NOMINALIZATIONS IN *PUNDIT*

Deborah A. Dahl, Martha S. Palmer, Rebecca J. Passonneau
Paoli Research Center
UNISYS Defense Systems[1]
Defense Systems, UNISYS
P.O Box 517
Paoli, PA 19301 USA

ABSTRACT

This paper describes the treatment of nominalizations in the PUNDIT text processing system. A single semantic definition is used for both nominalizations and the verbs to which they are related, with the same semantic roles, decompositions, and selectional restrictions on the semantic roles. However, because syntactically nominalizations are noun phrases, the processing which produces the semantic representation is different in several respects from that used for clauses. (1) The rules relating the syntactic positions of the constituents to the roles that they can fill are different. (2) The fact that nominalizations are untensed while clauses normally are tensed means that an alternative treatment of time is required for nominalizations. (3) Because none of the arguments of a nominalization is syntactically obligatory, some differences in the control of the filling of roles are required, in particular, roles can be filled as part of reference resolution for the nominalization. The differences in processing are captured by allowing the semantic interpreter to operate in two different modes, one for clauses, and one for nominalizations. Because many nominalizations are noun-noun compounds, this approach also addresses this problem, by suggesting a way of dealing with one relatively tractable subset of noun-noun compounds.

[1]Formerly SDC—A Burroughs Company.

1. Introduction

In this paper we will discuss the analysis of nominalizations in the PUNDIT text processing system.[2] Syntactically, nominalizations are noun phrases, as in examples (1)-(7).

(1) An ***inspection of lube oil filter*** revealed metal particles.

(2) ***Loss of lube oil pressure*** occurred during ***operation.***

(3) SAC received ***high usage.***

(4) ***Investigation*** revealed adequate lube oil.

(5) Request ***replacement of SAC.***

(6) ***Erosion of impellor blade tip*** is evident.

(7) Unit has low output air pressure, resulting in ***slow gas turbine starts.***

Semantically, however, nominalizations resemble clauses, with a predicate/argument structure like that of the related verb. Our treatment attempts to capture these resemblances in such a way that very little machinery is needed to analyze nominalizations other than that already in place for other noun phrases and clauses.

There are two types of differences between the treatment of nominalizations and that of clauses. There are those based on *linguistic* differences, related to (1) the mapping between syntactic arguments and semantic roles, which is

[2] The research described in this paper was supported in part by DARPA under contract N000014-85-C-0012, administered by the Office of Naval Research. APPROVED FOR

different in nominalizations and clauses, and (2) tense, which nominalizations lack. There are also differences in ***control***; in particular, control of the filling of semantic roles and control of reference resolution. All of these issues will be discussed in detail below.

2. Clause analysis

The semantic processing to be described in this paper is part of the PUNDIT [3] system for processing natural language messages. The PUNDIT system is a highly modular system, written in Prolog, consisting of distinct syntactic, semantic and discourse components. [Hirschman1985], and[Hirschman1986], describe the semantic components of PUNDIT, while[Dahl1986,Palmer1986,Passonneau1986], describe the semantic and pragmatic components. The semantic domain from which these examples are taken is that of reports of failures of the starting air compressors, or ***sac's,*** used in starting gas turbines on Navy ships.

The goal of semantic analysis is to produce a representation of the information conveyed by the sentence, both implicit and explicit. This involves 1) mapping the syntactic realization onto an underlying predicate argument representation, e.g., assigning referents of particular syntactic consituents to predicate arguments, and 2) making implicit argument fillers explicit. We are using an algorithm for semantic interpretation based on predicate decomposition that integrates the performance of these tasks. The integration is driven by the goal of filling in the predicate arguments of the decomposition.[Palmer1986].

In order to produce a semantic representation of a clause, its verb is first decomposed into a semantic predicate representation appropriate for the domain. The arguments of the predicates constitute the SEMANTIC ROLES of the verb, which are similar to cases[4] For example, ***fail*** decomposes into **become inoperative**, with **patient** as its only semantic role. Semantic roles can be filled either by a syntactic constituent or by reference resolution from default or contextual information.

We have categorized the semantic roles into three classes, based on how they are filled Semantic roles such as **theme, actor** and **patient** are syntactically OBLIGATORY, and must be filled by surface constituents. Semantic roles are categorized as semantically ESSENTIAL when they must be filled even if there is no syntactic constituent available.[5] In this case they can be filled pragmatically, making use of reference resolution, as explained below. The default categorization is NON-ESSENTIAL, which does not require that the role be filled. The algorithm in Figure 1 produces a semantic representation using this information. Each step in the algorithm will be illustrated at least once in the next section using the following (typical) CASREPS text.

Sac failed.
Pump sheared.
Investigation revealed metal contamination in filter.

2.1. A Simple Example

DECOMPOSE VERB - The first example uses the ***fail*** decomposition for ***Sac failed:***

```
fail <-
        becomeP(inoperativeP
                (patient(P))).
```

It indicates that the entity filling the OBLIGATORY **patient** role has or will become inoperative.

FOR **patient** ROLE -

PROPOSE SYNTACTIC CONSTITUENT FILLER - A mapping rule indicates that the syntactic subject is a likely filler for any **patient** role. The mapping rules make use of intuitions about syntactic cues for indicating semantic roles first embodied in the notion of case [Fillmore1968,Palmer1981]. The mapping rules can take advantage of general syntactic cues like "SUBJECT goes to PATIENT" while still indicating particular context sensitivities. (See [Palmer1985] for details.)

PUBLIC RELEASE, DISTRIBUTION UNLIMITED.

[3] PUNDIT UNDderstands and Integrates Text

[4] In this domain the semantic roles include: **agent, instigator, experiencer, instrument, theme, location, actor, patient, source, reference_pt** and **goal**. There are domain specific criteria for selecting a range of semantic roles. The criteria which we have used are described in[Passonneau1986]

[5] We are in the process of defining criteria for categorizing a role as ESSENTIAL. It is clearly very domain dependent,

CALL REFERENCE RESOLUTION - *Sac* is the subject of *sac failed*, and is suggested by the mapping rule as a likely filler of the **patient** role. At this point the semantic interpreter asks noun phrase analysis to provide a unique referent for the noun phrase subject. Since no *sacs* have been mentioned previously, a new name is created: **sac1**.

TEST SELECTION RESTRICTIONS - In addition to the mapping rules that are used to associate syntactic constituents with semantic roles, there are selection restrictions associated with each semantic role. The selection restrictions for *fail* test whether or not the filler of the **patient** role is a mechanical device. A *sac* is a mechanical device so the subject of the sentence *sac failed* maps straightforwardly onto the **patient** role, e.g., **becomeP(inoperativeP(patient(sac1)))**.

Since there are no other roles to be filled the algorithm terminates successfully at this point and the remaining steps are not applied. The next example illustrates further steps in the algorithm.

2.2. Unfilled Obligatory Roles

The second utterance in the example, *Pump sheared*, illustrates the effect of an unfilled obligatory role.

DECOMPOSE VERB -

```
shear
        <- causeP(instigator(I),
                 becomeP(shearedP
                        (patient(P))))
```

Shear is an example of a verb that can be used either transitively or intransitively. In both cases the **patient** role is filled by a mechanical device that becomes *sheared*. If the verb is used transitively, the **instigator** of the *shearing*, also a mechanical device, is mentioned explicitly, as in, ***The rotating drive shaft sheared the pump.*** If the verb is used intransitively, as in the current example, the **instigator** is not made explicit; however, the algorithm begins by attempting to fill it in.

FOR **instigator** ROLE - Working from left to right in the verb decomposition, the first role to be filled is the **instigator** role. A mapping rule indicates that the subject of the sentence, ***pump***, is a likely filler for this role. Reference resolution returns **pump1** as the referent of the noun phrase. Since pump is a mechanical device, the selection restriction test passes.

FOR **patient** ROLE - There are no syntactic constituents left, so a syntactic constituent cannot be proposed and tested.

UNFILLED OBLIGATORY ROLES - The **patient** role, a member of the set of obligatory roles, is still unfilled. This causes failure, and the binding of **pump1** to the **instigator** role is undone. The algorithm starts over again, trying to fill the instigator role.

FOR **instigator** ROLE- There are no other mapping rules for **instigator**, and it is non-essential, so Case 4 applies and it is left unfilled.[6] The algorithm tries again to fill in the patient role.

FOR **patient** ROLE - Two mapping rules can apply to the **patient** role, one of which suggests the subject, in this case, the pump, as a filler. Reference resolution returns **pump1** again, which passes the selection restriction of being a mechanical device. The final representation is:

causeP(instigator(I),
becomeP(shearedP(patient(pump1)))).

The last sentence in the text, "***Investigation revealed metal contamination in filter,***" is interesting mainly because of the occurrence of two nominalizations which are discussed in detail in a separate section.

2.3. Temporal Analysis of Tensed Clauses

The temporal component determines what kind of situation a predication denotes and what time it is asserted to hold for [Passonneau1986]. Its input is the semantic decomposition of the verb and its arguments, tense, an indication of whether the verb was in the perfect or progressive, and a list of unanalyzed constituents which may include temporal adverbials. It generates three kinds of output: an assignment of

and relies heavily on what can be assumed from the context.

[6]In other domains, the **instigator** might be an ESSENTIAL role and would get filled by pragmatics.

an actual time to the predication, if appropriate; a representation of the type of situation denoted by the predication as either a state, a process or a transition event; and finally, a set of predicates about the ordering of the time of the situation with respect to other times explicitly or implicitly mentioned in the same sentence. For the simple sentence, **sac failed**, the input would consist of the semantic decomposition and a past tense marker:

Decomposition:
become(inoperative(patient([sac1])))
Verb form: Past

The output would be a representation of a transitional event, corresponding to the ***moment*** of ***becoming inoperative***, and a resulting state in which the sac is inoperative for some ***period*** initiating at the ***moment*** of transition.

3. Nominalizations

Nominalizations are processed very similarly to clauses, but with a few crucial differences, both in linguistic information accessed and in the control of the algorithm. The first important linguistic characteristic of the nominalization algorithm is that the same predicate decomposition can be used as is used for the related verb. Secondly, different mapping rules are required since syntactically a nominalization is a noun phrase. For example, where a likely filler for the **patient** of *fail*, is the syntactic subject, a likely filler for the **patient** of *failure* is an *of* pp. Thirdly, nominalizations do not make use of the obligatory classification for semantic roles, since noun phrase modifiers are not syntactically obligatory.

In terms of differences in control structure, because nominalizations may themselves be anaphoric, there are two separate role-filling stages in the algorithm instead of just one. The first pass is for filling roles which are explicitly given syntactically; essential roles are left unfilled. If a nominalization is being used anaphorically some of its roles may have been specified or otherwise filled when the event was first described. The anaphoric reference to the event, the nominalization, would automatically inherit all of these role fillers, as a by-product of reference resolution. After the first pass, the interpreter looks for a referent, which, if found, will unify with the nominalization representation, sharing variable bindings. This is a method of filling unfilled roles pragmatically that is not currently available to clause analysis [8]. However, the first pass was important for filling roles with any explicit syntactic arguments of the nominalization before attempting to resolve its reference, since there may be more than one event in the context which nominalization could be specifying. For example, ***failure of pump*** and ***failure of sac*** can only be distinguished by the filler of the **patient** role. After reference resolution a second role-filling pass is made, where still unfilled roles may be filled pragmatically with default values in the same way that unfilled verb roles can be filled.

[7] This suggests the hypothesis that OBLIGATORY roles for clause decompositions automatically become ESSENTIAL roles for nominalization decompositions. This hypothesis seems to hold in the current domain; however, it will have to be tested on other domains. We are indebted to James Allen for this observation.

[8] Clauses can describe previously mentioned events, as discussed in [Dahl1987]. In order to handle cases like these, something analogous to reference resolution for clauses may be required. However a treatment of this has not yet been implemented in PUNDIT.

3.1. Temporal Analysis of Nominalizations

As with clauses, the temporal analysis of nominalizations takes place after the semantic analysis. Also as with clauses, one of the inputs to the temporal analysis of nominalizations is the semantic decomposition. The critical difference between the two cases is that a nominalization does not occur with tense. PUNDIT compensates by looking for relevant temporal information in the superordinate constituents in which the nominalization is embedded. Currently, PUNDIT processes nominalizations in three types of contexts.

The first context for which a nominalization is temporally processed is when it occurs as the prepositional object of a temporal connective (e.g., ***before, during, after***) and the matrix clause denotes an actual situation. For example, in the sentence ***sac lube oil pressure decreased below 60 psig after engagement***, the temporal component processes the main clause as referring to an actual event which happened in the past and which resulted in a new situation. When PUNDIT finds the temporal adverbial phrase ***after engagement***, it assumes that the ***engagement*** also has actual temporal reference. In such cases, the nominalization is processed using the meaning of the adverb and the tense of the main clause.

The second context in which a nominalization undergoes temporal analysis is where it occurs as the argument to a verb providing temporal information about situations. Such verbs are classified as aspectual. ***Occur*** is such a verb, so a sentence like ***failure occurred*** would be processed very similarly to a clause with the simple past tense of the related verb, i.e., ***something failed.***

Another type of verb whose nominalization arguments are temporally processed is a verb which itself denotes an actual situation that is semantically distinct from its arguments. For example, the sentence ***investigation revealed metal contamination in oil filter*** mentions three situations: the situation denoted by the matrix verb ***reveal***, and the two situations denoted by its arguments, ***investigation*** and ***contamination.*** If the situation denoted by ***reveal*** has actual temporal reference, then its arguments are presumed to as well.

3.2. Nominalization Mapping Rules

We will use the previous example, ***investigation revealed metal contamination in filter***, to illustrate the nominalization analysis algorithm. We will describe the ***contamination*** example first, since all of its roles are filled by syntactic constituents. The dotted line divides the algorithm in Figure 2 in the Appendix into the parts that are the same (above the line), and the parts that differ (below the line.)

DECOMPOSE VERB - ***Contaminate*** decomposes into a NON-ESSENTIAL **instrument** that contaminates an OBLIGATORY **location.**

```
contaminate <-
        contaminatedP(instrument(I),
        location(L))
```

FOR **instrument** role - In the example, ***metal*** is a noun modifier of ***contamination***, and **metal1** is selected as the filler of the **instrument** role.

FOR **theme** ROLE - The **theme** of a nominalization can be syntactically realized by an ***of pp*** or an ***in pp***. The role is filled with **filter1**, the referent of ***filter***.

At this point the temporal component is called for the nominalization ***metal contamination in oil filter*** with two inputs: the decomposition structure and the tense of the matrix verb, in this case the simple past. Because this predicate is stative, the representation of the **contamination** situation is a **state** predicate with the decomposition and a **period** time argument as well as the unique identifier S, (which will be eventually be instantiated by reference resolution as [**contaminate1**]):

```
state(S,
        contaminatedP
                (instrument(metal1),
                location(filter1)),
        (period(S))
```

In this context, the past tense indicates that at least one moment within the period of contamination precedes the time at which the report was filed.

CALL REFERENCE RESOLUTION FOR NOMINALIZATION - There are no previously mentioned ***contamination*** events, so a new referent, **contamination1** is created. There are no unfilled roles, so the analysis is completed.

3.3. Filling Essential Roles

The analysis of the other nominalization, ***investigation***, illustrates how essential roles are filled. The decomposition of ***investigate*** has two semantic roles, a NON-ESSENTIAL **agent** doing the investigation and an OBLIGATORY **theme** being investigated.[9]

```
investigate
        <- investigateP(agent(A),
                theme(T))
```

There are no syntactic constituents, so the mapping stage is skipped, and reference resolution is called for the nominalization. There are no previously mentioned investigative events in this example[10], so a new referent, **investigation1** is created. At this point, a second pass is made to attempt to fill any unfilled roles.

[9] In other domains, the **theme** can be essential, as in "I heard a noise. Let's investigate."

[10] If the example had been, ***A new engineer investigated the pump. The investigation occurred just before the complete breakdown.***, a previously mentioned event would have been found, and the **agent** and **theme** roles would have inherited the fillers **engineer1** and **pump1** from the reference to the previous event.

FOR **agent** ROLE - The role is NON-ESSENTIAL, so Case 4 applies, and it is left unfilled.

FOR **theme** ROLE - The selection restriction on the **theme** of an *investigation* is that it must be a **damaged** component or a **damage** causing event. All of the events and entities mentioned so far, the *sac* and the *pump*, the *failure of the sac* and the *shearing of the pump* satisfy this criteria. In this case, the item in focus, *the shearing of the pump*, would be selected [Dahl1986].
The final decomposition is:

investigateP(agent(A),theme(shear1))

4. Other Compounds

In addition to nominalizations, PUNDIT deals with three other types of noun-noun compounds. One is the category of nouns with arguments. These include *pressure* and *temperature*, for example. They are decomposed and have semantic roles like nominalizations; however, their treatment is different from that of nominalizations in that they do not undergo time analysis, since they do not describe temporal situations. As an example, the definition of *pressure*, **pressureP(theme(T),location(L))**, specifies **theme** and **location** as roles. The analysis of a noun phrase like *sac oil pressure* would fill in the **location** with the sac and the **theme** with the oil, resulting in the final representation, **pressureP(theme(oil1),location(sac1))**.
The syntactic mapping rules for the roles permit the theme to be filled in by either a noun modifier, such as *oil* in this case, or the object of an *of* prepositional phrase, as in *pressure of oil*. Similarly, the mapping rules for the location allow it to be filled in by either a noun modifier or by the object of an *in* prepositional phrase. Because of this flexibility, the noun phrases, *sac oil pressure, oil pressure in sac,* and **pressure of oil in sac,** all receive the same analysis.

The second class of compounds is that of nouns which do not have semantic roles. For these, a set of domain-specific semantic relationships between head nouns and noun modifiers has been developed. These include: **area of object,** for example, *blade tip,* **material-form,** such as *metal particles;* and **material-object,** such as *metal cylinder*. These relationships are assigned by examining the semantic properties of the nouns. The corresponding prepositional phrases, as in *tip of blade, particles of metal,* and *cylinder of metal,* have a similar analysis.

Finally, many noun-noun compounds are handled as idioms, in cases where there is no reason to analyze the semantics of their internal structure. Idioms in the CASREPS domain include *ships force, gear shaft,* and *connecting pin*. Our decision to treat these as idioms does not imply that we consider them unanalyzable, or noncompositional, but rather that, in this domain, there is no need to analyze them any further.

5. Previous Computational Treatments

Previous computational treatments of nominalizations differ in two ways from the current approach. In the first place, nominalizations have often been treated simply as one type of noun-noun compound. This viewpoint is adopted by [Finin1980,Leonard1984,Brachman(null)]. Certainly many nominalizations contain nominal premodifiers and hence, syntactically, are noun-noun compounds; however, this approach obscures the generalization that prepositional phrase modifiers in non-compound noun phrases often have the same semantic roles with respect to the head noun as noun modifiers. PUNDIT's analysis is aimed at a uniform treatment of the semantic similarity among expressions like *repair of engine, engine repair,* and *(someone) repaired engine* rather than the syntactic similarity of *engine repair, air pressure,* and *metal particles*. Of the analyses mentioned above, Brachman's analysis seems to be most similar to ours in that it provides an explicit link from the nominalization to the related verb to relate the roles of the noun to those of the verb. The second way in which our approach differs from previous approaches is that PUNDIT's analysis is driven by taking the semantic roles of the predicate and trying to fill them in any way it can. This means that PUNDIT knows when a role is not explicitly present, and consequently can call on the other mechanisms which we have described above to fill it in. Other approaches have tended to start by fitting the explicitly mentioned arguments into the role slots, thus they lack this flexibility.

6. Limitations

The current system has two main limitations. First, there is no attempt to build internal structure within a compound. Each nominal modifier is assumed to modify the head noun unless it is part of an idiom. For this reason, noun phrases like ***impellor blade tip erosion*** cannot be handled by our system in its current state because ***impellor blade tip*** forms a semantic unit and should be analyzed as a a single argument of ***erosion***. The second problem is related to the first. The system does not now keep track of the relative order of nominal modifiers. In this domain, this does not present serious problems, since there are no examples where a different order of modifiers would result in a different analysis. Generally, only one order is acceptable, as in ***sac oil contamination, *oil sac contamination***.

7. Conclusions

In this paper we have described a treatment of nominalizations in which the goal is to maximize the similarities between the processing of nominalizations and that of the clauses to which they are related. The semantic similarities between nominalizations and clauses are captured by making the semantic roles, semantic decompositions, and selectional restrictions on the roles the same for nominalizations and their related verbs. As a result, the same semantic representation is constructed for both structures. This similarity in representation in turn allows reference resolution to find referents for nominalizations which refer to events previously described in clauses. In addition, it allows the time component to integrate temporal relationships among events and situations described in clauses with those referred to by nominalizations.

On the other hand, where differences between nominalizations and clauses have a clear linguistic motivation, our treatment provides for differences in processing. PUNDIT recognizes that the semantic roles of nominalized verbs are expressed syntactically as modifiers of nouns rather than arguments of clauses by having a different set of syntactic mapping rules. It is also true in nominalizations that there are no syntactically obligatory arguments, so the analysis of a nominalization does not fail when there is an unfilled obligatory role, as is the case with clauses. Finally, the temporal analysis component is able to take into account the fact that nominalizations are untensed.

While there are many cases not yet covered by our system, in general, we believe this to be an approach to processing nominalizations which is both powerful and extensible, and which will provide a natural basis for further development.

Acknowledgements

We would like to thank Lynette Hirschman and Bonnie Webber for their helpful commments on this paper.

APPENDIX

```
DECOMPOSE VERB;

FOR EACH SEMANTIC ROLE

    CASE 1: IF THERE ARE SYNTACTIC CONSTITUENTS -
        PROPOSE SYNTACTIC CONSTITUENT FILLER
        & CALL REFERENCE RESOLUTION
        & TEST SELECTIONAL RESTRICTIONS

    CASE 2: IF ROLE IS OBLIGATORY AND SYNTACTICALLY UNFILLED -
        FAIL

    CASE 3: IF ROLE IS ESSENTIAL AND UNFILLED -
        CALL REFERENCE RESOLUTION TO HYPOTHESIZE A FILLER
        & TEST SELECTIONAL RESTRICTIONS

    CASE 4: IF ROLE IS NON-ESSENTIAL AND UNFILLED -
        LEAVE UNFILLED

CALL TEMPORAL ANALYSIS ON DECOMPOSITION
```

Figure 1. Clause Analysis Algorithm

```
DECOMPOSE NOMINALIZATION

FOR EACH SEMANTIC ROLE:

    IF THERE ARE SYNTACTIC CONSTITUENTS -
        PROPOSE SYNTACTIC CONSTITUENT FILLER
        & CALL REFERENCE RESOLUTION
        & TEST SELECTIONAL RESTRICTIONS

CALL TEMPORAL ANALYSIS ON DECOMPOSITION

CALL REFERENCE RESOLUTION FOR NOMINALIZATION NOUN PHRASE

FOR EACH SEMANTIC ROLE:

    IF ESSENTIAL ROLE AND UNFILLED
        CALL REFERENCE RESOLUTION TO HYPOTHESIZE A FILLER
        & TEST SELECTIONAL RESTRICTIONS
        ELSE LEAVE UNFILLED
```

Figure 2. Nominalization Analysis Algorithm

REFERENCES

[Brachman(null)]
Ronald J. Brachman, A Structural Paradigm for Representing Knowledge. In ***BBN Report No. 3605***, Bolt Beranek & Newman, Cambridge, Massachusetts.

[Dahl1986]
Deborah A. Dahl, Focusing and Reference Resolution in PUNDIT, Presented at AAAI, Philadelphia, PA, 1986.

[Dahl1987]
Deborah A. Dahl, Determiners, Entities, and Contexts, Presented at Tinlap-3, Las Cruces, New Mexico, January 7-9, 1987.

[Fillmore1968]
C. J. Fillmore, The Case for Case. In ***Universals in Linguistic Theory***, E. Bach and R. T. Harms (ed.), Holt, Rinehart, and Winston, New York, 1968.

[Finin1980]
Tim Finin, The Semantic Interpretation of Compound Nominals, PhD Thesis, University of Illinois at Urbana-Champaign, 1980.

[Hirschman1985]
L. Hirschman and K. Puder, Restriction Grammar: A Prolog Implementation. In ***Logic Programming and its Applications***, D.H.D. Warren and M. VanCaneghem (ed.), 1985.

[Hirschman1986]
L. Hirschman, Conjunction in Meta-Restriction Grammar. ***J. of Logic Programming***, 1986.

[Leonard1984]
Rosemary Leonard, ***The Interpretation of English Noun Sequences on the Computer***. North Holland, Amsterdam, 1984.

[Palmer1981]
Martha S. Palmer, A Case for Rule Driven Semantic Processing. ***Proc. of the 19th ACL Conference***, June, 1981.

[Palmer1985]
Martha S. Palmer, Driving Semantics for a Limited Domain, Ph.D. thesis, University of Edinburgh, 1985.

[Palmer1986]
Martha S. Palmer, Deborah A. Dahl, Rebecca J. [Passonneau] Schiffman, Lynette Hirschman, Marcia Linebarger, and John Dowding, Recovering Implicit Information, Presented at the 24th Annual Meeting of the Association for Computational Linguistics, Columbia University, New York, August 1986.

[Passonneau1986]
Rebecca J. Passonneau, A Computational Model of the Semantics of Tense and Aspect, Logic-Based Systems Technical Memo No. 43, Paoli Research Center, System Development Corporation, November, 1986.

[Passonneau1986]
Rebecca J. Passonneau, Designing Lexical Entries for a Limited Domain, Logic-Based Systems Technical Memo No. 42, Paoli Research Center, System Development Corporation, April, 1986.