CHAPTER FOUR

WEAVER++ and other computational models of lemma retrieval and word-form encoding

Ardi Roelofs
Max Planck Institute for Psycholinguistics, Nijmegen,
The Netherlands

INTRODUCTION

A basic skill of speakers is the access of words in memory. In producing utterances, speakers call on many facets of their stored knowledge about words, including their meaning, syntactic properties, morphological composition, and sound structure. Lexical access is the process by which this information about words is retrieved from memory in order to map a lexical concept onto an articulatory program. The access consists of two major steps: lemma retrieval and word-form encoding (cf., Levelt, 1989; but see Caramazza, 1997). In lemma retrieval, a lexical concept is used to retrieve the lemma of a corresponding word from memory. A lemma, as the term is used in this chapter, is a memory representation of the syntactic properties of a word, crucial for its use in sentences (Roelofs, 1992a, 1993). For example, the lemma of the Dutch word sigaar (cigar) says that it is a noun and that its grammatical gender is non-neuter. In word-form encoding, the lemma is used to retrieve the morpho-phonological properties of the word from memory in order to construct an articulatory program. For example, for sigaar the morpheme <sigaar> and the segments /s/, /i/, /ɣ/, /a/, and /r/ are retrieved and a phonetic plan for [si.'ɣar] is generated.

In this chapter, I provide a detailed comparison of the two most widely discussed computational models of lemma retrieval and word-form encoding: the classical spreading-activation model of Dell and colleagues (Dell, 1986, 1988, 1990; Dell & O’Seaghdha, 1991, 1992; Dell, Schwartz, Martin, Saffran, & Gagnon, 1997; Peterson, Dell, & O’Seaghdha, 1989) and the WEAVER++ model
(Levlt, Roelofs, & Meyer, 1999a,b; Roelofs, 1992a, 1993, 1994, 1996a,b,c, 1997a,b,c, 1998, 1999; Roelofs & Meyer, 1998; Roelofs, Meyer, & Levlt, 1996, 1998), which has been developed within Levlt’s (1989, 1992) general theoretical framework for speech production. The Dell model and WEAVER++ each have been designed to account for a wide variety of findings (see Levlt et al., 1999a, for a recent review of WEAVER++ applications). Unfortunately, the trend in the word production literature has been away from trying to simultaneously account for a lot of data in favour of a focus on one or two findings (e.g., Caramazza, 1997; Cutting & Ferreira, 1999; Harley, 1993; Schade & Berg, 1992; Starreveld & La Heij, 1996). Occasionally, I make reference to these more limited models, some of which have not been computationally implemented (e.g., Caramazza, 1997).

Dell’s model has been designed to account for speech errors, which constitute the traditional database for production research and modelling (e.g., Harley, 1993; Schade & Berg, 1992). In recent years, however, researchers have started to use chronometrical techniques and have collected production latency data. The WEAVER++ model recognises the key insights from speech errors, but has specifically been designed to provide a unifying account of the increasing body of chronometrical data. I start by describing Levlt’s (1989, 1992) general theoretical framework for lexical access. Next, I review the Dell model and its application to speech errors. Finally, I explain WEAVER++. I show that this latter model accounts for production latency data and that it is compatible with findings on speech errors.

Role of lexical access in speech production

Figure 4.1 illustrates the three major types of processes that underlie speaking: conceptualisation, formulation, and articulation. Conceptualisation processes generate so-called messages, that is, conceptual structures to be verbally expressed. In WEAVER++, messages are specified in terms of lexical concepts. Messages may, for example, be derived from external scenes via object perception. Formulation processes take the message, access appropriate words (i.e., nouns, verbs, adjectives, etc.) for the lexical concepts, and build a syntactic structure and a morpho-phonological structure. The result is a phonetic plan (articulatory program) for the utterance. In WEAVER++, the phonetic plan makes explicit motor programs for the syllables in the utterance. Finally, articulation processes execute the articulatory program, resulting in overt speech.

As indicated, lexical access consists of lemma retrieval and word-form encoding, which are stages of access that are part of the formulation stages of syntactic and morpho-phonological encoding, respectively. In lemma retrieval, a lexical concept is used to retrieve a lemma from memory, which is a representation of the syntactic properties of a word, crucial for its use in sentences. Lemma retrieval makes these properties available for syntactic encoding processes (e.g.,
Kempen & Hoenkamp, 1987). Furthermore, lemmas contain slots for the specification of abstract morpho-syntactic parameters such as mood (e.g., indicative), tense (e.g., past, present), number (i.e., singular, plural), and person (i.e., first, second, third). In word-form encoding, a lemma and its parameter values are used to recover the appropriate morpho-phonological properties from memory in order to construct a phonetic plan. Lexical access may be initiated by a message, but also by a perceived spoken or written word. WEAVER++ assumes that a perceived word activates lemmas and word forms in parallel (see Figure 4.1).

**Stages of lexical access**

Figure 4.2 illustrates the further division of stages of access in WEAVER++. Assume a Dutch speaker sees a cigar and wants to name it. Lexical access consists of mapping the lexical concept CIGAR(X) onto the articulatory program for *sigaar*. First, the lemma retriever takes CIGAR(X) and delivers the lemma.
Figure 4.2. Stages of lexical access in WEAVER++.

of \textit{sigaar}. That is, it makes available the syntactic class “noun”, grammatical gender “non-neuter”, and so forth. To derive the appropriate word form, singular [si.’\text{\textipa{yar}}] instead of plural [si.’\text{\textipa{yar}}\text{\textipa{r}}], the word’s number has to be specified. Therefore, the lemma retriever has to inspect the message for number information and to set the lemma’s number parameter. The lemma plus number diacritic are then input to word-form encoding. The articulatory program is derived in three major steps: morphological encoding, phonological encoding, and phonetic encoding (cf., Dell, 1986; Leve\text{"u}lt, 1989). The morphological encoder takes the lemma of \textit{sigaar} plus the parameter value “singular” and produces the stem morpheme <\textit{sigaar}>. The phonological encoder takes this morpheme, spells out its segments, syllabifies
them, and assigns a stress pattern. Thereby it produces a so-called phonological word representation. This representation describes the singular form of *siqaar* as a phonological word (σ) consisting of two feet (Σ), one metrically strong (s) and the other weak (w). The first syllable (σ) has /s/ as onset and /i/ as nucleus. The second, stressed syllable has /y/ as onset, /a/ as nucleus, and /r/ as coda. Finally, the phonetic encoder takes this phonological word representation, accesses a syllabary of learned motor programs for syllables (Levelt & Wheeldon, 1994), and delivers the corresponding articulatory program for [siˈyar].

Although this functional architecture has been developed on the basis of behavioural evidence (e.g., Levelt, 1989; Levelt et al., 1999a; Roelofs, 1992a, 1997c), it is receiving increasing support from studies such as MEG (magnetoencephalography: Levelt, Praamstra, Meyer, Helenius, & Salmelin, 1998), LRP (lateralised readiness potentials: Van Turennout, Hagoort, & Brown, 1997, 1998), PET (positron emission tomography), and fMRI (functional magnetic resonance imaging). Recently, Indefrey and Levelt (2000) performed a meta-analysis of 58 brain imaging studies in the literature, which anatomically located this functional architecture in the brain. As can be expected from the classical neuropsychology literature (e.g., Wernicke, 1874) and most later studies, the system is basically located in the left hemisphere. Visual and conceptual processing appears to involve the occipital, ventro-temporal, and anterior frontal regions of the brain; the middle part of the left middle temporal gyrus seems to be involved with lemma retrieval (the activity in these areas occurs within the first 275 ms after a to-be-named object is presented). Next, activation spreads to Wernicke’s area, where the phonological code of the word appears to be retrieved; activation is then transmitted to Broca’s area and the left mid superior temporal lobe for postlexical phonological processing such as syllabification (taking some 125 ms). Finally, phonetic encoding takes places (for the next 200 ms), with a contribution of the supplementary motor area (SMA) and the cerebellum, while the sensory-motor areas control articulation.

**Classical support for the stages**

The classical behavioural support for a distinction between lemma retrieval and word-form encoding comes from speech errors. A lemma level of encoding explains the different distribution of word and segment exchanges. Word exchanges such as the exchange of *roof* and *list* in “we completely forgot to add the *list* to the *roof*” (from Garrett, 1980) typically concern elements from different phrases and of the same syntactic category, here noun. By contrast, segment exchanges such as “she is a real *rack* pat” for “pack rat” (from Garrett, 1988) typically concern elements from the same phrase and do not respect lexical category. This finding is readily explained by assuming an exchange of lemmas during syntactic encoding and an exchange of segments during word-form encoding.
Speech errors also provide support for a morphological level of form encoding that is distinct from a lemma level with abstract morpho-syntactic parameters. Some morphemic errors appear to concern the lemma level, whereas others involve the form level (e.g., Dell, 1986; Garrett, 1975, 1980, 1988). For example, in “how many pies does it take to make an apple?” (from Garrett, 1988), the interacting stems belong to the same syntactic category (i.e., noun) and come from distinct phrases. Note that the plurality of apple is stranded, that is, it is realised on pie. Thus, the number parameter is set after the exchange. The distributional properties of these morpheme exchanges are similar to those of whole-word exchanges. This suggests that these morpheme errors and whole-word errors occur at the same level of processing, namely when lemmas in a developing syntactic structure trade places. Similarly, errors such as “I’d hear one if I knew it” for “I’d know one if I heard it” (from Garrett, 1980) suggest that syntactically specified lexical representations (lemmas) may trade places independently of their concrete morpho-phonological specifications. By contrast, the exchanging morphemes in an error such as “slieely thinned” (from Stemberger, 1985) belong to different syntactic categories (adjective and verb) and come from the same phrase, which is also characteristic of segment exchanges. This suggests that this second type of morpheme error and segment errors occur at the same level of processing, namely the level at which morphemes and segments are retrieved and the morpho-phonological form of the utterance is constructed. The errors occur when morphemes in a developing morpho-phonological structure trade places.


The model of Dell (1986, 1988) assumes stages of access similar to those of WEAVER++ described earlier, but there are also some important differences. For example, the Dell model does not make the claim that the output of phonological encoding is a phonological word representation and it does not assume a syllabary.

Basic tenets

Dell (1986, 1988; Dell & O’Searghda, 1991, 1992) assumes that the mental lexicon is a network that is accessed by spreading activation. Figure 4.3 illustrates the lexical entry of sigaar. The nodes in the network are associated to each other by equally weighed bidirectional connections. The network contains nodes for conceptual features (e.g., MADE-OF-TOBACCO(X), IS-FOR-SMOKING(X), etc.), lemmas (e.g., sigaar), morphemes (e.g., <sigaar>), syllables (e.g., si and yar), segments (marked for syllable position, e.g., /onset s/, /nucleus i/, /onset y/, /nucleus a/, and /coda r/), and phonological features (e.g., voiced). Phonological feature nodes are left out of the figure for reasons of simplicity. Furthermore,
there are wordshape header nodes and segment category nodes, specifying the abstract CV structure of a word (Dell, 1988).

Lexical access starts by supplying a jolt of activation to the set of conceptual features making up the intended thought. Activation then spreads through the network following a linear activation function with a decay factor. Lemma retrieval, morphological encoding, and phonological encoding are accomplished by successively selecting the most highly activated lemma node, morpheme nodes, and segment nodes at certain moments in time. The time interval between the successive selections has a constant duration whose value depends on the speech rate. Selected nodes are serially ordered by inserting them into slots of independently created frames (i.e., syntactic, morphological, and syllable frames). For example, the /onset s/ node is inserted into the onset slot of the first syllable frame of *sigaar*. To prevent that, for example, the /onset y/ of *sigaar* or the onset of another morpheme or word is inserted, lemmas, morphemes, and syllables making up an utterance are serially encoded. The serial encoding of the syllables of a polysyllabic word such as *sigaar* is achieved by invoking a procedure that temporarily increases the spreading rate $r$ between the morpheme node $<sigaar>$ and the syllable node for the first target syllable (i.e., si) and decreasing it for the other one (i.e., yar), and vice versa for the encoding of the second syllable.
Although the model includes a level of phonological feature nodes, it does not specify how phonetic encoding is accomplished.

Accounting for speech errors

The classical spreading-activation model has been designed to account for facts about speech errors: the kind of errors that occur and the constraints on their form and occurrence. Errors occur when, due to noise in the system, another node than the target is the most highly activated node and gets erroneously selected. The model accounts for a large variety of findings about speech errors. I mention a few major accomplishments (following Dell, 1986).

First, the model accounts for error biases such as the so-called phonological facilitation of semantic substitution errors (Dell & Reich, 1981). When cat is intended, the substitution rat for cat is more likely than dog for cat (if error opportunities are taken into account). Semantic substitution errors are taken to be failures in lemma node selection. The word rat shares segments with the target cat. So, the lemma node of rat receives feedback from these shared segments (i.e., /nucleus æ/, /coda t/), whereas the lemma node of dog does not. Consequently, the lemma node of rat will have a higher level of activation than the lemma node of dog, and rat is more likely to be involved in a lemma selection error resulting in a word substitution. Errors such as substituting rat for cat are called “mixed” errors because the erroneous words share both semantic and phonological properties with the target.

Second, the model accounts for lexical bias, that is, for the fact that speech errors tend to yield real words rather than nonwords (again, if error opportunities are taken into account). For example, in producing cat, the error /h/ for /k/ is more likely than /j/ for /k/, because hat is a word but jat is not. Again, this bias is due to feedback, now from shared segment nodes to morpheme nodes (e.g., from /nucleus æ/ and /coda t/ to <cat> and <hat>) and from these morpheme nodes to other segment nodes (i.e., from <cat> to /onset k/ and from <hat> to /onset h/). This will not occur for nonwords, because there are no morpheme nodes for such items in the network (i.e., there is no node <jat> to activate /onset j/).

Third, the model accounts for the relative frequencies of segmental substitution errors. Anticipations are more likely than perseverations, and exchanges have the lowest probability (Nootenboom, 1969). For example, the anticipation error sed sock for red sock is more likely than the perseveration red rock, which is in its turn more likely than the exchange sed rock. The anticipation bias is a built-in feature of the model: Upcoming words also receive a jolt of activation (less than the target). In the model, exchanges occur the least because they are double errors involving both an anticipation and a perseveration. It should be mentioned, however, that some researchers have argued that it is not clear that anticipations are more common than exchanges, since interrupted errors (by far
the most numerous form) could have been either. Shattuck-Hufnagel (1987) argued that exchanges are not simply a concatenation of an anticipatory and a perseveratory substitution, but occur under different conditions.

Finally, the model accounts for effects of speech rate on error probabilities. Errors are more likely when one speaks faster. It takes time for activation to spread through the network. At high speech rates, activation levels of nodes at the time of selection are not so high yet, so the system is more vulnerable to selection errors due to small random fluctuations of activation levels. Furthermore, if speech rate decreases, the likelihood of exchange errors decreases too (Dell, 1986). For an exchange to occur, first an anticipation error has to be made, /onset s/ for /onset r/ (yielding sed instead of red), and next the /onset r/ still has to be available for selection, /onset r/ for /onset s/ (yielding sock instead of rock). This is more likely at high speech rates, because the time interval between the successive selections is then small so that the activation of /onset r/ has not been decayed yet.

In summary, the model of Dell does a good job in accounting for many facts about speech errors, such as the statistical overrepresentation of mixed errors, lexical bias, the (presumed) relative frequency of segment anticipations, perseverations, and exchanges, and effects of speech rate on these errors. A number of problems with the model, however, have motivated the development of WEAVER++.

**PROBLEMS WITH THE CLASSICAL SPREADING-ACTIVATION MODEL**

**Convergence**

In the classical spreading-activation model, lemmas are retrieved on the basis of sets of conceptual features (cf., Bierwisch & Schreuder, 1992). This confronts the model with a number of computational issues concerning convergence that have not been appropriately dealt with yet within the classical framework (see Levelt, 1989, and Roelofs, 1992a,b, 1993, 1996a, and especially 1997a, for extensive discussion). The convergence problem consists of a number of sub-problems: (1) How to correctly dissect a thought into lexical concepts during message encoding, (2) how to avoid retrieving hyponyms or hyperonyms along with or instead of the intended words, and (3) how to correctly retrieve a single word instead of several words for a synonymous phrase.

Conceptually driven word retrieval does not appear to proceed by trial-and-error. Word meanings typically evade definition (e.g., Fodor, Garrett, Walker, & Parkes, 1980), but speakers seem to know exactly what conceptual information to prepare to access words efficiently. The dissection problem (Levelt, 1992) concerns the issue of how the message encoder knows which sets of conceptual feature nodes correspond to lexical concepts. For example, the thought MALE PARENT (i.e., FATHER) constitutes a lexical concept but the thought YOUNG
PARENT does not. But only lexical concepts can be verbalised by a single word. Furthermore, in the classical model, the lemma of sigaar is retrieved by activating features such as MADE-OF-TOBACCO(\textit{X}), IS-FOR-SMOKING(\textit{X}), and so forth. In the traditional view, the set of features of a more specific word such as Havanna (\textit{Havana cigar}) contains those of its hyperonyms sigaar (cigar) as a proper subset (i.e., Havanna has all the features of sigaar plus some extra ones such as ORIGINALLY-FROM-CUBA(\textit{X})). The classical model fails to explain how the system knows which subset of features of a word corresponds to the meaning of its hyperonyms. That is, which features should be given a jolt of activation in retrieving a word. The model has to be changed to solve this problem, perhaps by embedding the conceptual feature nodes in a constraint satisfaction network (cf., the model of Harley, 1993). Lexical concepts may then correspond to stable states of the network.

The words \textit{parent} and \textit{father} may refer to the same person. The hyponym problem is the question of how to avoid retrieving the hyponym \textit{father} along with or instead of the hyperonym \textit{parent}. The node of the conceptual primitive PARENT(\textit{X,Y}) is linked to the lemma nodes of \textit{parent} and \textit{father}, so both lemmas will attain the same level of activation. This problem may perhaps be solved by giving up the notion of a general spreading rate \textit{r} and tuning the weights on the links (e.g., by a learning process). The feature PARENT(\textit{X,Y}) should be strongly connected to \textit{parent} but weakly to \textit{father}.

The word \textit{father} and the phrase “male parent” may refer to the same person. The synonymy problem concerns the issue of how to avoid retrieving \textit{father} along with or instead of \textit{male} and \textit{parent} for the phrase “male parent”, or vice versa. Tuning weights is insufficient to solve this problem, because the same primitives are involved in producing the single word and the phrase. Perhaps this problem may be solved by sequentially activating the conceptual primitives in producing a phrase. However, there is empirical evidence that suggests that lemmas making up a phrase are planned in parallel (Meyer, 1996).

In summary, the conceptually decomposed retrieval of lemmas in the classical model confronts it with a number of convergence problems concerning the dissection of messages, hyponymy, and word-to-phrase synonymy. Perhaps, each of these problems may be solved within the model in one way or another, but the solutions are ad hoc. What is lacking is a principled solution to the class of convergence problems as a whole.

**Binding and latencies**

The binding problem is the issue of how to correctly retrieve the lemmas, morphemes, and segments of a word in the context of the activation of the lemmas, morphemes, and segments of another word. That is, how to keep the planning of a word insulated from interfering cross-talk, for example, from the concurrent
planning of other words in connected speech or from seeing words or hearing words spoken by an interlocutor (see Roelofs, 1997c, for an extensive discussion). In the classical spreading-activation model, binding is achieved by imposing severe temporal constraints on the planning process. Suppose that a Dutch speaker wants to produce the utterance “haar eens wat sigaren” (“just get me some cigars”). In order to produce this utterance, the speech segments of the morphemes <wat>, <sigaar>, and <en>, among others, have to be retrieved from memory. In retrieving the segments and assigning them to the correct production slots, the speech production system has to keep track of what goes with what. That is, it has to know that the /onset s/ is retrieved for <sigaar> and the /onset w/ for <wat>. Otherwise, the /onset w/ and /onset s/ may trade places and “gat wigaren” might be produced instead of “wat sigaren”. To avoid this indexing problem and the resulting segmental errors, the classical model assumes that only one morpheme is spelled out at a time. That is, during the encoding of <wat>, only /onset w/, /nucleus a/, and /coda t/ are available, so the /onset s/ of <sigaar> will not become selected. This may be called “binding by timing”. Binding by timing is at the heart of the classical account of speech errors. Errors occur during the rare event that, due to noise in the system, a segment of another word is the highest activated one and gets erroneously selected. For example, an error occurs when during the encoding of <wat>, the /onset s/ of <sigaar> has a higher level of activation than the /onset w/, and the /onset s/ gets selected as the onset of <wat>.

However, Meyer and Schriefers (1991) have shown that speakers do not make large numbers of errors when several words are available simultaneously under experimental conditions. For example, if Dutch speakers have to name a pictured object (e.g., a cigar) and they hear a word sharing some of its final segments (e.g., /yi.tar/, guitar), this perceived word helps (i.e., speeds up) naming the object instead of causing trouble in selecting the correct segments. Simulations by Peterson et al. (1989) have shown, however, that the classical model predicts massive amounts of errors. In a priming situation with high-frequency targets and distractors, the probability of selection of a critical target segment (at a time step determined by the lemma’s activation level) was $p = .45$. However, in picture–word interference experiments the error rate is about 5% rather than the 50% predicted by the classical model. Similarly, producing the hyperonym of the name of a picture (e.g., saying furniture to a table) is speeded up by a semantically related distractor word (e.g., chair). And the distractor chair inhibits the production of table, without yielding massive amounts of errors (e.g., Glaser, 1992; Glaser & Düngelhoff, 1984). In the classical model selection takes place on an ordinal basis as the most highly activated node is selected. So, priming may affect levels of activation, but it will not affect latencies. When it comes time to select nodes, the target node is either the most highly activated node and gets selected (priming will not affect this) or it is not the
most highly activated node (due to priming of another node) and an error occurs. So, the low error rate and the latency effects are not to be expected.

In summary, the Dell model attributes speech errors to sporadic interference from the concurrent planning of other words or the hearing of words spoken by interlocutors. However, when multiple words are activated under experimental conditions, planning times are affected but almost no errors are made. Yet, the Dell model predicts massive amounts of errors but no latency effect, exactly contrary to the empirical findings. This suggests that some other mechanism than timing solves the binding problem.

**Syllabification and phonetic encoding**

The task for a binding mechanism is often even more complex than keeping the segments of different words apart, because binding may be context dependent. That is, sometimes the binding of segments to slots has to ignore morpheme and word boundaries in that a segment of one morpheme or word has to be bound to another morpheme or word. This occurs in the production of polymorphic words and connected speech. By rigidly storing words as sequences of syllable nodes and storing each consonant as an onset or coda, models such as that of Dell (1988) and Schade and Berg (1992) have a difficult time dealing with the context-dependence of syllable membership (see Roelofs, 1997b, for an extensive discussion).

Consider the production of the plural form *sigaren* of the word *sigaar*. The plural *sigaren* is created by adding `<en>` ([ə]) to the stem `<sigaar>`. The resulting form is syllabified as `(si)o(ɣa)o(ɹə)o`. Thus, juxtaposing `-en` changes the syllabification of `/r/`. This segment occupies a coda position in *sigaar*, syllabified as `(si)o(ɣar)o`, but an onset position in *sigaren*. Or consider the production of connected speech involving clitics. Clitics are forms of function words such as pronouns, determiners, particles, auxiliary verbs, prepositions, and conjunctions that are phonologically dependent on a host (e.g., Booij, 1995; Levelt, 1989). For example, the reduced form '`s` [əs] of the Dutch adverb *eens* (just) cannot stand alone. In producing “probeer die sigaar '`s`” (“just try that cigar”), '`s` is adjoined to *sigaar*. This yields the new phonological word *sigaar '`s`, which is syllabified as `(si)o(ɣɑ)o(ɹə)s). In the classical model, segments are marked for syllable position, so we have `/coda r/` in the network for *sigaar*. The selection rule in this model prohibits selecting this coda node for the onset slot of a syllable frame, which would be required for the production of *sigaren* and *sigaar '`s`.

Word-form encoding does not end with syllabification. The generated phonological representation has to be mapped onto a context-dependent phonetic representation that can guide articulation. But the classical model does not have a phonetic level of encoding and therefore does not account well, for example, for assimilation and allophonic variation of speech segments. Although the model
has a level of phonological feature nodes connected to segment nodes, it does not say much of anything about phonetic encoding.

In summary, syllabification across morpheme and word boundaries poses difficulty to the classical model. This is because the model stores each word as a sequence of syllable nodes and each consonant as an onset or coda. Furthermore, the classical model lacks a level of phonetic encoding. In the remainder of this chapter, I first explain the WEAVER++ model and show how it solves the problems and how it accounts for production latencies. Next, I review several empirical tests of the model concerning production latencies. Finally, I show that WEAVER++ is compatible with speech error data. The WEAVER++ model has limitations (see Levelt et al., 1999a,b, and commentaries), but I believe it is a step towards a psychologically and computationally correct model of lexical access in spoken word production.

THE WEAVER++ MODEL

Over the past several years, a computational model has been developed on the basis of insights obtained with a chronometrical approach rather than the traditional approach of speech error analysis (e.g., Levelt, 1989, 1992; Levelt et al., 1991; Levelt & Wheeldon, 1994; Meyer, 1990, 1991; Meyer & Schriefers, 1991; Roelofs, 1992a, 1993, 1994, 1996a,b,c, 1997a,b,c, 1998, 1999; Roelofs & Meyer, 1998; Roelofs et al., 1996, 1998). The word-form encoding part of this model is called WEAVER (an acronym standing for Word-form Encoding by Activation and VERification) and the full model including lemma retrieval is called WEAVER++. As the classical spreading-activation model, WEAVER++ assumes that the mental lexicon is a network of nodes and links that is accessed by spreading of activation. In order to deal with the issues discussed earlier, a few new assumptions have been made.

The WEAVER++ model handles the dissection problem, the word-to-phrase synonymy problem, and the other convergence problems by assuming that each lexical concept is represented in the network by an independent node (cf., Collins & Loftus, 1975). For example, the network contains the nodes CIGAR(X) and TOBACCO(X) connected by a link labelled IS-MADE-OF. The node CIGAR(X) is connected to the lemma node for sigaar. Lemmas are accessed from the lexical concept nodes (cf., Fodor et al., 1980) rather than directly from sets of conceptual feature nodes. Since lexical concepts may differ between languages, this view entails that message encoding includes some form of "thinking for speaking". That is, speakers engage in language-specific thinking while they are speaking. I refer to Levinson (1997), Roelofs (1992b, 1997a), and Slobin (1996) for an extensive discussion of this view and its empirical justification.

The WEAVER++ model handles the binding problem, first, by assuming that the relationships between nodes are explicitly coded in the network by labels on
the links (e.g., the link between <sigaar> and /s/ says that /s/ is the first segment of <sigaar>), and second, by assuming that the links are verified by the access algorithm. This allows the algorithm to keep track of what goes with what, without the need for special temporal restrictions and without the need to change spreading rates.

The model handles the problem of syllabification across morpheme and word boundaries by assuming that syllable positions are not stored with words in memory as in the classical model (e.g., there are no /onset r/ and /coda r/ nodes), but that syllable positions are assigned on-line by a syllabification process (Levelt, 1992). The assignment of segments to syllable positions takes neighbouring morphemes and words into account. Syllable positions of segments are computed for phonological words rather than for lexical ones.

The model handles the problem of phonetic encoding by assuming that speakers have a syllabary, a store of motor programs for syllables, which is accessed on the basis of the phonological syllables constructed as part of phonological word representations (Levelt & Wheeldon, 1994). The programs make explicit the gestural scores for the articulatory movements (cf., Browman & Goldstein, 1986). The syllabary does not have to contain many programs. Statistical analyses have revealed that 500 different syllables already cover about 85% of the Dutch syllable tokens (cf., Levelt et al., 1999a). Since not all instances of a given syllable are articulated in exactly the same way, some further articulatory programming may be needed once the programs are retrieved from the syllabary (see Levelt, 1989), but this programming is limited.

The WEAYER++ model integrates a spreading-activation based network with a parallel object-oriented production-rule system (Levelt et al., 1999a; Roelofs, 1992a, 1994, 1996c, 1997c). The type of system is a mix of traditional AI, connectionism, and traditional cognitive modeling (cf., Anderson, 1983). Words are not planned by a central agent that overlooks the whole process but by teams of production rules (condition-action pairs) that work in parallel on small parts of the word (like several spiders making a single web). The production rules are stored with the nodes and have a limited overview only. Activation of nodes in the network triggers production rules that choose lemmas and incrementally build phonetic plans by selecting and connecting nodes. Upon activation of a node, a production rule verifies the link between the node and the selected nodes one level up in the network. Syntactic production rules select the lemma node linked to the target lexical concept node. Morphological production rules select the morpheme nodes that appropriately encode a selected lemma node and its tense, agreement, and mood parameters. Phonological production rules select the segments linked to the morpheme nodes and prosodyfy the segments in order to construct phonological word representations. And finally, phonetic production rules select the syllable program nodes that appropriately encode the constructed phonological syllables, and access the corresponding syllable programs in the syllabary.
Network structure

Figure 4.4 illustrates the structure of entries in WEAVER++'s lexical network, in particular, the memory representation of sigaar plus part of the representation of tabak (tobacco). A lexical network with nodes and labelled links is connected to a syllabary with learned motor programs for syllables. The lexical network consists of three major strata: a conceptual stratum, a syntactic stratum, and a word-form stratum. The conceptual stratum contains concept nodes and labelled conceptual links. Each lexical concept in the language, for example CIGAR(X), is represented by an independent node. The links specify conceptual relationships,
for example, between a concept and its properties such as TOBACCO(X). The syntactic stratum contains lemma nodes (sigaar), syntactic property nodes and labelled links (gender: non-neuter; lexical category: noun), and slots for the specification of parameters (number: {singular, plural}). The word-form stratum contains metrical structure, morpheme, segment, and syllable program nodes and links. Morpheme nodes are connected to a lemma and its diacritics. The links between morphemes and segments specify the serial position of the segments. The links between segments and syllable program nodes specify possible—as opposed to actual—syllabifications. The word-form stratum is connected to a syllabary, storing ready-made motor programs for syllables. I now explain the lexical access algorithm. First, I review WEAVER++'s assumptions about lemma retrieval and some applications of the model, and next the assumptions about word-form encoding and some applications.

Lemma retrieval

Basic tenets

A basic theoretical claim implemented in WEAVER++ is that lemmas are retrieved in a conceptually non-decomposed way, that is, for example, the noun sigaar is retrieved on the basis of the chunk CIGAR(X) instead of features such as MADE-OF-TOBACCO(X), IS-FOR-SMOKING(X), and so forth. Each lexical concept in the language is represented by an independent node in the network. Retrieval starts by enhancing the level of activation of the node of the target lexical concept. Activation then spreads through the network, each node sending a proportion of its activation to connected nodes. The most highly activated lemma node is selected. For example, in verbalising the thought CIGAR, the activation level of the lexical concept node CIGAR(X) is enhanced. Activation spreads through the network, whereby the lemma nodes sigaar and tabak, among others, will be activated. The sigaar node will become the most highly activated node, because it receives a full proportion of the activation of CIGAR(X), whereas tabak and other lemma nodes receive a proportion of a proportion of the activation of CIGAR(X). Upon verification of the link between the lemma node of sigaar and CIGAR(X), this lemma node will be selected. Roelofs (1997a) gives a mathematical convergence proof for this retrieval algorithm.

The equations that formalise WEAVER++ are given in Roelofs (1992a, 1993, 1994, 1996c, 1997c). There are simple equations for the spreading of activation and the instantaneous selection probability of lemma and syllable program nodes, that is, the hazard rate of the lemma retrieval and word-form encoding process. The selection probability of a lemma node (syllable program node) equals the ratio of its activation and that of all the other lemma nodes (syllable program nodes). Given the selection ratio, the mathematical expectation of the retrieval (and encoding) time can be computed.
Some applications

SOA curves of semantic effects. The retrieval algorithm explains, among other phenomena, the classical SOA curves of the semantic effects of picture and word distractors in picture naming, picture categorising, and word categorising. The basic experimental situation is as follows. Participants have to name pictured objects while trying to ignore written distractor words superimposed on the pictures. For example, they have to say chair to a pictured chair and ignore the word bed (semantically related) or the word fish (semantically unrelated). There is also a so-called stimulus onset asynchrony (SOA) manipulation. The written distractors are presented 400, 300, 200, 100 ms before (called negative SOAs), simultaneously with, or 100, 200, 300, 400 ms after picture onset (called positive SOAs). The classical finding is a semantic inhibition effect in a small SOA window ranging from −100 to +100 ms.

Panel A of Figure 4.5 gives the SOA curves. I have plotted the semantic effect (i.e., the difference between the naming latencies with related and unrelated distractors) against SOA. Thus, a positive difference indicates a semantic inhibition effect. The real data are from Glaser and Dänigl (1984). Semantic inhibition is obtained at SOA −100, 0, and 100 ms. The predictions by WEAVER++ are also indicated. As can be seen (and has been shown by a stringent statistical measure of fit), the model fits the data.

How does WEAVER++ explain these findings? I illustrate the explanation using the miniature network given in Figure 4.6. The figure illustrates the conceptual stratum and the syntactic stratum of two semantic fields: furniture and animals. Thus, there are lexical concepts nodes (e.g., BED(X)) and lemma nodes (e.g., bed). Pictures have direct access to the conceptual stratum and words have direct access to the syntactic stratum (see Figures 4.1 and 4.4). Assume chair is the target. Distractors are names of other pictures in the experiment. In the case of a pictured chair and distractor bed, activation from the picture and the distractor word will converge on the lemma of the distractor bed, due to the connections at the conceptual stratum. In the case of the unrelated distractor fish there will be no such convergence. Although the distractor bed will also activate the target chair, the pictured chair will prime the distractor lemma bed more than bed will prime the target lemma chair due to network distances: three links versus four links (pictured chair → CHAIR(X) → BED(X) → bed versus word bed → bed → BED(X) → CHAIR(X) → chair). Therefore, it will take longer before the activation of chair exceeds that of bed than that of fish. As a consequence, bed will be a stronger competitor than fish, which results in the semantic inhibition effect.

In WEAVER++, competition in lemma selection is “dynamic” rather than “hard wired”. That is, in lemma retrieval a shortlist of target lemmas can be defined depending on the task that is set for the retrieval system (e.g., in a categorisation task the response set consists of hyperonyms such as furniture,
animal, etc.). Competition is restricted to these shortlisted lemmas. The assumption is that speakers set up the shortlist before an experiment when they receive a booklet with the pictures and names to be used. Thus, in the model, only potential target responses will compete for selection. In case of picture or word categorisation, furniture and animal are the targets and will compete, but chair, bed, fish, and dog will not. The distractor bed superimposed on a pictured chair will activate the target furniture via the conceptual network, but bed will not be a competitor for furniture because bed is not a permitted response in the experiment. By contrast, fish on a pictured chair will activate animal, which is a
competitor of the target furniture. Thus, semantic facilitation is predicted and this is exactly what is empirically obtained.

Panel B of Figure 4.5 gives the results for picture categorising, for example, when participants have to say furniture to the pictured chair and ignore the distractor word. Again, the semantic effect is plotted against SOA. A negative difference indicates a semantic facilitation effect. The real data are from Glaser and D"ungelhoff (1984). WEAVER++ fits the data. The same prediction holds for the same reasons for word categorising, for example, when participants have to say furniture to the word chair and ignore the distractor picture. Panel C of Figure 4.5 gives the results for word categorising. Again, WEAVER++ fits the data.

The prediction of semantic facilitation also holds for picture naming with hyperonym, co-hyponym, and hyponym distractors that are not part of the response set. For example, in naming a pictured chair (the only pictured piece of furniture in the experiment), the distractor words furniture, bed, or throne are superimposed. Semantic facilitation is indeed what is obtained (Roelofs, 1992a). Panel D of Figure 4.5 plots the semantic facilitation against SOA. The semantic effect was the same for hyperonym (furniture), co-hyponym (bed), and hyponym (throne) distractors, so the curves represent means across these types of word.

The WEAVER++ model is not restricted to the retrieval of noun lemmas. Thus, the same effects should be obtained in naming actions using verbs, for example, when participants have to say drink to a drinking person and ignore eat or laugh (names of other actions in the experiment). Indeed, again semantic inhibition is obtained, as shown in Panel E of Figure 4.5 (Roelofs, 1993). Facilitation is also predicted for hyponym distractors, for example, when participants have to say drink to a drinking person and ignore booze or whimper (not permitted responses in the experiment). Again semantic facilitation is obtained, as shown in Panel F of Figure 4.5 (Roelofs, 1993).
The finding of semantic facilitation from hyponyms excludes one type of solution to the hyperonym problem in lemma retrieval. Bierwisch and Schreuder (1992) have proposed a model in which the convergence problem is solved by inhibitory links between hyponyms and hyperonyms (i.e., words inhibit all their hyperonyms). For example, in producing chair, activating the lemma of chair leads automatically to inhibition of the lemma of its hyperonym furniture. However, the existence of such inhibitory links predicts semantic inhibition from hyponym distractors (e.g., distractor throne should inhibit target chair), but facilitation is what has been empirically obtained. As we saw, distractor throne facilitates the production of chair (see Figure 4.5 panels D and F). Also, the finding of semantic facilitation in picture and word categorising refutes such an inhibitory link between words and their hyperonyms (e.g., between chair and furniture). In general, the semantic facilitation effects pose difficulty to models in which lemma selection is achieved by hardwired lateral inhibition among lemmas, as in the models of Cutting and Ferreira (1999), Schade and Berg (1992), and Stemberger (1985). Semantic facilitation is difficult to explain if all activated lemmas always compete, that is, if furniture and chair always compete due to their inhibitory links. Instead, the facilitation suggests that in lemma retrieval, a shortlist of target lemmas can be defined depending on the task set for the retrieval system (e.g., in a categorisation task consisting of hyperonyms such as furniture, animal, etc.), with competition restricted to these shortlisted lemmas.

The semantic effect is obtained for nouns, verbs, and adjectives (e.g., colour, which is related to the classical Stroop effect) in producing words (Glaser, 1992; Roelofs, 1992a, 1993), but also for lexical access in producing phrases and sentences, as has been shown by Meyer (1996) and Schriefers (1993). Furthermore, Schriefers obtained interference effects due to the selection of grammatical gender (a particular type of lemma information) in producing phrases.

Computing agreement in producing phrases. Schriefers (1993) asked Dutch participants to describe coloured objects using phrases. For example, they had to say “de groene tafel” (“the green table”) or “groene tafel” (“green table”). In Dutch, the grammatical gender of the noun (table) determines which definite article should be chosen (de for non-neuter and het for neuter) or the inflection on the adjective (groene versus groen). On the pictured objects, written distractor words were superimposed that were either gender congruent or incongruent with the target. For example, the distractor muis (mouse) takes the same gender as the target tafel, namely non-neuter, whereas distractor hemd (shirt) takes neuter gender.

Schriefers obtained a gender congruency effect, as predicted by WEAVERR++. Smaller production latencies were obtained when the distractor noun had the same gender as the target noun compared to a distractor with a different gender (see also Van Berkum, 1997). According to WEAVERR++, this gender congruency effect should only be obtained when agreement has to be computed, that is,
when the gender node has to be selected in order to choose the appropriate definite article or the gender marking on the adjective. It should not be obtained when participants have to produce bare nouns, that is, in "pure" object naming. WEAVER++ makes a distinction between activation of the lexical network and the actual selection of nodes. Lemma nodes point to grammatical gender nodes, but there are no backward pointers (see Figure 4.4). Thus, boosting the level of activation of the gender node by a gender-congruent distractor will not affect the level of activation of the target lemma node and therefore will not influence the selection of the lemma node. Consequently, priming a gender node will only affect lexical access when the gender node itself has to be selected, for example, when the gender node is needed for computing agreement. Thus, the gender congruency effect should only be obtained in producing gender-marked utterances, not in producing bare nouns. This corresponds exactly to what is empirically observed (Jescheniak, 1994; La Heij, Mak, Sander, & Willeboordse, 1998).

Interaction between semantic and orthographic factors. Starreveld and La Heij (1995, 1996) observed that the semantic inhibition effect in picture naming is reduced when there is an orthographic relationship between the target and distractor word. Damian and Martin (1999) observed the same for spoken distractors. For example, in naming a pictured cat, the semantic inhibition was less for distractor word calf compared to cap than for distractor word horse compared to house. According to Starreveld and La Heij, the interaction suggests that lemmas do not exist or that there is feedback from the output morphophonological level to the lemma level (as in the models of Dell, 1986, 1988, and Harley, 1993), contrary to the claim of WEAVER++ that the output morphophonological network contains forward links only. However, as argued by Roelofs et al. (1996), in deriving their predictions Starreveld and La Heij did not take into account that printed words may activate their lemma nodes and output morpho-phonological nodes in parallel (see Figure 4.1). In the lexical network of WEAVER++, the orthographic input stratum is connected to both the lemma level and the output morpho-phonological level (see the links labelled INLEX in Figure 4.4). Thus, printed words may affect lemma retrieval directly, and there is no need for backward links in the output morpho-phonological network. Computer simulations (reported in Roelofs et al., 1996) showed that WEAVER++ predicts that in naming a pictured cat, the semantic inhibition will be less for distractor calf compared to cap than for distractor horse compared to house, as empirically observed. Thus, WEAVER++ captures the interaction.

Summary and discussion

I have tried to show that WEAVER++ does a good job in accounting for the time course of semantic inhibition and facilitation effects of distractors in picture naming, picture categorising, and word categorising. Furthermore, it accounts
for the gender congruency effects and the interaction between semantic and form factors in picture naming. Other models in the literature have been less successful. For example, Starreveld and La Heij (1996) developed a new computational model without lemmas that was specifically designed to account for the interaction between semantic and form effects of distractors in picture naming. As its stands, their model can handle picture naming, but there is no provision in the model to deal with picture categorising and word categorising. And for picture naming, it incorrectly predicts semantic inhibition only, but, as we saw, semantic facilitation has also been obtained (Roelofs, 1992a, 1993). Also, there is no specification of syntax in the model of Starreveld and La Heij, and therefore the gender congruency effects fall outside the model. Moreover, by assuming a direct mapping of concepts onto word forms, the model fails to account for the speech error evidence for a lemma level of representation, namely the two types of morphemic errors discussed earlier. And by the direct mapping, the model predicts phonological activation of semantic relatives of a target (i.e., in planning the word cat, the form of dog should also become active), contrary to the empirical findings (Levett et al., 1991; Peterson & Savoy, 1998). The empirical data suggest that during the planning of the word cat, the lemma of dog also becomes active but not its form. This suggests that lemma retrieval and word-form are not only distinct but also discrete processes (i.e., only the form of a selected lemma becomes encoded), which is an assumption that has been implemented in WEAVER++ (see Levett et al., 1999a,b, for an extensive discussion).

Word-form encoding

Basic tenets

A basic theoretical claim implemented in WEAVER++ concerning word-form encoding is that lemmas are mapped onto learned syllable-based articulatory programs by serially grouping the segments of morphemes into phonological syllables. These phonological syllables are then used to address the programs in a phonetic syllabary.

Figure 4.4 illustrates the form representation of sigaar in WEAVER++. The non-metrical part of the form network consists of three layers of nodes: morpheme nodes, segment nodes, and syllable program nodes. Morpheme nodes stand for roots and affixes. Morpheme nodes are connected to the lemma and its parameters. The stem <sigaar> is connected to the lemma of sigaar and “singular”. A morpheme node points to the segments that make up its underlying form, and, for some words, to its metrical structure. For storing metrical structures, a principle of economy applies. WEAVER++ assumes that the main accent of Dutch words is on the first syllable containing a full vowel (which holds for more than 90% of the word tokens), unless the lexical form representation indicates otherwise (Levett et al., 1999a). Thus, for polysyllabic words that do not have main stress on the first stressable syllable, the metrical structure is stored as part of the
lexical entry, but for monosyllabic words and for all other polysyllabic words, it is not. For example, the metrical structure for *sigaar* [si.'yar] is stored, but for *tafel* ['ta.fæl] it is not. Stored metrical structures describe abstract groupings of syllables (o) into feet (Σ) and feet into phonological words (ω). Importantly, it is not specified which segments make up the syllables nor is the CV pattern specified. The links between morpheme and segment nodes indicate the serial position of the segments within the morpheme. Possible syllable positions (onset, nucleus, coda) of the segments are specified by the links between segment nodes and syllable program nodes. For example, the network specifies that /r/ is the coda of [yar] and the onset of [ra].

Encoding starts when a morpheme node receives activation from a selected lemma. Activation then spreads through the network in a forward fashion. The form encoders follow simple selection criteria. Attached to the nodes in the network, there are production rules that verify the links between the nodes and target nodes one level up. Production rules are triggered when the activation levels of their nodes exceed threshold. The production rules may be triggered and may fire in parallel.

The morphological encoder selects the morpheme nodes that are linked to a selected lemma and its parameters. Thus, *<sigaar>* is selected for *sigaar* and “singular”. The phonological encoder selects the segments and, if available, the metrical structures that are linked to the selected morpheme nodes. Next, the segments are input to a prosodification process that associates the segments to the syllable nodes within the metrical structure (for “exception” words) or constructs metrical structures based on segmental information. Thus, when stored, metrical structures are retrieved and woven into the speech plan, otherwise they are constructed on the spot. Like weaving a fabric, the process has a certain direction. The prosodification proceeds from the segment whose link is labelled first to the one labelled second, and so forth. In the prosodification, syllable positions (onset, nucleus, coda) are assigned to the segments following the syllabification rules of the language. Essentially, each vowel and diphtong is assigned to a different syllable node and consonants are treated as onsets unless phonotactically illegal onset clusters arise. In the encoding of *<sigaar>* , the /s/ is made syllable onset and the /i/ nucleus of the first syllable, and the /y/ onset, the /a/ nucleus, and the /r/ coda of the second syllable. The prosodification process provides for cross-morpheme and cross-word syllabification. In planning polymorphemic words or connected speech, the structures of adjacent morphemes or words may be combined (provided that certain phrase-structural conditions are satisfied, see Levelt, 1989, for discussion). This leads to new phonological words. For example, WEAVER++ may syllabify *<en>* with *<sigaar>* for the plural *sigaren* or it may prosodify the stem *<sigaar>* and the reduced form of *<eens>* together for the cliticisation *sigaar* ’s. Then, following the maximal onset principle in syllabification (e.g., Goldsmith, 1990), /r/ will be made onset of the third syllable instead of coda of the second syllable, yielding (si)o(ya)o(ra)o,
and \((si)_{o}(ya)_{o}(ras)_{o}\). In this way, WEAVER++ achieves syllabification across morpheme and word boundaries.

The phonetic encoder selects the syllable program nodes whose labelled links to the segments correspond with the syllable positions assigned to the segments. For example, \([yar]\) is selected for the second phonological syllable of \(sigaar\), because the link between \([yar]\) and /\(y/\) is labelled onset, between \([yar]\) and /\(a/\) nucleus, and between \([yar]\) and /\(r/\) coda. Similarly, the phonetic encoder selects \([ya]\) and \([ras]\) for the criticised form \(sigaar\ 's\) and \([ya]\) and \([ras]\) for the plural form \(sigaren\). Finally, the phonetic encoder addresses the syllable programs in the syllabary, thereby making the programs available to the articulators for the control of the articulatory movements (following Levelt, 1992; Levelt & Wheeldon, 1994). The encoder uses the metrical representation to set the parameters for loudness, pitch, and duration (see Levelt, 1989, for a discussion of the role of phrase-level prosody in setting the parameters). After programming some further articulatory adjustments, the phonetic plan will govern articulation.

In sum, word-form encoding is achieved by a spreading-activation based network with labelled links that is combined with a parallel production-rule system. WEAVER++ also provides for a suspend/resume mechanism that supports incremental generation of phonetic plans. Incremental production means that encoding processes can be triggered by a fragment of their characteristic input (Levelt, 1989). The three processing stages compute aspects of a word form in parallel from the beginning of the word to its end. For example, syllabification can start on the initial segments of a word without having all of its segments. Only initial segments and, for some words, the metrical structure are needed. When given partial information, computations are completed as far as possible, after which the computed representations are buffered and the process is put on hold. When given further information, the encoding processes continue from where they stopped.

**Some applications**

SOA curves of form effects. Meyer and Schriefers (1991) examined the effect of spoken distractor words on word-form encoding in object naming. The experiments were conducted in Dutch. The target and distractor words were either monomorphic monosyllables or disyllables. The monosyllabic targets and distractors shared either the syllable onset and nucleus (begin related) or the nucleus and coda (end related). For example, participants had to name a pictured bed (i.e., they had to say \(bed\), \([be\,l]\), where the distractor was \(bek\) ([\(b\,ek\)], begin related; \(beak\)), \(pet\) ([\(pet\], end related; \(cap\), or there was no distractor (silence condition). Recombining pictures and distractors created unrelated control conditions. The disyllabic targets and distractors shared either the first syllable (begin related) or the second syllable (end related). For example, the participants had to name a pictured table (i.e., they had to say \(tafel\), \([\,ta.\,f\,el\])\), where the
distractor was *tapiр* (['ta.pir], begin related; id.), *jofeľ* (['jo.fel], end related; **pleasant**), or there was no distractor. The distractors were presented just before (i.e., −300 or −150 ms), simultaneously with, or right after (i.e., +150 ms) picture onset.

The presentation of spoken distractors yielded longer object naming latencies compared to the situation without a distractor (cf., Glaser, 1992). The naming latencies were prolonged less with related distractors than with unrelated ones. Thus, a facilitatory effect was obtained from word-form overlap relative to the non-overlap situation. The difference between begin and end overlap for both the monosyllables and the disyllables was in the onset of the facilitatory effect. The onset of the effect in the begin condition was at SOA = −150 ms, whereas the onset of the effect in the end condition was at SOA = 0 ms. With both begin and end overlap the facilitatory effect was still present at the SOA of +150 ms.

According to WEAVER++, both begin-related (e.g., first-syllable) and end-related (e.g., second-syllable) spoken primes yield facilitation, because they will activate segments of the target word in memory and therefore speed up its encoding. Simply put, the onset difference between begin and end overlap reflects the rightward prosodification of segments. Computer simulations showed that WEAVER++ accounts for the empirical findings (Roelofs, 1997c). With begin overlap, the model predicts for SOA = −150 ms a facilitatory effect of −29 ms for the monosyllables (the real effect was −27 ms) and a facilitatory effect of −28 ms for the disyllables (real: −31 ms). In contrast, with end overlap, the effect for SOA = −150 ms was −3 ms for the monosyllables (real: −12 ms) and −4 ms for the disyllables (real: +10 ms). With both begin and end overlap the facilitatory effect was present at the SOAs of 0 and +150 ms. Thus, the model captures the basic findings.

**Implicit priming.** Meyer (1990, 1991) examined the planning of word forms using the so-called implicit priming paradigm. This paradigm involves producing words from learned paired-associates. The big advantage of this paradigm compared to the more widely used picture–word interference paradigm is that the responses do not need to be names of depictable entities, which puts less constraints on the selection of materials. Roelofs (1999) showed that word production from paired associates and picture naming yields equivalent outcomes. In Meyer’s experiments, participants first learned small sets of prompt-response pairs such as \{**blad**–**tabel**, **dier**–**tapiр**, etc.\} (\{**top**–**table**, **animal**–**tapiр**, etc.\}). After learning a set, the production of the response words was tested in a block of trials. On each trial, one of the prompts (the first word of a pair) was visually presented on a computer screen. The order of prompts across trials was random. The task for a participant was to produce the second word of a pair (e.g., the response **tabel**) upon the visual presentation of the first word (the prompt **blad**). The instruction was to respond as quickly as possible without making mistakes. The production latency (i.e., the interval between prompt onset and speech onset) was the main dependent variable. In the experiments there were homogeneous
and heterogeneous response sets. In a homogeneous set, the response words shared part of their form and in a heterogeneous set they did not. For example, the responses shared the first syllable (ia in tafel, tapir, etc.) or the second syllable (fel in tafel, jofel, etc.) or they were unrelated (tapir, jofel, etc.). The same prompt-response pairs were tested in the homogeneous and heterogeneous conditions; only their combinations into sets differed. Therefore, all uncontrolled item effects were kept constant across these conditions. Each participant was tested on all sets.

Meyer found shorter production latencies in homogeneous than in heterogeneous sets. However, this difference was dependent on serial order in that it was only obtained when the response words in homogeneous sets shared one or more word-initial segments, but not when they shared word-final segments. Thus, a facilitatory effect was obtained for the set that included tafel and tapir but not for the set that included tafel and jofel. Furthermore, facilitation increases with the number of shared segments. This holds not only for overlap within syllables but also across syllable and word boundaries, as shown by Roelofs (1998).

According to WEAVER++, the seriality phenomenon reflects the suspend-resume mechanism that underlies the incremental planning of an utterance. Assume the response set consists of tafel, tapir and so forth (i.e., the first syllable is shared). Before the beginning of a trial, the morphological encoder can do nothing, but the phonological encoder can construct the first phonological syllable (ta), and the phonetic encoder can recover the first motor program [ta]. When the prompt blad is presented, the morphological encoder will retrieve <tafel>. Segmental spellout makes available the segments of this morpheme, which includes the segments of the second syllable. The phonological and phonetic encoders can start working on the second syllable. In the heterogeneous condition (tapir, jofel, etc.), nothing can be prepared. There will be no morphological encoding, no phonological encoding, and no phonetic encoding before the beginning of a trial. In the end-homogeneous condition (tafel, jofel, etc.), nothing can be prepared either. Although the segments of the second syllable are known, the phonological word cannot be computed because the remaining segments are to the left of the suspension point. In WEAVER++, this means that the syllabification process has to go to the initial segments of the word, which amounts to restarting the whole process (like unravelling a woven fabric). Thus, a facilitatory effect will be obtained for the homogeneous condition relative to the heterogeneous condition for the begin condition only.

Computer simulations of these experiments supported this theoretical analysis (Roelofs, 1994, 1997c). Advance knowledge about a syllable was simulated by completing the phonological and phonetic encoding of the syllable before the beginning of a trial. For the begin condition, the model yielded a facilitatory effect of −43 ms (real: −49 ms, collapsed across trochaic feet and iambs), whereas for the end condition it predicted an effect of 0 ms (real: +5 ms). Thus, WEAVER++ captures the empirical phenomenon.
Implicit versus explicit priming. The results of implicit and explicit priming are different in an interesting way. In implicit priming experiments, the production of a disyllabic word like *tafel* is speeded up by advance knowledge about the first syllable (*ta*) but not by advance knowledge about the second syllable (*fel*), as shown by Meyer (1990, 1991). In contrast, when first-syllable or second-syllable spoken primes are presented during the production of a disyllabic word, both primes yield facilitation (Meyer & Schriefers, 1991). Only the onset of the facilitation differs. As we saw, WEAVERR++ resolves the discrepancy. According to the model, both first-syllable and second-syllable spoken primes yield facilitation, because they will activate segments of the target word in memory and therefore speed up its encoding. Implicit priming reflects the rightward prosodification of segments. Thus, later syllables cannot be prepared before earlier ones.

New experiments (cf., Roelofs, 1997c) tested WEAVERR++'s prediction that implicit and explicit primes yield independent effects. In the experiments, there were homogeneous and heterogeneous response sets (the implicit primes) as well as form-related and form-unrelated spoken distractors (the explicit primes). Participants had to produce single words such as *tafel*, simple imperative sentences such as “zoek op!” (“look up!”), or cliticisations such as “zoek ’s op!” (“just look it up”) where the reduced form ’s [əs] of *eens* is attached to the base verb. In homogeneous sets, the responses shared the first syllable, (e.g., *ta* in *tafel*), the base verb (e.g., *zoek—look* in “zoek op!”), or the base plus clitic (e.g., *zoek ’s* in “zoek ’s op!”). The spoken distractors consisted of the final syllables of the utterances, either a target syllable (e.g., *fel* for *tafel* or *op* for “zoek op!”), the related condition), a syllable of another item in the response set (the unrelated condition), or there was no distractor (the silence condition). The homogeneity variable (implicit) and the distractor variable (explicit) yielded main effects and the effects were additive (see Figure 4.7). Furthermore, as predicted by

![Graph showing production latency for implicit and explicit priming](image)

**Figure 4.7.** Production latencies for combining implicit and explicit primes: Empirical data (cf., Roelofs, 1997c) and predictions by WEAVERR++.
WEAVER++, the effects were the same for the production of single words, simple imperative sentences, and cliticisations. Recall that the prosodification of these three types of utterance proceeds in the same manner in the model.

Rightward incrementality and morphological decomposition. It is not only characteristic of WEAVER++ that its encoding algorithm operates in a rightward incremental fashion (like in weaving a fabric, the process goes from side to side), but also that it requires morphologically decomposed form entries. Morphological structure is needed, because some morphemes (e.g., prefixes) define independent domains of syllabification (cf., Booij, 1995). For example, without morphological structure, the /r/ of the prefix *ver-* of *vereisen* (demand) would incorrectly be syllabified with the base *eisen*, following the maximal onset principle.

Roelofs (1996c) tested effects of rightward incrementality and morphological decomposition using the implicit priming paradigm. WEAVER++ predicts that a larger facilitatory effect should be obtained when shared initial segments constitute a morpheme than when they do not. For example, the preparation effect should be larger for sharing the syllable *bij* in response sets including Dutch compounds such as *bijrol* (morphemes <bij> and <rol>, *supporting role*) than for sharing the syllable *bij* in sets including simple words such as *bijbel* (morpheme <bijbel>, * bible*). For sets with monomorphic words like *bijbel* consisting of the morpheme <bijbel>, sharing the first syllable *bij* allows phonological preparation only. In contrast, for sets with polymorphic words like *bijrol* consisting of the morphemes <bij> and <rol>, additional morphological preparation is possible.

When the monomorphic word *bijbel* is in a homogeneous condition where the responses share the syllable *bij*, the phonological syllable (bei)\_\_ and the motor program [bei] can be planned before the beginning of a trial. The morpheme <bijbel> and the second syllable *bel* will be planned during the trial itself. In a heterogeneous condition where the responses do not share part of their form, the whole monomorphic word *bijbel* has to be planned during the trial. When the polymorphic word *bijrol* is in a homogeneous condition where the responses share the syllable *bij*, the first morpheme <bij>, and the phonological syllable (bei)\_\_ and the motor program [bei] may be planned before the beginning of a trial. Thus, the second morpheme node <rol> has to be selected during the trial itself, and the second syllable *rol* has to be encoded at the phonological and the phonetic level. In the heterogeneous condition, however, the initial morpheme node <bij> has to be selected first, before the second morpheme node <rol> and its segments can be selected so that the second syllable *rol* can be encoded. Thus, in case of a polymorphic word such as *bijrol*, additional morphological preparation is possible before the beginning of a trial. Consequently, extra facilitation should be obtained. Thus, the facilitatory effect for *bij* in *bijrol* (consisting of the morphemes <bij> and <rol>) should be larger than the effect for *bij* in *bijbel* (<bijbel>).

The outcomes confirmed the predictions by WEAVER++. In producing disyllabic simple and compound nouns, a larger facilitatory effect was obtained
when a shared initial syllable constituted a morpheme than when it did not (see Figure 4.8). For example, the effect was larger for bij in bijrol (bij and rol) than for bij in bijbel (bijbel).

The outcomes of further experiments supported WEAVERT++'s claim that word forms are planned in a rightward fashion. In producing nominal compounds, no facilitation was obtained for non-initial morphemes. For example, no effect was obtained for rol in bijrol. In producing prefixed verbs, a facilitatory effect was obtained for the prefix but not for the non-initial base. For example, a facilitatory effect was obtained for the prefix <be> of behalen (to obtain), but not for the base <halen>.

According to WEAVERT++ morphological complexity can play a role in form planning without having a synchronic semantic motivation. As indicated, some morphemes, such as prefixes, indicate domains of syllabification and this is independent of semantic transparency. For example, the /t/ of the opaque prefixed verb verijdelen (frustrate) is syllabified with the prefix and not with the base as the maximal onset principle would predict. Indeed, Roelofs and Baayen (cf., Roelofs, 1996b) obtained the effect of morpheme preparation for semantically opaque compounds like bijval (bij<val> applause). In producing simple (bijbel<bijbel>) and compound nouns (bijrol <bij<rol> and bijval <bij<val>), a larger preparation effect was obtained when a shared initial syllable constituted a morpheme than when it did not, replicating Roelofs (1996c). Importantly, the size of the morphemic effect was identical for semantically transparent compounds (bijrol) and opaque compounds (bijval), which suggests that morphemes are present in the memory representations of opaque words. These findings support WEAVERT++'s modular view of form planning in which morphology operates "by itself" (cf., Aronoff, 1994). Garrett (1980) arrived at a similar
conclusion on the basis of speech error analyses, arguing for the representation of "pseudomorphs" in the mental lexicon.

**Metrical structure.** In developing WEAVER++, a specific role has been assigned to metrical structures in syllabification (see Roelofs & Meyer, 1998). For words like the trochee *tafel*, metrical structures are computed on-line, but for words like the iamb *sigaar*, metrical structures are stored. By contrast, Levelt (1992) does not make this differentiation. The stored metrical structures specify the number of syllables and the stress pattern, but not the precise CV structure of the syllables as the CV headers of Dell (1988) do. The prosodification process in WEAVER++ associates segments to the syllable nodes within the metrical structure for "exception" words (e.g., *sigaar*) or constructs syllable and metrical structures based on segmental information (e.g., for *tafel*). Roelofs and Meyer (1998) conducted a number of implicit-priming experiments designed to test this view on phonological encoding.

On each trial, participants had to produce one Dutch word out of a set of three, or four, as quickly as possible. In homogeneous sets, the responses shared a number of word-initial segments, whereas in heterogeneous sets they did not. As we saw, earlier research has shown that sharing initial segments reduces production latencies (Meyer, 1990, 1991; Roelofs, 1996c). The responses shared their metrical structure (the constant sets) or they did not (the variable sets).

A first series of experiments (Roelofs & Meyer, 1998) tested predictions of WEAVER++ about the role of metrical structure in the production of polysyllabic words that do not have main stress on the first stressable syllable, such as *sigaar*. According to the model, the metrical structures of these words are stored in memory. WEAVER's view of syllabification implies that preparation for word-initial segments should only be possible if such response words have an identical metrical structure. If the responses in a set have different metrical structures, segment-to-frame association cannot take place before the beginning of a trial, and no preparation effect should be obtained. This prediction was tested by comparing the preparation effect for response sets with a constant number of syllables such as \{manier (manner), matras (mattress), makreel (mackerel)\} (all two syllables) to that for sets having a variable number of syllables such as \{maaioor (major), materie (matter), malaria (malaria)\} (respectively, two, three, and four syllables). In the example, the responses share the first syllable *ma*. Word stress was always on the second syllable. As predicted, facilitation was obtained for the metrical constant sets but not for the variable sets. The same predictions were also tested by comparing the preparation effect for response sets with a constant stress pattern such as \{marine (navy), materie (matter), malaise (depression), madonna (madonna)\} (all responses having stress on the second syllable) to that for sets having a variable stress pattern such as \{marine (navy), materie (matter), manuscript (manuscript), madelief (daisy)\} (the first two responses having stress on the second syllable and the last two responses
having stress on the third syllable). All response words were trisyllabic. Again, as predicted, facilitation was obtained for the constant sets but not for the variable sets.

The results of the experiments already suggest that constancy in CV structure is not necessary to observe a preparation effect, because in none of the homogeneous sets of these experiments was this structure identical across response words (though one could perhaps argue that the response words in the metrically constant sets were more similar in CV structure than those in metrically variable sets). Results obtained by Meyer (1990, 1991) and Roelofs (1996b,c, 1997b, 1998) also suggest that implicit priming effects can be obtained for homogeneous sets with variable CV structures. However, though constancy in CV structure does not appear to be necessary for obtaining a facilitatory effect, it is still possible that stronger effects arise for sets with constant than with variable CV structure. This is not predicted by WEAVER++, but, as explained earlier, other computational models such as those of Dell (1988) and Schade and Berg (1992) assume an explicit representation of CV structure. Therefore, we tested whether the size of the preparation effect was affected by the constancy versus variability of the CV structure of the response words. We compared the effect of segmental overlap for response sets having a constant CV structure such as \{bres (breach), bril (glasses), brok (piece), brug (bridge)\} (responses all CCVC) to that for sets having a variable CV structure such as \{brij (porridge), brief (letter), bron (source), brand (fire)\} (responses respectively, CCVV, CCVVC, CCVC, CCVCC). In the example, the responses share the onset cluster br. Facilitation from segmental overlap was obtained for both the constant and the variable sets. The size of the preparation effect was the same for both types of set.

These results suggest that the exact CV structure is not stored, thereby refuting the CV headers of the Dell model. With constant CV structure, the system might select a header (i.e., CCVC) and work its way through the sequence of segment categories corresponding to the shared segments. After the last shared category node has been reached and the corresponding segment has been selected, the encoding process is suspended. The process is resumed when the first non-shared segment has been made available by the morpheme that is derived from the prompt. With variable CV structure, however, the system cannot select a header and work its way through the sequence of segment categories corresponding to the shared segments. For example, suspension in a CCVV sequence leads to wrong results if it turns out that a CCVVC word has actually to be produced. But empirically the same amount of facilitation from segmental overlap was obtained for the constant and the variable sets.

WEAVER++ explains why preparation for word-initial segments is only possible for response words with identical number of syllables and stress pattern, and why an identical CV structure is not needed. Figure 4.9 gives the results of simulations comparing the effect of segmental overlap for response sets with a
constant number of syllables such as \{manier, matras, makreel\} to that for sets having a variable number of syllables such as \{majoor, materie, malaria\}. Varying the place of stress while keeping the number of syllables fixed gives the same results. As can be seen, WEAVER++ accounts for the key empirical finding concerning metrical structure. In contrast, if metrical structures are not involved in advance planning or if metrical structures are computed on-line on the basis of segments for these words, sharing metrical structure should be irrelevant for preparation. Then, preparation (i.e., on-line computing the syllable \textit{ma} before the beginning of a trial and computing the remainder of the word form during the trial itself) should be possible for both the metrically constant and variable sets.

We also tested predictions of WEAVER++ about the role of metrical structure in producing monosyllabic words and polysyllabic words whose main stress is on the first syllable like \textit{tafel} (see Levelt et al., 1999a). According to the model, the metrical structures of these words are computed on-line by the prosodification process. That is, syllabification and stress assignment (i.e., footing) is done on the basis of retrieved segments. Consequently, implicit priming of initial segments should now be possible for both metrically constant and variable sets. This prediction was tested by comparing the preparation effect for response sets with a constant number of syllables such as \{borstel (brush), botsing (crash), bochel (hump), bonje (fight)\} (all disyllables stressed on the first syllable) to that for sets having a variable number of syllables such as \{borstel, botsing, bok (goat), bom (bomb)\} (two disyllables stressed on the first syllable and two monosyllables, respectively). In the example, the responses share the onset and nucleus \textit{bo}. As predicted, facilitation was obtained (in an equal amount) for both the constant and the variable sets. The same result is predicted for varying the number of syllables of polysyllabic words with an unstressable first syllable (i.e., words
with a schwa as the first vowel) and stress on the second syllable. This was tested by comparing the preparation effect for such words in response sets with a constant number of syllables such as \{gebit (teeth), gezin (family), getal (number), gewei (antlers)\} (all disyllables having stress on the second syllable) to that for sets having a variable number of syllables such as \{geraanie (skeleton), getuige (witness), gebit, gezin\} (two trisyllables stressed on the second syllable and two disyllables stressed on the second syllable, respectively). As predicted, facilitation was obtained (in an equal amount) for both the constant and the variable sets.

Finally, we tested predictions of WEAVER++ about the production of cliticisations and suffixed forms. According to Levelt (1992), cliticisations like sigaar 's (e.g., “probeer deze sigaar 's”, “just try this cigar”) are produced by first combining the retrieved metrical frames for sigaar and 's, followed by segment-to-frame association. By contrast, WEAVER++ has implemented the claim that first sigaar is syllabified and later 's is adjoined (Roelofs, 1997b). The same holds for the production of the plural form sigaren: First the stem sigaar is syllabified, and later the plural suffix –en is attached and prosodified. These different views were tested by comparing the effect of segmental overlap for response sets that combine disyllabic nouns (e.g., dozijn–dozen) with disyllabic verb stems (e.g., doneer–donate) to that of sets combining the disyllabic nouns with trisyllabic cliticised forms of the verbs (doneer's) or with trisyllabic infinitival forms of the verbs (doneren). If metrical frames are combined before segment-to-frame association, then the sets with the cliticisations (e.g., the set including disyllabic dozijn and trisyllabic doneer 's) and the sets with the infinitives (e.g., the set including disyllabic dozijn and trisyllabic doneren) would be metrically variable, and preparation should not be possible. According to WEAVER++, however, all three types of set are metrically constant, because the clitic and the plural suffix are metrically independent elements that are adjoined after syllabification of the verb stem. The outcomes of the experiment supported the predictions of WEAVER++. The sets with the cliticisations (e.g., the set including dozijn and doneer 's) and the sets with the infinitives (e.g., the set including dozijn and doneren) yielded an equal amount of segmental facilitation, and the size of the facilitation was the same as that for the sets including the disyllabic verb stems (e.g., the set including dozijn and doneer).

In summary, I have reviewed empirical evidence that supports the claim (implemented in WEAVER++) that syllable structure is computed on-line and in a left-to-right fashion (Levelt, 1992). The evidence suggests that syllable structure is computed by associating retrieved segments to the syllable nodes within retrieved metrical frames for polysyllabic words that do not have main stress on the first stressable syllable (which holds for roughly 10% of the words) and by constructing syllable and metrical structures based on segmental information for monosyllabic words and for all other polysyllabic words (the remaining 90% of the words). Furthermore, I reviewed evidence that supports the claim that
syllabification across morpheme and word boundaries is achieved by adjoining a suffix or clitic to an already partly syllabified base rather than by first combining the metrical frames for the base and for the suffix or clitic followed by segment-to-frame association.

**Frequency effects.** WEAVER++ can capture frequency effects in word production. Frequency effects in the model originate from differences in the speed of production-rule application. Speed depends on frequency of usage (more experienced spiders work faster in making a web).

Experiments by Jescheniak and Levelt (1994; Jescheniak, 1994) have shown that when lemma information such as grammatical gender is accessed, a frequency effect is obtained. For example, Dutch participants had to decide on the gender of a picture’s name (e.g., they had to decide that the grammatical gender of *tafel* is non-neuter), which was done faster for high-frequency words than for low-frequency ones. The effect disappeared over repetitions, contrary to a “robust” frequency effect obtained in naming the pictures.

Jescheniak and Levelt (1994) provided evidence that the locus of the robust frequency effect is the form level. When participants had to respond to an English probe word by producing its Dutch translation equivalent, the production latency of a low-frequency homophone was determined by the sum frequency of that word and its high-frequency counterpart. For example, participants had to produce the Dutch word *bos* in response to *bunch* (low-frequency reading). The production latencies for these homophones were compared to the latencies for two types of other words. First, there were low-frequency control words whose frequency was matched to that of the low-frequency reading of the homophone. The low-frequency control for *bos* was *hok* (*kennel*). Second, there were high-frequency control words whose frequency was matched to the sum frequency of the low-frequency reading (i.e., *bunch*) and high-frequency reading (i.e., *forest*) of *bos*. The high-frequency control for *bos* was *hoek* (*corner*). Producing the homophones (*bos*) in their low-frequency reading went as fast as producing the high-frequency controls (*hoek*), and it went faster than producing the low-frequency controls (*hok*). A low-frequency homophone inherits the frequency of its high-frequency counterpart. In WEAVER++, homophones share their form nodes in common but not the lemma (Dell, 1990 and Jescheniak & Levelt, 1994 also take this point of view). By sharing form nodes, a low-frequency homophone inherits the frequency properties of its high-frequency counterpart. This explains the homophone effect observed by Jescheniak and Levelt.

WEAVER++ predicts also an effect of morpheme frequency for the constituents of polymorphemic lexical items. This was tested by Roelofs (1996b, 1998) using the implicit priming paradigm. High-frequency morphemes are retrieved faster from memory than morphemes of low frequency, so the benefit from preparation should be larger for low-frequency morphemes than for high-frequency
ones. This prediction was empirically confirmed. For example, in producing compounds (Roelofs, 1996b), the facilitatory effect was larger for response sets sharing a low-frequency morpheme like \textless bloem\textgreater{} (\textit{flower})—as in bloemkool (\textit{cauliflower})—than for response sets sharing a high-frequency morpheme like \textless bloed\textgreater{} (\textit{blood})—as in bloedspoor (\textit{trace of blood}). Also, in producing particle verbs (Roelofs, 1998), the facilitatory effect was larger for veeg (low frequency) in “veeg op!” (“clean up!”) than for geef (high frequency) in “geef op!” (“give up!”).

Finally, Levelt and Wheeldon (1994) observed that in word-form encoding an effect of syllable frequency is obtained that is independent from the word-frequency effect. Furthermore, in producing disyllabic words, the syllable-frequency effect was confined to the second syllable. Again, if a selection procedure for a syllable program node is run faster for a high-frequency syllable than for a low-frequency one, then the finding of an independent effect of syllable frequency is readily explained. Furthermore, during the encoding of disyllables, the encoding of the first syllable will have a head-start as a result of the left-to-right prosodification of segments. Thus, according to the model, the second syllable typically sets the pace of the encoding of the whole word form. Therefore, one expects the syllable-frequency effect to be confined to the second syllable, as is observed by Levelt and Wheeldon.

The syllable frequency effects obtained by Levelt and Wheeldon (1994) were very small. Furthermore, in some experiments, syllable and segment frequency were correlated. In recent experiments by Levelt and Meyer (reported in Meyer, 1997) that controlled for a number of possible confounds, effects of syllable, and segment frequency were not obtained. Clearly, the speed of accessing syllables does not depend very much on their frequency.

Simulations showed that WEAVER++ captures the word-frequency effects (obtained by Jescheniak and Levelt), the small syllable frequency effects (obtained by Levelt and Wheeldon), and their independence. In the earlier simulations, the frequency of the targets was held constant in the model. In the current simulations, frequency was explicitly manipulated by varying verification times as a function of the frequency of the item, so that different items took different periods of encoding time within the network. For example, in producing the word \textit{wortel} (\texttt{[ˈwɔrtɛl]}, \textit{carrot}), doubling the duration of the selection procedure of the first syllable [\texttt{wɔ}] increased the word-form encoding latency by +6 ms (Levelt & Wheeldon observed a non-significant frequency effect for the first syllable of −3 ms), whereas doubling the duration of the procedure of the second syllable [\texttt{tɛl}] increased the encoding latency by +19 ms (real: +12). This difference between first-syllable and second-syllable frequency was independent of the effect of word frequency. For example, tripling the duration of a morpheme test increased the word-form encoding latency by +25 ms, but the frequency-effect difference between the first syllable and the second syllable remained the same.
Segments and features. According to WEAVERR++, preparation in an implicit-priming experiment critically depends on shared segments rather than shared articulatory movements or phonological features. In support of this, it has been shown empirically that preparation requires that the responses share their initial segments fully and that sharing features only does not allow for preparation (Roelofs, 1999). For example, the initial segment of the words boat, bird, and boy is the same, whereas the initial segment of the words paint, boat, and bird is the same except for one feature, namely voicing (i.e., /b/ is voiced, /p/ is voiceless). However, a preparation effect was only obtained for the sets with full segment overlap, but no effect was observed at all (not even a reduction of latencies) for the sets with feature overlap. The same result has been obtained in comparing sets of disyllabic words sharing the first syllable fully (e.g., te in tennis, terrace, teddy) and sets of disyllabic words whose first syllable differs in one feature only (e.g., devil, tennis, teddy). Although syllables were shared except for one feature in the first segment, this feature difference completely blocked preparation. Also, the same results were obtained when words were produced in response to pictured objects, and when place of articulation rather than voicing was manipulated (e.g., /m/ versus /n/).

To conclude, the special status of identity suggests that segments are planning units independent of their features. The findings support segmental models such as WEAVERR++, but they argue against models without an explicit representation of segments, such as the PDP model proposed by Dell, Juliano, and Govindjee (1993).

Summary and discussion

I have reviewed empirical evidence for the claim that lemmas are mapped onto syllable-based articulatory programs by serially grouping the segments of morphemes into phonological syllables, after which these phonological syllables are used to address the programs in a syllabary. In agreement with the speech error evidence, the chronometrical findings (i.e., Roelofs, 1996b,c, 1998) support, among other things, the assumption that there is a level of concrete morphemes in addition to the level of syntactically specified lemmas. By contrast, Caramazza (1997) proposed a model with only a single, syntactically specified level intermediate between concepts and segments (the model of Starreveld & La Heij, 1996, is similar to that of Caramazza but contains no syntax).

There are several problems with Caramazza’s proposal that a single lexical level suffices. First of all, by assuming just one lexical level, his model fails to account for the speech error evidence that suggests two types of morpheme error (Dell, 1986; Garrett, 1975, 1980, 1988). Furthermore, Caramazza's model has difficulty accounting for the observation that low-frequency homophones inherit the frequency properties of their high-frequency twins (Jescheniak & Levelt, 1994). The Caramazza model assumes separate lexical nodes for homophones of
different syntactic classes, so *more* (adjective) and *moor* (noun) each have their own lexical node, which fails to explain why low-frequency *moor* behaves like high-frequency *more*. To save the model, Caramazza and Miozzo (1998) have suggested that feedback from segments to lexical nodes may give rise to the inherited effect (cf., Dell, 1990). However, this is no option for their model, because it contains no backward links. Also, since syntactic properties and segments are accessed through the same lexical node, the model cannot explain the dissociation between the frequency-effect in gender decision and in picture naming. If the frequency effect resides at the lexical node, the effect should be robust both in picture naming and gender decision, contrary to the empirical findings. For an extensive discussion of these and other issues, refer to Roelofs et al. (1998) and Caramazza and Miozzo (1998).

**Speech errors**

Although WEAVER++ has not been designed to account for speech errors, it can be shown that the model is compatible with speech error data (see Roelofs, 1997c). Here, I briefly indicate how the model copes with the error data reviewed in discussing the Dell model. I address the relative frequencies of segmental substitution errors (e.g., the anticipation error *sed sock* for *red sock* is more likely than the perseveration *red rock*, which is in its turn more likely than the exchange *sed rock*), effects of speech rate on error probabilities (e.g., more errors at higher speech rates), the phonological facilitation of semantic substitution errors (e.g., *calf* for *cat* is more likely than *dog* for *cat*), and lexical bias (i.e., errors tend to be real words rather than nonwords).

In WEAVER++, phonological errors may be due to indexing failures by the phonetic encoder. For example, in the planning of "red sock", the production rule of [sed] might find its condition satisfied. It wants to have an onset /s/, a nucleus /e/, and a coda /d/, which is present in the phonological representation. The error is of course that the /s/ is in the wrong phonological syllable. If the production rule of [red] does its job well, there will be a race between [red] and [sed] to become the first syllable in the articulatory program for the utterance. If [sed] wins the race, the speaker will make an anticipation error. If this indexing error occurs, instead, for the second syllable, a perseveration error will be made, and if the error is made both for the first syllable and the second one, an exchange error will be made. Errors may also occur when WEAVER++ skips verification to gain speed in order to obtain a higher speech rate. Thus, more errors are to be expected at high speech rates.

Figure 4.10 gives some simulation results concerning segmental anticipations, perseverations, and exchanges. The real data are from the Dutch error corpus of Nooteboom (1969), which are typical. As can be seen, WEAVER++ can capture some of the basic findings about the relative frequency of these types of substitution errors in spontaneous speech. The anticipation error *sed sock* for *red sock* is
more likely than the perseveration *red rock*, which is in turn more likely than the exchange *sed rock*. The model predicts almost no exchanges, which is, of course, a weakness of the model. However, comparison of WEAVER++'s performance with the error-based model of Dell (1986) shows that WEAVER++ does not behave poorly. At a low speech rate \((r = 8)\), the Dell model predicts no exchanges and at high speech rate \((r = 3)\) the model predicts incorrectly more perseverations than anticipations.

Lexical bias has traditionally been taken as an argument for backward links in a lexical network. Backward links are present in the classical model (Dell, 1986) but absent in WEAVER++. Segmental errors tend to create words rather than nonwords. For example, the selection of /h/ resulting in *hat* for *cat* is more likely than the selection of /z/ resulting in *zat* for *cat*. In the classical model, lexical bias is a result of feedback from segment nodes to word nodes (i.e., lemma nodes or morpheme nodes) and from word nodes back to segment nodes. Strings of segments that make up words receive feedback from word nodes but nonword strings do not. So, /h/ receives feedback from *hat* but /z/ not from *zat*, because the latter is not part of the lexicon. Reverberation of activation in the network takes time, so lexical influences on errors take time to develop, as empirically observed (Dell, 1986).

The classical account of lexical bias meets, however, with a difficulty. In this view, lexical bias is an automatic effect. The seminal study of Baars, Motley,
and MacKay (1975), however, has already shown that lexical bias is not a necessary effect. When all the target and filler items in an error-elicitation experiment are nonwords, word and nonword slips occur equally often. Only when some words are included as filler items does the lexical bias appear. The account of Baars et al. of lexical bias was in terms of speech monitoring by speakers. Just before articulation, speakers monitor their phonetic plan for errors. If an experimental task exclusively deals with nonwords, speakers do not bother to attend to the lexical status of their phonetic plan. As proposed by Levelt (1989), the monitoring may be achieved by feeding the phonetic plan to the speech comprehension system. On this account, there is no direct feedback in the output form lexicon, but only indirect feedback via the speech comprehension system. Feedback via the comprehension system takes time, so lexical influences on errors take time to develop.

Similarly, the phonological facilitation of semantic substitutions may be a monitor effect. The target cat will be in the comprehension cohort of calf but not of dog. Consequently, it is more likely that calf will pass the monitor than that dog will. There exists also another potential error source within a forward model such as WEAVER++. Occasionally, the lemma retriever may erroneously select two lemmas instead of one, the target and an intruder. This assumption is independently motivated by the occurrence of blends such a close combining clear combining close and near (Roelofs, 1992a). In WEAVER++, the selection of two lemmas instead of one will lead to the parallel encoding of two word forms instead of one. The encoding time is a random variable, whereby the word form that is ready first will control articulation. In WEAVER++, it is more likely that the intruder wins the form race when there is phonological overlap between target and intruder (i.e., when the form of the target primes the intruder) than when there is no phonological relation. Thus, WEAVER++ predicts that the substitution calf for cat is more likely than dog for cat, which is the phonological facilitation of semantic substitution errors. This also explains the syntactic category constraint on substitution errors, that is, like in word exchanges, in substitution errors the target and the intruder are typically of the same syntactic category.

Wider implications for understanding speech production

The basic design feature of WEAVER++ that gives rise to its psychologically and computationally desirable properties is the integration of a spreading-activation based lexical network with a parallel production-rule system embodying linguistic rules or constraints. Words are not planned by a central agent that overlooks the whole process but by teams of production rules that work in parallel on small parts of the word. The production rules are stored with the relevant data structures and have a limited overview only. Retrieval and encoding in the model are lexically
driven, that is, production rules operate on the basis of information stored with individual words in the lexicon.

These design features of WEAVER++ are similar to those of the algorithm that Kempen and Hoenkamp (1987) proposed for the planning of syntactic structures. A retrieved lemma (e.g., a noun lemma) triggers procedures that build the appropriate syntactic environment for the word (e.g., an NP, S, etc.). Syntactic encoding in their model is lexically driven, that is, it operates on the basis of information stored with individual words in the lexicon. Also, there is no central agent that overlooks the whole process, but syntactic encoding is achieved by a team of expert procedures which work in parallel on small parts of the sentence. To conclude, lexically driven production systems are, in many ways, beginning to form a coherent theoretical paradigm for understanding the planning of speech, both at the syntactic and the word-form encoding level.

WEAVER++ uses production rules in addition to spreading activation. Back in the dark days of behaviourism, any reference to mental notions was considered to be unscientific. Nowadays, in some circles it is held that theorising may involve connectionist notions only and that production rules are forbidden. Usually, this conviction remains operative in the background but sometimes it is publicised. For example, in their peer commentaries on a BBS target article explaining WEAVER++ (Levett et al., 1999a), Santiago and MacKay (1999) called production rules “homunculi”, others held that production rules involve “unnatural computation” (Roberts, Kalish, Hird, & Kirsner, 1999), or that they are “too declarative” for the brain (O’Seaghdha, 1999). The problem with these criticisms is that our knowledge of how the brain works is still so rudimentary that it seems premature to ban any reference to production rules and other symbolic entities. Moreover, these criticisms overlook that the line between production rules and many connectionist constructs is rather thin. For example, production rules without variables can easily be implemented using a connectionist network. Furthermore, proposals have been made for how a connectionist system may achieve variable binding (e.g., Shastri & Ajjanagadde, 1993; Touretzky & Hinton, 1988), in effect implementing productions rules with variables. Production rules mean nothing more than the operations that they specify. Crucial for the issue of “neural plausibility” is whether we can exclude that the brain performs such operations—and the criticisms do not bring forward evidence against that.

There is no direct way to observe how lexical access happens in the mind of speakers. We can only make use of indirect evidence coming from behavioural and neural studies. In the present chapter, I evaluated existing evidence from behavioural studies. On the basis of these data, I made a case for the WEAVER++ model. This does not mean that other models are by necessity fatally flawed. Models in cognitive science are moving targets and it remains to be seen whether the problems that confront other models can ultimately be solved, which is just another way of saying that more research needs to be done.
REFERENCES


