

Cycles 349-355
The first 7 letters are processed one by one by shifting focus across the FEATURE set, and encoding C-O-N-T-A-I-N.

Cycle 356
The focus is on the S in “contains.” The first 7 letters are recognized as a sub-word, CON-TAIN. This also acts as a request to check if the remaining letters constitute a sub-word.

Cycle 357
The final S is recognized as a sub-word suffix.

Cycle 358
Enough evidence is available to construct the proposition that (WORD5 : IS CONTAINS), with a low initial activation level.

Cycles 359-372
The proposition that (WORD5 : IS CONTAINS) is repeatedly activated (by directing activation to itself) until its activation level reaches the standard threshold. The number of cycles necessary to reach threshold depends on the base-activation level of the word-concept CONTAINS.

Cycle 373
At threshold, all the stored meanings and associations of the word-concept become available in working memory, including the fact that it is a verb, singular, is usually a physical property, has a container and a containee, and so on.

whole words. These percepts are representations of letters or letter clusters that can be matched to entries in the lexicon. Matches to lexical entries repeatedly activate internal word concepts on successive cycles until at least one concept reaches threshold (as detailed below, in the description of lexical access). The matching process is accompanied by operations that attempt to segment prefixes and suffixes, and work with the remaining symbol (Taft & Forster, 1976). There is some left-to-right bias in constructing the word percept. Two of the segmentation productions are:

- Sub-word percepts that correspond to known word concepts are recognized as the word features are encoded from left to right, provided that the complement of any subword is also a subword. For example, the word-percept corresponding to “flywheels” is decom-
posed into "fly" and "wheel". Since "wh" is not a word, the word "eel" is not developed. Subword concepts are activated at a much lower level than whole-word concepts.

- A word-percept composed of a subword and an affix that both correspond to known word concepts is decomposed into the sub-word percept and the affix-percept. Thus the "er" is treated as a suffix in "worker" but not in "her".

The segmentation and recognition processes have implications for syntactic and semantic processes as well. For example, if READER needs to fill a verb slot and encounters a word with the suffix "-ing," then recognizing the suffix contributes to identifying that word as a verb.

There are some minor ways in which READER's encoding of words is knowingly different from human readers. First, READER encodes all the content words of the text, while humans fixate about 80%, and READER encodes almost all function words, while humans fixate only about 40%. There are several mutually compatible accounts of why humans don't fixate some words. One reason is that human readers can probably encode short words, particularly function words like the, and, and of, in parafoveal or foveal vision while fixating on the rightmost part of the preceding word (Rayner, 1975; Carpenter & Just, in press). Another reason for not fixating certain words is that they may be inferred easily on the basis of the context, so there is no need to encode them. Finally, some non-fixated words might be ignored. Quite simply, we do not yet know the fate of the words that are not directly fixated by human readers.

READER has no peripheral vision but does skip over words it can infer with high certainty, which turn out to be some of the short function words, like of. Since our primary focus has not been on the perceptual aspects of reading, we have given READER no peripheral vision. This does not do much violence to simulation of human readers, and adding peripheral vision to READER would not be inconsistent with its operation. In fact, a newer version of READER is beginning to use this ability in skimming a text.

**Lexical Access Processes**

The lexical access processes activate word meanings that are associated with the encoded word percepts. At this point, other processes can help raise the activation level of one or another word sense. Moreover, productions whose conditions involve knowing some word meaning are now enabled. This activation process somewhat resembles Morton's (1969) logogen model, in that subthreshold word meanings are activated to threshold level by an accumulation of appropriate information. Thus, lexical access is seen more as a sampling process than a search.
**Word Frequency Effects.** The gaze duration data clearly indicate a linear relation between the logarithm of a word’s normative frequency and its received gaze duration, with longer times associated with less frequent words (Carpenter & Just, in press). READER’s lexical access time is also logarithmically related to normative frequency, using an interesting mechanism that is compatible with the CAPS architecture. READER’s representations of word meanings have base-activation levels that are directly proportional to the words’ normative frequency. (In principle, these should be proportional to the frequency of the word meanings, and not the words, but counts of word meanings are unavailable.) Human readers also seem to possess this type of frequency information. When asked to make subjective estimates of word frequencies, subjects produce ratings that are linearly related to normative frequency (Tryk, 1968). The knowledge may have resulted from previous exposures to the word, which in turn is generally closely related to normative frequency.

READER produces the logarithmic effects using the “veridically” distributed baseline activations, and a nonlinear, short-term activation scheme. The activation scheme involves the self-activation of the proposition, as described previously. There is a production that says, roughly,

-If there is uncertainty (a subthreshold activation level) about what the current word-concept is (say, (WORD5 :IS CONTAINS)), then take the current confidence (activation level) that (WORD5 :IS CONTAINS) and multiply it by the activation weight (1.0 in this case) and add that number to the current activation level. Reiterate this process until the activation level reaches threshold.

This representation and process yield logarithmically distributed lexical access times for linearly distributed baseline activations. Starting with the exponential which characterizes the activation, Y, after N cycles, and with the baseline activation level b, and the activation weight (proportion) w, we have:

\[ Y = b(1 + w)^N, \]

or alternatively, taking logarithms,

\[ K \log(Y) - K \log(b) = N \]

But, Y is set to the conventional threshold value and is therefore constant in READER. Therefore N, the number of READER cycles needed to reach threshold, is linearly related to the log of the baseline activation, just as the human gaze duration is.

**Lexical Ambiguity.** The final interpretation of a word can be influenced by all aspects of the word’s context. The semantic and syntactic con-
text that precedes a polysemous word will differentially activate its various senses, and its various senses will have different base levels of activation, corresponding to their relative frequency of occurrence. READER disambiguates simply by choosing the sense that first reaches the criterial activation level. In keeping with the principle of immediacy, this selection occurs when the word is first encountered, as opposed to waiting for disambiguation by later parts of the text. This implies that occasionally incorrect selections will be made, but READER has other heuristics that make error recovery relatively straightforward. This approach to polysemy is dictated by recent eye fixation studies of reading of garden path sentences (Carpenter & Daneman, 1981). That research suggests that human readers choose a meaning while fixating a polysemous word, but if the selected meaning later turns out to be incorrect, they have efficient strategies for recovery. The method of dealing with multiple alternatives outlined here for polysemy is also used to resolve ambiguity at other levels of processing.

**Novel Words.** In scientific passages, there are some words that are totally novel to the reader. Human readers spend considerable time on such words, an average of 800 msec above what could be expected purely on the basis of the word's length and infrequency. In READER, there is an initial attempt to identify the word on the basis of sub-word units. If that fails, READER establishes a new lexical entry with the syntactic and semantic properties inferrable to that point associated with the new word's perceptual representation. It is assumed that the unsuccessful parsing process, establishing the new lexical entry, and consolidating its properties consumes most of the 800 msec.

**Repetition Effects.** If a word is encountered more than once in a lexical decision task (Scarborough, Cortese, & Scarborough, 1977) or in a text-search task (Dixon & Rothkopf, 1979), then it is usually processed faster on the second encounter, and this facilitation is generally attributed to savings in the lexical access process. Word repetition effects have not been examined in any task as natural as reading prose. An initial analysis of the gaze duration suggested that the second and subsequent times that an infrequent word is read take 49 msec less than the first time (Just & Carpenter, 1980), but this characterization is not a good one. A repetition effect exists, but only for words directly related to the topic of the passage, and irrespective of the word's normative frequency. The gaze duration on the topic words in all of the passages decreases from the first occurrence to all subsequent occurrences, by an average of 181 msec. Thus, most of the word repetition effects observed in normal reading seem attributable to topic repetition. READER's text integration processes (described later) account for this phenomenon.
PARSING: SEMANTIC AND SYNTACTIC ANALYSIS
OF SENTENCES

READER not only processes individual words, but also has a collection of parsing productions that transform an input stream of word concepts into an unordered conceptual representation of the sentence, and a representation of the things the sentence refers to. We will describe the operation of the parser by first listing the attributes it shares with well-known parsers, and then describing its novel attributes. In addition, we will present some brief annotated traces of the parser's operation.

One facet of the parser analyzes the conceptual dependencies relating the elements of a sentence, much like the parsers written by Schank and his colleagues (Riesbeck, 1975; Schank, 1975), and resembling case-grammatical analyses (Fillmore, 1968). This analysis attempts to determine what action or process is being described, what people or objects are participating in the action, and what each of their roles is. For example, the sentence fragment "This flywheel stores...energy..." would be analyzed as an instance of the conceptual predicate *CONTAIN*, with the flywheel as instrument (the CONTAINER) and the energy as object (the CONTAINEE). READER's stored representations of word-meanings indicate the possible meanings a word can have, the syntactic roles it can play, and the case roles it is associated with. Unlike the Schank family of parsers, READER's conceptual dependency analysis does not necessarily use predicates that are semantic primitives, but instead uses higher-level predicates, like *CONTAIN*, which are easier to relate to the schema, as we will describe later.

READER's analysis of syntax makes use of sequential constraints in English, knowing what classes of words and larger units can follow other units, and permitting recursive embedding of units within units. READER's parser progresses from left to right in a sentence while assigning the words to surface syntactic categories, sometimes anticipating a certain class of syntactic or conceptual unit, such as anticipating a nominal after encountering a "the." The parser's progression from left to right across the words of a sentence somewhat resembles an ATN (augmented transition network) parser (Woods, 1970; 1980).

READER's construction of a referential representation resembles Winograd's (1972) SHRDLU's attention to what was occurring in the toy blocks world. READER keeps track of what objects are being described, what is being said about them, and how they interact with each other. As READER progresses from sentence to sentence in a passage, it elaborates its own representation of the referential world. When a new object is introduced in the passage, a corresponding model referent is introduced into the referential representation. When a previously-mentioned referent is reintroduced, and more is said about it, the model referent is correspondingly
elaborated. More cycles are generally required to establish a new referent and associate some properties with it than to associate additional properties with an already established referent. This is consistent with psychological findings that passages of a given length that introduce many new arguments are read slower (Kintsch, Kozminsky, Streby, McKoon, & Keenan, 1975), and sentences that introduce a new argument are read slower than those that refer to an established referent (Haviland & Clark, 1974).

READER also shares some properties with other parsers. READER is word-oriented like Rieger's parser (Rieger, 1979), so that it tries to analyze each word, rather than necessarily operating on larger units. READER constructs and fills slots with an open-slot request scheme like McCord's parser (1980), and integrates levels of processing, like FRUMP (Schank, Lebowitz, & Birnbaum, 1980). Like HEARSAY (Reddy & Newell, 1974), READER performs concurrent analyses of different levels of the text. Unlike HEARSAY, READER's analyses do not necessarily operate modularly. As soon as an intermediate proposition is constructed, it appears in READER's working memory and becomes equally accessible to all levels of processing, no matter which level produced it.

In contrast to these shared attributes, READER differs from other parsers in ways that reflect the sequential and chronometric characteristics of human readers. One major constraint is the immediacy of processing, requiring that all levels of processing generally be attempted on one word before reading the next word. This contrasts with other more conventional understanding systems that commonly have a multi-pass understanding system. In a multi-pass parser, first one process operates on all the words of a sentence (or clause or paragraph), and then another process or several processes operate on that same group of words. Often, the output of the first process is the input to the second. One example of such a system is Schank's (1975) combination of ELI which converts English sentences into meaning representations, and SAM, which is a script-applying program. This system violates the immediacy assumption because SAM does not operate on any part of a sentence until ELI has completely finished with it. At that point, SAM operates on ELI's output. By contrast, all of READER's component processes attempt to operate on each word as it is encountered in the text. The source of this constraint is the finding that the effects of many levels of processing can be observed on the gaze duration on the very word that enables the various levels (Carpenter & Just, 1977; Just & Carpenter, 1980).

Another distinguishing feature of READER's parser is its non-reliance on any single aspect of the analysis. Most other parsers have a particular focus (e.g., syntax vs. semantics) and tend to rely on that focus for most of its progress. READER has no tension among different foci because it pursues them concurrently, and lets them collaborate on formulating the interpretation. If one level of analysis (e.g., the conceptual dependency) fails to
analyze a given word or structure, READER may still be able to derive an adequate representation based on its knowledge of the syntax and the referent. Collaboration and non-dependence on any one focus is also an accurate characterization of the relation among the grosser levels of analysis, like parsing, lexical access, and schema integration. For example, if READER can determine how a piece of text fits into the schema, then it can in principle fit it in without knowing its syntactic form. This has allowed us to work on versions of READER as models of skimming and speedreading, by having very incomplete sentence representations when substantial portions of the text are skipped, but still being able to extract some schema-level information for easy texts.

**Parsing Noun Phrases.** The interplay of various types of parsing processes can be illustrated with some examples, such as how the syntactic and semantic analyses of a noun phrase contribute to its comprehension. A transition network approach is used to recognize the ordered sequence of words within a noun phrase (Thibadeau, Just & Carpenter, 1980). That is, various productions recognize possible sequences of word types that can constitute a noun phrase, such as a determiner followed by a noun [DET N], or a determiner followed by any number of adjectives followed by a noun [DET ADJ(repeated) N]. The following productions are among those that process the sequential constraints among the words of a noun phrase:

- Prepositions have slots for nominals they modify: If you see a word that is a preposition, assume it has a head noun.
- Determiners have slots for nominals they modify: If you see a word that is a determiner, assume it has a head noun.
- Adjectives have slots for nominals they modify: If you see a word that looks like an adjective, assume it has an associated noun.
- Determiner-generated slots absorb adjectives: If a word expectation from a determiner-generated slot lands on an adjective, move the word expectation forward to the next word position.
- Adjective-generated slots absorb adjectives: (This resembles the immediately preceding production, but is kept distinct because determiner-generated slots absorb certain types of adverbs that adjective-generated slots don’t.)
- A noun seeking-slot gets its noun: When a noun is encountered in the presence of a determiner-generated slot or an adjective-generated slot, construct and activate a proposition that says that the noun is the head of the noun phrase.

Table 6 shows an example of how these productions are activated in processing the phrase “of the oldest mechanical devices.” For example, the preposition “of” is processed by establishing a slot for the expected nomi-
nal. It is initially hypothesized that the nominal will be the next word, but this expectation is disconfirmed with the processing of "the," and the expectation is moved forward to the next word. At the same time, the determiner "the" also generates an expectation of a nominal, and so on.

**TABLE VI**
Description of the parser's processing of the phrase "of the oldest mechanical devices" from the sentence "Flywheels are one of the oldest mechanical devices known to man."

**Cycle 91**
The preposition OF establishes a slot for the nominal that is the object of the preposition. This slot is called a PREPOSITION-MATE4. The resulting representation is: (WORD4 :HAS PREPOSITION-MATE4). The absence of a proposition asserting what PREPOSITION-MATE4 :IS acts as a request to fill the PREPOSITION-MATE4 slot.

**Cycle 92**
The PREPOSITION-MATE4 generates a WORD-CONCEPT-EXPECTATION4 that the next word will be the MATE, resulting in: (PREPOSITION-MATE4 :HAS WORD-CONCEPT-EXPECTATION4) (WORD-CONCEPT-EXPECTATION4 :IS WORD5). The next word, WORD5, will be evaluated in terms of its potential role as all or part of the PREPOSITION-MATE4.

**Cycle 104**
By this time, the word "the" has been recognized. It is not in PREPOSITION-MATE4 relation to "of," but some word which follows "the" might still bear that relation. Thus the expectation of a word to fill the PREPOSITION-MATE4 slot is moved forward to the word that follows "the"—(PREPOSITION-MATE4 :IS WORD6).

**Cycle 105**
The determiner THE generates a syntactic relation, called a DETERMINER-MATE5, that is a slot for the noun head of a noun-phrase.

**Cycle 106**
A WORD-CONCEPT-EXPECTATION5 is established for the DETERMINER-MATE5 that establishes an expectation for the very next word of text.

**Cycle 128-130**
After the word "oldest" is recognized, both the PREPOSITION-MATE4 and the DETERMINER-MATE5 relations are moved rightward to the next word. The adjective OLDEST establishes a slot for an ADJECTIVE-MATE6. A WORD-CONCEPT-EXPECTATION for the next word is established to anticipate the filler of the empty ADJECTIVE-MATE6 slot, similarly to the way it was done for the PREPOSITION-MATE4 and the DETERMINER-MATE5.

**Cycle 160-162**
The adjective MECHANICAL is recognized and treated just like the adjective OLDEST and the expectations for the PREPOSITION-MATE4, DETERMINER-MATE5, ADJECTIVE-MATE6 (generated by OLDEST) and ADJECTIVE-MATE7 (generated by MECHANICAL) are moved rightward to the next word-concept, WORD8.

**Cycle 189**
With the recognition of the noun DEVICES, a number of grammatical relations become known: the word-concept DEVICES satisfies the -MATE relations for the concepts OF, THE, OLDEST, and MECHANICAL. The propositions expressing these relations are constructed, and the expectations that anticipated them, having been satisfied, are now deactivated.
The noun DEVICES acquires a relation to a referent (as yet unknown). The WORD-CONCEPT-EXPECTATION3 that was noted for the OBJECT3 of the complement ONE has been fulfilled because it was tied to finding the PREPOSITION-MATE4 for the word-concept for OF. Since that tail has been found, the OBJECT3 is recognized as the word-concept, DEVICES, which is the MATE.

Cycle 191
Since ONE describes a NUMBER in the construction “one of . . . .”, the proposition is formed that the MODEL-REFERENT3 for ONE (FLYWHEEL1) is an example of the MODEL-REFERENT8 for DEVICES1.

Cycle 192
The conceptualization for the main clause is now transformed into a single predication which can be translated roughly as “flywheels are devices.”

Cycle 193
A proposition is constructed stating that the sentence TOPIC is “flywheels” and, with lower confidence, that it is a topic of the entire paragraph as well. This inference is made when the main conceptualization in the first sentence of a text is fundamentally a predication (‘x is a y’) about some object x, in this case, flywheels. (It is from this information that READER will derive the summary statement “the text was about flywheels.”)

Collaboration. READER has the advantage of interweaving productions concerning order constraints, such as those above, with order-free productions, such as those found in conceptual parsers like Riesbeck’s. The commingling of the different types of productions allows the system to correctly process text that would be difficult for a parser that depended exclusively on syntax or semantics. For example, a purely syntactic parser, ATN or other, might reject a sequence like [DET ADJ DET N] as an instance of a noun phrase. An example of such a sequence that occurs in the “Flywheels” passage is “the greater the mass . . . .” Although this phrase is syntactically complex, READER correctly parses it with the aid of conceptually-based productions.

- A comparative adjective preceded by “the” signals a definite comparison: This production associates the definiteness of the determiner with the comparison itself, rather than with the compared object, with an intermediate level of activation. “The greater . . . .” or “The more . . . .” establish the possibility of a definite comparison.
- Definite comparatives expect a nominal that can vary in magnitude along the dimension of the comparative adjective: This is a conceptual expectation that is not associated with a particular word-slot in the surface structure; in this way it is dissimilar from syntactic parsing productions.
A nominal that can vary in magnitude along the dimension of the comparative adjective fulfills the conceptual expectation: In this case, the mass is taken as the argument of "greater."

A definite comparative denotes magnitude of change in the head, to be related to magnitude of change in another entity: So "the greater the mass..." will denote magnitude of change in mass, to be related to magnitude of change in something else (in this case, the energy that can be stored).

The normal parsing of a noun phrase is considerably altered by these conceptual-level productions. The first production tries to fill the determiner-generated slot with a comparative change denoted by an adjective. The second production generates a conceptual expectation that can countervail the syntactic slot generated by the adjective. The second "the" in the phrase reinforces the conceptual expectation and inhibits the syntactic slot, using another production not shown here.

The productions above illustrate several of the themes of the parsing process. The main one is that various levels of processing can be executed concurrently, with collaboration between the processes in understanding the text. The potential for collaboration exists on every cycle, not just when various processes have completed their work. The intermediate products of parsing, involving all facets of the analysis, are potential loci of collaborative activity.

Contrast with Augmented Transition Networks. READER's parser somewhat resembles an ATN in its use of sequential information to recognize a syntactic unit. However, READER and an ATN differ in how they distribute their processing load across the words of a syntactic unit, such as a noun phrase. An ATN typically buffers each successive word of a noun phrase by pushing it onto a stack until the end of the noun phrase is reached. Then, the stack is unpacked and the representation of the noun phrase is constructed. Any interpretation of the ATN as a model of human processing would postulate more processing during the unpacking than during the encounter with successive words belonging to the same syntactic unit (traversal of nonterminal arcs). If this processing scheme corresponded to human processing, one would expect the gaze duration on the head noun of a noun phrase (or the word following the head) to be longer than otherwise expected. The gaze duration data clearly disconfirm this expectation.

READER's immediacy of processing distinguishes it in at least one way from an ATN. READER buffers nothing unless it is absolutely necessary. This entails making intelligent guesses and working with incomplete information at times, but it also means that processing is distributed more uniformly across the words of a text. This is not to say that READER never
postpones some aspects of processing; there are instances in which aspects of comprehending a given word cannot possibly be performed until a later word is read (e.g. assigning a pronominal reference when the pronoun precedes the referent). But if READER does postpone, it is out of necessity, and not out of a design feature of its architecture.

As noted above, the gaze duration data from human readers show no increase in processing time at the end of a noun phrase. In fact, the data indicate a decrease in the gaze duration on noun-phrase ends, as the number of modifiers of the noun in the phrase increases. This was true for various types of modifiers, such as prepositions, determiners, adjectives, and noun classifiers. The gaze duration on the head noun decreased by about 10 msec with each additional modifier in the phrase.

Although this finding is unexpected in most parsing schemes, READER attributes the facilitation to the processes that search for model referents. When READER encounters a noun phrase, a slot for a model referent is established, with immediate attempts to determine the slot filler’s identity (whether it is a previously established object or a new object) and some of the slot filler’s properties. Each modifier in the noun phrase contributes to this process in one of several ways. For example, in a passage describing red fire ants, on all but the first occurrence of the phrase “red fire ant,” the human gaze duration on “ant” is shorter than would otherwise be expected. READER would also exhibit such savings because “red” and “fire” provide excellent cues to what the model referent of the phrase will be, and READER makes use of such cues as soon as they become available, rather than waiting for the end of the noun phrase. Modifiers can also reduce the uncertainty of subsequent nouns based on semantic entailment. For example, in the phrase “one of the oldest mechanical devices known to man,” mechanical makes devices largely redundant, and known to fairly well determines that the head of the prepositional phrase will be human. Most of the expectations that READER generates are at the level of conceptual objects (e.g., expect an agent) and at the world model level (e.g., expect that this utterance elaborates on the properties of the physical object under discussion). The expectations are generally not expectations of particular words.

Other Parsing Effects. Parsing effects are large under two circumstances. The first is when the sentence structure is awkward, ambiguous or in some other way difficult. One example of this occurs at the beginning of the Flywheel passage in the introductory phrase “Flywheels are one...” The word one receives an extended gaze, probably because both the subject noun, Flywheels and the verb, are, are plural and disagree in number with the predicate one until Flywheels is reinterpreted as referring to a generic concept. An extreme form of such parsing difficulty can be observed with garden path sentences, in which subjects are led to assign one interpretation
that is incompatible with the structure or semantics of later parts of the sentence. There is an extended gaze duration on the first word that reveals the inconsistency (Carpenter & Daneman, 1981). The other kind of situation in which parsing processes have a noticeable effect on gaze durations is one in which the parsing process can foreshorten other comprehension activities, such as the decreased time on a modified head noun described above.

Although we have stressed the immediacy of READER's parsing and other levels of processing, there are clearly some limits imposed by the text on how immediately some things can be interpreted. Both human readers and READER are sometimes unable to even guess at the interpretation of a word or phrase or clause, or its significance to the schema. Human readers spend extra processing time at the ends of sentences, particularly those that contain difficult constructions, novel words, unexpected changes of topic, or contradictory information. They also pause at the ends of sentences in which the information necessary for referential assignment and schema integration are not available until the entire sentence has been read. But they do not pause at the end of a sentence that is a well-written, well-structured part of a relatively easy text. READER also executes wrap-up processes (particularly schema integration) when it reaches the end of a sentence, and completes unfinished processes from any other levels.

While parsing effects are readily found in the gaze duration data and are of considerable theoretical interest, they account for much less of the variance than encoding and lexical access, and the text-level integration processes. This suggests that parsing processes are concurrent with other much longer or much more variable processes, so that parsing effects are not visible under most circumstances. This type of temporal relation between parsing and other processes develops fairly naturally within the CAPS architecture.

READER is not a complete parser of English text, or even of scientific English text. The parser is complete enough to analyze one 140-word passage. Even so, it contains about 130 productions for syntactic and semantic analysis. About 30 of these constitute a core syntactic parser (to analyze surface syntactic relations) that is robust over passages. If the parser can be made complete without qualitative changes in its design, then it is a sufficient system. We currently see no obvious inherent obstacles to completeness.

TEXT INTEGRATION

Any natural-language understanding system, human or machine, must integrate the representations of words, clauses, and sentences to produce a text-level representation of the passage. READER constructs a representation that is adequate to produce a summary, qualitatively and quantitatively similar to human readers' recall. Equally important, READER's processes
exemplify three properties that appear to characterize human integrative processing. First, READER uses its previous knowledge, contained in a schema, to organize the information in the passage as it reads along. Second, like human readers, READER spends different amounts of time on various parts of the passage, depending on that part’s role in the schema. For example, in the scientific passages, human readers and READER spend more time on the names and operating principles of the mechanisms being described, and less time on who made it or physical motions. Finally, READER uses the immediacy principle for text-level integration, as it does for all other levels of processing. As each successive word of a text is processed, READER attempts to integrate the newly read text with the representation of the preceding text, just as human readers do (Carpenter & Daneman, 1981; Dee-Lucas, Just, Carpenter, & Daneman, in press; Just & Carpenter, 1980).

The MECHANISM Schema

The representation of the text-level material makes use of a schema—a frame and slot structure that helps to organize the main information from a scientific exposition. The schema slots are instantiated with the information from the passage during comprehension, and can be used in a later recall or question-answering task. To read the Flywheel passage, READER uses a schema, called the MECHANISM schema, that specifies the kinds of information one might expect to read about man-made devices and biological mechanisms, as they are used by human or animal agents. This schema is sufficient to comfortably handle not only the Flywheel passage, but most of the 14 other test passages as well. The schema consists of slots that identify the role of various kinds of information and these can be filled by one or more propositions. The slots for the MECHANISM schema are given in Table 7, with some examples of slot fillers. The notation for the slot fillers has been abbreviated or altogether eliminated here for expository convenience.

The slots are ordered a priori with a default ordering that reflects normative relative importance. The slots marked as most important are NAME, GOALS, and PRINCIPLES, while MADE-BY and USED-BY are marked as less important. “Importance” is not an emergent property of the text (as topicality is), but instead reflects relevance of various parts of a text to a reader’s goals. The relative importance of various text categories can change with task goals, and the relative time spent reading them also changes (Carpenter & Just, 1981). In READER, the task can cause a reordering of the slots. For example, if READER were given the goal of determining the users of various mechanisms, it would attach the highest importance to the USED-BY slot. The importance ordering affects how carefully the slot filler is read, as described below.
TABLE VII
READER's Schema for a MECHANISM Passage

1. NAMES contains the words that have been established to refer to this schema, such as "flywheels" and "device."

2. PRINCIPLES indicates the operating principles that relate the mechanism's physical properties and actions to its goals. Examples of these are:

   CAUSE*1 with antecedent (a comparative increase in speed of spin) will yield a consequent (a comparative increase in stored energy)
   CAUSE*2 with antecedent (a comparative increase in mass) will yield a consequent (a comparative increase in stored energy)

3. GOALS specifies the end states that the mechanisms are used to achieve. The goal of a flywheel is energy storage, represented as:

   (FLYWHEEL1 :GOALS1 *CONTAIN*2)
   (*CONTAIN*2 :SUBJECT FLYWHEEL1)
   (*CONTAIN*2 :OBJECT ENERGY1).

4. PHYSICAL-PROPERTIES specifies the enduring physical properties of the mechanism, such as color, shape, or size. An example is:

   (FLYWHEEL3 :PHYSICAL-PROPERTIES2 *MADE-OF*2)
   (*MADE-OF*2 :SUBJECT FLYWHEEL3)
   (*MADE-OF*2 :OBJECT FIBERGLAS1)
   (*MADE-OF*2 :OBJECT RUBBER1)

5. PHYSICAL-MOVEMENTS gives the physical actions that are typically associated with the mechanism. For example, flywheels may spin, wings may "beat", a hammer may be moved by hand.

6. MADE-BY identifies who brought this mechanism into existence, for example (FLYWHEEL1 :MADE-BY MAN1). There may also be an embedded proposition specifying HOW or WHEN the mechanism was made.

7. USED-BY tells who or what uses the mechanism. For example, this slot in the flywheel passage is (FLYWHEEL1 :USED-BY MAN1), indicating that the mechanism is used by people.

8. EXEMPLARS contains another MECHANISM schema embedded within it. There are four flywheels referred to in the passage, a generic one and three exemplars: the car-engine flywheel, the small-space one, and the superflywheel.

The Process of Slot-Filling. A central component of the integration process is relating the clause and sentence-level representations to the text-level organization provided by the schema. There are several ways that this is accomplished in READER. The most important way that READER relates the sentence-level representations to the appropriate schema slots is on the basis of its knowledge of the meanings of the sentence-level predicates. The sentence-level representations are composed of predicates like *CONTAIN*, *CAUSE*, *CONVERT*, *PURPOSE*, and *HAVE-AS-PART*, while the schema-level slots are NAMES, MADE-BY, USED-BY, PHYSICAL-PROPERTIES, PHYSICAL-MOVEMENTS, GOALS, PRINCIPLES, and EXEMPLARS. As part of its lexicon, READER knows enough
about the properties of the predicates to make some reasonable matches to the schema-level slots. For example, READER knows that *CONTAIN* and *HAVE-AS-PART* specify physical properties. Consequently, information about the component parts of a flywheel represented by *CONTAIN* or *HAVE-AS-PART* would likely be inserted into the PHYSICAL-PROPERTIES slot.

In the example above, the processing is bottom up, from sentence-level predicate to schema slot. In other cases, the processing is more symmetrically collaborative between the levels. For example, in the sentence "The greater the mass of a flywheel, the faster it spins, the more energy can be stored in it," the relation between mass and storage capacity must be inferred. Since mass is a physical property, spin a behavior, and energy accumulation a positive outcome associated with purpose (inferred from the second sentence), a causal link is inferred relating physical properties and actions to the GOALS of the flywheel.

Sometimes, the text explicitly states how a given piece of information is related to the schema. Phrases like "X is called," when predicated of an object presumed to be the topic of the passage, explicitly cue the NAMES slot. READER uses various heuristics to identify the topic of a passage, such as hypothesizing that the surface subject of the first or second sentence of the passage is also the topic of the passage.

Finally, the schema slots can be filled with default values for a given type of mechanism. For example, any mechanism that is a "device" is automatically known to be (MADE-BY MANI). Any mechanism that is part of a living thing is (MADE-BY NATURE1) and (USED-BY LIVINGTHING1).

Thus, there are a number of means that READER uses to determine how a particular piece of information in that passage fits into the schema. Like other knowledge structures, READER's inferences at this level have varying degrees of activation, reflecting the strength of the evidence for a particular idea. In addition, various processes collaborate, so that there is often more than one factor arguing in favor of a given schema slot. Some types of information may not fit into any schema slot, and in this case the information will not be recalled, although it may possibly remain in working memory for a while.

The Time Course of Text Integration

READER uses the immediacy principle for text-level integration, as it does for the other levels of processing. As each successive word of a text is encountered, it is evaluated as a potential basis for action by all the productions, including those that attempt to fit the new information into schema slots. The schema-level integration actions are actually evoked as soon as enough of the sentence has been processed to indicate how it fits into the
schema. The point in the sentence at which the first schema-level production will be evoked depends on the sentence wording and structure. Sometimes, an entire clause or sentence must be read in order to determine the relation of it and its constituents to the higher-level structure. At other times, a particular word located at an arbitrary point in a sentence can trigger the text-integration process. This is consistent with the results of several eye-fixation studies that examined the point in a sentence at which integrative processing is done. The two main processing loci are at the word that provides sufficient information to logically permit the integration and at the last word in the sentence (Carpenter & Just, 1977; Dee-Lucas et al., 1981; Just & Carpenter, 1978). Thus, schema-level integration is done by READER as soon as possible, but this sometimes turns out to be the end of the sentence.

The time course of text integration is especially important to READER because it adjusts its reading rate to the importance of the information it encounters, just as human readers do. READER goes slower if it is processing information that it believes matches an important slot, and it continues at the slower reading rate for the rest of the sentence or until it determines that a different slot-filler has been encountered. Currently, READER slows down for the NAME, PRINCIPLES, and GOALS slots, since we have assumed that the default goals in reading the short scientific passages are to determine what the mechanism is called (NAME), how it works (PRINCIPLES), and what it accomplishes (GOALS).

Slowing down the reading rate improves READER's comprehension and recall of the more important parts of the passage. READER slows down by globally decreasing the activation weights. When this occurs, comprehension changes qualitatively as well as quantitatively, because the decrement in activation weights changes the relative influence of two types of collaborative processes. The two types are first, the main interpretive processes, and second, the processes that check for consistency among various parts of the interpretation. These consistency-checking processes include checking for subject-verb agreement or for unique fillers for single-argument predicates. The consistency-checking processes start later than the interpretive processes because they are enabled by the presence of the interpretations. Given this asynchrony in onset, the consistency-checking processes have only a small chance of intercepting an erroneous interpretation. This disadvantage is somewhat offset by the fact that the consistency-checking processes have higher activation weights, and so have some potential to intercept incorrect interpretations. Interception occurs when the consistency check raises the activation of a previously unfavored interpretation to threshold level before any other candidates reach threshold. Consistency-checking processes become more influential at slower reading rates (i.e., when RE-WEIGHT uniformly decreases all the activation weights). This is because the late-starting consistency-checking processes have a better opportunity to
intercept inconsistencies before they cause conflict or before an incorrect interpretation is accepted. Thus, slower reading is more likely to initially produce a correct interpretation.

Another reason that slower reading leads READER to better comprehension has to do with the method of resolving above-threshold conflicts. If two or more conflicting interpretations rise above threshold, then their activation levels are all greatly decremented, resulting in an absence of knowledge about that aspect of the text. If no process is actively seeking that knowledge, then READER continues reading, having failed to understand the point that produced conflicting interpretations. (If the absent knowledge, say a noun, is actively being sought by some process, such as a noun phrase analysis that has strongly activated a slot for the head noun, then the relevant part of the text must be reread more slowly). Since the schema slot-fillers that are considered important are read slowly, they are more likely to be comprehended without conflict, and hence not dismissed. The important parts also are more likely to be interpreted correctly in the first place, for the reasons outlined in the paragraph above. So READER's slowing down on important parts of the text not only emulates the human slowdown, but also contributes to the same superior comprehension and recall of the important parts that human readers exhibit.

An Example. We will illustrate the integration processing by considering how the second sentence in the Flywheel passage is integrated with the first. The first sentence is:

*Flywheels are one of the oldest mechanical devices known to man.*

Both the form and content of the sentence suggest that the passage topic is "flywheels," since this is the topic of the opening sentence and it is also identified as a mechanical device. READER uses these cues to hypothesize that "flywheels" is indeed the passage topic, and READER starts to evoke relevant schema knowledge. Similarly, human readers pause on the first mention of the words that designate the passage topic. READER understands the reference to the "oldest mechanical device known to man" as the beginning of an exposition on "flywheels." Since the focal aspect is a mechanism, the MECHANISM schema is fully evoked when FLYWHEEL is understood to be a DEVICE, namely after the word "devices." After having read the first sentence, READER has instantiated a prototype of flywheels and predicated of this prototype that it is a device, that it is mechanical, that it is old in man's knowing, and that it is made by and used by man.

The second sentence is:

*Every internal-combustion engine contains a small flywheel that converts the jerky motion of the pistons into the smooth flow of energy that powers the drive shaft.*
Table 8 describes some of the integration processes initiated by the first clause, cycle by cycle. Some of the major integrative processes shown in the table are how model referents are constructed for ENGINE and FLYWHEEL, how the predicates "contains" and "converts" are related to high-level predicates, *CONTAIN* and *CONVERT*, and how these, in turn, are related to schema-slots. An example of these processes, and some of the issues they address, is how READER determines the model referent of "a small flywheel" in cycle 444. Substantives such as "flywheels" and "flywheel" require associated model-referents, and if they don't exist, new ones are created. The referent of "flywheel" in the second sentence is not the prototypical flywheel mentioned in the first sentence. Instead, the second sentence refers to a particular example of a prototype flywheel, namely the kind found in internal-combustion engines. READER constructs a proposition that the model-referent of "a small flywheel" is a new, previously unmentioned referent, and calls it FLYWHEEL2. It makes this inference because of the indefinite article that modifies "flywheel," and because internal combustion engines are modern, while the prototype flywheel is ancient.

A second example of the integrative processes are those that relate specific conceptual dependency representations to the schema slots. Two examples occur in Table 8. In one, which begins in cycle 375, the word-concept CONTAIN is recognized as the conceptual predicate *CONTAIN*, and at this time CAR-ENGINE1 (the model referent of internal-combustion engine) can be assigned to the agent slot for *CONTAIN*. During a later cycle, after "a small flywheel" is processed, it is assigned the role of OBJECT of *CONTAIN*. With this information, it can be inferred that the *CONTAIN* predicate fills a slot for a property of the flywheel—in this case its location. Then the entire proposition that a CAR-ENGINE *CONTAINS* a FLYWHEEL is entered in the PHYSICAL-PROPERTIES slot of the schema.

Recall and Forgetting. If human readers are asked to recall a passage immediately after having read it, their recall protocols are incomplete and sometimes inaccurate, but they do reflect the schema structure (Just & Carpenter, 1980). The human recall protocol below illustrates how the schema is reflected:

The topic of this paragraph was flywheels and the basic function of a flywheel is to store energy, they come in various types, such as, well they can come in various types of materials, you can have rubber flywheels or plastic ones, etc. and they're also, well your engine in a car has a flywheel because it takes the jerky motion of a piston and converts into smooth energy if you will. Flywheels are one of the oldest form of energy producers known to man and they basically function by the wind turning them and that's about it.
TABLE VIII
A description of some of READER's text integration on the sentence
"Every internal-combustion engine contains a small flywheel that converts..."

The MODEL-REFERENT3 of ENGINE is interpreted as CAR-ENGINE1 (not some unspecified engine), because ENGINE activates CAR-ENGINE in the context of "internal-combustion." The proposition that the model referent is a car engine initially has a low activation level, but the activation is later increased by the mention of "pistons" and "drive shaft."

The verb CONTAINS activates a conceptual relation called *CONTAIN*1, which has a CONTAINER1 argument, instantiated here as CAR-ENGINE1. The instantiation is made because the SUBJECT4 of CONTAINS is ENGINE and the MODEL-REFERENT3 of ENGINE is CAR-ENGINE1.

The phrase "a small flywheel" activates a new model-referent, FLYWHEEL2, which is an *EXAMPLE-OF*1 the prototype, whose referent is FLYWHEEL1. The inference is made that the model referent is new because the article is indefinite and because internal-combustion engines are modern, while the prototype flywheel is ancient.

The CONTAINEE1 of *CONTAIN*1 is FLYWHEEL2 because the verb "contains" has FLYWHEEL as its OBJECT4. The model-referent for FLYWHEEL2 is paired with a MECHANISM sub-schema, namely, that it has a goal, principles, physical properties, and so on.

FLYWHEEL2 acquires the PHYSICAL-PROPERTY that it is the CONTAINEE1 of CAR-ENGINE1. Since it is an exemplar of a prototypical flywheel, it inherits the properties of the prototype by duplicating them in its sub-schema.

The word "that" refers to the model-referent FLYWHEEL2. A cue to this reference is the proximity of the pronoun and the previous mention of the model-referent in the surface structure.

The verb CONVERTS activates a conceptualization called *CONVERT*1, which has a CONVERTER1 argument, instantiated here as FLYWHEEL2. The instantiation is made because "that" (which refers to FLYWHEEL2) is the SUBJECT9 of the verb CONVERT.

It is inferred with low activation that *CONVERT*1 is a PRINCIPLE for FLYWHEEL2. At this point, only the agent of the conversion, the flywheel, is known. Since the mechanism under discussion is FLYWHEEL2 and it is agent of *CONVERT*1, it is inferred with low activation that the transformation will be relevant to how FLYWHEEL2 works.
READER’s recall protocol is produced by outputting the contents of the various schema slots, in order of the slots’ importance: the NAME, the flywheels’ main GOALS, and some PRINCIPLES of operation. Similarly, the human recall protocols include responses like “this passage was about flywheels...,” “flywheels store energy,” or “if a flywheel spins too fast it will break up.” Physical properties are not recalled in isolation, but in the context of their relation to a PRINCIPLE, if at all. There are no recalls like “Flywheels spin...but that’s all I can remember.” Rather we find recalls like “The faster a flywheel can spin the better it is.” Examples, physical properties, and motions that are not intrinsic to principles are generally of lesser importance. We can tell how “important” a particular part of the text is to the human readers by how slowly that part is read and how likely it is to appear in the recall protocols. The information that fits in the more important schema slots is read more slowly and is recalled better (Just & Carpenter, 1980).

Neither the human nor READER’s recall is perfect. There are three main reasons for READER’s imperfect recall. First, facts that READER comprehended but were not integrated into the schema are not recalled because the retrieval plan does not access them. Thus READER knows more than it recalls, and this additional knowledge could be accessed if READER were asked specific questions or asked to perform a recognition test. A second reason for the imperfect recall is imperfect comprehension. If an erroneous interpretation is accepted and integrated with the schema, then there will be an error of commission in the recall. Moreover, if there was an above-threshold conflict between competing interpretations that was resolved by dismissing all the interpretations, and there was no rereading to fill in the gap, there will be an error of omission in the recall due to imperfect comprehension. This is especially likely at higher reading speeds. Finally, there may be imperfect recall because of some forgetting at the time of comprehension. For example, READER numbers and remembers the word tokens only until it finishes processing the current sentence, (cf. Jarvella, 1971). Thus there is some forgetting of verbatim information in working memory during comprehension that might affect the final representation of the passage and hence the recall.

**DISCUSSION**

Since the gaze duration characteristics of human readers provide the driving force behind READER’s design, it is appropriate to assess the model partially in terms of its fidelity to the human gaze duration. Both quantitative and qualitative comparisons are involved, and both made with the aid of a multiple linear regression analysis whose dependent variable is the mean gaze duration on each word.
One analysis compared the number of CAPS processing cycles on a word to the observed gaze duration. A regression analysis on only the 140 words of the *Flywheels* passage, in which the independent variable is the number of CAPS processing cycles, and the dependent variable is human gaze duration, accounts for 67% of the variance. Much of this fit is attributable to the word encoding and lexical access processes. In the absence of alternative models of this fine grain, it is difficult to evaluate the significance of the 67%, but it does provide a benchmark that tells us not so much how far we have come, but how far we have yet to go.

There are a number of ways that READER's fit to the data might be improved. First, a number of READER's free parameters can be tuned to bring the number of cycles and the gaze duration into closer correspondence. These parameters are the initial activation levels of propositions, the activation weights of productions, and various threshold levels. To the extent that these parameters might correspond to human characteristics that could be independently estimated, such a tuning exercise might be fruitful. Second, the fit might be improved by giving READER the ability to encode a short familiar word to the right of the current word without fixating it, thereby approximating the frequent human non-fixation of short function words. But to make the fit a great deal better, we believe it will be necessary to know more about individual readers' prior knowledge of the content area and their reading style, and then to fit a model to their individual data. The READER cycles currently account for 22-46% of the variance in the gaze durations of the individual human subjects.

Another way of using READER to account for the human gaze duration is to develop a multiple linear regression model whose independent variables can be associated with some of READER's processes. Some of these independent variables have long been known to affect human processing time. For example, it is well known that people take longer to recognize less familiar words, although the exact function relating word frequency and reading time was previously unknown. Other variables came to light when READER or another understanding system strongly suggested particular loci of processing ease or difficulty, such as the effect of the length of a noun phrase on the time spent on a head noun, or the extra time spent on the first occurrence of the word that designates the passage topic.

There are 11 independent variables in this regression analysis, as shown in Table 9. The table is divided into three parts, corresponding to the three main types of processing, but the assignment of some of the independent variables to a particular processing stage is somewhat arbitrary. For example, the wrap-up processes on the last word of a sentence are assigned to text-integration processes, but could just as well have been assigned to semantic and syntactic analysis, since both levels of processing consume extra cycles on the last word of a sentence. The analysis reported in Table 9 is
based on the mean gaze durations on the 1,936 words in all 15 passages, and averaged over 14 readers. The $R^2$ value is .79 and the intercept is 2.8 msec.

### TABLE IX
Application of the Regression Model to the Gaze Duration on Each Word of the Scientific Texts

<table>
<thead>
<tr>
<th>Processing Stage</th>
<th>Factor</th>
<th>Regression Weight (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>no. of letters</td>
<td>32</td>
</tr>
<tr>
<td>Encoding and</td>
<td>log frequency</td>
<td>31</td>
</tr>
<tr>
<td>Lexical Access</td>
<td>beginning of line</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>novel word</td>
<td>686</td>
</tr>
<tr>
<td></td>
<td>digits</td>
<td>21</td>
</tr>
<tr>
<td>Semantic and</td>
<td>skipped word</td>
<td>-17</td>
</tr>
<tr>
<td>Syntactic Analysis</td>
<td>head noun modification</td>
<td>-10</td>
</tr>
<tr>
<td>Text</td>
<td>last word in sentence</td>
<td>41</td>
</tr>
<tr>
<td>Integration</td>
<td>last word in paragraph</td>
<td>155</td>
</tr>
<tr>
<td></td>
<td>first mention of topic</td>
<td>183</td>
</tr>
<tr>
<td></td>
<td>first content word in paragraph</td>
<td>66</td>
</tr>
</tbody>
</table>

Note: All regression weights are reliably different from zero, $p < .01$.

Many of the results found in the human performance data above can be interpreted in terms of READER's processes.

1. The gaze duration increases by 32 msec for each additional letter in a word—a result that READER matches quantitatively. There is one additional CAPS cycle for each additional letter in a word.
2. The gaze duration increases logarithmically as the word's normative frequency decreases, by 31 msec for each log unit (base 10) decrease. READER matches this result quantitatively. The number of CAPS cycles on a word increases logarithmically as its normative frequency decreases.
3. There is an extra 16-msec gaze duration on the first word in each line of print. The explanation probably lies in corrective eye movements that compensate for the inaccuracy of the large saccades from the end of one line of print to the beginning of the next, and not in any comprehension process. READER does not provide an account of this result.
4. The gaze duration on novel words, ones the human readers are unlikely to have encountered before, extends for an extra 686 msec. READER matches this behavior qualitatively, taking extra cycles to determine the interpretation of a novel word.
5. The gaze duration is somewhat longer on digits than on words. READER does not account for this result, but it seems to be inter-
pretatable in terms of phonemic recoding of digits, which may take extra time.

6. The gaze duration is 17 msec less on those words that READER would have skipped because they were highly predictable function words.

7. The gaze duration on the head noun of a noun phrase decreases by 10 msec with each additional modifier of the noun. READER matches this behavior qualitatively, executing fewer cycles on the head noun as the number of modifiers increases, for the reasons described previously.

8. There is a pause of 41 msec on the last word of a sentence—a behavior READER matches qualitatively. READER's additional cycles are spent on schema-level computations that had to be postponed until the sentence end, as well as on some wrap-up processes at the syntactic and conceptual level.

9. There is a pause on the last word of a paragraph and READER also uses many cycles on the last word of the paragraph, primarily for schema integration. Since READER processes only one paragraph, it is difficult to quantitatively assess this aspect of READER, as it is for the two remaining results.

10. There is a pause on the first mention of the topic in the human gaze duration data. The first mention of the topic evokes schema-level processes in READER, such as retrieving some relevant content knowledge and fitting new information into the NAMES slot of the schema.

11. There is a pause on the first content word of the passage. In the "Flywheels" passage, the first mention of the topic and the first content word are coincident, so it is difficult to separate these two effects in READER, although they are separable in the data of the human readers over the 15 passages.

This regression model accounts for 79% of the variance, compared to 72% for the model reported by Just & Carpenter (1980). Of the 7% improvement, most (5%) is due to the measurement of word length in letters, rather than in syllables. The other 2% gain is due to a better characterization of the independent variables, particularly those pertaining to semantic and syntactic analysis. But the main appeal of the current regression model is that the independent variables are grounded in processing characteristics existing in an explicit process model, READER.

The interpretation of the regression results must take into account that READER posits concurrent execution of many processes. READER performs some processes in sequence (e.g., word encoding followed by lexical access) and in those cases the interpretation of the regression results is
straightforward. The regression weight indicates the amount of extra processing time per stimulus unit, such as the time to encode another letter. For concurrent processes, a reliable regression weight means that when that process occurs, the gaze duration is longer (or shorter), but the magnitude of the regression weight does not directly correspond to the duration of the process. For example, the regression analysis shows that the first mention of a topic is gazed at for an extra 183 msec, but we cannot conclude that the extra processes associated with such words take 183 msec to be executed. If they are concurrent, they could take much more time to be executed, but only the last 183 msec would extend beyond the execution of other processes.

Future Directions. In the new implementation (Franzlisp on a VAX), we are currently developing READER in two directions. The first is to make READER a more flexible processor, allowing it to skim and speedread in a way that is similar to human speedreaders we have recently studied. READER might be able to speedread by changing its goals, by lowering some of its thresholds (i.e. being content with less complete comprehension of a given piece of text), by relying less on syntactic analysis, by sampling the text the way human readers do, and by relying more on the schema in a familiar content domain to provide default values for some slots. READER's comprehension and recall will of course diminish, as does the performance of human readers when they are speedreading.

A second direction of development is to account for individual differences in reading ability among people, by varying some of READER's properties that would make it a better or a poorer comprehender. Motivated by recent research on individual differences in human readers (Daneman & Carpenter, 1980), we have been varying the size of the surface structure unit that can be encoded at one time, as well as manipulating the parameters associated with the processes assumed to account for individual differences (such as encoding and lexical access). Finally, readers probably differ a great deal in their knowledge structures and reading strategies. READER provides us with a model that can be correspondingly manipulated to examine if the resulting performance parallels differences among human readers.

A number of questions arise if one considers how READER might be expanded into a more general language comprehension system, although the issues appear as soluble within the CAPS and READER framework as within any other. Many of these issues concern large knowledge bases. For instance, READER knows only one schema and reads only one passage, so the problem of deciding which schema to evoke and the possibility of using more than one schema at a time do not arise. Another way that READER was sheltered is that it needed only a small knowledge base to understand one passage and many of READER's schema-level productions are fairly
content-specific. A much more general understanding system would require a great deal more knowledge, yet the larger data base would have to be used as effectively as the smaller one. So a more general model would have to be able to reason and make inferences from a larger data base than READER does. But none of these factors should limit READER in principle.

What READER has in its favor is plausibility as a psychological model. The immediate, collaborative nature of the processes mimic important features of human reading. In addition, CAPS and READER provide the potential to model the time course of processing. READER already shows some resemblances to human chronometric data and the potential exists to further modify READER to increasingly resemble this aspect of human processing. More generally, modeling the time course of comprehension appears to be a useful way to limit the range of possible mechanisms (cf. Kieras, 1981; Miller & Kintsch, 1980). Studying the temporal course of various processes is one way of learning how the processes are organized and coordinated. Finally, the scope of the model, from perceptual to schema-level processes is sufficiently broad to accommodate many other temporal properties of human reading, both those that are known and those that are yet to be learned.

REFERENCES


