Internal Use Only (非公開)

TR-I-0254

情報伝達行為解析システム CATE
Communicative Act Type Extractor

山岡孝行
Takayuki YAMAOKA

1992.3

概要

本報告では、情報伝達行為システム CATE(Communicative Act Type Extractor) について述べる。CATE は、発話文解析後の意味表現から、発話の意図を情報伝達行為タイプとして抽出する。

CATE への入力は、日本語解析システム NADINE の解析結果、素性構造意味表現であり、その中心となる推論メカニズムは、素性構造書き換え処理である。そして、抽出した情報伝達行為タイプに基づき、対話構造解析システムの入力となる発話の表現を構成し、出力するインタフェースを持つ。従って、このシステムにより、発話文字列からの対話構造構築、すなわち対話理解が、可能となる。

ATR 自動翻訳電話研究所
ATR Interpreting Telephony Research Laboratories
©(株) ATR 自動翻訳電話研究所 1992
©1992 by ATR Interpreting Telephony Research Laboratories

目次

1 概説 2

1.1 はじめに 2

1.2 何を求めるのか? 3

1.3 どの様に求めるのか? 4

1.4 何を使うのか? 6

2 情報伝達行為 8

2.1 協調的目標指向型対話における情報伝達行為とは 8

2.2 情報伝達行為タイプの設定 9

3 システム構成 13

3.1 データ構造 13

3.1.1 素性構造 13

3.1.2 発話の表現 15

3.2 システム構成 18

3.2.1 発話モダリティ解析 18

3.2.2 発話表現変換 18

3.3 処理のながれ 22

4 知識・データベース 23

4.1 語用論知識 23

4.1.1 文末表現規則 23

4.1.2 固定化表現規則 28

4.1.3 デフォールト規則 28

4.2 話題属性規則 29

4.2.1 情報伝達行為分析 29

4.2.2 概念の集合 29

4.2.3 話題属性規則詳説 31

4.3 命題格要素データベース 34

5 実験 36
5.1 入力 36
5.2 実験結果 37

6 おわりに 41

A 関数リファレンス 43
A.1 ファイル 43
A.2 主関数 44
A.2.1 大域変数 45
A.2.2 知識・データ・規則 46
A.2.3 発話モダリティ変換 52
A.2.4 発話表現変換 53
A.3 実験用インタフェース 57
A.3.1 入出力インタフェース 57
A.3.2 解析用関数 61
A.3.3 表示 62
A.3.4 バッチ処理 63
A.4 素性構造書換えシステムの拡張 64
A.4.1 外部関数 64
A.4.2 規則管理 66

B 実験システム利用例 69

C 語用論知識集 73
C.1 固定化表現規則 73
C.2 文末表現規則 79
C.3 デフォールト規則 86

D 命題格要素データベース 88

E 実験結果 (会話 A,B,1–10) 90

表目次

2.1 発話モダリティのタイプの分類 10
2.2 話題の属性タイプの分類 11
2.3 情報伝達行為のタイプと表層表現 12

3.1 CATE 固有の素性 14
3.2 発話の表現の構造 15
3.3 書き換え環境一覧 22

4.1 発話行為タイプと情報伝達行為タイプの対応 30
4.2 話題属性記述名 30

5.1 実験入力 37
5.2 利用規則・データ一覧 38
5.3 実験出力 39
5.4 実験結果の分類 40

図目次

3.1 命題内容リストの例 17
3.2 発話の表現のリストの例 17
3.3 CATE のシステム構成 18

4.1 語用論規則の記述例 25
4.2 固定化表現の規則記述例 35

第 1 章

概説

1.1 はじめに

高品質な機械翻訳には、文脈の理解が不可欠である。自然言語理解における文脈処理技術は、その問題領域において着実な進歩が遂げられている。しかし、文脈処理技術を機械翻訳システムへ応用するときに出現する様々な問題については、あまり目が向けられていないばかりか、入力文字列から一貫して文脈理解を行なうシステムはほとんどない。およそ、文脈処理システムは、その入力に関してはなんらかの仮定の上で獲得できるものとしている。そうでなければ、予め決められた入力（形式言語）に対する処理に限られる。さらに、音声言語処理システムにおいて、入力音声から対話の構造まで一貫して処理できるシステムは皆無であろう。

我々は、協調的目標指向型対話の文脈処理機構として、対話運用の知識と話題領域の知識を利用して、対話構造構築を行なう階層型プラン認識モデルを提案した。[1] また、異言語間の話し言葉翻訳システムとして、意図翻訳方式に基づく翻訳機構 NADINE が開発されている。階層型プラン認識モデルは、その入力として、発話行為の一種である情報伝達行為に基づく発話の表現を扱う。一方、NADINE は発話文の意味表現を出力するのみである。今、自動翻訳電話システムにおいて、音声入力から自動的に文脈処理を行なうためには、この発話文の意味表現から、上記の発話の表現を得る機構が必要となる。すなわち、一文章の記述から、その文章が表わしている抽象的な情報を得る計算手法を考えなければならない。

本報告では、階層型プラン認識モデルによる対話構造解析を自動翻訳電話システムにおける文脈処理機構として組み込む際の、文解析－対話構造解析インタフェースとして機能する情報伝達行為解析システム (CATE:Communicative Act Type Extractor) について述べる。CATE は日本語解析システム NADINE の出力である素性構造意味表現 (sem 素性) 記述 (prag 素性: 語用論素性を含む) のみから、発話の解釈として可能な限りの発話の表現を抽出する。対話の文脈における発話解釈の曖昧性の削減は、対話構造解析の仕事であり、CATE 内部ではその問題についての腐心は行なわ

ない。

1.2 何を求めるのか?

設計思想 情報伝達行為解析の目的は、発話の持つ情報伝達行為を求めることであるが、あくまでもこれは局所的な目的であり、その最終的目的は、対話構造解析、つまり文脈の理解にある。従って、CATE の設計思想では、常に対話構造解析との関連性を考慮する。具体的には、階層型プラン認識モデルにおける対話構造の最小単位である発話対構成の効率性・明瞭性の実現を念頭においたモデル化を行なう。

発話の表現 CATE の出力である発話の表現は、階層型プラン認識モデルの入力であり、以下の構造で表現される [2]:

> 発話の表現 ::= (情報伝達行為タイプ 話し手 聞き手 トピック 命題内容)

情報伝達行為タイプ 情報伝達行為 (communicative act) は、対象としている協調的な対話における情報交換の際に観察される発話行為の一種であり、そのタイプは、いわゆる発話のモダリティ部分と発話が伝達する情報内容 (命題内容) の話題の属性のコンビネーションから決定する。ここで話題の属性の情報を導入する理由は、同様の発話行為タイプを持つ発話であっても、その話題の属性の違いにより、特に応答発話の表層表現に違いが出るからである。応答発話表現に違いが出るということは、発話対構成において、曖昧な (選言的な) 記述を余儀なくされるということになる。発話対は対話構造解析における基本的構成素なので、その記述が曖昧であれば、対話構造の解釈において曖昧さが出ることになるし、また対話構造解析の処理効率を落とすことになる。

例えば、「お願いします」という表現は、依頼 (request) の発話行為を具体化した表現であるが、「お名前をお願いします」のようにその話題の属性が、具体的値を持つオブジェクトであれば、行為依頼に対する一般的応答と考えられる「分かりました」や「そう致します」といった拘束 (promise) で表現することが困難となる。

もちろん「お名前をお願いします」は行為依頼であり、受諾や拒否の意味の発話を行なうことはできる。しかしながら、ここでは次の 3 つの観点からこれを行為依頼と解析し出力することに大きな利点を見いださない: 1) 電話対話での情報交換、2) 協調的な情報交換、3) 目標達成を目的とする情報交換において、この様な情報交換行為自体の行為依頼・拘束に関するやり取りを規定する事は冗長である。同様に、情報交換では「教える」や「聞く」といった述語を含む要求発話の情報伝達行為は、いわゆる間接発話行為を求める方が現実的である。

命題内容 命題内容は、発話により伝達される情報内容であり、その具体的内容は、対話全体の目標達成に寄与していることが期待される。

命題内容は、発話の主述語とその格要素で表現する。例えば、「登録用紙を送って下さい」の命題内容は;

((predicate 送る) (agent 誰か) (recipient 他の誰か) (object 登録用紙))

のように表現できる。(この情報伝達行為タイプは行為依頼である) これは、日本語解析機構の意味表現から比較的簡単に求めることができる。

トピック トピックは、発話表現中で「は」相当格によってマークされた概念である。[1]

トピックは、対話構造解析において話題領域に関する制約として利用できる。具体的には、ドメインプラン探索の探索空間の制限を可能とする [2]。また、情報伝達行為解析においては、話題の属性の決定に貢献する。[2]例えば、「締切は、3月15日です。」の「締切」は具体的値を持つオブジェクトについての話題であるが、「支払は、銀行振込です。」の「支払」はそれよりも行為自体を話題としている。従って、後者は、単なる情報提示と言うよりも、より行為依頼に近い情報伝達行為を持つと認定できる。

トピックの導出及びトピックと話題の属性の厳密な相関モデルは、今後の課題である。

話し手・聞き手 話し手及び聞き手は、言語外情報であり、システム全体の中で何らかの形で得られると仮定する。(例えば、入力チャネルの違いによるなど。)

1.3 どの様に求めるのか?

基本的アプローチ CATEにおける発話の表現の導出の基本的アプローチは、前節で述べてきた点を鑑みると以下のようになる:

- 発話対に基づく情報伝達行為タイプの設定
- 情報伝達行為導出における2つのフェイズ
  - 発話のモダリティ解析

[1]階層型プラン認識モデルでは、「は」相当格に該当するものがない場合、発話の先頭の格要素を取るという定義が為されているが、現在の意味表現からこれを行なうことは困難である。

[2]現在のCATEでは、このトピックとさきに述べた話題の属性における話題の間に強い相関を持たせていない。(強い相関を持たせることは危険である。)

– 話題の属性による解析

- 命題内容と情報伝達行為タイプの統合的・相互作用的処理

素性構造書換えによる発話モダリティ解析　発話のモダリティ部分の解析処理では、素性構造書換え処理を行なう。発話文の記述から発話行為に関する情報を解析する手法として、久米ら [3] は、文の発語内行為 (IFT:Illocutionary Force Type) を抽出する手法を提案している。そこでは、素性構造書換えシステム [4] の基で、言語使用における社会常識的な知識を素性構造書換え規則として記述し、入力素性構造意味表現のモダリティ部分の記述を、特定の IFT 記述に変換する。CATE では、基本的にこの手法を採用する。ただし、素性構造書換えシステムには、長谷川システム [5] を利用する。このシステムでは、書換え環境による規則適用の制御が可能であり、上で示したような複数フェイズの知識 (書換え規則) を比較的容易に記述することができる。[3]

発話モダリティ解析では、可能な限り情報伝達行為タイプを導出する。つまり、モダリティ部分でそのタイプが決定できるような意味記述は、極力このフェイズで変換されるように書き換え規則の設定を行なう。この部分だけで決定できないものについては、IFT のみを求め、次のフェイズにその結果を渡す。

話題の属性による解析　話題の属性による情報伝達行為の決定は、特定環境下での IF-THEN 規則に基づく。例えば、「お名前をお願いします」の例では、行為依頼という環境の下で、話題である「名前」の属性を求め、その属性に関する規則が存在すれば、そこに記述された情報伝達行為タイプ (例えば, 質問 (question)) が新たな解となる。

命題内容の変換　命題内容は、発話文の主述語とその格要素を抽出することによって得る。

主述語の決定は、基本的には、モダリティ解析で解析されなかった部分で最も外にある主辞 (動詞) を選ぶ。このような要素がなければ、とりあえず変項を与えておく。但し、モダリティ解析により、特定の命題内容が決定できる時は、それを採用する。(例えば、対話開始・終了の挨拶などにはこれが適用できる。) [4]主述語が決定できれば、素性構造意味表現にしたがって、その格要素の内容を取り出す。

現在のところ CATE では、簡単化のため、必須格相当の格要素のみを命題内容の

---

[3]なお、1.4節で述べるように、最新の CATE では、命題部分及びそれに付随する話題の属性による情報伝達行為の決定は、LISP code になっている。

[4]Searle によれば、挨拶 (greet) に命題内容は存在しない。ここでは、特定の発話行為に命題内容が存在するか否かという議論は抜きにして、単に変項を与えることによる対話構造解析における解釈の自由度を制限するために、その情報内容が明かであろうものについては、特定の命題内容を記号として与える。

構成素として扱う。また、各々の格要素の内容は、オブジェクトに関する書換えにより、その主辞構造のみを取り出す。従って、任意格や格要素の修飾語に重要な意味が含まれている発話については弱い。(例えば、「会議は八月二十二日から二十五日まで京都国際会議場で開催されます」は、

((predicate 開催する) (agent ?) (object 会議))

となってしまう。)

トピックの抽出 トピックは、存在すれば、素性構造意味表現の topic 素性の値とする。なお、「お名前をお願いします」のような特定の表現では、名詞句そのものが命題内容になることがあるが、そのような場合は、その命題名詞句をトピックとして、命題は変項とする。

1.4 何を使うのか?

CATE が利用する知識・データは以下のようになる。

発話モダリティに関する語用論知識 語用論知識は、発話モダリティと情報伝達行為タイプを対応づける知識である。発話表現のモダリティ部分は、比較的反射的な表現が行なわれていると考えられる。つまり、命題内容のように、構成要素の配列や格助詞の利用といった構成的な知識が用いられず、話し手が持っているある意図に対応した表現を直観的に引き出しているということになる。もちろん文法的可否により判断される部分は多くあろうが、日本語の文末表現のように補助動詞・助動詞・接続助詞さらには活用変化といった複雑な構造と組合せを、真面目に解析することは効率的ではない [6]。従って、CATE では、このような対応知識を網羅的に書き下す という方針をとる。

語用論知識は、書換え規則のフォーマリズムで記述する。従って、情報伝達行為タイプの特定は、それに対応する表現パターンと入力との書換えシステムで定義されたパターンマッチングによりなされる。この処理は、決定的な処理ではない。従って、複数のパターンにマッチすれば、潜在的に複数の情報伝達行為を持っていると解釈し、そのすべてを出力する。[5]

話題の属性による変換規則 話題の属性による規則は、IF-THEN の形で記述する。前節までで述べたように、CATE での情報伝達行為抽出では、話題の属性を考慮に入れるが、

[5]現在の長谷川システムでは、マッチングの競合において、その規則記述中の素性パスの長い規則が優先されるようである。それはそれで、よい。

1. 対話に現われる述語や概念をすべて書換え規則として設定するのが面倒であること、
2. たとえそれらを設定したとしても多量の規則の元で動作するため処理効率が低下すること、
3. 命題の格要素に関するデータベースや概念のネットワークが対話構造解析機構と共有できること、

といった理由から、書換え規則による記述は行なわない。(以下の2つのデータベースについてもこれらの理由から特有のものを利用するに至った。) もちろん、特定の環境下での規則適用という考えは、継承する。すなわち、この規則は独立して成立するものではなく、発話モダリティ解析の結果に従った記述になる。

基本的に、語や語の意味に関する分析は、文解析の仕事である。解析や変換部の辞書から自動的に命題内容や話題の属性に関する書換え規則を設定する道具を作ることができれば統一性の高いシステム実現が可能である。しかし、ここではそこまで行なわない。[6]

命題格要素のデータベース 命題格要素データは、単純に述語とその格要素のラベルのリストで表現する。これらは、命題内容の表現を行なう際に必要な構成素を取り出してくる時に利用する。[7]

概念シソーラス 概念シソーラスは、述語 (動詞) や格要素・話題 (名詞句) となる概念の階層関係を表した知識である。これは、ネットワークで表現する [7]。

概念の階層関係は、具体的には、命題内容や話題の属性を決定する時に役立つ。例えば、1.2節で触れた情報交換自体の行為を意味する述語、「教える」などは“体面動作”という概念の下位概念であり、この概念に属する命題内容は、その目的格の属性によって、情報伝達行為タイプに変化を与える。例えば、「お名前を教えて下さい」は質問になるが、「英語を教えて下さい」などは (行為) 依頼になる。

概念シソーラスのより詳しい説明と利用は、[10] に譲る。

これらの知識・データは、ユーザや対象問題によりカスタマイズできることが望ましい。つまり、CATE を生かすも殺すもこれらの知識やデータの記述にかかってくる。各々の知識・データについて、現在どのようなものを準備しているか、またどのようにカスタマイズするかは、第4章で詳説する。

---

[6] 旧 CATE では、これらはすべて書換え規則として記述されていた。

[7] 実は、このような規定を行なう最大の理由は、対話構造解析のプラン認識におけるマッチングメカニズム (単一化) がグラフ構造を扱えないことである (term unify である)。つまり、同じ命題内容は、同じ構造として入力されてくることを期待している。今後改良すべきである。

第 2 章

情報伝達行為

CATE の目的は、一発話の意味表現から情報伝達行為を抽出することである。本章では、その出力目的の中心となる情報伝達行為の概念について再考し、ATR サンプル対話のための情報伝達行為と発話の表現のモデルを設定する。

2.1 協調的目標指向型対話における情報伝達行為とは

協調的目標指向型対話は、ある目標の達成に向けて、対話参加者が協調的に情報交換を行なう対話である。

情報伝達行為は、協調的目標指向型対話において話し手が何らかの情報を聞き手に与える発話を行なう時に観察される発話行為である。情報伝達行為は、話し手の信念に基づく意図の遂行の形態である。したがって、それは話し手の信念の内容と聞き手の信念に与える影響に関する話し手の信念の内容からで記述することができる。(情報伝達行為のスキーマ記述については、[2] を参照のこと。)

情報伝達行為のクラス

情報伝達行為は、その聞き手に与える影響 (効果:effect) により大きく 2 つに区別することができる。『要求 (Demand)』と『応答 (Response)』である。『要求』の情報伝達行為とは、effect が聞き手の与えられた命題内容に関する発話を行なうこと (動機:motivation) を促すものである。ここで、協調的に行なわれる情報交換の仮定において、『要求』の情報伝達行為の受け手 (すなわち聞き手) には、応答の義務が生じる。すなわち、受け手が『要求』により受けた motivation に対して、何らかの形で情報を与えることにより、情報交換が成立する。この『応答の義務』は (協調的) 会話の協働原則により保証される。『要求』の情報伝達行為に対し、それが伝達している命題内容について、純粋に言明する発話の情報伝達行為を、『応答』の情報伝達行為とする。すなわち、この定義においては、ある『要求』の effect がそのまま motiva-

tion となる受け手の発話がその『要求』に対する『応答』になる。

例えば、質問 (要求の発話) 「お名前をお願いします」に対する「はい」は応答ではない。なぜなら、前の発話の持つ effect

```
(BELIEVE( hearer WANT( speaker KNOW(speaker ?name))))
```

に対して、この発話は何ら貢献していないと考えられるからである。(敢えていえば、"I believe." の意味での発話かも知れないが、情報交換は成立しない。) この質問に対する『応答』の発話とは、KNOW(speaker ?name) を成就する発話でなければならない。

すべての発話が、『要求』・『応答』でとらえられるとは限らない。情報交換の成立の確認の意味で発せられる発話の情報伝達行為を、『確認 (Acknowledgement)』index 確認の情報伝達行為とする。その他、上記 3 つに含まれない発話の情報伝達行為を『陳述 (inform)』の情報伝達行為とする。

このようにここでは、情報伝達行為のクラスとして『要求』『応答』『確認』『陳述』の 4 つを考える。

発話対

協調的目標指向型対話では、『要求』と『応答』の発話により、ある命題内容についての情報交換が成立することになる。この 2 つの発話の作る構造を『発話対 (utterance pair)』と呼ぶ。発話対によって成立する情報交換を、対話中の最小の談話セグメント[1]としてとらえ、それらの上位目標への関係により表される構造が、対話構造である。従って、我々の文脈理解は、発話の情報伝達行為と命題内容から、対話構造を構築することが目標となる。

本報告の 1 つの目標は、対象対話における情報伝達行為の分類にある。そして、その分類に際しての基本的立場は、『情報交換を成就する発話対の構成』である。よって、分類された情報伝達行為は、『要求』から『応答』への写像関係 (必ずしも 1 対 1 でない) を持つものでなければならない。

以下では、以上の考えに基づいて、ATR サンプル対話を対象に情報伝達行為の分類を行なう。なお、より広範な対話を対象とした形式的考察としては、参考文献 [8][6] を参考されたい。

2.2 情報伝達行為タイプの設定

情報伝達行為タイプは、発話モダリティのタイプと話題の属性で表現する。

[1] この談話セグメント (発話クラスタ [8]) は、『確認』の発話を伴う場合がある。

表 2.1: 発話モダリティのタイプの分類

INFORM	: 事柄の事実について述べる。	(平叙文)
ASK	: 未知の事柄について質問する。	(疑問詞を伴う疑問文)
CONFIRM	: 事柄の真偽について質問する。	(yes/no 疑問文)
REQUEST	: 相手に行動を依頼する。	(命令文)
OFFER	: 自分の行為を拘束する。	(動作動詞を述語とした平叙文)
GREETING	: あいさつ。	(おもに固定化された表現)
() 内は代表的な文形式を表している。		
INFORM の下位属性のタイプ		
AFFIRMATIVE	: 命題内容を肯定する。	
NEGATIVE	: 命題内容を否定する。	
ACCEPT	: 行為を受け入れる。	
REJECT	: 行為を拒む。	
CONFIRMATION	:: 情報交換成立の確認。	

発話モダリティのタイプ 表 2.1に、発話モダリティのタイプを示す。さらに、INFORM の下位に属すると考えられるが、対話での機能上特定の情報伝達行為有すると考えられるものを区別する。

[2] では、これらの他に、意味的に対話の流れに関与した発話モダリティを設定している。(例えば、INTRODUCE, NOTICE, REQUIRE) これらもまた、INFORM の下位に属する (REQUIRE は REQUEST の下位) と考えられるが、表層上の形態から区別して規定することは困難である。(なお、INTRODUCE は、この後の WANT と INFORM による組合せが近いかも知れない。しかし、INFORM-WANT が全て INTRODUCE の意味を持つかということは甚だ疑問である。) それよりも、これらは対話構造に照らし合わせることによりその役割 (情報伝達行為) が決定されるものと考え、ここでは考慮しないことにした。[2]一方、表に示した INFORM の下位タイプは、対話において比較的一定の表現形式が用いられるものである。(固定化表現のところ参照)

話題の属性タイプ 表 2.2に、話題の属性タイプの分類を示す。

なお、WANT は STATEMENT の下位であるが、先に述べたように対話中で、通常の状態提示とはことなる、重要な振舞いを見せるため区別した。また、GREETING は第 1 章で述べたように、特定の発話行為 (上の発話モダリティのタイプ GREETING) で成立する話題としてここにあげた。これは、具体値として、OPEN(ing), CLOSE(ing) を設定する。

[2] このような対話のストラテジックな流れを統御する知識が、コミュニケーションプランであろう。

表 2.2: 話題の属性タイプの分類

ACTION	:	ある行為・行動についての話題
VALUE	:	ある事物の属性についての話題
STATEMENT	:	ある事物の状態についての話題
(WANT)	:	話し手の欲求状態について
(GREETING)	:	対話の開始・終了

情報伝達行為タイプ 表 2.1と表 2.2のような分類から、表 2.3のような情報伝達行為のタイプが設定できる。また、これらの情報伝達行為に対応した代表的な表層表現を併せて示す。

固定化表現の設定 問い合わせ電話対話などにおいては、慣用的に発話される固定化された表現が良く現れる。これらを『固定化表現 (frozen term)』と呼び、効率的な解析の手助けとする。(表 2.3右欄で "*" のついた表現)

固定化表現は、表で示した情報伝達行為による発話をなす時、対話中に断片的な一発話として現れることが多い。従って、発話モダリティの書換え規則として記述する際の負担も大きくない。

表 2.3: 情報伝達行為のタイプと表層表現

1.Demand Class:		
ASK-ACTION	:	「*ACT* は *WH* ですか。」
CONFIRM-ACTION	:	「*ACT*（する／できるの）ですか。」
REQUEST-ACTION	:	「*ACT* して下さい。」
	:	「*ACT* していただけますか。」
	:	「*ACT* していただきたいのですが。」
OFFER-ACTION	:	「*ACT* します。」
	:	「*ACT* させていただきます。」
ASK-VALUE	:	「*OBJ* は *WH* ですか。」
	:	「*OBJ* をお願いします。」
	:	「*OBJ* を教えて／聞かせて下さい。」
	:	「*OBJ* を聞くことができますか。」
	:	「*OBJ* をお聞きしたいのですが。」
CONFIRM-VALUE	:	「*OBJ* は *VAL* ですか。」
	:	「*OBJ* は *VAL* ですね。」
ASK-STATEMENT	:	「*STA* は *WH* ですか。」
CONFIRM-STATEMENT	:	「*STA* ですか。」「*STA* ですね。」
GREETING-OPEN	:	「もしもし。」*
GREETING-CLOSE	:	「失礼します。」*「さようなら。」*
	:	「ありがとうございました。」
2.Response Class:		
INFORM-ACTION	:	「*ACT* して下さい。」
	:	「*ACT* しなくてはいけません。」
	:	「*ACT* です。」
INFORM-VALUE	:	「*OBJ* は *VAL* です。」
	:	「*VAL* を／が *STA* ます／です。」
INFORM-STATEMENT	:	「*STA* です（が）。」
INFORM-WANT	:	「*STA* したい（のですが）。」
AFFIRMATIVE	:	「はい。」*「そうです。」*
NEGATIVE	:	「いいえ。」*
	:	「（否定表現）。」
ACCEPT-ACTION	:	「わかりました。」*
	:	「*ACT* は問題ありません。」
REJECT-ACTION	:	「*ACT* できません。」
ACCEPT-OFFER	:	「ありがとうございます。」*
	:	「（よろしく）*ACT* 願います。」
REJECT-OFFER	:	「（いいえ）結構です。」
GREETING-OPEN	:	「はい。」*
GREETING-CLOSE	:	「失礼します。」*「さようなら。」*
	:	「どういたしまして。」*
3.Confirm Class:		
CONFIRMATION	:	「わかりました。」*「そうですか。」*

表層表現内のイタリック文字は変項を表し、それぞれ *ACT* は ACTION に関する内容、*OBJ* は OBJECT に関する内容、*STA* は STATEMENT に関する内容、*WH* は疑問詞を表す。また、／は選言を表し、() 内は付いても付かなくても良い表現である。

第 3 章

システム構成

本章では、CATE のデータ構造と、システム構成とその中の各モジュールの概要および基本的な動作について述べる。

3.1 データ構造

CATE は入力として素性構造をとり、出力として発話の表現を出す、いわば変換システムである。ここでは、これら 2 つの基本的なデータの内容の取り扱いについて述べる。

なお、知識ベース・データベース内のデータ記述については、第 4章を参照されたい。

3.1.1 素性構造

CATE が扱う素性構造は、長谷川システムで定義された素性構造である。素性構造の内部構造については、参考文献 [5] を参照。

素性構造中の任意の素性パタンは、そのタイプにより以下の意味を持たせる:

`:atomic`	何らかの概念を示すシンボル (symbol)
`:leaf`	任意の構造と単一化可能な変項 (variable: 次節参照)
`:variable`	`:leaf` に同じ
`:complex`	上記 2 項の組合せによる構造体

CATE の処理では、入力として、文解析 NADINE から出力される素性構造の意味表現 (sem 素性) および語用論素性 (prag 素性) のみを扱う。

表 3.1: CATE 固有の素性

素性名	データタイプ	内容
CAT	:atomic	情報伝達行為タイプ／発話行為のタイプ
TPC	:atomic/variable	トピック
PRP	–	命題内容の素性構造
VAL	:atomic	(複合) 概念の主辞
MOD	:atomic	発話モダリティのエキストラ情報 (例えば、否定 (nagate)・欲求 (want) 等)

中間データのための素性の拡張

第 3.2節に示すように、CATE の処理には 2 つのフェイズがあり、その間で素性構造の受渡しが行なわれる。この時点での素性構造 (以後中間素性構造と呼ぶ) は、入力意味表現を抽象化したものになっている。従って、ここで、そのデータを表現する際に新しい素性を導入する。

CATE で新しく導入する素性の素性名は、3 文字のアルファベットで表す。(例外もある) 表 3.1に、その中で重要なものの一覧を示す。なお、表中データタイプのキーワードは、長谷川素性構造のタイプに対応する。また、 variable は :leaf/:variable を、– はいずれのタイプでも良いことを表す。

- CAT には、第 2.2節で定義した情報伝達行為タイプが入る。文解析の失敗などで、入力素性構造の発話モダリティ解析ができない時 (すなわち発話行為のタイプが求められない時) は、:unknown にする。[1]

- TPC には、トピックの定義 (1.2節) を満足するものがあれば、その主辞の概念を表すシンボルが入る。でなければ、変項である。

- PRP には、発話の命題内容部分の素性構造が入る。これはしばしば、名詞句を表現しているものになることがある。そのような時は :atomic タイプになる。命題内容部分は存在しない時は、変項である。

- VAL は、主に名詞句の主辞を表すことに用いる。素性構造意味表現における名詞句の表現は、様々な情報を包含し表す形になっているので、複雑である [9]。とりあえず、対話構造解析への情報としては、名詞句の中心的な意味のみで十分であると考え、その主辞の情報のみを受け渡すことにする。

  この素性には、具体的には、“登録用紙 -1” といった概念 (を表現している記号: RELN 素性, 関係名) が入り、素性構造意味表現でいう ENTITY の値ではない。

[1]:unknown は 変項ではない。

表 3.2: 発話の表現の構造

要素	データタイプ	内容
1 情報伝達行為タイプ (cat)	symbol	第 2.2節で定義したもの
2 話し手 (sp)	symbol	話し手を表す任意のシンボル
3 聞き手 (hr)	symbol	聞き手を表す任意のシンボル
4 トピック (tpc)	symbol/variable	第 1.2節参照
5 命題内容 (prp)	list/variable	下記参照
(6 発話文字列 (str)	string	発話文の文字列

もちろん、発話の厳密な解釈を行なうには、これだけの情報では物足りない場合の方が多い。付加語や修飾語の情報の取り扱いは、今後の課題である。

- MOD には、いわゆる発話のボイス・ムードといった情報を入れる。これらは、通常発話モダリティ解析のための情報として利用されるべきであるが、現在の CATE が、モダリティ部分を完全に解析できる能力を備えていないために、このような素性を持っておき、後々利用できるようにする。この素性は以下のような場合、中間素性構造に付加される:
  - 発話モダリティ解析において、そのムードに顕著な特徴がある時;
    例えば、2.2節で触れた INFORM-STATMENT の欲求:WANT や、否定:NEGATE など。
  - 発話モダリティ解析の対象に入らず、かつ命題内容部分でないボイスなど:
    これらは、3.2で触れるが、命題内容変換の前処理の際に抽出する。例えば、可能:POSSIBLE, 受益:PASSIVE など。

これらの素性構造は、入力素性構造の発話表現への変換への中間構造として、情報の受渡しをするのみならず、話題の属性による解析や (我々の) 発話行為の解釈モデルに含まれない情報 (表の MOD 素性など) を利用した情報伝達行為タイプの絞り込みに対しての、情報提供を行なっている。逆にいえば、発話表現変換部 (3.2節) では、単にデータの変換を行なうに留まらず、このような情報を利用してより的確な情報伝達行為・発話の表現を生成していることになる。

3.1.2 発話の表現

第 1.2節に示したように、CATE の出力は発話の表現である。一つの発話の表現は、一本のリストで表す。このリストは、発話の一つの解釈に相当する。従って、一つの発話・一つの素性構造意味表現から、ただ一つのリストが導かれるわけではない。発話の表現のリストの構造を図 3.2に示す。(発話文字列は、オプショナルである。)

発話の表現のリストには、情報伝達行為タイプ・話し手／聞き手・概念を表すシンボルの他に、以下のようなデータが入りえる:

- 変項
  変項 (variable) は、対話構造解析において任意の構造と単一化可能なデータ構造である。CATE では、入力素性構造において以下の条件を満たすものを変項として扱う:
  - 素性値が :leaf/:variable タイプの素性構造
  - 疑問詞相当句が主辞である格要素

  以下具体的変項の記述は、接頭辞 "?" を付して表す。

- 命題内容リスト

  命題内容は、一本のリストで表す。命題内容のリストは、命題格要素データの記述にしたがった長さ (1 以上) を持つ。命題内容の先頭要素は、その述語を表すシンボルである。その他の要素 (cdr) は、その格要素の並びである。格要素の並びの順序は、命題格要素データにより規定されている。(命題格要素データの構造については、第 4.3節参照) 任意の格要素は変項になり得る。

  命題内容リストは、以下のような場合、それ自体変項になり得る:
  - 発話モダリティ解析の結果、命題内容に相当する素性構造が残らなかった時
    (多くは、「はい」「そうです」等の固定化表現)
  - 話題の属性による解析により、従来の命題内容が消滅した時
    (例えば、「参加方法について教えて下さい」など。これは、「参加方法」がトピックとして残るのみである。)

図 3.1に「登録用紙をお送りします」の命題内容部分の素性構造意味表現、命題内容リスト表現、および「送る」の命題格要素データの記述例を示す。

また、図 3.2に「登録用紙を送ってください」の素性構造意味表現と発話の表現のリストの記述例を示す。(この例では、情報伝達行為に多義が出る。)

```
;;; 素性構造意味表現 ;;;
[[RELN  送る-1]
 [AGEN []]
 [RECP []]
 [ASPT  UNRL]
 [OBJE [[PARM !X12[]]
        [RESTR [[RELN  登録用紙-1]
                [ENTITY !X12]]]]]]

;;; 命題内容リスト ;;;
(送る-1 ?agen ?recp  登録用紙-1)

;;; 命題格要素データ ;;;
(送る-1  AGEN  RECP  OBJE)
```

図 3.1: 命題内容リストの例

```
;;; 素性構造意味表現 ;;;
[[SEM [[RELN  下さい-REQUEST]
       [AGEN [[LABEL  *SPEAKER*]]]
       [RECP !X02[[LABEL  *HEARER*]]]
       [ASPT  UNRL]
       [OBJE [[RELN  送る-1]
              [AGEN !X02]
              [RECP []]
              [OBJE [[PARM !X01[]]
                     [RESTR [[RELN  登録用紙-1]
                             [ENTITY !X01]]]]]]]]]]

;;; 発話の表現 ;;;
(REQUEST-ACTION *SPEAKER* *HEARER* ?TPC (送る-1 *HEARER* ?RECP  登録用紙-1))
(INFORM-ACTION  *SPEAKER* *HEARER* ?TPC (送る-1 *HEARER* ?RECP  登録用紙-1))
```

図 3.2: 発話の表現のリストの例

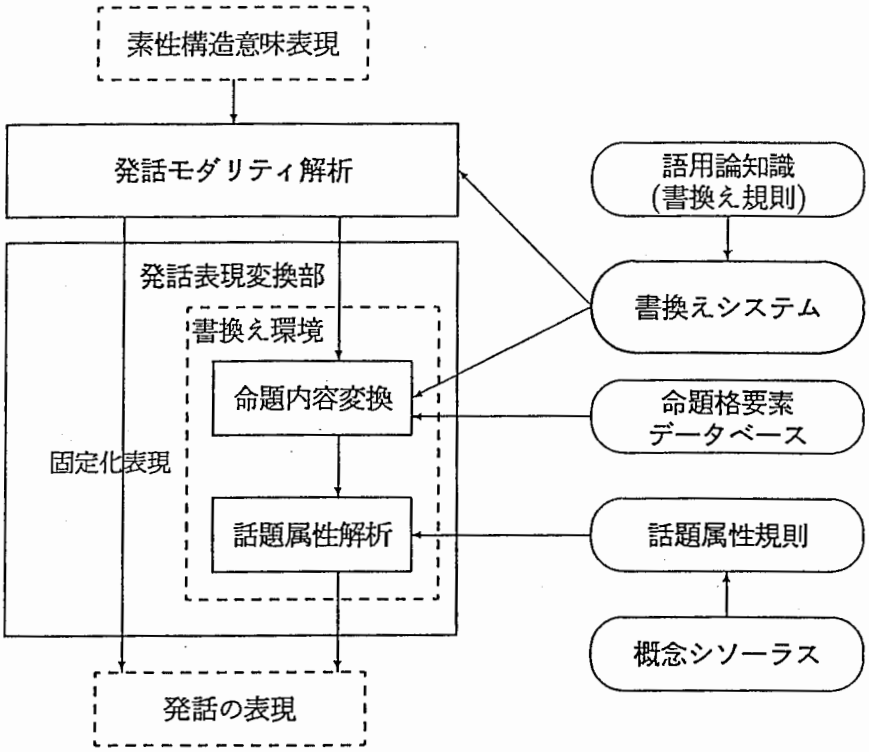

図 3.3: CATE のシステム構成

3.2 システム構成

図 3.3に、CATE の基本的なモジュール構成を示す。

CATE の処理は、大まかに、発話モダリティ解析と発話表現変換の 2 つのフェイズに分かれる:

3.2.1 発話モダリティ解析

入力素性構造意味表現を、書換え規則の適用により書換え、発話行為のタイプあるいは情報伝達行為タイプを含む素性構造 (中間素性構造) を出力する。

3.2.2 発話表現変換

このフェイズでは、中間素性構造を発話の表現のデータ構造 (リスト) に変換する。その過程において、以下のことを順次行ない、発話の表現のリストを構成し、出力する:

1. 命題内容変換
   中間素性構造の命題内容部分 (PRP 素性) を、命題格要素データを参照して、命題内容リストに変換する。

   前段の発話モダリティ解析では、発話の意図に関する部分しか変換されず、特定の発話行為における命題内容を除いては、素性構造で受け渡される。現在、対話構造解析機構が素性構造そのものを扱うことができないため、この命題内容部分の素性構造を先に述べた命題内容リストに変換せねばならない。

   この処理では、命題内容素性構造の主辞 (述語節の関係名) を主述語として、命題格要素データに記述された格要素だけを、対話構造解析への入力として抽出する。結果として、命題内容素性構造が構造を持つものであれば、命題内容リストが返る。

2. 話題属性解析
   話題属性による再解析を行なう。ここでは、発話モダリティ解析により 2.2節で定義した情報伝達行為タイプが出なかったもの、すなわち発話行為タイプの抽出に留まったものが、話題属性規則により再解析される。

   話題属性規則は、発話行為タイプに特有のものであり、その中で可能な情報伝達行為タイプを決定する規則である。つまり発話行為タイプを環境と見て、話題となっている部分をキーに書き換える操作に等しい。話題属性規則内部での話題の属性の判断は、概念シソーラスを参照して行なう。

   ここで、発話行為タイプとは、久米ら CATEKume89 の表層発話行為タイプ (SIFT) に近いものであり、以下のようなものがある (詳細な表現の例については、語用論規則 (付録 C) 参照):

タイプ名	文型 (表層表現の例)
INFORM	平叙文 (... です, ... ます)
CONFIRM	yes/no 疑問文 (... か?, ... ね?)
ASK	(疑問詞のある) 疑問文 (... は何ですか?)
REQUEST	命令文, 依頼文 (... をお願いします, ... して下さい)

   また、話題の属性には、2.2節で設定した 3 つ (細かく分ければ 5 つ) のものがある。そして、発話中で話題となり得るエンティティについての階層関係の知識が、概念シソーラスに記述されている。

   従って、話題属性規則には、上の発話行為タイプに対応する 4 つの規則がある。発話行為タイプと可能な情報伝達行為タイプの対応の詳細と、その処理規則については、4.2節で詳説する。

その他の処理

CATE の中心となる処理は以上で述べたが、その他に以下のような処理を行なっている。これらは、現状の環境の制約から来るものが多く、将来は統一的な形で改善・消滅していくことが望まれる:

1. **命題内容素性構造の前処理**
   語用論知識 (発話モダリティ書き換え規則) 適用後、命題内容に付加している情報を変換する。知識適用後の命題内容部分 PRP 素性に対する、素性構造書き換え処理である。

   3.1節 MOD 素性のところで触れたように、現在の CATE の語用論知識では、すべてのモダリティ部分を処理できない。従って、命題内容部分に、助動詞・補助動詞などの情報が残っている場合がある。このよう名名だいないよう部分から、純粋な命題内容部分を取り出すための処理を前処理として行なう。ここでは、その対象によって、MOD 素性を追加する操作を行なうことがある。

   現状では、「...でしょう」の う -GUESS や NEGATE、 *-SHOULD 等の助動詞に関する 12 規則がある。

2. **オブジェクトの処理**
   主に名詞句の素性構造意味表現から、その主辞の概念 (関係名) を取り出すための書き換え操作である。基本的には、以下の 2 つの書き換え規則による書換えである:

```
(rws:defrwschema2 gobj-0 gobj main
  "on <RELN> :unspecified
   in :Phase :object :Type :general
      in= [[RELN   ?val]
           [ENTITY ?x]
           ?rest]
      out= ?val
   end")
(rws:defrwschema2 gobj-1 gobj main
  "on <RESTR RELN> :unspecified
   in :Phase :object :Type :general
      in= [[PARM  ?ent]
           [RESTR ?obj]
           ?rest]
      if ?ent is []
        then => ?obj
```

```
        out= ?obj
         else => ?ent
        out= ?ent
        endif
    end")
```

なお、この他に、関係節や並立助詞・形式名詞などに対応する特定の規則を設定している。

また、疑問詞に対しては、上の環境と別環境に設定している。これは、疑問詞に関する書き換えは、発話モダリティ解析の際の条件として利用しているからである。従って、上のタイミングで疑問詞の書き換えは行なわない。

3. トピック抽出
トピックは、基本的に、入力素性構造の prag 素性の topic 素性を参照する。この処理は、素性構造書換えであり、発話モダリティ解析の後処理として行なう。この書き換え規則は単に、 topic 素性のオブジェクトの内容を取り出すのみである。

4. 話し手・聞き手情報
対話構造解析において、話し手・聞き手は、区別された agent として各々ユニークに表されなければならない。なぜなら、典型的な発話対は、違った話者による情報交換であるからである。話し手・聞き手の情報は、システム外から与えられることが期待されるが、現状の NADINE では、その区別が明示的ではない。

現状の CATE では、話し手・聞き手は、 SP1/SP2 という 2 つのシンボルを与えている。そして、サンプル会話について、その中の発話 ID と話し手の対応テーブルを準備しておき、それを参照することにより、話し手のユニークなシンボルを獲得する。また、聞き手は、得られた話し手の逆のシンボルを与える。従って、現状の CATE 話し手・聞き手情報を扱う際には、以下の注意が必要である:

(a) 処理対象の発話 ID は指定されたものでなければならない

この情報の獲得は、発話表現変換の始めに行なう。

5. 書き換え前処理・後処理
現在の長谷川システムでは、その規則定義中の素性値 (素性パスの値) に "*" を含むシンボルを書くことができない。従って、書き換え前処理として、 "*" を含むような素性構造 (実際には *SPEAKER*, *HEARER*) をそれを含まないユニークな素性構造に変換している。 (情報内容に変化はない。)

また、書き換え後処理として、複数命題を含む一つの素性構造を、その命題内容の数に振り分ける処理を行なっている。

(これらの詳細については、付録参照)

3.3 処理のながれ

現状のCATEでは、以上説明した処理モジュールの処理順序をまとめると、以下のようになっている:

1. 発話モダリティ解析部(素性構造書き換え処理)
   (a) 書き換え前処理
   (b) 発話モダリティ解析(語用論知識適用)
   (c) 命題内容素性構造前処理
   (d) オブジェクト変換処理
   (e) トピック抽出
   (f) 書き換え後処理
2. 発話表現変換部
   (a) 話し手・聞き手情報獲得
   (b) 命題内容変換
   (c) 話題属性解析

発話モダリティ解析部は、素性構造書き換え処理であり、その書き換え環境を表3.3のように設定して、規則適用の効率化と規則管理のモジュラリティ向上を狙っている:

各処理部の詳細と実装については、付録A: 関数リファレンス参照。

表 3.3: 書き換え環境一覧

処理	環境名 :Phase	環境名 :Type	規則数
1. 書き換え前処理	:preedit	:speaker	2
2. 発話モダリティ解析	:intention	:general, :default	49
3. 命題内容素性構造前処理	:proposition	:ignore	12
4. オブジェクト変換処理	:object	:general, :wh-phrase	28
5. トピック抽出	:topic	:generral	2
6. 書き換え後処理	:postedit	:topic	4

第 4 章

知識・データベース

4.1 語用論知識

発話のモダリティ表現部分と情報伝達行為タイプ(あるいは表層発話行為タイプ)を対応付ける知識である語用論知識は、長谷川システムの書換え規則のフォーマリズム [5] で記述する。

現在の CATE では、実際の素性構造意味表現と対話の流れの対応分析から 49 の規則を設定している。(巻末付録 C にその一覧を掲げる[1])

4.1.1 文末表現規則

文末表現に対する規則の基本的な記述戦略は:

- 発話モダリティ部分の書換え環境は、`'((:Phase :intention))` とする。
- 素性パスには、できる限り深い位置にありかつ特定的な素性値を記述する。
- 対象入力素性構造パタン (in=) には、入力素性構造意味表現の発話モダリティ部分を記述する。[2]
- 適用結果素性構造 (out=) には:
  - CAT 素性に、情報伝達行為タイプ (あるいは表層発話行為タイプ) の明示的な値 (:atomic) を,

[1]規則定義マクロ rws:defrwschema2 については、付録 A.4: 『素性構造書換えシステムの拡張』を参照

[2]例外として、"疑問詞 + 教える (対面動作動詞)" は、ASK を表す発話モダリティとして考えている。

- TPC 素性に、入力の prag 素性構造を (これは後のトピック抽出で書き換えられる),
- PRP 素性に、発話モダリティの OBJE 素性構造を

を記述するようにする。

発話モダリティを表す文末表現 “ ください ” と “(疑問詞) ですか?” に対する規則の例を図 4.1 に示す。

図 4.1の規則例 1 を以下の入力素性構造 (発話文:「登録用紙で手続きをして下さい」) に適用すれば[3];

```
[[SEM [[RELN  ください-REQUEST]
       [ASPT  UNRL]
       [AGEN !X04[[LABEL  *SPEAKER*]]]
       [RECP !X03[[LABEL  *HEARER*]]]
       [OBJE [[RELN  する-1]
              [AGEN !X03]
              [OBJE [[PARM !X06[]]
                     [RESTR [[RELN  手続き-1]
                             [ENTITY !X06]]]]]
              [INST [[PARM !X05[]]
                     [RESTR [[RELN  登録用紙-1]
                             [ENTITY !X05]]]]]]]]
 [PRAG [[RESTR [[IN [[FIRST [[RELN  RESPECT]
                             [AGEN !X04]
                             [RECP !X03]]]
                     [REST [[FIRST [[RELN  POLITE]
                                    [AGEN !X04]
                                    [RECP !X03]]]
                            [REST !X02[]]]]]]
                [OUT !X02]]]
        [TOPIC [[IN []]
                [OUT []]]]
        [PRSP-TERMS [[IN []]
                     [OUT []]]]
        [SPEAKER !X04]
        [HEARER !X03]
        [ASPE [[IN !X01[]]
               [OUT !X01]]]]]]
```

以下の出力素性構造が得られる;

[3]この中の *SPEAKER*, *HEARER* は、書き換え前処理により、それぞれ SPEAKER, HEARER に書き換わっている。

```
;;; 規則例 1 ;;;
(rws:defrwschema2 req-2-1 req kudasai
  "on <SEM RELN>   ください-REQUEST              ;; 素性パスと素性値
   in :Phase :intention :Type :general           ;; 環境設定
      in= [[SEM [[RELN  ください-REQUEST]        ;;
                [AGEN [[LABEL  SPEAKER]]]        ;; 行為者は話し手
                [RECP [[LABEL  HEARER]]]         ;; 受益者は聞き手
                [OBJE ?prp]                      ;; 命題内容
                ?rest]]
           [PRAG  ?prag]
           ?rests]
      out= [[CAT REQUEST]                         ;; 表層発話行為タイプ
            [PRP ?prp]                            ;; 命題内容
            [TPC ?prag]]                          ;; トピック
   end")

;;; 規則例 2 ;;;
(rws:defrwschema2 ask-1 ask INFORMREF
  "on <SEM OBJE RELN> INFORMREF                   ;; 特定の素性値で深いものは
                                                  ;; INFORMREF である
   in :Phase :intention :Type :general            ;; 環境設定
      in= [[SEM [[RELN  S-REQUEST]
                [AGEN !SP[[LABEL  SPEAKER]]]
                [RECP !HR[[LABEL  HEARER]]]
                [OBJE [[RELN  INFORMREF]
                       [AGEN !HR]
                       [RECP !SP]
                       [OBJE [[PARM ?wh]          ;; 疑問詞であること
                              [RESTR ?prp]        ;; 命題内容
                              ?rest2]]
                       ?rest3]]
                ?rest4]]
           [PRAG ?prag]
           ?rests]
      ==> ?wh  with :Phase :object  :Type :wh-phrase
                                                  ;; 上の ?wh が真に疑問詞で
                                                  ;; あることを確認する
      out= [[CAT ASK]                             ;; 表層発話行為タイプ
    [WH  ?wh]                              ;; 疑問詞の具体的値(エキストラ情報)
            [TPC ?prag]                           ;; トピック
            [PRP ?prp]]                           ;; 命題内容
   end")
```

図 4.1: 語用論規則の記述例

```
[[CAT REQUEST]
 [PRP [[AGEN !X4[[LABEL HEARER]]]
       [INST [[PARM !X1[]]
              [RESTR [[ENTITY !X1]
                      [RELN 登録用紙-1]]]]]
       [OBJE [[PARM !X2[]]
              [RESTR [[ENTITY !X2]
                      [RELN 手続き-1]]]]]
       [RELN する-1]]]
 [TPC [[ASPE [[IN !X3[]]
              [OUT !X3]]]
       [HEARER !X4]
       [PRSP-TERMS [[IN []]
                    [OUT []]]]
       [RESTR [[IN [[FIRST [[AGEN !X6[[LABEL SPEAKER]]]
                            [RECP !X4]
                            [RELN RESPECT]]]
                    [REST [[FIRST [[AGEN !X6]
                                   [RECP !X4]
                                   [RELN POLITE]]]
                           [REST !X5[]]]]]]
              [OUT !X5]]]
       [SPEAKER !X6]
       [TOPIC [[IN []]
              [OUT []]]]]]]
```

ちなみに、3章で述べたその他の処理を行なった後の中間素性構造出力は以下のようになる;

```
[[CAT REQUEST]
 [PRP [[AGEN HEARER]
       [INST 登録用紙-1]
       [OBJE 手続き-1]
       [RELN する-1]]]
 [TPC []]]
```

同様に、図の規則例2の発話文:「どのようなご用件でしょうか?」への適用例を以下に示す;

```
;;; 入力素性構造 ;;;
[[SEM [[RELN  S-REQUEST]
       [AGEN !X02[[LABEL  *SPEAKER*]]]
       [RECP !X03[[LABEL  *HEARER*]]]
       [OBJE [[RELN  INFORMREF]
              [AGEN !X03]
```

```
            [RECP !X02]
            [OBJE [[PARM !X05[[RELN  どのような-1]
                              [ARG-1 [[PARM !X04[]]
                                      [RESTR [[RELN  用件-1]
                                              [ENTITY !X04]]]]]]]
                  [RESTR [[RELN  う-GUESS]
                          [ASPT  STAT]
                          [EXPR !X02]
                          [OBJE [[RELN  だ-IDENTICAL]
                                 [OBJE !X06[]]
                                 [IDEN !X05]]]]]]]]]]
 [PRAG [[RESTR [[IN [[FIRST [[RELN  POLITE]
                            [AGEN !X02]
                            [RECP !X03]]]
                  [REST !X01[]]]]
               [OUT !X01]]]
       [TOPIC [[IN []]
               [OUT []]]]
       [PRSP-TERMS [[IN []]
                    [OUT []]]]
       [SPEAKER !X02]
       [HEARER !X03]
       [ASPE [[IN []]
              [OUT []]]]]]]

;;; 適用結果 ;;;
 [[CAT ASK]
  [PRP [[ASPT STAT]
        [EXPR !X3[[LABEL SPEAKER]]]
        [OBJE [[IDEN !X4 用件-1]
               [OBJE []]
               [RELN だ-IDENTICAL]]]
        [RELN う-GUESS]]]                ;; 命題内容前処理により落ちる
  [TPC [[ASPE [[IN []]
               [OUT []]]]
        [HEARER !X2[[LABEL HEARER]]]
        [PRSP-TERMS [[IN []]
                     [OUT []]]]
        [RESTR [[IN [[FIRST [[AGEN !X3]
                             [RECP !X2]
                             [RELN POLITE]]]
                     [REST !X1[]]]]
                [OUT !X1]]]
        [SPEAKER !X3]
```

```
        [TOPIC [[IN []]
                [OUT []]]]]]
  [WH !X4]]

;;; 中間素性構造 ;;;
 [[CAT ASK]
  [PRP [[IDEN !X1 用件-1]
        [OBJE []]
        [RELN だ-IDENTICAL]]]
  [TPC []]
  [WH !X1]]
```

4.1.2 固定化表現規則

また、固定化表現に対する規則については:

- 固定化表現は、入力素性構造そのもの(全体)が対象であり、発話のモダリティであると考える。
- 命題内容(トピック)が特定的なものについては、適用結果素性構造にその値を記述する(現在は、GREETING のみ)。

固定化表現「もしもし」の規則記述例を図4.2に示す。

4.1.3 デフォールト規則

発話文の中には、その素性構造意味表現に発話モダリティ部分を持たないような発話がある。例えば、以下のような素性構造は、sem 素性の直下に命題内容部分が来ている;

```
;;; それでは登録用紙をお送り致します
[[SEM [[RELN  送る-1]
       [ASPT  UNRL]
       [AGEN !X04[[LABEL  *SPEAKER*]]]
       [RECP !X05[[LABEL  *HEARER*]]]
       [OBJE [[PARM !X03[]]
              [RESTR [[RELN  登録用紙-1]
                      [ENTITY !X03]]]]]
       [INST [[PARM !X02[]]
              [RESTR [[RELN  それ-1]
                      [ENTITY !X02]]]]]]]
```

このような例には、上のような規則を設定することができないので、情報伝達行為タイプがでないことになる。従って、このような入力に対する救済措置が必要となる。そこで、(sem 素性に) 発話モダリティ部分のない素性構造に対する規則として、デフォールト規則を設ける。現在、基本的デフォールト規則には 2 つのものがある (詳細は付録 C.3参照);

1. 命題内容の行為者 AGEN 素性が話し手かつアスペクトが UNRL なものは、行為拘束 (OFFER-ACTION) である。

2. 1 以外は、単に陳述的 INFORM である。

4.2 話題属性規則

話題属性規則は、発話の話題の属性により、より詳細な情報伝達行為タイプ導出を行なうための、知識記述である。発話モダリティ解析の結果として、情報伝達行為タイプの他に 4 つの (表層) 発話行為タイプが現れる可能性があることは、3.2.2節で述べた。最終的には、これらも情報伝達行為タイプに変換する必要がある。ここでは、これら 4 つの表層発話行為タイプに対応するための規則の内容について説明する。

4.2.1 情報伝達行為分析

まず、対象対話中の発話の、表層発話行為タイプと 2.2節の情報伝達行為タイプとの対応について分析を行なった。その対応の一部を、表 4.1に示した。

4.2.2 概念の集合

発話の話題の属性の値に関する情報は、名詞句概念ネットワーク [7] の考えを述語その他の概念にも拡張した知識 (概念シソーラス) から得る (概念シソーラスの詳細については、[10] 参照)。各発話の話題属性の認定は、ここから得られた集合を参照することにより行なう。

ここでは、話題とトピックは異なる概念として捉えている (1.2節)。ここでいう話題には、命題内容の構成要素全般が含まれる。つまり、焦点があたっているオブジェクトのみならず、述語の概念 (どのようなことを使用としているのか)、行為者・対象の概念なども参照する。表 4.2に、名詞句・疑問詞・述語別の話題の属性の記述を示す。これらの具体的要素は、概念ネットワークインタフェース ([10] 参照) を利用することによって、得ることができる。[5]

[5]現在は、処理効率化のため、これらの集合はリストとして各々の集合を示す大域変数に保持している。これらの変数の設定は、システム初期化時に行なっている。従って、概念シソーラスを変更した際

表 4.1: 発話行為タイプと情報伝達行為タイプの対応

発話行為タイプ	情報伝達行為タイプ	表現の例
ASK	ASK-VALUE	参加料はいくらですか?
	ASK-STATEMENT	どのようなご用件ですか?
4	ASK-ACTION	どうすればよろしいでしょうか?
CONFIRM	CONFIRM-VALUE	そちらは会議事務局ですか?
	CONFIRM-STATEMENT	登録用紙はお持ちですか?
	CONFIRM-ACTION	参加なさいますか?
	ASK-VALUE	お名前をお伺いできますか?
	ASK-ACTION	参加方法をお伺いできますか?*
REQUEST	REQUEST/INFORM-ACTION	用紙を送って下さい
	ASK-VALUE	お名前をお願いします
	ASK-ACTION	参加方法を教えて頂けますか?*
	ACCEPT-OFFER	(よろしく) お願いします
INFORM	INFORM-VALUE	名前は鈴木真弓です
	INFORM-STATEMENT	京都プリンスホテルが近いんですが
	INFORM-ACTION	こちらで審査を行ないます
	ASK-VALUE	参加料について教えて頂きたいのですが
	ASK-ACTION	参加方法について教えて頂きたいのですが
	(INFORM-WANT)	会議に申し込みたいのですが

表現に * を付したものは、サンプル対話には出てこない

表 4.2: 話題属性記述名

略号	: 内容	例
NP-value	: 特定の値を持つオブジェクトを表す名詞句概念の集合	名前 -1, 参加料 -1
NP-behavior	: 動作・行動を表す名詞句概念の集合	参加 -1, 交通手段 -1
WH-value	: 特定の値を尋ねる疑問詞の集合	いくら -1, いつ -1
WH-behavior	: 動作・行動の内容・方法を尋ねる疑問詞の集合	どのように -1, どう -1
V-action	: 行為動詞概念の集合	する -1, 送る -1
V-statement	: 状態を表す述語概念の集合	近い -1, 持つ -1
V-interaction	: 対面動作を表す述語概念の集合	教える -1, 聞く -3
IS-verb	: いわゆるダ文を表す述語概念の集合	だ -IDENTICAL

4.2.3 話題属性規則詳説

以上のような分析から現状の CATE に設定した環境 (表層発話行為タイプ) 別の話題属性規則の処理の詳細を以下で説明する:

パラメータ記述 話題属性規則処理の対象 (入力) は中間素性構造と命題内容変換時までにでき上がっている一時的な発話の表現である。以下はそれら入力データに含まれている情報であり、説明では以下の略号を利用する (これらを入力から獲得する時のパラメータ設定については、付録 A.2.2節: 話題属性関連参照);

略号:	内容
prp :	命題内容リスト
pred:	命題内容の主述語
mod :	中間素性構造の MOD 素性値 (発話ムード: エキストラ情報)
wh :	中間素性構造の WH 素性値 (疑問詞対象概念: エキストラ情報)
aspt:	命題内容素性構造の ASPT 素性値 (命題アスペクト)

また、概念シソーラスから求められた特定の話題の属性の概念の集合については、4.2.2節の記述を利用する。

ASK 規則 発話モダリティ解析結果表層発話行為タイプが ASK の時;

1. pred が IS-verb であり、かつ wh が WH-value(あるいは NP-value) であれば、
   → ASK-VALUE,
   (例): “参加料はいくらですか?”
2. pred が V-action であり、かつ wh が WH-behavior(あるいは NP-behavior) であり、かつ mod が義務的 (must, should) であれば、
   → ASK-ACTION,
   (例): “参加料はどのようにお支払いしたらよろしいのでしょうか?”
3. それ以外の時 (このステップは複数出力が可能);
   (a) wh が WH-value(あるいは NP-value) であれば ASK-VALUE を与える,
   (例): “誰と出席なさいますか?”
   (b) wh が WH-behavior(あるいは NP-behavior) であれば ASK-ACTION を与える,
   (例): “それはどのような方法でしょうか?”
   (c) pred が V-statement であれば ASK-STATEMENT を与える.
   (例): “どちらが近いですか?”
4. 以上の処理で情報伝達行為タイプがでないものには、デフォールトとして、ASK-VALUE, ASK-ACTION, ASK-STATEMENT を与える。

は、システムの初期かが必要となる。

CONFIRM 規則 発話モダリティ解析結果表層発話行為タイプが CONFIRM の時;

1. pred が IS-verb であり、その値 (IDEN 格) が変項でなければ、
   → CONFIRM-VALUE,
   (例): “そちらは会議事務局ですか?”
   (a) さらに、その値が NP-behavior であれば、
   → CONFIRM-ACTION,
   (例): “支払いは銀行振り込みですか?”
2. mod が可能的 (POSSIBLE) である時 (複数出力可能);
   (a) pred が V-interaction であり、かつその対象格 (目的格) の値が変項でないとき;
   i. その値が NP-value であれば、
   → ASK-VALUE,
   (例): “参加料について教えて頂けますか?”
   ii. その値が NP-behavior であれば、
   → ASK-ACTION,
   (例): “支払い方法について教えて頂けますか?”
   (b) pred が V-behavior であれば、
   → CONFIRM-ACTION,
   (例): “まだ参加できますか?”
3. それ以外の時 (複数出力可能);
   (a) aspt が UNRL かつ pred が V-action であれば、 CONFIRM-ACTION を与える,
   (例): “参加なさいますか?”
   (b) aspt が UNRL 以外であるか、または pred が V-statement であれば、 CONFIRM-STATEMENT を与える,
   (例): “登録用紙はお持ちですか?”
4. 以上の処理で情報伝達行為タイプがでないものには、デフォールトとして、CONFIRM-VALUE, CONFIRM-ACTION, CONFIRM-STATEMENT を与える。

REQUEST 規則 発話モダリティ解析結果表層発話行為タイプが REQUEST の時;

1. prp が変項であれば、
   → REQUEST-ACTION/ACCEPT-OFFER,
   (例): “(よろしく) お願いします”
2. prp が名詞句命題内容の時 (複数出力可能) [6];

[6] この時 (名詞句命題内容の時) は、当該名詞句をトピックに移して命題内容は変更にする。

(a) その値 (prp) が NP-value であれば、
→ ASK-VALUE,
(例): “ ご住所をお願いします ”

(b) その値が NP-behavior であれば、
→ REQUEST-ACTION,
(例): “ 支払をお願いします ”

3. pred が V-interaction であり、かつその対象格 (目的格) の値が変項でないとき (複数出力可能);

(a) その値 (prp) が NP-value であれば、
→ ASK-VALUE,
(例): “ ご住所を教えて下さい ”

(b) その値が NP-behavior であれば、
→ ASK-ACTION,
(例): “ 支払方法を教えて下さい ”

4. 以上の処理で情報伝達行為タイプがでないものには、デフォールトとして、REQUEST-ACTION, INFORM-ACTION を与える。

INFORM 規則 発話モダリティ解析結果表層発話行為タイプが INFORM の時;

1. pred が IS-verb であり、その値 (IDEN 格) が変項でなければ、
→ INFORM-VALUE,
(例): “ こちらは会議事務局です ”

(a) さらに、その値が NP-behavior であれば、
→ INFORM-ACTION,
(例): “ 支払いは銀行振り込みです ”

2. mod が欲求 (WANT) であれば (複数出力可能)、
→ INFORM-WANT,
(例): “ 会議に参加したいのですが ”
さらにこの場合;

(a) pred が V-interaction であり、かつその対象格 (目的格) の値が変項でないとき (複数出力可能)

i. その値が NP-value であれば、
→ ASK-VALUE,
(例): “ ご住所を伺いたいのですが ”

ii. その値が NP-behavior であれば、
→ ASK-ACTION,
(例): “ 支払方法について教えて欲しいのですが ”

3. mod が可能的であり、かつ pred が V-action であれば、
   または、 aspt が UNRL であり、かつ pred が V-action であれば、
   → INFORM-ACTION,
   (例): “参加できます”, “こちらで審査を行ないます”
4. それ以外の時 (複数出力可能);
   (a) mod が否定的 (NEGATE) であれば、
   NEGATIVE を与える,
   (例): “持っていません”
   (b) aspt が UNRL 以外であるか、または pred が V-statement であれば、 INFORM-STATEMENT を与える,
   (例): “持っています”
5. 以上の処理で情報伝達行為タイプがでないものには、デフォールトとして、INFORM-STATEMENT, INFORM-VALUE, INFORM-ACTION を与える。

4.3 命題格要素データベース

命題格要素データは、命題内容の主述語とその格要素の関係を定義したリストデータである。この定義に従い、命題内容部分の素性構造が命題内容リストに変換される。変換された命題内容は、定義にしたがった (格の位置が一定となっている) リスト構造となり、プラン認識機構で単一化可能なデータ構造である命題内容リストとして出力される。

このデータベースは、本来不要である。プラン認識機構の単一化を柔軟にすれば、解決する問題である。

しかし、現在の文脈処理機構では必要なので、付録Dにサンプル会話をカバーするデータの一覧を掲げておく。これらは、 ATR 日本語基本辞書 [11] 記述中の必須格を参考に定義した。

```
;;; 規則例 3 ;;;
(rws:defrwschema2 gop-1 gop moshimoshi
  "on <SEM RELN> もしもし-OPEN_DIALOGUE
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  もしもし-OPEN_DIALOGUE]
                [AGEN [[LABEL  SPEAKER]]]
                [RECP [[LABEL  HEARER]]]
                ?rest]]
           ?prag]
      out= [[CAT GREETING-OPEN]
            [TPC opening]                    ;; 特定のトピックと
            [PRP [[RELN open-dialogue]]]]    ;; 命題内容を設定
   end")
```

図 4.2: 固定化表現の規則記述例

第 5 章

実験

以上述べてきたシステムにより、ATR サンプル会話 A,B,1-10 の情報伝達行為解析実験を行なった。本章では、その実験の概要と実験結果について述べる。

5.1 入力

実験の入力はサンプル会話内の 261 発話の日本語解析結果である。注意すべき点は;

- 日本語解析結果は、省略補完を行なったものである。
- 日本語解析結果は、基本的に 1991.11 月現在の解析実験結果[1]を利用し、解析できていないものについては、再度解析した結果を利用した。
- 解析結果が出なかった発話文は 1 つだけ; “ 発表を希望されるのでしたら三月二十日までに要約を提出して下さい ”

表 5.1に一覧を示す。

知識・規則・データ

実験に利用した知識・規則・データは、第 4章で説明したものである。その数量の内訳を表 5.2に示す。[2]

[1]ディレクトリ: as04:/as23/jikken-kekka-11.11/nadine.log.1-10

[2]概念ネットワークのノード総数は、名詞句・動詞概念の他に余分なものも含んでいる。

表 5.1: 実験入力

会話	発話数	延べ解析結果
合計	261	645
A	19	39
B	18	30
1	20	34
2	21	40
3	16	66
4	22	28
5	22	52
6	23	68
7	21	50
8	25	58
9	17	78
10	38	102

5.2 実験結果

表 5.3に情報伝達行為解析後の、中間素性構造数・最終発話表現の数を示す。また、表 5.4に実験後の情報伝達行為を分類したものを掲げる。[3]

実験結果は、おおむね期待された情報伝達行為を含む発話表現を出力していた。実験結果の詳細は、付録 Eに掲げるので参照されたい。

なお、仕様マシンは、 SparcStationII, 主記憶 64Mb, スワップ 200Mb (as21) である。

[3] この分類の数と、先の出力の数の差は、複数命題を考慮していないことによる。つまり、先の出力では、複数命題の発話表現は一つとして数えたが、分類の方では各々の情報伝達行為をカウントに入れた。

表 5.2: 利用規則・データ一覧

項目	数
書換え規則: 92 規則	
語用論知識	49
(固定化表現)	22
(文末表現)	27
その他	48
(前処理)	2
(後処理)	4
(命題前処理)	12
(オブジェクト)	28
(トピック)	2
命題格要素データ: 69 述語	
概念ネットワーク: 472 ノード	
NP-value	93
NP-behavior	25
WH-value	8
WH-behavior	4
V-action	58
V-statement	25
V-interaction	9
IS-verb	4
話題属性規則: 4 規則 (4.2.3節)	

表 5.3: 実験出力

会話	発話数	解析結果	中間	(平均)	最終	(平均)
合計	261	645	375	1.44	413	1.58
A	19	39	23	1.21	24	1.26
B	18	30	23	1.28	25	1.39
1	20	34	31	1.55	33	1.65
2	21	40	28	1.33	35	1.67
3	16	66	25	1.56	24	1.50
4	22	28	28	1.27	31	1.41
5	22	52	33	1.50	35	1.59
6	23	68	37	1.61	37	1.61
7	21	50	26	1.24	30	1.43
8	25	58	38	1.52	42	1.68
9	17	78	23	1.35	25	1.47
10	38	102	77	2.03	72	1.89

表 5.4: 実験結果の分類

CAT	Number
TOTAL	431
ASK-ACTION	12
ASK-STATEMENT	4
ASK-VALUE	36
CONFIRM-ACTION	6
CONFIRM-STATEMENT	10
CONFIRM-VALUE	11
REQUEST-ACTION	31
OFFER-ACTION	18
INFORM-ACTION	41
INFORM-STATEMENT	35
INFORM-VALUE	59
INFORM-WANT	17
AFFIRMATIVE	31
NEGATIVE	12
ACCEPT-ACTION	19
REJECT-ACTION	4
ACCEPT-OFFER	11
CONFIRMATION	23
GREETING-OPEN	29
GREETING-CLOSE	21
UNKNOWN	1

第 6 章

おわりに

本報告では、情報伝達行為解析システム (CATE) について説明した。

まず、対象対話の分析に基づいた情報伝達行為モデルの設定を行ない、処理の目標を定めた。そして、素性構造書換えによる発話モダリティ表現部分の解析手法と知識を提示した。また、情報伝達行為解析における話題の属性情報の利用を提案し、その処理手法と規則記述について説明した。

CATE を利用することにより、日本語解析機構 NADINE と対話構造解析機構 LAYLA の結合が可能となり、入力発話から文脈処理を自動的に行なうことができるようになった。また、CATE は、様々な知識・データ・規則のカスタマイズを行なうことにより、他の対話ドメインにも利用していくことができるであろう。

単に文字列のみから、発話の意図を解析することには限界がある。また、発話表現は、それ自体膨大なバリエーションを持つものである。今後は、より多くの発話文を分析・解析実験することによって、より精密で高品質の知識を蓄えていくことが望まれる。

最後に、文脈処理を統語・意味処理、さらには音声処理、と真の意味で統合することができれば、CATE はいらなくなるであろう。

参考文献

[1] 飯田 仁, 有田 英一：“4 階層プラン認識モデルを使った対話の理解”, 情処学論, **31**, 6, pp.810-821(1990)

[2] 有田 英一, 飯田 仁: “対話翻訳のための階層型プラン認識モデル”, ATR テクニカルレポート, TR-I-0067, (1989)

[3] 久米 雅子, 小暮 潔: “発話意図の翻訳のための発話行為推論部”, 第 38 回情処学全大, (1989)

[4] Zajac, R.: “A Transfer Model Using a Typed Feature Structure Rewriting System with Inheritance”, 27th ACL, (1989)

[5] 長谷川敏郎: “素性構造書換えシステムマニュアル (改定版)”, ATR テクニカルレポート, TR-I-0187, (1990)

[6] 山岡 孝行, 久世谷 正則, 山田 恭寛: “協調的な目標指向型対話における文形式と発話者の意図との対応 – 文末形式規則による情報伝達行為判定システム –”, ATR テクニカルレポート, TR-I-0252, (1992)

[7] 有田 英一, 山岡 孝行, 飯田 仁：“電話対話における次発話内の名詞句表現の予測”, 情処学自然言語処理技法, NL-81-13 (1991)

[8] 定延 利之, 山岡 孝行, 飯田 仁: “協調的な目標指向型対話における文形式と発話者の意図との対応 – 文形式を重視した情報伝達行為の分類 –”, ATR テクニカルレポート, TR-I-0220, (1991)

[9] 久米 雅子, 永田 昌明: “日本語解析文法の意味表現について”, ATR テクニカルレポート, TR-I-0155, (1990)

[10] 山岡 孝行, 西村 仁志, 飯田 仁: “階層型プラン認識システム LAYLA(LAyered pLAn recognition system)”, ATR テクニカルレポート, TR-I-0255, (1992)

[11] 永田 昌明, 衛藤 純司: “日本語基本辞書の概要”, ATR テクニカルレポート, TR-I-0250, (1992)

付録 A

関数リファレンス

ここでは、CATE の主な内部関数のいくつかを紹介する。

A.1 ファイル

現在の、CATE のソースの所在は以下の通りである:

内容	ファイル名
プログラムソース	``as21:/home/yamaoka/System/III/CATE/''
システムロード用	load.lisp
発話モダリティ変換部	cate-main.lisp
発話表現変換部	output.lisp
話題属性パラメータ設定	parameters.lisp
素性構造操作用	fs.lisp
実験用インタフェース	examine.lisp
RWS 拡張プログラム	rws-if.lisp
語用論知識 (書換え規則)	``as21:/home/yamaoka/System/III/Rules/''
固定化表現	frozen.lisp
文末表現	intention.lisp
命題前処理	proposition-ignore.lisp
トピック抽出	topic.lisp
オブジェクト変換	object-general.lisp
オブジェクト疑問詞	object-wh.lisp
前処理	preedit.lisp
後処理	postedit.lisp
話題属性規則	output-rule.lisp
知識・データベース	``as21:/home/yamaoka/System/III/Data/''
命題格要素データ	case-grammar.lisp
概念ネットワーク	concept-node.lisp
話題属性・概念集合設定	parameter-set.lisp
話し手対応テーブル	speaker.list

A.2 主関数

---

load-cate [*Function*]

引数: なし
リターン値:

CATE をロードする. (`load.lisp` をロードすればこれは自動的に行なわれる。) (`load.lisp`)

---

compile-cate [*Function*]

引数: なし
リターン値:

CATE のソースファイルをコンパイルする. 同時にそのコンパイルコードをロードする. (`load.lisp`)

---

initialize-cate [*Function*]

引数: なし
リターン値: T

CATE の初期化. 以下のことを行なう:

1. 書換え規則のロード (renew-rws-rules)
2. 概念ネットワークの設定 (load-concept)
3. 命題格要素データの設定 (load-proposition-grammar)
4. 規則パラメータの設定 (parameter-set)
5. 話題の属性規則のロード (load-output-rules)

(`load.lisp`)

A.2.1 大域変数

---

program-path [*Variable*]

タイプ: string
初期値: "/home/yamaoka/System/III/CATE/"

プログラムソース・バイナリファイルのパス名. (`load.lisp`)

---

data-path [*Variable*]

タイプ: string
初期値: "/home/yamaoka/System/III/Data/"

データファイルのパス名. (`load.lisp`)

---

file-list [*Variable*]

タイプ: list of strings
初期値: (list "parameters" "cate-main" "fs" "output" "examine")

プログラムソースファイル名のリスト. (`load.lisp`)

---

rws-load-filename [*Variable*]

タイプ: string/pathname
初期値: "/home/yamaoka/System/III/CATE/rws-load-as21"

(長谷川) 素性構造書換えシステムロード用のファイル名. (`load.lisp`)

---

rws-patch-filename [*Variable*]

タイプ: string/pathname
初期値: (merge-pathnames rws-id *program-path*)

素性構造書換え拡張システム (RWS2) ロード用のファイル名. (`load.lisp`)

A.2.2 知識・データ・規則

書換え規則

---

rule-path [*Variable*]

タイプ: string/pathname
初期値: "/home/yamaoka/System/III/Rules/"

書換え規則ファイルのあるパス名. (`load.lisp`)

---

rule-file-list [*Variable*]

タイプ: list of strings
初期値: (list "preedit" "frozen" "intention" "proposition-ignore" "object-general" "object-wh" "topic" "postedit")

書換え規則ファイル名のリスト. (`load.lisp`)

---

load-rule-file *file &optional (dir *rule-path*)* [*Function*]

引数:

1. *file* : 規則ファイル名

2. *dir (&optional)*: 規則ファイルのパス名

リターン値:

引数で指定された規則ファイルをロードする. (`cate-main.lisp`)

---

load-rules *&optional (dir *rule-path*)* [*Function*]

引数:

1. *dir (&optional)*: 規則ファイルのパス名

リターン値:

書換え規則の全ロード. 引数 *dir* の, 大域変数 *rule-file-list* の要素で指定されたファイルすべてをロードする. (`cate-main.lisp`)

---

clear-rules [*Macro*]

引数: なし
リターン値:

書換え規則の全削除. (関数 rws::remove-all-rw-rules2 に同じ) (`cate-main.lisp`)

---

renew-rws-rules *&optional (dir *rule-path*)* [*Function*]

引数:

1. *dir (&optional)*: 規則ファイルのパス名

リターン値:

規則の再ロード. 現在定義されている規則を全削除し, 新たに load-rules を行なう. (`cate-main.lisp`)

命題格要素データ

---

proposition-grammar-filename [*Variable*]

タイプ: string/pathname
初期値: "/home/yamaoka/System/III/Data/case-grammar.lisp"

命題格要素データベースのファイル名. (load.lisp)

---

proposition-grammar [*Variable*]

タイプ: alist
初期値: nil

命題格要素のデータベース. (output.lisp)

---

load-proposition-grammar [*Function*]

引数: なし
リターン値:
命題格要素データベースをファイル:*proposition-grammar-filename*)
からロードし、*proposition-grammar* に設定する。(output.lisp)

---

get-pg *pred* [*Function*]

引数:

1. *pred* : 述語名

リターン値: list (命題格要素データ)

pred の命題格要素データが *proposition-grammar* にあれば、そのデータを返す。それ以外は nil を返す。(output.lisp)

話題属性関連

(概念・表現ネットーワーク検索システム本体については、参考文献 [10] 参照)

---

np-load-filename [*Variable*]

タイプ: string/pathname
初期値: "/home/yamaoka/System/III/NP/load"

概念・表現ネットーワーク検索システムロード用のファイル名. (`load.lisp`)

---

np-concept-filename [*Variable*]

タイプ: string/pathname
初期値: "/home/yamaoka/System/III/Data/concept-nodes.lisp"

概念ネットワークの辞書ファイル名. (`load.lisp`)

---

load-concept *&optional (file *np-concept-filename*)* [*Function*]

引数:

1. *file (&optional)*: 概念ネットワーク辞書ファイル名

リターン値: numeric(設定したネットワークのノード数)

file をロードし, 概念ネットワークを設定する. 引数は optional であり, 明示的に与えられなければ, *np-concept-filename* の値が利用される。話題属性解析に利用される概念集合の変数は更新しないので、そこまで更新する場合は、(parameter-set) を行なう必要がある。(`load.lisp`)

---

parameter-set-filename [*Variable*]

タイプ: string/pathname
初期値: "/home/yamaoka/System/III/Data/parameter-set.lisp"

話題属性パラメータ初期設定用のファイル名. (load.lisp)

---

parameter-set *&optional (package *read-package*)* [*Function*]

引数:

1. *package (&optional)*: 読み込みパッケージ名[1]

リターン値: loaded file's path

話題属性解析用のパラメータを設定する。実際には、*package* 下において、*parameter-set-filename* をロードするだけである。解析の入力となる中間素性構造を参照するためのパラメータについては、以下の表を参照. (parameters.lisp)

中間素性構造参照用パラメータ 変数名	現在の値 (備考)
cate-cat-path	'(CAT) ; 情報伝達行為タイプ (表層発話行為タイプ) の素性名
cate-topic-path	'(TPC) ; トピックの素性名
cate-proposition-path	'(PRP) ; 命題内容の素性名
cate-sem-feature-path	'(SEM) ; 意味表現の素性名 (情報伝達行為が出ていない時利用)
cate-object-main-feature	'VAL ; 主に名詞句概念の主辞を表す素性名
cate-predicate-feature-name	'RELN ; 命題内容の主述語を表す素性名
cate-agent-value-list	'(SPEAKER HEARER) ; 対話参加者を表す素性値の集合
cate-speaker-value	'(SPEAKER) ; 発話の話し手を表す素性値の集合
cate-hearer-value	'(HEARER) ; 発話の聞き手を表す素性値の集合

[1]この『読み込みパッケージ』は、CATE, LAYLA では、しばしば現れる。何のために利用されているかというと、パッケージの違うシステムで、同じシンボルを同じパッケージのものとして扱うために外部のデータを読み込む際に利用している。システムごとにパッケージが異なっていることがこの手間をつくっている。

---

output-rule-filename [*Variable*]

タイプ: string/pathname
初期値: 初期値: "/home/yamaoka/System/III/Rules/output-rules.lisp"

話題の属性を利用した抽出規則のソースファイル. (現在は LISP code で書かれている.) (`load.lisp`)

---

load-output-rules *&optional (filename *output-rule-filename*)* [*Function*]

引数:

1. *filename (&optional)*: 話題属性規則ファイル名

リターン値:

話題属性規則をロードする. 現在の CATE では、この規則は LISP code なので、単に *filename* をロードするのみ。(`output.lisp`)

話し手

---

speaker-filename [*Variable*]

タイプ: string/pathname
初期値: "/home/yamaoka/System/III/Data/speaker.list"

サンプル会話内の発話の ID と話し手の対応データのファイル名. (`load.lisp`)

---

cate-speaker-list [*Function*]

引数: なし
リターン値: alist

speaker-filename から、発話 ID・話しての対応データを読み込み、その A リストを返す。(`parameters.lisp`)

---

get-utterance-speaker *snum* [*Function*]

引数:

1. *snum* : 発話 ID

リターン値: symbol (現在は、SP1/SP2)

snum に対応する発話の話し手を表すシンボルを返す。そのようなものがなければ nil. (parameters.lisp)

---

get-utterance-hearer *snum* [*Function*]

引数:

1. *snum* : 発話 ID

リターン値: symbol (現在は、SP1/SP2)

snum に対応する発話の聞き手を表すシンボルを返す。そのようなものがなければ nil. (parameters.lisp)

---

cate-speaker-p *fs* [*Function*]

引数:

1. *fs* : 素性構造

リターン値: T/nil

fs が話し手を表すシンボル (:atomic 型素性構造) であれば T それ以外であれば nil を返す。(parameters.lisp)

A.2.3 発話モダリティ変換

---

transfer *node* [*Function*]

引数:

1. *node* : 素性構造

リターン値: 中間素性構造のリストのリスト

node の発話モダリティ解析を行なう。(3.3節1 のすべての処理を行なう。)
リターン値は, 以下の構造である:

```
(list (list 命題1についての中間素性構造1 命題1についての中間素性構造2 ...)
      ...
      (list 命題nについての中間素性構造1 命題nについての中間素性構造2 ...))
```

つまり, 入力発話が複数命題を持つものでなければ, リターン値の長さは1
である.

A.2.4 発話表現変換

---

utterance-representation *snum fs &optional (string "none")* [*Function*]

引数:

1. *snum* : 発話 ID
2. *fs* : 素性構造
3. *string (&optional)*: 発話文字列

リターン値: list (発話の表現)

fs(素性構造) を発話の表現 (リスト) に変換したものを返す.
現在のところ, この中で, 話題の属性による情報伝達行為の解析 (get-cats-from-fs) を行なっている.

string は、オプショナルであり、発話の表現のリストの第6要素に加える。
(主にデバッグ表示用に利用) (`output.lisp`)

命題内容変換

---

get-proposition-from-fs *fs* [*Function*]

引数:

1. *fs* : 素性構造 (命題内容)

リターン値: symbol/variable/list (命題内容リスト) /nil

命題内容部分素性構造 *fs* を発話表現リスト用のデータ構造 (命題内容リスト) に変換する。(`output.lisp`)

---

fs-to-urep-list-with-pg *pg fs* [*Function*]

引数:

1. *pg* : 命題格要素データ
2. *fs* : 素性構造 (命題内容)

リターン値: list (命題内容リスト)

pg の記述に従って、素性構造 *fs* から命題内容リストを構成する。(`output.lisp`)

話題属性解析

---

get-cats-from-fs *fs cdr* [*Function*]

引数:

1. *fs* : 素性構造 (中間素性構造)
2. *cdr* : 発話の表現の cdr

リターン値: list (発話の表現)

話題の属性による書換えを行なった結果の発話の表現を返す。

引数 *cdr* は、話題の属性を鑑みる以前の発話表現のデータである。(`output.lisp`)

---

get-caterule *cat* [*Function*]

引数:

1. *cat* ：表層発話行為タイプ

リターン値: function (規則の関数定義)/nil

cat の話題の属性の規則を定義した関数があれば、それを返す. それ以外は nil. (output.lisp)

---

urep-with-apply-caterule *rule fs urep* [*Function*]

引数:

1. *rule* ：話題属性規則
2. *fs* ：素性構造 (中間素性構造)
3. *urep* ：発話の表現リスト

リターン値: list (発話表現リスト)

fs, urep に *rule* を 適用した後の発話の表現リストを返す。(output.lisp)

---

urep-with-atomic-prp *urep* [*Function*]

引数:

1. *urep* ：発話の表現リスト

リターン値: list (発話の表現)

urep の命題内容部分が命題データで表されていないものについて、データの変換を行なう。

具体的には、atomic な命題内容 (例えば、“銀行振り込み”) を変項とし、その値をトピックの位置に置き換えている。(output.lisp)

オブジェクト抽出

---

fs-to-urep-value *fs feature &key (without-sp nil)* [*Function*]

引数:

1. *fs* ：素性構造
2. *feature* ：素性パス
3. *without-sp (&key)*:

リターン値: symbol/variable (概念の主辞)

fs の *feature* で指定された素性値の主辞のシンボル (多くの場合は、*cate-object-main-feature* で与えられた素性構造の素性値) を返す。

キーワード *without-sp* が nil 以外であれば、*cate-speaker-value*, *cate-hearer-value* に指定されたシンボルを、システム特有の話し手・聞き手を表すシンボル (SP1/SP2) に変換して返す。それ以外であれば変換しない。(output.lisp)

変項

---

genvar *&optional string-list* [*Function*]

引数:

1. *string-list (&optional)*: 文字列のリスト

リターン値: variable (変項を表すシンボル)

変項を表すシンボルを生成し、それを返す。現在 CATE(LAYLA) では、prefix として "?" を付したシンボルを変項としている。

string-list が明示的に与えられれば、その要素である文字列を連結した文字列を変項名 (prefix に続く文字の並び) として変項を生成する。与えられなければ、内部的に文字列を生成し、それを変項名とする。(output.lisp)

---

urep-var-p *atom* [*Function*]

引数:

1. *atom* ：任意のデータ

リターン値: T/nil

atom が変項であれば、T を、それ以外ならば nil を返す。(output.lisp)

A.3 実験用インタフェース

ここでは、CATE を単体で実験する際の環境と、便利な関数について説明する。なお説明注で明示的にファイル名が提示されていないものは、`examine.lisp` にある。

また、実際の利用例については、付録 B を参照。

A.3.1 入出力インタフェース

内部データ

---

sentence-obj [*Structure*]

CATE 実験用の内部データ構造で、以下のようなスロットに内部処理における情報を格納している:

スロット名	初期値; 内容
number	nil ; 発話 ID(symbol)
speaker	nil ; 話し手 (symbol)
string	nil ; 発話文文字列 (string)
input-fs	nil ; 入力素性構造 (日本語解析結果: list of fs)
results	nil ; 中間素性構造 (発話モダリティ解析結果: list of fs)
cat-list	nil ; 発話表現リスト (最終的結果: list of list)

本実験用インタフェースの多くの関数では、基本的にこの内部データを対象として扱う。

---

kaiwa-input-data [*Variable*]

タイプ: hash-table
初期値: (make-hash-table)

内部データ sentence-obj を発話 ID をキーとして保持するデータプール。

---

init-kaiwa-input-data [*Function*]

引数: なし

kaiwa-input-data の全ての内部データを消去する。(初期化)

---

kaiwa-input-id-list [*Macro*]

引数: なし
リターン値: list (発話 ID のリスト)

kaiwa-input-data 内に保持されている全ての内部データの発話 ID のリストを返す。

---

make-inputdata *id sp string node-list* [*Macro*]

引数:

1. *id* : 発話 ID
2. *sp* : 話し手
3. *string* : 発話文文字列
4. *node-list* : 素性構造のリスト

リターン値: 内部データ構造 sentence-obj

引数を入力の情報として、内部データ構造 sentence-obj を作成し、それを返す。この時点では、結果 (results, cat-list スロット) は何もない。

---

find-inputdata *id* [*Function*]

引数:

1. *id* : 発話 ID

リターン値: 内部データ構造 sentence-obj

id に対応する内部データが *kaiwa-input-data* に存在すれば、その内部データを返す。そうでなければ nil.

入力

---

kaiwa-input-path [*Variable*]

タイプ: string/pathname
初期値: "/home/yamaoka/Nadine/Results/"

サンプル会話の日本語解析結果素性構造出力ファイルがあるパス名. (`load.lisp`)

---

kaiwa-input-file [*Variable*]

タイプ: string
初期値: "sem-1.fs"

デフォールト処理対象となる会話の素性構造ファイル名.

---

kaiwa-input-file-list [*Variable*]

タイプ: list of strings
初期値: (list "sem-1.fs" "sem-2.fs" "sem-3.fs" "sem-4.fs" "sem-5.fs" "sem-6.fs" "sem-7.fs" "sem-8.fs" "sem-9.fs" "sem-10.fs" "sem-A.fs" "sem-B.fs")

サンプル会話の日本語解析結果素性構造出力ファイル名のリスト. 現在は、以上の 12 会話。 (`load.lisp`)

---

load-kaiwa-input-file *&optional (file *kaiwa-input-file*) &key (print t)* [*Function*]

引数:

1. *file (&optional)* : 入力ファイル名
2. *print (&key)* : 表示フラグ

リターン値: *kaiwa-input-data*

file のデータを読み込み (長谷川システムの rws::push-fs-from-file を利用) 発話ごとに内部データを作成して、 *kaiwa-input-data* に格納する。

もし、 *print* に nil 以外が与えられれば、作成された内部データの発話 ID と文字列を、端末出力に表示する。

ここでは、内部環境 *kaiwa-input-data* の初期化を行なわないので、その時点までで保持されている内部データに追加する形で、データが読み込まれる。従って、それまでの内部データをクリアしたい場合は、明示的に (init-kaiwa-input-data) を行なう必要がある。

出力

---

kaiwa-output-path [*Variable*]

タイプ: string/pathname
初期値: "/home/yamaoka/System/III/Log/"

情報伝達行為解析後の出力, 発話の表現のリスト (のリスト), を書き出す際のパス名. (`load.lisp`)

---

kaiwa-data *&optional file* [*Function*]

引数:

1. *file (&optional)* : 出力ファイル名

リターン値: T

その時点で保持されている全ての内部データ ((kaiwa-input-id-list) で得られる発話 ID) の結果を、階層型プラン認識システム入力用のデータフォーマットにして、出力する。

出力先は、*file* に明示的にファイル名が与えられれば、*kaiwa-output-path* で指定されたディレクトリの下のそのファイル、そうでなければ、端末出力である。

---

make-and-print-a-urep *snum &optional (stream t)* [*Function*]

引数:

1. *snum* : 発話 ID
2. *stream (&optional)*出力ストリーム

snum に対応する内部データを、発話表現変換を行なった後、階層型プラン認識システム入力用のデータフォーマットにして、*stream* に出力する。

A.3.2 解析用関数

---

tran *id* [*Function*]

引数:

1. *id* : 発話 ID

リターン値: 内部データ

id に対応する内部データについて、発話モダリティの解析を行なう。解析結果の中間素性構造のリストは、results スロットに setf する。

この関数は、内部データの input-fs の全ての入力素性構造について、解析を行なった後に、重複する結果を削除したリストを格納する。

---

tran-1 *id* [*Function*]

引数:

1. *id* : 発話 ID

リターン値: 内部データ

id に対応する内部データについて、発話モダリティの解析を行なう。解析結果の中間素性構造のリストは、results スロットに setf する。

この関数は、内部データの input-fs の先頭 (car) の入力素性構造一つについてのみ、解析を行なった後に、重複する結果を削除したリストを格納する。

---

urep *id* [*Function*]

引数:

1. *id* : 発話 ID

リターン値: 発話表現リスト (のリスト)

id に対応する内部データについて、発話表現変換 (話題属性解析を含む) を行なう。結果の発話表現のリストは、cat-list スロットに setf する。

この関数は、内部データの results の全ての中間素性構造について、変換を行なった後に、重複する結果を削除したリストを格納する。

A.3.3 表示

---

show-input-fs *sentence-obj* [*Function*]

引数:

1. *sentence-obj* : 内部データ / 発話 ID

sentence-obj の全ての入力素性構造 (input-fs スロット) を、端末出力に表示する。引数 *sentence-obj* は、それが内部データであればそれを、発話 ID であればそれに対応する内部データが利用される。

---

show-results *sentence-obj* [*Function*]

引数:

1. *sentence-obj* ：内部データ / 発話 ID

sentence-obj の全ての中間素性構造 (results スロット) を、端末出力に表示する。引数 *sentence-obj* は、それが内部データであればそれを、発話 ID であればそれに対応する内部データが利用される。

---

show-urep *id* [*Function*]

引数:

1. *id* ：内部データ / 発話 ID

id の全ての入力素性構造 (cat-list スロット) を、端末出力に表示する。引数 *sentence-obj* は、それが内部データであればそれを、発話 ID であればそれに対応する内部データが利用される。

---

show-kaiwa [*Function*]

引数: なし

その時点で保持されている全ての内部データの、発話 ID と発話文文字列を、端末出力に表示する。

A.3.4 バッチ処理

---

examine-file *&optional (file *kaiwa-input-file*)* [*Function*]

引数:

1. *file (&optional)* : 入力ファイル名

file のデータを読み込み、CATE の全ての解析を行なった後に、その結果を LAYLA 入力用データフォーマットにして、端末出力に出力する。

A.4 素性構造書換えシステムの拡張

長谷川システムでは、規則の (編集による) 追加・削除が面倒であった。特にある特定の規則を編集して置き換えたい場合、その規則の KEY と同じ KEY を持つ規則すべてを消さなければならなかったり、規則特定の NAME を検索する方法がなかった。

機能: 書換え規則に (ユーザ定義による) ユニークな ID を付加できる。また、その ID にしたがって規則の管理を行なうことができる。

注意: 現在のところ、本インタフェースと基システムの関数（あるいはマクロ）を混在して使うことは、想定していない。従って、混在して利用した場合は、規則ベース内部あるいはインタフェース機能の整合性を保証できない。

利用方法: 基システムをロードした後、`rws-id.lisp` をロードする。

A.4.1 外部関数

---

defrwschema2 *id type sort-key body &optional (check nil)* [*Macro*]

引数:

1. *id* : 規則 ID(symbol)
2. : 第 2 引数以降は defrwschema に同じ
3. *check (&optional)* 置換検査フラグ

リターン値: defrwschema に同じ

書換え規則を規則ベースに登録する。

check が nil であれば、もし *id* に対応する規則が既に規則ベース内に存在すれば、その規則を削除しここで定義される規則を追加する (置換)。*check* に nil 以外が与えらた場合、*id* の規則が存在すれば、ワーニングを出して登録をしない。規則が存在しなければ、登録する。

---

remove-all-rw-rules2 [*Function*]

引数: なし
リターン値: remove-all-rw-rules に同じ

規則ベース内のすべての規則を削除する。(バグあり)

---

remove-rw-rule2 *id* [*Function*]

引数:

1. *id* : 規則 ID

リターン値: remove-rw-rule に同じ

id に対応する規則を (もし存在すれば) 規則ベースから削除する。

---

print-all-rw-rules2 [*Function*]

引数: なし

規則ベース内のすべての規則を端末出力に表示する。

表示内容は、一つの規則に対して、規則 ID,*sort-key*, 素性パス, 素性値。
またこの際、余計なお世話であるが、規則 ID によりソートする。

---

print-rw-rule2 *id* [*Function*]

引数:

1. *id* : 規則 ID

id に対応する規則の内容を端末出力に表示する。表示内容は、規則 ID,*sort-key*, 素性パス, 素性値、および本体.

A.4.2 規則管理

書換え規則固有の ID は、基システムとは独立して、固有の管理を行なっている。従って、本システムの規則ベースの内容と、規則 ID の内容がずれることが危惧される。今後改良されることを望む。

---

rws-IDDB [*Variable*]

タイプ: hash-table
初期値: 空のハッシュテーブル

規則 ID を管理するハッシュテーブル。

IDDB-id-list [*Macro*]

引数: なし
リターン値: 規則 ID のリスト

その時点で *rws-IDDB* が管理している規則 ID の一覧リストを返す。

---

initiate-IDDB *db* [*Function*]

引数:

1. *db* : ハッシュテーブル (*rws-IDDB*)

リターン値: *db*

db を初期化する。

---

count-rw-rules [*Function*]

引数: なし
リターン値: numeric

rws-IDDB に登録されている規則 ID の数を返す。

---

check-id-in-IDDB *id flag* [*Function*]

引数:

1. *id* : 規則 ID
2. *flag* : フラグ (T/nil)

リターン値: T/nil

flag が T の時、既に *id* の規則が存在すれば、ワーニングを出して nil を返す。それ以外の時は、T を返す。

---

put-id-to-IDDB *id* [*Function*]

引数:

1. *id* : 規則 ID

リターン値: list of IDs

id のみを *rws-IDDB* に登録する。(規則本体は登録しない)

---

remove-id-from-IDDB *id* [*Function*]

引数:

1. *id* : 規則 ID

リターン値: list of IDs

id のみを *rws-IDDB* から削除する。(規則本体は削除しない)

---

find-rw-rule2 *id* [*Function*]

引数:

1. *id* : 規則 ID

リターン値: 規則定義リスト

id に対応する規則定義リスト (本システム defrwschema に与える引数のリスト) を返す。そのようなのものがなければ、nil.

---

remove-rw-rule-from-IDDB *id* [*Function*]

引数:

1. *id* : 規則 ID

リターン値: T/nil

id とそれに対応する規則定義リストを *rws-IDDB* から削除する。そのようなものがなければ何もしない。

付録 B

実験システム利用例

CATE を起動するには, Common Lisp 上で, ロードファイル (load.lisp: 付録 A参照) をロードすれば良い。なお、長谷川システムが、 UNIX マシン上でしか作動しないために[1]、他のリスプマシンでは動作しない。

```
Starting /usr/local/bin/lisp ...
;;; Sun Common Lisp, Development Environment 4.0.0 , 6 July 1990
;;;
;;; Sun, Sun-4, and Sun Common Lisp are trademarks of Sun Microsystems Inc.

>
;;
;; CATE システムの起動
;;
> (load "/home/yamaoka/System/III/CATE/load")
;;; Loading source file "/home/yamaoka/System/III/CATE/load.lisp"
--------- CATE SYSTEM 2.0 ---------
;;; Loading source file "/home/yamaoka/System/III/CATE/rws-load-as21.lisp"

(途中省略)

;;; Warning: File "output-rule.lisp" does not begin with IN-PACKAGE.
 Loading into package "CATE"

----- CATE 2.0 for demonstration Dialogue 1-10 ----
;;
;; プログラムをロードし終ると自動的に
;; 書換えシステムのデバッグモードを聞いてくる
;;
Print Input Feature Structure [RWS::YES] (Y or N): n
```

[1]シンボリクス上で動くシステムは、あるという話しであるが、分からない。

```
Print Rewritten Results [RWS::YES] (Y or N): n
Trace Applying Rules to FS and Result FSs [RWS::NO](Yes, No or Verbose) n
Trace Rewriting Call [RWS::NO](Yes, No or Verbose) n
Trace Pattern Matching[RWS::NO] (Y or N): n
Trace Parameter Setting [RWS::NO](Yes, No or Verbose) n
#P"/home/yamaoka/System/III/CATE/load.lisp"
;;
;; ロード終了
;;
> (in-package :cate)
#<Package "CATE" 63449E>
> *kaiwa-input-file*
"sem-1.fs"
;;
;; ファイル読み込み
;;
> (load-kaiwa-input-file)
D1-1: もしもし,
D1-2: そちらは会議事務局ですか,
D1-3: はい,
D1-4: そうです,
D1-5: どのようなご用件でしょうか,
D1-6: 会議に申し込みたいのですが, 2 3
D1-7: どのような手続きをすればよろしいのでしょうか, 2 3
D1-8: 登録用紙で手続きをして下さい, 2 3
D1-9: 登録用紙は既にお持ちでしょうか,
D1-10: いいえ,
D1-11: まだです,
D1-12: 分かりました, 2 3 4
D1-13: それでは登録用紙をお送り致します, 2
D1-14: ご住所とお名前をお願いします,
D1-15: 住所は大阪市北区茶屋町二十三です,
D1-16: 名前は鈴木真弓です,
D1-17: 分かりました, 2 3 4
D1-18: 登録用紙を至急送らせて頂きます,
D1-19: よろしくお願いします,
D1-20: それでは失礼します, 2
#<Hash-Table 11515E6>
;;
;; D1-9 を発話モダリティ解析
;;
> (tran 'd1-9)
D1-9 登録用紙は既にお持ちでしょうか
;;; Result 1
```

```
 [[CAT CONFIRM]
  [PRP [[AGEN []]
        [OBJE !X1 登録用紙-1]
        [RELN 持つ-1]]]
  [TPC !X1]]
#<Structure SENTENCE-OBJ BAB4F6>
;;
;; 続いて D1-9 を発話表現変換
;;
> (urep 'd1-9)
(((CONFIRM-STATEMENT SP2 SP1 登録用紙-1 (持つ-1 ?AGEND1-9 登録用紙-1)
 "登録用紙は既にお持ちでしょうか")))

;;
;; 1会話ファイルのバッチ実験
;;
> *kaiwa-input-file*
"sem-1.fs"
;;
;; デフォールトで上のファイルの実験を行なう
;;
> (examine-file)
;;
;; ファイル読み込み
;;
D1-1: もしもし,
D1-2: そちらは会議事務局ですか,
D1-3: はい,
D1-4: そうです,
D1-5: どのようなご用件でしょうか,
D1-6: 会議に申し込みたいのですが, 2 3
D1-7: どのような手続きをすればよろしいのでしょうか, 2 3
D1-8: 登録用紙で手続きをして下さい, 2 3
D1-9: 登録用紙は既にお持ちでしょうか,
D1-10: いいえ,
D1-11: まだです,
D1-12: 分かりました, 2 3 4
D1-13: それでは登録用紙をお送り致します, 2
D1-14: ご住所とお名前をお願いします,
D1-15: 住所は大阪市北区茶屋町二十三です,
D1-16: 名前は鈴木真弓です,
D1-17: 分かりました, 2 3 4
D1-18: 登録用紙を至急送らせて頂きます,
D1-19: よろしくお願いします,
```

```
D1-20: それでは失礼します, 2
;;
;; 発話モダリティ解析開始
;;
D1-1 もしもし
;;; ------------ Transfer Input ------------
 [[PRAG [[HEARER !X1[[LABEL *HEARER*]]]
         [SPEAKER !X2[[LABEL *SPEAKER*]]]]]
  [SEM [[AGEN !X2]
        [ASPT -]
        [RECP !X1]
        [RELN もしもし-OPEN_DIALOGUE]]]]

(途中大きく省略)

;; D1-20 の結果
;;
;;; Result 1
 [[CAT GREETING-CLOSE]
  [PRP [[RELN CLOSE-DIALOGUE]]]
  [TPC CLOSING]]
;;
;; 発話表現変換
;;
(D1-1
  (
    (GREETING-OPEN      SP1 SP2 OPENING
                      (OPEN-DIALOGUE)
                      "もしもし")
  )
)

(途中大きく省略)

(D1-20
  (
    (GREETING-CLOSE     SP1 SP2 CLOSING
                      (CLOSE-DIALOGUE)
                      "それでは失礼します")
  )
)
T
>
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

付録 C

語用論知識集

以下に、サンプル会話の発話モダリティ解析で利用する語用論知識 (長谷川書換え規則) の一覧を示す。これらは、現在、以下のファイルにある:

対象	ファイル名
固定化表現	as21:/home/yamaoka/System/III/Rules/frozen.lisp
文末表現	as21:/home/yamaoka/System/III/Rules/intention.lisp
その他	同上 (環境名 :Type = :default

C.1 固定化表現規則

```
; Response Class ;
;;
;; AFFIRMATIVE
;;
;;; 1. はい*
(rws:defrwschema2 afm-1 afm hai
  "on <SEM RELN> はい-AFFIRMATIVE
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  はい-AFFIRMATIVE]
                 [AGEN [[LABEL  SPEAKER]]]
                 [RECP [[LABEL  HEARER]]]
                 ?rest]]
            ?prag]
      out= [[CAT AFFIRMATIVE]
             [TPC []]
             [PRP []]]
   end")
;;; 2. そうです*
(rws:defrwschema2 afm-2-1 afm soudesu
  "on <SEM MANN RESTR RELN> そう-1
   in :Phase :intention :Type :general
      in= [[SEM [[RELN だ-STATEMENT]
;                  [ASPT  STAT]
                 [MANN [[RESTR [[RELN そう-1]
                                ?rest1]]
                        ?rest2]]
                 ?rest3]]
            ?prag]
```

```
      out= [[CAT AFFIRMATIVE]
            [TPC []]
            [PRP []]]
   end")
(rws:defrwschema2 afm-2-2 afm soudesu
  "on <SEM IDEN RESTR RELN> そう-1
   in :Phase :intention :Type :general
      in= [[SEM [[RELN だ-STATEMENT]
;;                  [ASPT  STAT]
                  [IDEN [[RESTR [[RELN そう-1]
                                 ?rest1]]
                         ?rest2]]
                  ?rest3]]
           ?prag]
      out= [[CAT AFFIRMATIVE]
            [TPC []]
            [PRP []]]
   end")

;;
;; NEGATIVE
;;
;;; 1. いいえ*
(rws:defrwschema2 ngt-1 ngt iie
  "on <SEM RELN> いいえ-NEGATIVE
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  いいえ-NEGATIVE]
                  [AGEN [[LABEL  SPEAKER]]]
                  [RECP [[LABEL  HEARER]]]
                  ?rest]]
           ?prag]
      out= [[CAT NEGATIVE]
            [TPC []]
            [PRP []]]
   end")

;;; 2. まだです*
(rws:defrwschema2 ngt-2 ngt madadesu
  "on <SEM TLOC RESTR RELN> まだ-1
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  だ-STATEMENT]
;;                  [ASPT  STAT]
                  [TLOC [[RESTR [[RELN  まだ-1]
                                 ?rest1]]
                         ?rest2]]
                  ?rest3]]
           ?prag]
      out= [[CAT NEGATIVE]
            [TPC []]
            [PRP []]]
   end")
(rws:defrwschema2 ngt-2-2 ngt madadesu
  "on <SEM IDEN RESTR RELN> まだ-1
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  だ-STATEMENT]
;;                  [ASPT  STAT]
                  [IDEN [[RESTR [[RELN  まだ-1]
                                 ?rest1]]
                         ?rest2]]
                  ?rest3]]
           ?prag]
      out= [[CAT NEGATIVE]
            [TPC []]
            [PRP []]]
   end")
```

```
;;
;; ACCEPT-ACTION
;;
;;; 1. 分かりました *
(rws:defrwschema2 aca-1-1 aca wakarimashita
  "on <SEM RELN> 分かる-1
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  分かる-1]
                 [ASPT ?asp]
                 [OBJE ?obj]
                 ?rest1]]
            [PRAG ?prag]
            ?rests]
      if ?asp is $UNRL then fail endif
      out= [[CAT ACCEPT-ACTION]
             [TPC ?prag]
             [PRP ?obj]]
   end")
(rws:defrwschema2 aca-1-2 aca wakarimashita
  "on <SEM RELN> 分かる-2
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  分かる-2]
                 [ASPT ?asp]
                 [OBJE ?obj]
                 ?rest1]]
            [PRAG ?prag]
            ?rests]
      if ?asp is $UNRL then fail endif
      out= [[CAT ACCEPT-ACTION]
             [TPC ?prag]
             [PRP ?obj]]
   end")

;;; 2. 問題ありません *
(rws:defrwschema2 aca-2 aca mondainai
  "on <SEM OBJE RELN> 問題ある-1
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  NEGATE]
                 [ASPT  STAT]
                 [OBJE [[RELN  問題ある-1]
                        [LOCT ?prp]
                        ?rest1]]
                 ?rest2]]
             [PRAG  ?prag]
             ?rests]
      out= [[CAT ACCEPT-ACTION]
             [TPC ?prag]
             [PRP ?prp]]
   end")

;;; 3. いいですよ
(rws:defrwschema2 aca-3 aca iidesuyo
  "on <SEM RELN> いいですよ-CONFIRMATION
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  いいですよ-CONFIRMATION]
                 [ASPT  -]
                 [AGEN [[LABEL  SPEAKER]]]
                 [RECP [[LABEL  HEARER]]]]]
            ?prag]
      out= [[CAT ACCEPT-ACTION]
             [PRP []]
             [TPC []]]
   end")
```

```
;;
;; ACCEPT-OFFER
;;
;;; 1. ありがとう*
(rws:defrwschema2 aco-1 aco thanks
  "on <SEM RELN> ありがとう-THANKING
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  ありがとう-THANKING]
                 [ASPT  -]
                 [AGEN [[LABEL  SPEAKER]]]
                 [RECP [[LABEL  HEARER]]]
                 ?rest]]
            [PRAG  ?prag]
            ?rests]
      out= [[CAT ACCEPT-OFFER]
            [PRP []]
            [TPC []]]
   end")

;; 2. お願いします*
(rws:defrwschema2 aco-2 aco please
  "on <SEM RELN> 願う-REQUEST
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  願う-REQUEST]
;;                  [ASPT  UNRL]
                 [AGEN [[LABEL  SPEAKER]]]
;;                  [RECP [[LABEL  HEARER]]]
                 [OBJE []]
                 ?rest1]]
            ?rests]
      out= [[CAT ACCEPT-OFFER]
            [PRP []]
            [TPC []]]
   end")

; CONFIRM Class ;
;;
;; CONFIRMATION
;;
;;; 1. 分かりました*
(rws:defrwschema2 cfm-1 cfm wakarimashita
  "on <SEM RELN> 分かる-1
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  分かる-1]
                 [ASPT ?asp]
                 [OBJE ?obj]
                 ?rest1]]
           [PRAG ?prag]
           ?rests]
      if ?asp is $UNRL then fail endif
      out= [[CAT CONFIRMATION]
            [TPC ?prag]
            [PRP ?obj]]
   end")
(rws:defrwschema2 cfm-2 cfm wakarimashita
  "on <SEM RELN> 分かる-2
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  分かる-2]
                 [ASPT ?asp]
                 [OBJE ?obj]
                 ?rest1]]
           [PRAG ?prag]
           ?rests]
      if ?asp is $UNRL then fail endif
      out= [[CAT CONFIRMATION]
```

```
            [TPC ?prag]
            [PRP ?obj]]
   end")

;;; 2. そうですか*
(rws:defrwschema2 cfm-3 cfm soudesuka
  "on <SEM OBJE OBJE MANN RESTR RELN> そう-1
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  S-REQUEST]
                 [AGEN !SP[[LABEL  SPEAKER]]]
                 [RECP !HR[[LABEL  HEARER]]]
                 [OBJE [[RELN  INFORMIF]
                        [AGEN !HR]
                        [RECP !SP]
                        [OBJE [[RELN  だ-STATEMENT]
                               [MANN [[RESTR [[RELN  そう-1]
                                              ?rest0]]
                                      ?rest1]]
                              ?rest2]]
                       ?rest3]]
                 ?rest4]]
            ?prag]
      out= [[CAT CONFIRMATION]
            [TPC []]
            [PRP []]]
   end")
(rws:defrwschema2 cfm-3-1 cfm soudesuka
  "on <SEM OBJE OBJE IDEN RESTR RELN> そう-1
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  S-REQUEST]
                 [AGEN !SP[[LABEL  SPEAKER]]]
                 [RECP !HR[[LABEL  HEARER]]]
                 [OBJE [[RELN  INFORMIF]
                        [AGEN !HR]
                        [RECP !SP]
                        [OBJE [[RELN  だ-STATEMENT]
                               [IDEN [[RESTR [[RELN  そう-1]
                                              ?rest0]]
                                      ?rest1]]
                              ?rest2]]
                       ?rest3]]
                 ?rest4]]
            ?prag]
      out= [[CAT CONFIRMATION]
            [TPC []]
            [PRP []]]
   end")

; OTHERS ;
;;
;; GREETING-OPEN
;;
;;; 1 もしもし*
(rws:defrwschema2 gop-1 gop moshimoshi
  "on <SEM RELN> もしもし-OPEN_DIALOGUE
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  もしもし-OPEN_DIALOGUE]
                 [AGEN [[LABEL  SPEAKER]]]
                 [RECP [[LABEL  HEARER]]]
                 ?rest]]
            ?prag]
      out= [[CAT GREETING-OPEN]
            [TPC opening]
            [PRP [[RELN open-dialogue]]]]
   end")
```

```
;;; 2. はい*
(rws:defrwschema2 gop-2 gop hai
  "on <SEM RELN> はい-AFFIRMATIVE
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  はい-AFFIRMATIVE]
                 [AGEN [[LABEL  SPEAKER]]]
                 [RECP [[LABEL  HEARER]]]
                 ?rest]]
           ?prag]
      out= [[CAT GREETING-OPEN]
            [TPC opening]
            [PRP [[RELN open-dialogue]]]]
   end")

;;
;; GREETING-CLOSE
;;
;;; 1. さようなら*
(rws:defrwschema2 gcl-1 gcl sayonara
  "on <SEM RELN> さようなら-CLOSE_DIALOGUE
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  さようなら-CLOSE_DIALOGUE]
                 [AGEN [[LABEL  SPEAKER]]]
                 [RECP [[LABEL  HEARER]]]
                 ?rest]]
           ?prag]
      out= [[CAT GREETING-CLOSE]
            [TPC closing]
            [PRP [[RELN close-dialogue]]]]
   end")

;;; 2. 失礼します*
(rws:defrwschema2 gcl-2 gcl shitureisuru
  "on <SEM RELN> 失礼する-CLOSE_DIALOGUE
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  失礼する-CLOSE_DIALOGUE]
                 [AGEN [[LABEL  SPEAKER]]]
                 [RECP [[LABEL  HEARER]]]
                 ?rest]]
           ?prag]
      out= [[CAT GREETING-CLOSE]
            [TPC closing]
            [PRP [[RELN close-dialogue]]]]
   end")

;;; 3. ありがとうございました*
(rws:defrwschema2 gcl-3 gcl arigatou
  "on <SEM RELN> ありがとう-THANKING
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  ありがとう-THANKING]
                 [AGEN [[LABEL  SPEAKER]]]
                 [RECP [[LABEL  HEARER]]]
                 ?rest]]
           ?prag]
      out= [[CAT GREETING-CLOSE]
            [TPC closing]
            [PRP [[RELN close-dialogue]]]]
   end")

;;; 4. どういたしまして*
(rws:defrwschema2 gcl-4 gcl douitashimashite
  "on <SEM RELN> どういたしまして-YOUR-WELCOME
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  どういたしまして-YOUR-WELCOME]
```

```
                [AGEN [[LABEL  SPEAKER]]]
                [RECP [[LABEL  HEARER]]]
                ?rest]]
            ?prag]
      out= [[CAT GREETING-CLOSE]
             [TPC closing]
             [PRP [[RELN close-dialogue]]]]
   end")

;;;; end of frozen expressions ;;;
```

C.2 文末表現規則

```
; DEMAND Class ;
;;
;; ASK (SIFT)
;;
;;; 1. SUBJ は WH ですか?
(rws:defrwschema2 ask-1 ask INFORMREF
  "on <SEM OBJE RELN> INFORMREF
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  S-REQUEST]
                [AGEN !SP[[LABEL  SPEAKER]]]
                [RECP !HR[[LABEL  HEARER]]]
                [OBJE [[RELN  INFORMREF]
                       [AGEN !HR]
                       [RECP !SP]
                       [OBJE [[PARM ?wh]
                              [RESTR ?prp]
                              ?rest2]]
                       ?rest3]]
                ?rest4]]
            [PRAG ?prag]
            ?rests]
      ==> ?wh  with :Phase :object  :Type :wh-phrase
      out= [[CAT ASK]
    [WH  ?wh]
            [TPC ?prag]
            [PRP ?prp]]
   end")

;;; 2. WH か, 教えて欲しいのですが*
(rws:defrwschema2 ask-2 ask tellme
  "on <SEM OBJE RELN> 欲しい-DESIRE
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  が-MODERATE]
                [OBJE [[RELN  欲しい-DESIRE]
                       [EXPR !SP[[LABEL  SPEAKER]]]
                       [RECP !HR[[LABEL  HEARER]]]
                       [OBJE [[RELN  教える-1]
                              [AGEN !HR]
                              [OBJE [[RELN  INFORMREF]
                                     [OBJE [[PARM  ?wh]
                                            [RESTR ?prp]
                                            ?rest1]]
                                     ?rest2]]
                              ?rest3]]
                       ?rest4]]
                ?rest5]]
            [PRAG ?prag]
            ?rests]
      ==> ?wh  with :Phase :object      :Type :wh-phrase
```

```
      out= [[CAT ASK]
            [WH  ?wh]
            [TPC ?prag]
    [PRP ?prp]]
   end")

;;
;; CONFIRM (SIFT)
;;
;;; 1. です{か|ね}?
(rws:defrwschema2 cof-1 cof INFORMIF
  "on <SEM OBJE RELN> INFORMIF
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  S-REQUEST]
                 [AGEN !SP[[LABEL  SPEAKER]]]
                 [RECP !HR[[LABEL  HEARER]]]
                 [OBJE [[RELN  INFORMIF]
                        [AGEN !HR]
                        [RECP !SP]
                        [OBJE ?prp]
                        ?rest2]]
                 ?rest3]]
           [PRAG ?prag]
           ?rests]
      out= [[CAT CONFIRM]
            [TPC ?prag]
            [PRP ?prp]]
   end")
(rws:defrwschema2 cof-2 cof ne
  "on <SEM RELN> ね-CONFIRMATION
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  ね-CONFIRMATION]
                 [OBJE ?prp]
                 ?rest]]
           [PRAG ?prag]
           ?rests]
      out= [[CAT CONFIRM]
            [TPC ?prag]
            [PRP ?prp]]
   end")

;;; 2. できますか?
(rws:defrwschema2 cof-3-1 cof dekiruka
  "on <SEM OBJE OBJE RELN> できる-POSSIBLE
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  S-REQUEST]
                 [AGEN !SP[[LABEL  SPEAKER]]]
                 [RECP !HR[[LABEL  HEARER]]]
                 [OBJE [[RELN  INFORMIF]
                        [AGEN !HR]
                        [RECP !SP]
                        [OBJE [[RELN  できる-POSSIBLE]
                               [OBJE ?prp]
                               ?rest2]]
                        ?rest3]]
                 ?rest4]]
           [PRAG ?prag]
           ?rests]
      out= [[CAT CONFIRM]
            [MOD POSSIBLE]
            [TPC ?prag]
            [PRP ?prp]]
   end")
(rws:defrwschema2 cof-3-2 cof kiku
  "on <SEM OBJE OBJE OBJE RELN> できる-POSSIBLE
```

```
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  S-REQUEST]
                 [AGEN !SP[[LABEL  SPEAKER]]]
                 [RECP !HR[[LABEL  HEARER]]]
                 [OBJE [[RELN  INFORMIF]
                        [AGEN !HR]
                        [RECP !SP]
                        [OBJE [[RELN  う-GUESS]
                               [OBJE [[RELN  できる-POSSIBLE]
                                      [OBJE ?prp]
                                      ?rest1]]
                               ?rest2]]
                        ?rest3]]
                 ?rest4]]
            [PRAG ?prag]
            ?rests]
      out= [[CAT CONFIRM]
            [MOD POSSIBLE]
            [TPC ?prag]
            [PRP ?prp]]
   end")

;;
;; REQUEST (SIFT)
;;
;;; 1. OBJ をお願いします
(rws:defrwschema2 req-1 req negau
  "on <SEM RELN> 願う-REQUEST
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  願う-REQUEST]
;                 [ASPT  UNRL]
                 [AGEN [[LABEL  SPEAKER]]]
                 [OBJE ?obj]
                 ?rest]]
            [PRAG ?prag]
            ?rests]
      out= [[CAT REQUEST]
            [TPC ?prag]
    [PRP ?obj]]
   end")

;; 2. ACT (して)ください
(rws:defrwschema2 req-2-1 req kudasai
  "on <SEM RELN>  ください-REQUEST
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  ください-REQUEST]
                 [AGEN [[LABEL  SPEAKER]]]
                 [RECP [[LABEL  HEARER]]]
;                 [ASPT  UNRL]
                 [OBJE ?prp]
                 ?rest]]
            [PRAG  ?prag]
            ?rests]
      out= [[CAT REQUEST]
            [PRP ?prp]
            [TPC ?prag]]
   end")
(rws:defrwschema2 req-2-2 req kudasai
  "on <SEM RELN>  下さい-REQUEST
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  下さい-REQUEST]
                 [AGEN [[LABEL  SPEAKER]]]
                 [RECP [[LABEL  HEARER]]]
;                 [ASPT  UNRL]
                 [OBJE ?prp]
```

```
                ?rest]]
            [PRAG  ?prag]
            ?rests]
      out= [[CAT REQUEST]
             [PRP ?prp]
             [TPC ?prag]]
   end")

;;; 3. ACT してもらう
(rws:defrwschema2 req-3-0 req receive
  "on <SEM OBJE OBJE RELN> てもらう-RECEIVE_FAVOR
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  が-MODERATE]
                [OBJE [[RELN  たい-DESIRE]
                       [OBJE [[RELN  てもらう-RECEIVE_FAVOR]
                              [OBJE ?prp]
                              ?rest2]]
                       ?rest3]]
                ?rest4]]
            [PRAG ?prag]
            ?rests]
      out= [[CAT REQUEST]
             [TPC ?prag]
             [PRP ?prp]]
   end")
(rws:defrwschema2 req-3-1 req sitemoraemasuka
  "on <SEM OBJE OBJE OBJE RELN>   てもらう-RECEIVE_FAVOR
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  S-REQUEST]
                [AGEN !SP[[LABEL  SPEAKER]]]
                [RECP !HR[[LABEL  HEARER]]]
                [OBJE [[RELN  INFORMIF]
                       [AGEN !HR]
                       [RECP !SP]
                       [OBJE [[RELN  える-POSSIBLE]
                              [OBJE [[RELN  てもらう-RECEIVE_FAVOR]
                                     [OBJE ?prp]
                                     ?rest1]]
                              ?rest2]]
                       ?rest3]]
                ?rest4]]
            [PRAG  ?prag]
            ?rests]
      out= [[CAT REQUEST]
             [PRP ?prp]
             [TPC ?prag]]
   end")
(rws:defrwschema2 req-3-2 req sitemoraemasuka
  "on <SEM OBJE OBJE OBJE OBJE RELN>   てもらう-RECEIVE_FAVOR
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  S-REQUEST]
                [AGEN !SP[[LABEL  SPEAKER]]]
                [RECP !HR[[LABEL  HEARER]]]
                [OBJE [[RELN  INFORMIF]
                       [AGEN !HR]
                       [RECP !SP]
                       [OBJE [[RELN  NEGATE]
                              [OBJE [[RELN  える-POSSIBLE]
                                     [OBJE [[RELN  てもらう-RECEIVE_FAVOR]
                                            [OBJE ?prp]
                                            ?rest2]]
                                     ?rest3]]
                              ?rest4]]
                       ?rest5]]
                ?rest6]]
```

```
            [PRAG  ?prag]
            ?rests]
      out= [[CAT REQUEST]
             [PRP ?prp]
             [TPC ?prag]]
   end")
(rws:defrwschema2 req-3-3 req sitemoraenaidesuka
  "on <SEM OBJE OBJE OBJE OBJE OBJE RELN>   てもらう-RECEIVE_FAVOR
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  S-REQUEST]
                 [AGEN !SP[[LABEL  SPEAKER]]]
                 [RECP !HR[[LABEL  HEARER]]]
                 [OBJE [[RELN  INFORMIF]
                        [AGEN !HR]
                        [RECP !SP]
                        [OBJE [[RELN  う-GUESS]
                               [OBJE [[RELN  NEGATE]
                                      [OBJE [[RELN  える-POSSIBLE]
                                             [OBJE [[RELN  てもらう-RECEIVE_FAVOR]
                                                    [OBJE ?prp]
                                                    ?rest1]]
                                             ?rest2]]
                                      ?rest3]]
                               ?rest4]]
                        ?rest5]]
                 ?rest6]]
            [PRAG  ?prag]
            ?rests]
      out= [[CAT REQUEST]
             [PRP ?prp]
             [TPC ?prag]]
   end")

;;
;; Offer
;;
;;; 1. ACT させていただきます
(rws:defrwschema2 ofa-2 ofa saseteitadakimasu
  "on <SEM RELN>  てもらう-RECEIVE_FAVOR
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  てもらう-RECEIVE_FAVOR]
;;                [RECP !HR[[LABEL  HEARER]]]
                 [OBJE [[RELN  させる-PERMISSIVE]
;;                         [AGEN !HR]
;;                         [RECP [[LABEL  SPEAKER]]]
                        [OBJE ?prp]
                        ?rest1]]
                 ?rest2]]
            [PRAG  ?prag]
            ?rests]
      out= [[CAT OFFER-ACTION]
             [PRP ?prp]
             [TPC ?prag]]
   end")

; RESPONSE Class
;;
;; Inform
;;
;;; 1. したい(のですが)
(rws:defrwschema2 inf-2-1 inf want
  "on <SEM OBJE RELN> たい-DESIRE
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  が-MODERATE]
                 [OBJE [[RELN  たい-DESIRE]
```

```
                        [OBJE ?prp]
                        ?rest2]]
                  ?rest3]]
            [PRAG ?prag]
            ?rests]
      out= [[CAT INFORM]
             [MOD WANT]
             [TPC ?prag]
             [PRP ?prp]]
   end")
(rws:defrwschema2 inf-2-2 inf WANT
  "on <SEM OBJE RELN>  たい-DESIRE
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  思う-1]
                  [OBJE [[RELN  たい-DESIRE]
                        [OBJE ?prp]
                        ?rest2]]
                  ?rest3]]
            [PRAG ?prag]
            ?rests]
      out= [[CAT INFORM]
             [MOD WANT]
             [TPC ?prag]
             [PRP ?prp]]
   end")
(rws:defrwschema2 inf-2-3 inf WANT
  "on <SEM OBJE OBJE RELN>  たい-DESIRE
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  が-MODERATE]
                  [OBJE [[RELN  思う-1]
                        [OBJE [[RELN  たい-DESIRE]
                              [OBJE ?prp]
                              ?rest1]]
                        ?rest2]]
                  ?rest3]]
            [PRAG ?prag]
            ?rests]
      out= [[CAT INFORM]
             [MOD WANT]
             [TPC ?prag]
             [PRP ?prp]]
   end")
;; the deference from the above is CONT feature
(rws:defrwschema2 inf-2-4 inf WANT
  "on <SEM OBJE CONT RELN>  たい-DESIRE
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  が-MODERATE]
                  [OBJE [[RELN  思う-1]
                        [CONT  [[RELN  たい-DESIRE]
                              [OBJE ?prp]
                              ?rest1]]
                        ?rest2]]
                  ?rest3]]
            [PRAG ?prag]
            ?rests]
      out= [[CAT INFORM]
             [MOD WANT]
             [TPC ?prag]
             [PRP ?prp]]
   end")
(rws:defrwschema2 inf-2-6 inf WANT
  "on <SEM OBJE OBJE RESTR RELN>  たい-DESIRE
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  が-MODERATE]
                  [OBJE [[RELN  ある-1]
```

```
                              [OBJE [[RESTR [[RELN  たい-DESIRE]
                                             [OBJE ?prp]
                                             ?rest0]]
                                     ?rest1]]
                              ?rest2]]
                        ?rest3]]
                  [PRAG ?prag]
                  ?rests]
             out= [[CAT INFORM]
                   [MOD WANT]
                   [TPC ?prag]
                   [PRP ?prp]]
        end")

;; 2. しなければなりません
(rws:defrwschema2 inf-3-1 inf must
   "on <SEM OBJE RELN>  なくてはいけない-MUST
    in :Phase :intention :Type :general
       in= [[SEM [[RELN  が-MODERATE]
                  [OBJE [[RELN  なくてはいけない-MUST]
                         [OBJE ?prp]
                         ?rest1]]
                  ?rest2]]
            [PRAG  ?prag]
            ?rests]
       out= [[CAT INFORM]
             [MOD MUST]
             [PRP ?prp]
             [TPC ?prag]]
    end")
(rws:defrwschema2 inf-3-2 inf must
   "on <SEM RELN>  なくてはいけない-MUST
    in :Phase :intention :Type :general
       in= [[SEM [[RELN  なくてはいけない-MUST]
                  [OBJE ?prp]
                  ?rest2]]
            [PRAG  ?prag]
            ?rests]
       out= [[CAT INFORM]
             [MOD MUST]
             [PRP ?prp]
             [TPC ?prag]]
    end")

;;
;; REJECT
;;
;;; 1. できません
(rws:defrwschema2 rja-1 rja dekinai
   "on <SEM OBJE RELN>  できる-POSSIBLE
    in :Phase :intention :Type :general
       in= [[SEM [[RELN  NEGATE]
                  [OBJE [[RELN  できる-POSSIBLE]
                         [OBJE ?prp]
                         ?rest1]]
                  ?rest2]]
            [PRAG  ?prag]
            ?rests]
       out= [[CAT REJECT-ACTION]
             [PRP ?prp]
             [TPC ?prag]]
    end")

;;; OTHERS ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(rws:defrwschema2 etc-1 etc compound-s
```

```
  "on <SEM RELN>  連用形接続
   in :Phase :intention :Type :general
      in= [[SEM [[RELN  連用形接続]
                 [ARG1 ?arg1]
                 [ARG2 ?arg2]
                 ?rest1]]
           [PRAG  ?prag]
           ?rests]
      set ?new1 to [[SEM ?arg1] [PRAG ?prag]]
      set ?new2 to [[SEM ?arg2] [PRAG ?prag]]
      out= [[CAT COMPOUND-CONT]
            [ARG-1 ?new1]
            [ARG-2 ?new2]
            [TPC  ?prag]]
   end")
```

C.3 デフォールト規則

```
;;
;; Type :DEFAULT
;;
;;; OFFER-ACTION ; ACT します
(rws:defrwschema2 ofa-1 ofa simasu
  "on <SEM AGEN LABEL> SPEAKER
   in :Phase :intention :Type :default
      in= [[SEM [[ASPT  UNRL]
                 [AGEN [[LABEL  SPEAKER]]]
 ?rest1]]
           [PRAG  ?prag]
           ?rests]
      set ?prp to ?input.sem
      out= [[CAT OFFER-ACTION]
            [PRP ?prp]
    [TPC ?prag]]
   end")

;;; INFORM (SIFT)
(rws:defrwschema2 inf-1-1 inf inform
  "on <SEM RELN>  :unspecified
   in :Phase :intention :Type :default
      in= [[SEM ?prp]
           [PRAG ?prag]
           ?rests]
      out= [[CAT INFORM]
            [TPC ?prag]
            [PRP ?prp]]
   end")
(rws:defrwschema2 inf-1-2 inf inform
  "on <SEM RELN>  が-MODERATE
   in :Phase :intention :Type :default
      in= [[SEM [[RELN  が-MODERATE]
                 [OBJE  ?prp]
                 ?rest2]]
           [PRAG ?prag]
           ?rests]
      if input.SEM has ASPT then
        set ?aspt to input.SEM.ASPT
        add {[ASPT ?aspt]} to ?prp
      endif
      set ?output to [[SEM  ?prp]
                      [PRAG ?prag]]
```

```
      => ?output
      out= ?output
   end")

;; NEGATE ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(rws:defrwschema2 ngm-1 ngt NEGATE
  "on <SEM RELN> NEGATE
   in :Phase :intention :Type :default
      in= [[SEM [[RELN  NEGATE]
                 [OBJE  ?prp]
                 ?rest]]
           [PRAG ?prag]
           ?rests]
      if input.SEM has ASPT then
        set ?aspt to input.SEM.ASPT
        add {[ASPT ?aspt]} to ?prp
      endif
      set ?output to [[SEM  ?prp]
                      [PRAG ?prag]]
      => ?output
      add {[MOD NEGATE]} to ?output
      out= ?output
   end")

;;;; END RWS RULES ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

付録 D

命題格要素データベース

現在定義している命題格要素データの一覧を示す。

なお、述語が英語表現であるものは、一段階抽象的な命題内容を表しており、これらはプラン認識機構のプランスキーマ定義記述で利用され、プラン認識の単一化の際に参照される。

注) このデータベースは不完全である。今後いっそうの整備が望まれる。

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(IS             OBJE IDEN)
(だ-IDENTICAL   OBJE IDEN)
(だ-STATEMENT   OBJE IDEN)
(GO             AGEN DEST)
(行く-1         AGEN DEST)
(行く-3         AGEN DEST)
(PARTICIPATE    AGEN LOCT)
(参加する-1     AGEN LOCT)
(REGISTER       AGEN OBJE)
(申し込む-1     AGEN RECP OBJE)
(申し込む-2     AGEN LOCT OBJE)
(SEND           AGEN RECP OBJE)
(送る-1         AGEN RECP OBJE)
(同封する-1     AGEN OBJE)
(WRITE          AGEN OBJE LOCT)
(書く-1         AGEN OBJE LOCT)
(記入する-1     AGEN OBJE LOCT)
(PAY            AGEN RECP OBJE)
(支払う-1       AGEN RECP OBJE)
(振り込む-1     AGEN DEST OBJE)
(HAVE           AGEN OBJE)
(持つ-1         AGEN OBJE)
(ある-1         OBJE)
(DO             AGEN OBJE)
(する-1         AGEN OBJE)
(行なう-1       AGEN OBJE)
(よい-1         OBJE)
(いい-1         OBJE)
(PRESENT        AGEN OBJE LOCT)
(発表する-1     AGEN OBJE LOCT)
(HOLD           AGEN OBJE)
(含む-1         OBJE LOCT)
(用意する-1     AGEN RECP OBJE)
```

```
(NEED           OBJE)
(要る -1        OBJE)
(必要だ -1      OBJE)
(する -2        AGEN OBJE RESL)
(する -4        AGEN RECP OBJE)
(行く -2        OBJE DEST)
(かかる -1      OBJE DEGR)
(なる -2        OBJE RESL)
(分かる -1      EXPR OBJE)
(教える -1      AGEN RECP OBJE RANG)
(教える -2      AGEN OBJE RANG)
(教える -3      AGEN RANG)
(近い -1        OBJE)
(見る -1        AGEN OBJE)
(取る -1        AGEN LOCT OBJE)
(尋ねる -1      AGEN RECP RANG)
(待つ -1        AGEN OBJE)
(泊まる -1      AGEN LOCT)
(聞く -1        AGEN OBJE)
(聞く -3        AGEN RECP OBJE)
(開催する -1    AGEN OBJE)
(紹介する -1    AGEN OBJE RECP)
(利用する -1    AGEN PURP OBJE)
(いる -1        OBJE LOCT)
(載る -1        OBJE LOCT)
(知らせる -1    AGEN RECP OBJE)
(調べる -1      AGEN OBJE)
(答える -1      AGEN OBJE)
(払い戻す -1    AGEN OBJE)
(取り消す -1    AGEN OBJE)
(見学する -1    AGEN OBJE TLOC)
(質問する -1    AGEN RECP RANG)
(問題ある -1    LOCT)
(予約する -1    AGEN OBJE)
(OPEN-DIALOGUE)
(CLOSE-DIALOGUE)


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

付録 E

実験結果 (会話 A,B,1–10)

ここでは、紙面の都合上、最終発話表現リストのみを掲載する。より細かい情報 (中間素性構造) は、以下のファイルを参照されたい:

- “as21:/home/yamaoka/System/III/Log/CATE-test-ALL”

```
;;;
;;; 会話-A
;;;
DA-1 : "もしもし"
 (((GREETING-OPEN        SP1 SP2 OPENING       (OPEN-DIALOGUE))))

DA-2 : "そちらは会議事務局ですか"
 (((CONFIRM-VALUE        SP1 SP2 そちら-1      (だ-IDENTICAL そちら-1 会議事務局-1))))

DA-3 : "はい"
 (((GREETING-OPEN        SP2 SP1 OPENING       (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP2 SP1 ?TPCDA-3      ?PRPDA-3)))

DA-4 : "そうです"
 (((AFFIRMATIVE          SP2 SP1 ?TPCDA-4      ?PRPDA-4)))

DA-5 : "会議に申込みたいのですが"
 (((INFORM-WANT          SP1 SP2 ?TPCDA-5      (申し込む-1 SP1 ?RECPDA-5 会議-1))))

DA-6 : "登録用紙は既にお持ちでしょうか"
 (((CONFIRM-STATEMENT    SP2 SP1 登録用紙-1    (持つ-1 SP1 登録用紙-1))))

DA-7 : "いいえ"
 (((NEGATIVE             SP1 SP2 ?TPCDA-7      ?PRPDA-7)))

DA-8 : "まだです"
```

```
 (((NEGATIVE            SP1 SP2 ?TPCDA-8     ?PRPDA-8)))

DA-9 : "分かりました"
 (((CONFIRMATION        SP2 SP1 ?TPCDA-9     ?PRPDA-9))
  ((ACCEPT-ACTION       SP2 SP1 ?TPCDA-9     ?PRPDA-9)))

DA-10 : "それでは登録用紙をお送り致します"
 (((OFFER-ACTION        SP2 SP1 ?TPCDA-10    (送る-1 SP2 SP1 登録用紙-1))))

DA-11 : "ご住所とお名前をお願いします"
 (((ASK-VALUE           SP2 SP1 名前-1       ?PRPDA-11)
   (ASK-VALUE           SP2 SP1 住所-1       ?PRPDA-11)))

DA-12 : "住所は大阪市北区茶屋町二十三です"
 (((INFORM-VALUE        SP1 SP2 住所-1       (だ-IDENTICAL 住所-1 ADDRESS))))

DA-13 : "名前は鈴木真弓です"
 (((INFORM-VALUE        SP1 SP2 名前-1       (だ-IDENTICAL 名前-1 NAME))))

DA-14 : "分かりました"
 (((CONFIRMATION        SP2 SP1 ?TPCDA-14    ?PRPDA-14))
  ((ACCEPT-ACTION       SP2 SP1 ?TPCDA-14    ?PRPDA-14)))

DA-15 : "登録用紙を至急送らせていただきます"
 (((OFFER-ACTION        SP2 SP1 ?TPCDA-15    (送る-1 SP2 SP1 登録用紙-1))))

DA-16 : "分からない点がございましたらいつでもお聞き下さい"
 (((REQUEST-ACTION      SP2 SP1 ?TPCDA-16    (聞く-1 SP1 ?OBJEDA-16)))
  ((INFORM-ACTION       SP2 SP1 ?TPCDA-16    (聞く-1 SP1 ?OBJEDA-16))))

DA-17 : "有難うございます"
 (((GREETING-CLOSE      SP1 SP2 CLOSING      (CLOSE-DIALOGUE)))
  ((ACCEPT-OFFER        SP1 SP2 ?TPCDA-17    ?PRPDA-17)))

DA-18 : "それでは失礼します"
 (((GREETING-CLOSE      SP1 SP2 CLOSING      (CLOSE-DIALOGUE))))

DA-19 : "どうも失礼致します"
 (((GREETING-CLOSE      SP2 SP1 CLOSING      (CLOSE-DIALOGUE))))
```

```
;;;
;;; 会話-B
;;;
DB-1 : "もしもし"
 (((GREETING-OPEN        SP2 SP1 OPENING      (OPEN-DIALOGUE))))

DB-2 : "こちらは会議事務局です"
 (((INFORM-VALUE         SP2 SP1 SP2          (だ-IDENTICAL SP2 会議事務局-1))))

DB-3 : "会議に参加したいのですが"
 (((INFORM-WANT          SP1 SP2 ?TPCDB-3     (参加する-1 SP1 ?LOCTDB-3))))

DB-4 : "どうすればよろしいですか"
 (((ASK-ACTION           SP1 SP2 ?TPCDB-4     (する-1 ?AGENDB-4 ?OBJEDB-4))))

DB-5 : "まず登録用紙で手続きをしていただかなくてはなりませんが"
 (((INFORM-STATEMENT     SP2 SP1 ?TPCDB-5     (する-1 SP1 手続き-1)))
  ((INFORM-VALUE         SP2 SP1 ?TPCDB-5     (する-1 SP1 手続き-1)))
  ((INFORM-ACTION        SP2 SP1 ?TPCDB-5     (する-1 SP1 手続き-1))))

DB-6 : "もう登録用紙はお持ちでしょうか"
 (((CONFIRM-STATEMENT    SP2 SP1 登録用紙-1   (持つ-1 SP1 登録用紙-1))))

DB-7 : "まだです"
 (((NEGATIVE             SP1 SP2 ?TPCDB-7     ?PRPDB-7)))

DB-8 : "用紙を送ってください"
 (((REQUEST-ACTION       SP1 SP2 ?TPCDB-8     (送る-1 SP2 SP1 用紙-1)))
  ((INFORM-ACTION        SP1 SP2 ?TPCDB-8     (送る-1 SP2 SP1 用紙-1))))

DB-9 : "ではご住所とお名前をお願いします"
 (((ASK-VALUE            SP2 SP1 名前-1       ?PRPDB-9)
   (ASK-VALUE            SP2 SP1 住所-1       ?PRPDB-9)))

DB-10 : "住所は大阪市東区徳井町一の二です"
 (((INFORM-VALUE         SP1 SP2 住所-1       (だ-IDENTICAL 住所-1 ADDRESS)))
  ((INFORM-VALUE         SP1 SP2 住所-1       (だ-IDENTICAL 住所-1 二-1))))

DB-11 : "名前は清水太郎です"
 (((INFORM-VALUE         SP1 SP2 名前-1       (だ-IDENTICAL 名前-1 NAME))))

DB-12 : "分かりました"
 (((CONFIRMATION         SP2 SP1 ?TPCDB-12    ?PRPDB-12))
  ((ACCEPT-ACTION        SP2 SP1 ?TPCDB-12    ?PRPDB-12)))
```

DB-13 : "参加料は要るのでしょうか"
(((CONFIRM-STATEMENT SP1 SP2 参加料-1 (要る-1 参加料-1))))

DB-14 : "はい"
(((GREETING-OPEN SP2 SP1 OPENING (OPEN-DIALOGUE)))
((AFFIRMATIVE SP2 SP1 ?TPCDB-14 ?PRPDB-14)))

DB-15 : "登録費としてお一人三万五千円が必要です"
(((INFORM-STATEMENT SP2 SP1 ?TPCDB-15 (必要だ-1 三万五千-1))))

DB-16 : "そうですか"
(((CONFIRMATION SP1 SP2 ?TPCDB-16 ?PRPDB-16)))

DB-17 : "どうも有難うございました"
(((GREETING-CLOSE SP1 SP2 CLOSING (CLOSE-DIALOGUE)))
((ACCEPT-OFFER SP1 SP2 ?TPCDB-17 ?PRPDB-17)))

DB-18 : "失礼致します"
(((GREETING-CLOSE SP2 SP1 CLOSING (CLOSE-DIALOGUE))))

```
;;;
;;; 会話-1
;;;
D1-1 : "もしもし"
 (((GREETING-OPEN       SP1 SP2 OPENING      (OPEN-DIALOGUE))))

D1-2 : "そちらは会議事務局ですか"
 (((CONFIRM-VALUE       SP1 SP2 そちら-1     (だ-IDENTICAL そちら-1 会議事務局-1))))

D1-3 : "はい"
 (((GREETING-OPEN       SP2 SP1 OPENING      (OPEN-DIALOGUE)))
  ((AFFIRMATIVE         SP2 SP1 ?TPCD1-3     ?PRPD1-3)))

D1-4 : "そうです"
 (((AFFIRMATIVE         SP2 SP1 ?TPCD1-4     ?PRPD1-4)))

D1-5 : "どのようなご用件でしょうか"
 (((ASK-STATEMENT       SP2 SP1 ?TPCD1-5     (だ-IDENTICAL ?OBJED1-5 用件-1))))

D1-6 : "会議に申し込みたいのですが"
 (((INFORM-WANT         SP1 SP2 ?TPCD1-6     (申し込む-1 SP1 ?RECPD1-6 ?OBJED1-6)))
  ((INFORM-WANT         SP1 SP2 ?TPCD1-6     (申し込む-2 ?AGEND1-6 会議-1 ?OBJED1-6)))
  ((INFORM-WANT         SP1 SP2 ?TPCD1-6     (申し込む-1 ?AGEND1-6 ?RECPD1-6 ?OBJED1-6))))

D1-7 : "どのような手続きをすればよろしいのでしょうか"
 (((ASK-ACTION          SP1 SP2 ?TPCD1-7     (する-1 ?AGEND1-7 手続き-1)))
  ((ASK-ACTION          SP1 SP2 ?TPCD1-7     (する-2 ?AGEND1-7 手続き-1 ?RESLD1-7)))
  ((ASK-ACTION          SP1 SP2 ?TPCD1-7     (する-4 ?AGEND1-7 ?RECPD1-7 手続き-1))))

D1-8 : "登録用紙で手続きをして下さい"
 (((REQUEST-ACTION      SP2 SP1 ?TPCD1-8     (する-1 SP1 手続き-1)))
  ((INFORM-ACTION       SP2 SP1 ?TPCD1-8     (する-1 SP1 手続き-1)))
  ((REQUEST-ACTION      SP2 SP1 ?TPCD1-8     (する-2 SP1 手続き-1 ?RESLD1-8)))
  ((INFORM-ACTION       SP2 SP1 ?TPCD1-8     (する-2 SP1 手続き-1 ?RESLD1-8)))
  ((ASK-ACTION          SP2 SP1 手続き-1     ?PRPD1-8)))

D1-9 : "登録用紙は既にお持ちでしょうか"
 (((CONFIRM-STATEMENT   SP2 SP1 登録用紙-1   (持つ-1 ?AGEND1-9 登録用紙-1))))

D1-10 : "いいえ"
 (((NEGATIVE            SP1 SP2 ?TPCD1-10    ?PRPD1-10)))

D1-11 : "まだです"
 (((NEGATIVE            SP1 SP2 ?TPCD1-11    ?PRPD1-11)))
```

```
D1-12 : "分かりました"
 (((CONFIRMATION        SP2 SP1 ?TPCD1-12    ?PRPD1-12))
  ((ACCEPT-ACTION       SP2 SP1 ?TPCD1-12    ?PRPD1-12)))

D1-13 : "それでは登録用紙をお送り致します"
 (((OFFER-ACTION        SP2 SP1 ?TPCD1-13    (送る-1 SP2 SP1 登録用紙-1)))
  ((OFFER-ACTION        SP2 SP1 ?TPCD1-13    (送る-1 SP2 ?RECPD1-13 登録用紙-1))))

D1-14 : "ご住所とお名前をお願いします"
 (((ASK-VALUE           SP2 SP1 名前-1       ?PRPD1-14)
   (ASK-VALUE           SP2 SP1 住所-1       ?PRPD1-14)))

D1-15 : "住所は大阪市北区茶屋町二十三です"
 (((INFORM-VALUE        SP1 SP2 住所-1       (だ-IDENTICAL 住所-1 ADDRESS))))

D1-16 : "名前は鈴木真弓です"
 (((INFORM-VALUE        SP1 SP2 名前-1       (だ-IDENTICAL 名前-1 NAME))))

D1-17 : "分かりました"
 (((CONFIRMATION        SP2 SP1 ?TPCD1-17    ?PRPD1-17))
  ((ACCEPT-ACTION       SP2 SP1 ?TPCD1-17    ?PRPD1-17)))

D1-18 : "登録用紙を至急送らせて頂きます"
 (((OFFER-ACTION        SP2 SP1 ?TPCD1-18    (送る-1 SP2 ?RECPD1-18 登録用紙-1))))

D1-19 : "よろしくお願いします"
 (((REQUEST-ACTION      SP1 SP2 ?TPCD1-19    ?PRPD1-19))
  ((ACCEPT-OFFER        SP1 SP2 ?TPCD1-19    ?PRPD1-19)))

D1-20 : "それでは失礼します"
 (((GREETING-CLOSE      SP1 SP2 CLOSING      (CLOSE-DIALOGUE))))
```

```
;;;
;;; 会話-2
;;;
D2-1 : "はい"
 (((GREETING-OPEN        SP2 SP1 OPENING      (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP2 SP1 ?TPCD2-1     ?PRPD2-1)))

D2-2 : "こちらは会議事務局です"
 (((INFORM-VALUE         SP2 SP1 SP2          (だ-IDENTICAL SP2 会議事務局-1))))

D2-3 : "会議の参加料について教えて頂きたいのですが"
 (((ASK-VALUE            SP1 SP2 参加料-1     ?PRPD2-3))
  ((REQUEST-ACTION       SP1 SP2 ?TPCD2-3     (教える-3 ?AGEND2-3 参加料-1)))
  ((INFORM-ACTION        SP1 SP2 ?TPCD2-3     (教える-3 ?AGEND2-3 参加料-1))))

D2-4 : "いま会議に申し込めば参加料はいくらですか"
 (((ASK-VALUE            SP1 SP2 ?TPCD2-4     (だ-IDENTICAL 参加料-1 ?IDEND2-4))))

D2-5 : "はい"
 (((GREETING-OPEN        SP2 SP1 OPENING      (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP2 SP1 ?TPCD2-5     ?PRPD2-5)))

D2-6 : "参加料は現在お一人三万五千円です"
 (((INFORM-VALUE         SP2 SP1 参加料-1     (だ-IDENTICAL 参加料-1 三万五千-1))))

D2-7 : "来月お申込みになりますと四万円です"
 (((INFORM-VALUE         SP2 SP1 ?TPCD2-7     (だ-IDENTICAL ?OBJED2-7 四万-1))))

D2-8 : "参加料には予稿集代と歓迎会費が含まれています"
 (((INFORM-STATEMENT     SP2 SP1 ?TPCD2-8     (含む-1 歓迎会費-1 参加料-1))
   (INFORM-STATEMENT     SP2 SP1 ?TPCD2-8     (含む-1 予稿集代-1 参加料-1))))

D2-9 : "わたしは情報処理学会の会員なのですが"
 (((INFORM-VALUE         SP1 SP2 SP1          (だ-IDENTICAL SP1 会員-1))))

D2-10 : "参加料の割引はないのですか"
 (((CONFIRM-VALUE        SP1 SP2 割引-1       ?VALD2-10))
  ((CONFIRM-STATEMENT    SP1 SP2 割引-1       ?VALD2-10))
  ((CONFIRM-ACTION       SP1 SP2 割引-1       ?VALD2-10)))

D2-11 : "今回は割引を行なっておりません"
 (((INFORM-STATEMENT     SP2 SP1 今回-1       (行なう-1 ?AGEND2-11 割引-1)))
  ((NEGATIVE             SP2 SP1 今回-1       (行なう-1 ?AGEND2-11 割引-1))))
```

D2-12 : "そうですか"
(((CONFIRMATION SP1 SP2 ?TPCD2-12 ?PRPD2-12)))

D2-13 : "参加料はどのようにお支払いしたらよいのですか"
(((ASK-ACTION SP1 SP2 参加料-1 (支払う-1 ?AGEND2-13 ?RECPD2-13 参加料
-1)))
((ASK-STATEMENT SP1 SP2 参加料-1 (よい-1 ?OBJED2-13)))
((ASK-ACTION SP1 SP2 参加料-1 (よい-1 ?OBJED2-13))))

D2-14 : "参加料は銀行振り込みです"
(((INFORM-ACTION SP2 SP1 参加料-1 (だ-IDENTICAL 参加料-1 銀行振り込み-1)))
((INFORM-VALUE SP2 SP1 参加料-1 (だ-IDENTICAL 参加料-1 銀行振り込み-1))))

D2-15 : "案内書に記載されている口座番号に振り込んで下さい"
(((REQUEST-ACTION SP2 SP1 ?TPCD2-15 (振り込む-1 SP1 口座番号-1 ?OBJED2-15)))
((INFORM-ACTION SP2 SP1 ?TPCD2-15 (振り込む-1 SP1 口座番号-1 ?OBJED2-15))))

D2-16 : "また期限は今年いっぱいです"
(((INFORM-VALUE SP2 SP1 ?TPCD2-16 (だ-IDENTICAL 期限-1 今年-1))))

D2-17 : "分かりました"
(((CONFIRMATION SP1 SP2 ?TPCD2-17 ?PRPD2-17))
((ACCEPT-ACTION SP1 SP2 ?TPCD2-17 ?PRPD2-17)))

D2-18 : "どうもありがとうございました"
(((GREETING-CLOSE SP1 SP2 CLOSING (CLOSE-DIALOGUE)))
((ACCEPT-OFFER SP1 SP2 ?TPCD2-18 ?PRPD2-18)))

D2-19 : "どういたしまして"
(((GREETING-CLOSE SP2 SP1 CLOSING (CLOSE-DIALOGUE))))

D2-20 : "分からない点がございましたらいつでもお聞き下さい"
(((REQUEST-ACTION SP2 SP1 ?TPCD2-20 (聞く-1 SP1 ?OBJED2-20)))
((INFORM-ACTION SP2 SP1 ?TPCD2-20 (聞く-1 SP1 ?OBJED2-20))))

D2-21 : "失礼致しました"
(((GREETING-CLOSE SP2 SP1 CLOSING (CLOSE-DIALOGUE))))

```
;;;
;;; 会話-3
;;;
D3-1 : "はい"
 (((GREETING-OPEN        SP2 SP1 OPENING       (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP2 SP1 ?TPCD3-1      ?PRPD3-1)))

D3-2 : "こちらは会議事務局です"
 (((INFORM-VALUE         SP2 SP1 SP2           (だ-IDENTICAL SP2 会議事務局-1))))

D3-3 : "会議に論文を発表したいと思っているのですが"
 (((INFORM-WANT          SP1 SP2 ?TPCD3-3      (発表する-1 ?AGEND3-3 論文-1 会議-1)))
  ((INFORM-WANT          SP1 SP2 ?TPCD3-3      (発表する-1 ?AGEND3-3 論文-1 ?LOCTD3-3))))

D3-4 : "会議の内容について教えて下さい"
 (((ASK-VALUE            SP1 SP2 内容-1        ?PRPD3-4))
  ((REQUEST-ACTION       SP1 SP2 ?TPCD3-4      (教える-3 SP2 内容-1)))
  ((INFORM-ACTION        SP1 SP2 ?TPCD3-4      (教える-3 SP2 内容-1))))

D3-5 : "今回の会議は通訳電話に関連する広範な研究分野を含んでいます"
 (((INFORM-STATEMENT     SP2 SP1 今回-1        (含む-1 研究分野-1 今回-1))))

D3-6 : "言語学や心理学を専攻する方にも参加して頂く予定です"
 (((INFORM-VALUE         SP2 SP1 ?TPCD3-6      (だ-IDENTICAL ?OBJED3-6 予定-1))))

D3-7 : "分かりました"
 (((CONFIRMATION         SP1 SP2 ?TPCD3-7      ?PRPD3-7))
  ((ACCEPT-ACTION        SP1 SP2 ?TPCD3-7      ?PRPD3-7)))

D3-8 : "ところで会議での公式言語は何ですか"
 (((ASK-VALUE            SP1 SP2 ?TPCD3-8      (だ-IDENTICAL 公式言語-1 ?IDEND3-8))))

D3-9 : "英語と日本語です"
 (((INFORM-VALUE         SP2 SP1 ?TPCD3-9      (だ-IDENTICAL ?OBJED3-9 日本語-1))
   (INFORM-VALUE         SP2 SP1 ?TPCD3-9      (だ-IDENTICAL ?OBJED3-9 英語-1))))

D3-10 : "わたしは日本語が全然分からないのですが"
 (((NEGATIVE             SP1 SP2 SP1           (分かる-1 SP1 日本語-1))))

D3-11 : "発表が日本語で行なわれる場合英語への同時通訳はあるのですか"
 (((CONFIRM-STATEMENT    SP1 SP2 ?TPCD3-11     (ある-1 同時通訳-1))))

D3-12 : "はい"
 (((GREETING-OPEN        SP2 SP1 OPENING       (OPEN-DIALOGUE)))
```

```
  ((AFFIRMATIVE          SP2 SP1 ?TPCD3-12    ?PRPD3-12)))

D3-13 : "英語への同時通訳を用意しております"
 (((INFORM-STATEMENT     SP2 SP1 ?TPCD3-13    (用意する-1 SP2 ?RECPD3-13 同時通訳-1))))

D3-14 : "分かりました"
 (((CONFIRMATION         SP1 SP2 ?TPCD3-14    ?PRPD3-14))
  ((ACCEPT-ACTION        SP1 SP2 ?TPCD3-14    ?PRPD3-14)))

D3-15 : "どうもありがとうございました"
 (((GREETING-CLOSE       SP1 SP2 CLOSING      (CLOSE-DIALOGUE)))
  ((ACCEPT-OFFER         SP1 SP2 ?TPCD3-15    ?PRPD3-15)))

D3-16 : "さようなら"
 (((GREETING-CLOSE       SP1 SP2 CLOSING      (CLOSE-DIALOGUE))))
```

```
;;;
;;; 会話-4
;;;
D4-1 : "こちらは会議事務局です"
 (((INFORM-VALUE        SP2 SP1 SP2          (だ-IDENTICAL SP2 会議事務局-1))))

D4-2 : "会議について詳しいことを教えて下さい"
 (((ASK-VALUE           SP1 SP2 会議-1       ?PRPD4-2)))

D4-3 : "会議の案内書はお持ちですか"
 (((CONFIRM-STATEMENT   SP2 SP1 案内書-1     (持つ-1 ?AGEND4-3 案内書-1))))

D4-4 : "いいえ"
 (((NEGATIVE            SP1 SP2 ?TPCD4-4     ?PRPD4-4)))

D4-5 : "持っていません"
 (((INFORM-STATEMENT    SP1 SP2 ?TPCD4-5     (持つ-1 ?AGEND4-5 ?OBJED4-5)))
  ((NEGATIVE            SP1 SP2 ?TPCD4-5     (持つ-1 ?AGEND4-5 ?OBJED4-5))))

D4-6 : "そうですか"
 (((CONFIRMATION        SP2 SP1 ?TPCD4-6     ?PRPD4-6)))

D4-7 : "会議は八月二十二日から二十五日まで京都国際会議場で開催されます"
 (((OFFER-ACTION        SP2 SP1 会議-1       (開催する-1 SP2 会議-1)))
  ((INFORM-ACTION       SP2 SP1 会議-1       (開催する-1 ?AGEND4-7 会議-1))))

D4-8 : "参加料は四万円です"
 (((INFORM-VALUE        SP2 SP1 参加料-1     (だ-IDENTICAL 参加料-1 四万-1))))

D4-9 : "発表を希望されるのでしたら三月二十日までに要約を提出して下さい"
 (((UNKNOWN             SP2 SP1 ?TPCD4-9     ?PRPD4-9)))

D4-10 : "会議の案内書をお送り致しますのでそれをご覧下さい"
 (((REQUEST-ACTION      SP2 SP1 ?TPCD4-10    (見る-1 SP1 それ-1)))
  ((INFORM-ACTION       SP2 SP1 ?TPCD4-10    (見る-1 SP1 それ-1))))

D4-11 : "失礼ですがお名前とご住所をお願いします"
 (((ASK-VALUE           SP2 SP1 住所-1       ?PRPD4-11)
   (ASK-VALUE           SP2 SP1 名前-1       ?PRPD4-11)))

D4-12 : "アダムスミスです"
 (((INFORM-VALUE        SP1 SP2 ?TPCD4-12    (だ-IDENTICAL ?OBJED4-12 NAME))))

D4-13 : "住所は大阪市東区玉造二丁目二十七の七です"
```

```
 (((INFORM-VALUE        SP1 SP2 住所-1       (だ-IDENTICAL 住所-1 ADDRESS))))

D4-14 : "分かりました"
 (((CONFIRMATION        SP2 SP1 ?TPCD4-14    ?PRPD4-14))
  ((ACCEPT-ACTION       SP2 SP1 ?TPCD4-14    ?PRPD4-14)))

D4-15 : "電話番号もお聞きしたいのですが"
 (((INFORM-WANT         SP2 SP1 電話番号-1   (聞く-3 SP2 SP1 電話番号-1)))
  ((ASK-VALUE           SP2 SP1 電話番号-1   ?PRPD4-15)))

D4-16 : "はい"
 (((GREETING-OPEN       SP1 SP2 OPENING      (OPEN-DIALOGUE)))
  ((AFFIRMATIVE         SP1 SP2 ?TPCD4-16    ?PRPD4-16)))

D4-17 : "三七二の八〇一八です"
 (((INFORM-VALUE        SP1 SP2 ?TPCD4-17    (だ-IDENTICAL ?OBJED4-17 八〇一八-1))))

D4-18 : "三七二の八〇一八でございますね"
 (((CONFIRM-VALUE       SP2 SP1 ?TPCD4-18    (だ-IDENTICAL ?OBJED4-18 八〇一八-1)))
  ((CONFIRM-STATEMENT   SP2 SP1 ?TPCD4-18    (ある-1 ?OBJED4-18))))

D4-19 : "はい"
 (((GREETING-OPEN       SP1 SP2 OPENING      (OPEN-DIALOGUE)))
  ((AFFIRMATIVE         SP1 SP2 ?TPCD4-19    ?PRPD4-19)))

D4-20 : "そうです"
 (((AFFIRMATIVE         SP1 SP2 ?TPCD4-20    ?PRPD4-20)))

D4-21 : "それではよろしくお願いします"
 (((REQUEST-ACTION      SP1 SP2 ?TPCD4-21    ?PRPD4-21))
  ((ACCEPT-OFFER        SP1 SP2 ?TPCD4-21    ?PRPD4-21)))

D4-22 : "失礼します"
 (((GREETING-CLOSE      SP1 SP2 CLOSING      (CLOSE-DIALOGUE))))
```

```
;;;
;;; 会話-5
;;;
D5-1 : "はい"
 (((GREETING-OPEN        SP2 SP1 OPENING       (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP2 SP1 ?TPCD5-1      ?PRPD5-1)))

D5-2 : "こちらは会議事務局でございます"
 (((INFORM-VALUE         SP2 SP1 SP2           (だ-IDENTICAL SP2 会議事務局-1)))
  ((INFORM-STATEMENT     SP2 SP1 SP2           (ある-1 SP2))))

D5-3 : "ちょっとお願いがあるのですが"
 (((INFORM-STATEMENT     SP1 SP2 ?TPCD5-3      (ある-1 お願い-1))))

D5-4 : "私は会議に申込みをした者です"
 (((INFORM-VALUE         SP1 SP2 SP1           (だ-IDENTICAL SP1 者-1))))

D5-5 : "参加を取り消したいのですが"
 (((INFORM-WANT          SP1 SP2 ?TPCD5-5      (取り消す-1 SP1 参加-1))))

D5-6 : "お名前をお伺いできますでしょうか"
 (((CONFIRM-ACTION       SP2 SP1 ?TPCD5-6      (聞く-3 ?AGEND5-6 ?RECPD5-6 名前-1)))
  ((ASK-VALUE            SP2 SP1 名前-1        ?PRPD5-6)))

D5-7 : "はい"
 (((GREETING-OPEN        SP1 SP2 OPENING       (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP1 SP2 ?TPCD5-7      ?PRPD5-7)))

D5-8 : "ベル研のジムワイベルです"
 (((INFORM-VALUE         SP1 SP2 ?TPCD5-8      (だ-IDENTICAL ?OBJED5-8 NAME))))

D5-9 : "既に登録料の八万五千円を振り込まれておられますね"
 (((CONFIRM-STATEMENT    SP2 SP1 ?TPCD5-9      (振り込む-1 ?AGEND5-9 ?DESTD5-9 八万五
千-1))))

D5-10 : "はい"
 (((GREETING-OPEN        SP1 SP2 OPENING       (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP1 SP2 ?TPCD5-10     ?PRPD5-10)))

D5-11 : "そうです"
 (((AFFIRMATIVE          SP1 SP2 ?TPCD5-11     ?PRPD5-11)))

D5-12 : "登録料を払い戻して頂けますか"
 (((REQUEST-ACTION       SP1 SP2 ?TPCD5-12     (払い戻す-1 ?AGEND5-12 登録料-1)))
```

```
  ((INFORM-ACTION        SP1 SP2 ?TPCD5-12    (払い戻す-1 ?AGEND5-12 登録料-1))))

D5-13 : "お気の毒ですができません"
 (((REJECT-ACTION        SP2 SP1 ?TPCD5-13    ?PRPD5-13)))

D5-14 : "案内書にも書いていますが"
 (((INFORM-STATEMENT     SP2 SP1 ?TPCD5-14    (書く-1 ?AGEND5-14 ?OBJED5-14 案内書-1))))

D5-15 : "九月二十七日以後の取り消しに対する払い戻しはできません"
 (((REJECT-ACTION        SP2 SP1 ?TPCD5-15    (払い戻す-1 ?AGEND5-15 ?OBJED5-15))))

D5-16 : "後日プログラムと予稿集をお送り致します"
 (((OFFER-ACTION         SP2 SP1 ?TPCD5-16    (送る-1 SP2 SP1 予稿集-1))
   (OFFER-ACTION         SP2 SP1 ?TPCD5-16    (送る-1 SP2 SP1 プログラム-1))))

D5-17 : "では誰かが私の代わりに参加することはできますか"
 (((CONFIRM-ACTION       SP1 SP2 ?TPCD5-17    (参加する-1 ?AGEND5-17 ?LOCTD5-17))))

D5-18 : "それは別に問題ありません"
 (((ACCEPT-ACTION        SP2 SP1 それ-1       ?PRPD5-18)))

D5-19 : "代理人が参加する場合はあらかじめこちらまでお知らせ下さい"
 (((REQUEST-ACTION       SP2 SP1 場合-CONDITIONAL (知らせる-1 SP1 SP2 ?OBJED5-19)))
  ((INFORM-ACTION        SP2 SP1 場合-CONDITIONAL (知らせる-1 SP1 SP2 ?OBJED5-19)))
  ((REQUEST-ACTION       SP2 SP1 場合-CONDITIONAL (知らせる-1 SP1 ?RECPD5-19 ?OBJED5-19)))
  ((INFORM-ACTION        SP2 SP1 場合-CONDITIONAL (知らせる-1 SP1 ?RECPD5-19 ?OBJED5-19)))
  ((REQUEST-ACTION       SP2 SP1 ?TPCD5-19    (知らせる-1 SP1 ?RECPD5-19 ?OBJED5-19)))
  ((INFORM-ACTION        SP2 SP1 ?TPCD5-19    (知らせる-1 SP1 ?RECPD5-19 ?OBJED5-19))))

D5-20 : "分かりました"
 (((CONFIRMATION         SP1 SP2 ?TPCD5-20    ?PRPD5-20))
  ((ACCEPT-ACTION        SP1 SP2 ?TPCD5-20    ?PRPD5-20)))

D5-21 : "代理人が決まりましたらお知らせ致します"
 (((OFFER-ACTION         SP1 SP2 ?TPCD5-21    (知らせる-1 SP1 SP2 ?OBJED5-21)))
  ((OFFER-ACTION         SP1 SP2 ?TPCD5-21    (知らせる-1 SP1 ?RECPD5-21 ?OBJED5-21))))

D5-22 : "では失礼します"
 (((GREETING-CLOSE       SP1 SP2 CLOSING      (CLOSE-DIALOGUE))))
```

```
;;;
;;; 会話-6
;;;
D6-1 : "はい"
 (((GREETING-OPEN        SP2 SP1 OPENING       (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP2 SP1 ?TPCD6-1      ?PRPD6-1)))

D6-2 : "こちらは会議事務局ですが"
 (((INFORM-VALUE         SP2 SP1 SP2           (だ-IDENTICAL SP2 会議事務局-1))))

D6-3 : "会議の間に市内観光があるそうですが"
 (((INFORM-STATEMENT     SP1 SP2 ?TPCD6-3      (ある-1 市内観光-1))))

D6-4 : "まだ参加できますか"
 (((CONFIRM-ACTION       SP1 SP2 ?TPCD6-4      (参加する-1 ?AGEND6-4 ?LOCTD6-4))))

D6-5 : "はい"
 (((GREETING-OPEN        SP2 SP1 OPENING       (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP2 SP1 ?TPCD6-5      ?PRPD6-5)))

D6-6 : "まだ参加可能です"
 (((INFORM-STATEMENT     SP2 SP1 ?TPCD6-6      (参加する-1 ?AGEND6-6 ?LOCTD6-6))))

D6-7 : "八月五日の午後に清水寺、金閣寺、龍安寺などを見学します"
 (((INFORM-ACTION        SP2 SP1 ?TPCD6-7      (見学する-1 ?AGEND6-7 龍安寺-1 TIME))
   (INFORM-ACTION        SP2 SP1 ?TPCD6-7      (見学する-1 ?AGEND6-7 金閣寺-1 TIME))))

D6-8 : "参加なさいますか"
 (((CONFIRM-ACTION       SP2 SP1 ?TPCD6-8      (参加する-1 ?AGEND6-8 ?LOCTD6-8))))

D6-9 : "参加料はいくらですか"
 (((ASK-VALUE            SP1 SP2 参加料-1      (だ-IDENTICAL 参加料-1 ?IDEND6-9))))

D6-10 : "八千円です"
 (((INFORM-VALUE         SP2 SP1 ?TPCD6-10     (だ-IDENTICAL ?OBJED6-10 八千-1))))

D6-11 : "参加料には夕食代も含まれています"
 (((INFORM-STATEMENT     SP2 SP1 ?TPCD6-11     (含む-1 夕食代-1 参加料-1))))

D6-12 : "講演者も参加されるのですか"
 (((CONFIRM-ACTION       SP1 SP2 講演者-1      (参加する-1 講演者-1 ?LOCTD6-12))))

D6-13 : "講演者の何人かは参加する予定になっています"
 (((INFORM-STATEMENT     SP2 SP1 講演者-1      (だ-STATEMENT 予定-1 ?IDEND6-13)))
```

```
  ((INFORM-VALUE        SP2 SP1 講演者-1     (だ-IDENTICAL ?OBJED6-13 予定-1)))
  ((INFORM-VALUE        SP2 SP1 講演者-1     (だ-IDENTICAL 講演者-1 予定-1))))

D6-14 : "そうですか"
 (((CONFIRMATION        SP1 SP2 ?TPCD6-14    ?PRPD6-14)))

D6-15 : "それでは参加したいと思います"
 (((INFORM-WANT         SP1 SP2 ?TPCD6-15    (参加する-1 ?AGEND6-15 ?LOCTD6-15))))

D6-16 : "ではお名前と人数をお願い致します"
 (((ASK-VALUE           SP2 SP1 人数-1       ?PRPD6-16)
   (ASK-VALUE           SP2 SP1 名前-1       ?PRPD6-16)))

D6-17 : "ケンブラウンと申します"
 (((INFORM-VALUE        SP1 SP2 ?TPCD6-17    (だ-IDENTICAL ?OBJED6-17 NAME)))
  ((INFORM-STATEMENT    SP1 SP2 ?TPCD6-17    (だ-IDENTICAL ?OBJED6-17 ?IDEND6-17))))

D6-18 : "家内と参加します"
 (((OFFER-ACTION        SP1 SP2 ?TPCD6-18    (参加する-1 SP1 ?LOCTD6-18))))

D6-19 : "集合場所は会議場の受付の前になっております"
 (((INFORM-VALUE        SP2 SP1 集合場所-1   (だ-IDENTICAL ?OBJED6-19 集合場所-1)))
  ((INFORM-STATEMENT    SP2 SP1 集合場所-1   (だ-IDENTICAL 集合場所-1 ?IDEND6-19)))
  ((INFORM-STATEMENT    SP2 SP1 集合場所-1   (だ-STATEMENT ?OBJED6-19 ?IDEND6-19)))
  ((INFORM-STATEMENT    SP2 SP1 集合場所-1   (なる-2 集合場所-1 ?RESLD6-19)))
  ((INFORM-STATEMENT    SP2 SP1 集合場所-1   (なる-2 ?OBJED6-19 集合場所-1)))
  ((INFORM-VALUE        SP2 SP1 集合場所-1   (だ-IDENTICAL 集合場所-1 前-1)))
  ((INFORM-STATEMENT    SP2 SP1 集合場所-1   (なる-2 集合場所-1 前-1))))

D6-20 : "参加料は当日集合場所でお支払い下さい"
 (((REQUEST-ACTION      SP2 SP1 参加料-1     (支払う-1 SP1 ?RECPD6-20 参加料-1)))
  ((INFORM-ACTION       SP2 SP1 参加料-1     (支払う-1 SP1 ?RECPD6-20 参加料-1))))

D6-21 : "分かりました"
 (((CONFIRMATION        SP1 SP2 ?TPCD6-21    ?PRPD6-21))
  ((ACCEPT-ACTION       SP1 SP2 ?TPCD6-21    ?PRPD6-21)))

D6-22 : "ありがとうございました"
 (((GREETING-CLOSE      SP1 SP2 CLOSING      (CLOSE-DIALOGUE)))
  ((ACCEPT-OFFER        SP1 SP2 ?TPCD6-22    ?PRPD6-22)))

D6-23 : "ではお待ちしております"
 (((INFORM-STATEMENT    SP2 SP1 ?TPCD6-23    (待つ-1 SP2 ?OBJED6-23))))
```

```
;;;
;;; 会話-7
;;;
D7-1 : "はい"
 (((GREETING-OPEN        SP2 SP1 OPENING       (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP2 SP1 ?TPCD7-1      ?PRPD7-1)))

D7-2 : "こちらは会議事務局です"
 (((INFORM-VALUE         SP2 SP1 SP2           (だ-IDENTICAL SP2 会議事務局-1))))

D7-3 : "会議で扱う話題に関して質問したいんですが"
 (((INFORM-WANT          SP1 SP2 ?TPCD7-3      (質問する-1 SP1 SP2 話題-1)))
  ((ASK-VALUE            SP1 SP2 話題-1        ?PRPD7-3)))

D7-4 : "はい"
 (((GREETING-OPEN        SP2 SP1 OPENING       (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP2 SP1 ?TPCD7-4      ?PRPD7-4)))

D7-5 : "何でしょうか"
 (((ASK-VALUE            SP2 SP1 ?TPCD7-5      (だ-IDENTICAL ?OBJED7-5 ?IDEND7-5))))

D7-6 : "機械翻訳という話題が案内書に載っていますが"
 (((INFORM-STATEMENT     SP1 SP2 ?TPCD7-6      (載る-1 話題-1 案内書-1))))

D7-7 : "具体的にこれはどういう内容のものなんですか"
 (((ASK-VALUE            SP1 SP2 ?TPCD7-7      (だ-IDENTICAL これ-1 もの-1))))

D7-8 : "申し訳ありませんがこちらでは専門的な質問にお答えできません"
 (((REJECT-ACTION        SP2 SP1 ?TPCD7-8      (答える-1 ?AGEND7-8 ?OBJED7-8)))
  ((REJECT-ACTION        SP2 SP1 ?TPCD7-8      (答える-1 ?AGEND7-8 質問-1))))

D7-9 : "第二版の案内書に会議で発表される論文の題目が載っております"
 (((INFORM-STATEMENT     SP2 SP1 ?TPCD7-9      (載る-1 論文-1 版-1))))

D7-10 : "そちらを見て頂けないでしょうか"
 (((REQUEST-ACTION       SP2 SP1 ?TPCD7-10     (見る-1 ?AGEND7-10 そちら-1)))
  ((INFORM-ACTION        SP2 SP1 ?TPCD7-10     (見る-1 ?AGEND7-10 そちら-1))))

D7-11 : "いいですよ"
 (((ACCEPT-ACTION        SP1 SP2 ?TPCD7-11     ?PRPD7-11)))

D7-12 : "それでは早急にその案内書を送って下さい"
 (((REQUEST-ACTION       SP1 SP2 ?TPCD7-12     (送る-1 SP2 SP1 案内書-1)))
  ((INFORM-ACTION        SP1 SP2 ?TPCD7-12     (送る-1 SP2 SP1 案内書-1))))
```

```
D7-13 : "送り先は大阪市東区城見二の一の六十一渡辺明です"
 (((INFORM-VALUE        SP1 SP2 送り先-1      (だ-IDENTICAL 送り先-1 NAME))
   (INFORM-VALUE        SP1 SP2 送り先-1      (だ-IDENTICAL 送り先-1 ADDRESS))))

D7-14 : "大阪市東区城見二の一の六十一渡辺明様ですね"
 (((CONFIRM-VALUE       SP2 SP1 ?TPCD7-14     (だ-IDENTICAL ?OBJED7-14 NAME))
   (CONFIRM-VALUE       SP2 SP1 ?TPCD7-14     (だ-IDENTICAL ?OBJED7-14 ADDRESS))))

D7-15 : "はい"
 (((GREETING-OPEN       SP1 SP2 OPENING       (OPEN-DIALOGUE)))
  ((AFFIRMATIVE         SP1 SP2 ?TPCD7-15     ?PRPD7-15)))

D7-16 : "早速送らせて頂きます"
 (((OFFER-ACTION        SP2 SP1 ?TPCD7-16     (送る-1 SP2 ?RECPD7-16 ?OBJED7-16))))

D7-17 : "他に何かございますか"
 (((CONFIRM-STATEMENT   SP2 SP1 ?TPCD7-17     (ある-1 ?OBJED7-17))))

D7-18 : "いいえ"
 (((NEGATIVE            SP1 SP2 ?TPCD7-18     ?PRPD7-18)))

D7-19 : "ありません"
 (((INFORM-STATEMENT    SP1 SP2 ?TPCD7-19     (ある-1 ?OBJED7-19)))
  ((NEGATIVE            SP1 SP2 ?TPCD7-19     (ある-1 ?OBJED7-19))))

D7-20 : "ありがとうございました"
 (((GREETING-CLOSE      SP1 SP2 CLOSING       (CLOSE-DIALOGUE)))
  ((ACCEPT-OFFER        SP1 SP2 ?TPCD7-20     ?PRPD7-20)))

D7-21 : "失礼します"
 (((GREETING-CLOSE      SP1 SP2 CLOSING       (CLOSE-DIALOGUE))))
```

```
;;;
;;; 会話-8
;;;
D8-1 : "はい"
 (((GREETING-OPEN        SP2 SP1 OPENING       (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP2 SP1 ?TPCD8-1      ?PRPD8-1)))

D8-2 : "会議事務局です"
 (((INFORM-VALUE         SP2 SP1 ?TPCD8-2      (だ-IDENTICAL ?OBJED8-2 会議事務局-1))))

D8-3 : "ちょっとお聞きしたいことがあるんですが"
 (((INFORM-WANT          SP1 SP2 ?TPCD8-3      (聞く-3 ?AGEND8-3 ?RECPD8-3 こと-1)))
  ((ASK-VALUE            SP1 SP2 こと-1        ?PRPD8-3))
  ((INFORM-STATEMENT     SP1 SP2 ?TPCD8-3      (ある-1 ?OBJED8-3))))

D8-4 : "私は今度の会議に発表したいと思っているんですが"
 (((INFORM-WANT          SP1 SP2 SP1           (発表する-1 SP1 ?OBJED8-4 今度-1)))
  ((INFORM-WANT          SP1 SP2 SP1           (発表する-1 ?AGEND8-4 ?OBJED8-4 今度-1))))

D8-5 : "どのような手続きをすればよろしいでしょうか"
 (((ASK-ACTION           SP1 SP2 ?TPCD8-5      (する-1 ?AGEND8-5 手続き-1)))
  ((ASK-ACTION           SP1 SP2 ?TPCD8-5      (する-2 ?AGEND8-5 手続き-1 ?RESLD8-5)))
  ((ASK-ACTION           SP1 SP2 ?TPCD8-5      (する-4 ?AGEND8-5 ?RECPD8-5 手続き-1))))

D8-6 : "先ず二百字の要約を三月二十日までにこちらまでお送り下さい"
 (((REQUEST-ACTION       SP2 SP1 ?TPCD8-6      (送る-1 SP1 SP2 要約-1)))
  ((INFORM-ACTION        SP2 SP1 ?TPCD8-6      (送る-1 SP1 SP2 要約-1))))

D8-7 : "こちらで審査を行ない、五月二十日までに結果をお送り致します"
 (((OFFER-ACTION         SP2 SP1 ?TPCD8-7      (送る-1 SP2 ?RECPD8-7 結果-1))
   (INFORM-STATEMENT     SP2 SP1 ?TPCD8-7      (行なう-1 ?AGEND8-7 審査-1)))
  ((OFFER-ACTION         SP2 SP1 ?TPCD8-7      (送る-1 SP2 ?RECPD8-7 結果-1))
   (INFORM-VALUE         SP2 SP1 ?TPCD8-7      (行なう-1 ?AGEND8-7 審査-1)))
  ((OFFER-ACTION         SP2 SP1 ?TPCD8-7      (送る-1 SP2 ?RECPD8-7 結果-1))
   (INFORM-ACTION        SP2 SP1 ?TPCD8-7      (行なう-1 ?AGEND8-7 審査-1))))

D8-8 : "投稿が受理された場合原稿用紙を同封いたします"
 (((OFFER-ACTION         SP2 SP1 ?TPCD8-8      (同封する-1 SP2 原稿用紙-1))))

D8-9 : "六月三十日までに原稿の送付をお願いします"
 (((REQUEST-ACTION       SP2 SP1 送付-1        ?PRPD8-9)))

D8-10 : "分かりました"
 (((CONFIRMATION         SP1 SP2 ?TPCD8-10     ?PRPD8-10))
```

```
  ((ACCEPT-ACTION        SP1 SP2 ?TPCD8-10    ?PRPD8-10)))

D8-11 : "契約はどのような書式で書けばいいんですか"
 (((ASK-VALUE            SP1 SP2 契約-1       (書く-1 ?AGEND8-11 契約-1 ?LOCTD8-11)))
  ((ASK-STATEMENT        SP1 SP2 契約-1       (いい-1 ?OBJED8-11)))
  ((ASK-VALUE            SP1 SP2 契約-1       (いい-1 ?OBJED8-11))))

D8-12 : "所定の申込み用紙がありますのでそれに記入して下さい"
 (((REQUEST-ACTION       SP2 SP1 ?TPCD8-12    (記入する-1 SP1 ?OBJED8-12 それ-1)))
  ((INFORM-ACTION        SP2 SP1 ?TPCD8-12    (記入する-1 SP1 ?OBJED8-12 それ-1))))

D8-13 : "それでは申込み用紙を送りますので送り先をお願いします"
 (((ASK-VALUE            SP2 SP1 送り先-1     ?PRPD8-13)))

D8-14 : "分かりました"
 (((CONFIRMATION         SP1 SP2 ?TPCD8-14    ?PRPD8-14))
  ((ACCEPT-ACTION        SP1 SP2 ?TPCD8-14    ?PRPD8-14)))

D8-15 : "人工知能研究所のジョージオハラです"
 (((INFORM-VALUE         SP1 SP2 ?TPCD8-15    (だ-IDENTICAL ?OBJED8-15 NAME))))

D8-16 : "住所は東京都豊島区東池袋三丁目二番五号です"
 (((INFORM-VALUE         SP1 SP2 住所-1       (だ-IDENTICAL 住所-1 ADDRESS))))

D8-17 : "人工知能研究所のジョージオハラ様ですね"
 (((CONFIRM-VALUE        SP2 SP1 ?TPCD8-17    (だ-IDENTICAL ?OBJED8-17 NAME))))

D8-18 : "ご住所は東京都豊島区東池袋三丁目二番五号でよろしいですね"
 (((CONFIRM-VALUE        SP2 SP1 住所-1       (だ-IDENTICAL 住所-1 ADDRESS))))

D8-19 : "はい"
 (((GREETING-OPEN        SP1 SP2 OPENING      (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP1 SP2 ?TPCD8-19    ?PRPD8-19)))

D8-20 : "そうです"
 (((AFFIRMATIVE          SP1 SP2 ?TPCD8-20    ?PRPD8-20)))

D8-21 : "それでは申込み用紙の送付をよろしくお願いします"
 (((REQUEST-ACTION       SP1 SP2 送付-1       ?PRPD8-21)))

D8-22 : "はい"
 (((GREETING-OPEN        SP2 SP1 OPENING      (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP2 SP1 ?TPCD8-22    ?PRPD8-22)))
```

```
D8-23 : "分かりました"
 (((CONFIRMATION        SP2 SP1 ?TPCD8-23    ?PRPD8-23))
  ((ACCEPT-ACTION       SP2 SP1 ?TPCD8-23    ?PRPD8-23)))

D8-24 : "では早速お送り致します"
 (((OFFER-ACTION        SP2 SP1 ?TPCD8-24    (送る-1 SP2 SP1 ?OBJED8-24))))

D8-25 : "失礼致します"
 (((GREETING-CLOSE      SP2 SP1 CLOSING      (CLOSE-DIALOGUE))))
```

```
;;;
;;; 会話-9
;;;
D9-1 : "そちらは会議事務局ですか"
 (((CONFIRM-VALUE        SP1 SP2 そちら-1     (だ-IDENTICAL そちら-1 会議事務局-1))))

D9-2 : "はい"
 (((GREETING-OPEN        SP2 SP1 OPENING      (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP2 SP1 ?TPCD9-2     ?PRPD9-2)))

D9-3 : "会議事務局です"
 (((INFORM-VALUE         SP2 SP1 ?TPCD9-3     (だ-IDENTICAL ?OBJED9-3 会議事務局-1))))

D9-4 : "何のご用件でしょうか"
 (((ASK-VALUE            SP2 SP1 ?TPCD9-4     (だ-IDENTICAL ?OBJED9-4 用件-1))))

D9-5 : "会議場へどうやって行ったらいいか教えて欲しいんですが"
 (((ASK-ACTION           SP1 SP2 ?TPCD9-5     (行く-1 ?AGEND9-5 会議場-1))))

D9-6 : "いま京都駅にいるんです"
 (((INFORM-STATEMENT     SP1 SP2 ?TPCD9-6     (いる-1 ?OBJED9-6 京都駅-1))))

D9-7 : "地下鉄で北大路駅まで行って下さい"
 (((REQUEST-ACTION       SP2 SP1 ?TPCD9-7     (行く-3 SP1 北大路駅-1)))
  ((INFORM-ACTION        SP2 SP1 ?TPCD9-7     (行く-3 SP1 北大路駅-1)))
  ((REQUEST-ACTION       SP2 SP1 ?TPCD9-7     (行く-2 SP1 北大路駅-1)))
  ((INFORM-ACTION        SP2 SP1 ?TPCD9-7     (行く-2 SP1 北大路駅-1)))
  ((REQUEST-ACTION       SP2 SP1 ?TPCD9-7     (行く-1 SP1 北大路駅-1)))
  ((INFORM-ACTION        SP2 SP1 ?TPCD9-7     (行く-1 SP1 北大路駅-1))))

D9-8 : "そこから国際会議場へ行くバスが利用できます"
 (((INFORM-ACTION        SP2 SP1 ?TPCD9-8     (利用する-1 ?AGEND9-8 ?PURPD9-8 バス-1))))

D9-9 : "北大路駅ではタクシーも利用できます"
 (((INFORM-ACTION        SP2 SP1 ?TPCD9-9     (利用する-1 ?AGEND9-9 ?PURPD9-9 タクシー
-1))))

D9-10 : "京都駅からタクシーで会議場まで行くにはいくらぐらいかかりますか"
 (((ASK-VALUE            SP1 SP2 ?TPCD9-10    (かかる-1 ?OBJED9-10 ?DEGRD9-10))))

D9-11 : "京都駅からですとおよそ六千円かかります"
 (((INFORM-STATEMENT     SP2 SP1 ?TPCD9-11    (かかる-1 ?OBJED9-11 六千-1))))

D9-12 : "では北大路駅からですといくらぐらいかかりますか"
```

```
 (((ASK-VALUE            SP1 SP2 ?TPCD9-12    (かかる-1 ?OBJED9-12 ?DEGRD9-12))))

D9-13 : "北大路駅からですとおよそ九百円です"
 (((INFORM-VALUE         SP2 SP1 ?TPCD9-13    (だ-IDENTICAL ?OBJED9-13 九百-1))))

D9-14 : "分かりました"
 (((CONFIRMATION         SP1 SP2 ?TPCD9-14    ?PRPD9-14))
  ((ACCEPT-ACTION        SP1 SP2 ?TPCD9-14    ?PRPD9-14)))

D9-15 : "どうもありがとうございました"
 (((GREETING-CLOSE       SP1 SP2 CLOSING      (CLOSE-DIALOGUE)))
  ((ACCEPT-OFFER         SP1 SP2 ?TPCD9-15    ?PRPD9-15)))

D9-16 : "いいえ"
 (((NEGATIVE             SP2 SP1 ?TPCD9-16    ?PRPD9-16)))

D9-17 : "どういたしまして"
 (((GREETING-CLOSE       SP2 SP1 CLOSING      (CLOSE-DIALOGUE))))
```

```
;;;
;;; 会話-10
;;;
D10-1 : "もしもし"
 (((GREETING-OPEN        SP1 SP2 OPENING       (OPEN-DIALOGUE))))

D10-2 : "はい"
 (((GREETING-OPEN        SP2 SP1 OPENING       (OPEN-DIALOGUE)))
  ((AFFIRMATIVE          SP2 SP1 ?TPCD10-2     ?PRPD10-2)))

D10-3 : "会議事務局でございます"
 (((INFORM-VALUE         SP2 SP1 ?TPCD10-3     (だ-IDENTICAL ?OBJED10-3 会議事務局-1)))
  ((INFORM-STATEMENT     SP2 SP1 ?TPCD10-3     (ある-1 ?OBJED10-3))))

D10-4 : "会議の宿泊施設についてお尋ねしたいのですが"
 (((INFORM-WANT          SP1 SP2 ?TPCD10-4     (尋ねる-1 SP1 SP2 宿泊施設-1)))
  ((ASK-VALUE            SP1 SP2 宿泊施設-1    ?PRPD10-4)))

D10-5 : "そちらでどこか紹介して頂けますか"
 (((REQUEST-ACTION       SP1 SP2 ?TPCD10-5     (紹介する-1 SP2 ?OBJED10-5 SP1)))
  ((INFORM-ACTION        SP1 SP2 ?TPCD10-5     (紹介する-1 SP2 ?OBJED10-5 SP1))))

D10-6 : "私どもでご紹介できるホテルは京都ホテルと京都プリンスホテルです"
 (((INFORM-VALUE         SP2 SP1 ホテル-1      (だ-IDENTICAL ホテル-1 京都プリンスホテ
ル-1))
   (INFORM-VALUE         SP2 SP1 ホテル-1      (だ-IDENTICAL ホテル-1 京都ホテル-1))))

D10-7 : "一人部屋の値段は一晩七千円から一万円です"
 (((INFORM-VALUE         SP2 SP1 ?TPCD10-7     (だ-IDENTICAL 値段-1 FEE)))
  ((INFORM-VALUE         SP2 SP1 ?TPCD10-7     (だ-IDENTICAL 値段-1 一万-1)))
  ((INFORM-VALUE         SP2 SP1 値段-1        (だ-IDENTICAL 値段-1 FEE)))
  ((INFORM-VALUE         SP2 SP1 値段-1        (だ-IDENTICAL 値段-1 一万-1))))

D10-9 : "二人部屋の値段は九千五百円から六万円です"
 (((INFORM-VALUE         SP2 SP1 ?TPCD10-9     (だ-IDENTICAL 値段-1 九千五百-1)))
  ((INFORM-VALUE         SP2 SP1 ?TPCD10-9     (だ-IDENTICAL 値段-1 六万-1)))
  ((INFORM-VALUE         SP2 SP1 値段-1        (だ-IDENTICAL 値段-1 六万-1))))

D10-10 : "そうですか"
 (((CONFIRMATION         SP1 SP2 ?TPCD10-10    ?PRPD10-10)))

D10-11 : "どちらのホテルが会議場に近いですか"
 (((ASK-STATEMENT        SP1 SP2 ?TPCD10-11    (近い-1 ホテル-1)))
  ((ASK-VALUE            SP1 SP2 ?TPCD10-11    (近い-1 ホテル-1))))
```

D10-12 : "京都プリンスホテルが会議場には近いんですが"
(((INFORM-STATEMENT SP2 SP1 ?TPCD10-12 (近い-1 京都プリンスホテル-1))))

D10-13 : "それでは京都プリンスホテルを予約したいのですが"
(((INFORM-WANT SP1 SP2 ?TPCD10-13 (予約する-1 SP1 京都プリンスホテル-1)))
((INFORM-WANT SP1 SP2 ?TPCD10-13 (予約する-1 ?AGEND10-13 京都プリンスホテル-1))))

D10-14 : "ホテルの手配もして頂けるのですか"
(((ASK-ACTION SP1 SP2 手配-1 ?PRPD10-14))
((REQUEST-ACTION SP1 SP2 手配-1 (する-2 ?AGEND10-14 ?OBJED10-14 手配-1)))
((INFORM-ACTION SP1 SP2 手配-1 (する-2 ?AGEND10-14 ?OBJED10-14 手配-1)))
((REQUEST-ACTION SP1 SP2 手配-1 (する-2 ?AGEND10-14 手配-1 ?RESLD10-14)))
((INFORM-ACTION SP1 SP2 手配-1 (する-2 ?AGEND10-14 手配-1 ?RESLD10-14)))
((REQUEST-ACTION SP1 SP2 手配-1 (する-1 ?AGEND10-14 手配-1)))
((INFORM-ACTION SP1 SP2 手配-1 (する-1 ?AGEND10-14 手配-1))))

D10-15 : "はい"
(((GREETING-OPEN SP2 SP1 OPENING (OPEN-DIALOGUE)))
((AFFIRMATIVE SP2 SP1 ?TPCD10-15 ?PRPD10-15)))

D10-16 : "京都ホテルと京都プリンスホテルは予約できます"
(((INFORM-ACTION SP2 SP1 京都プリンスホテル-1 (予約する-1 ?AGEND10-16 京都プリンスホテル-1))
(INFORM-ACTION SP2 SP1 京都ホテル-1 (予約する-1 ?AGEND10-16 京都ホテル-1))))

D10-17 : "そうですか"
(((CONFIRMATION SP1 SP2 ?TPCD10-17 ?PRPD10-17)))

D10-18 : "では京都プリンスホテルの七千円の一人部屋をお願いします"
(((ASK-VALUE SP1 SP2 一人部屋-1 ?PRPD10-18))
((ASK-VALUE SP1 SP2 部屋-1 ?PRPD10-18)))

D10-19 : "はい"
(((GREETING-OPEN SP2 SP1 OPENING (OPEN-DIALOGUE)))
((AFFIRMATIVE SP2 SP1 ?TPCD10-19 ?PRPD10-19)))

D10-20 : "京都プリンスホテルの七千円の一人部屋ですね"
(((CONFIRM-VALUE SP2 SP1 ?TPCD10-20 (だ-IDENTICAL ?OBJED10-20 一人部屋-1)))
((CONFIRM-VALUE SP2 SP1 ?TPCD10-20 (だ-IDENTICAL ?OBJED10-20 部屋-1))))

D10-21 : "はい"
(((GREETING-OPEN SP1 SP2 OPENING (OPEN-DIALOGUE)))

```
  ((AFFIRMATIVE          SP1 SP2 ?TPCD10-21   ?PRPD10-21)))

D10-22 : "そうです"
 (((AFFIRMATIVE          SP1 SP2 ?TPCD10-22   ?PRPD10-22)))

D10-23 : "いつからお泊まりになりますか"
 (((ASK-VALUE            SP2 SP1 ?TPCD10-23   (泊まる-1 ?AGEND10-23 ?LOCTD10-23))))

D10-24 : "八月四日の夜からです"
 (((INFORM-STATEMENT     SP1 SP2 ?TPCD10-24   (だ-STATEMENT 夜-1 ?IDEND10-24))))

D10-25 : "八日の朝までお願いします"
 (((REQUEST-ACTION       SP1 SP2 ?TPCD10-25   ?PRPD10-25))
  ((ACCEPT-OFFER         SP1 SP2 ?TPCD10-25   ?PRPD10-25)))

D10-26 : "分かりました"
 (((CONFIRMATION         SP2 SP1 ?TPCD10-26   ?PRPD10-26))
  ((ACCEPT-ACTION        SP2 SP1 ?TPCD10-26   ?PRPD10-26)))

D10-27 : "少々お待ちください"
 (((REQUEST-ACTION       SP2 SP1 ?TPCD10-27   (待つ-1 SP1 ?OBJED10-27)))
  ((INFORM-ACTION        SP2 SP1 ?TPCD10-27   (待つ-1 SP1 ?OBJED10-27))))

D10-28 : "お部屋が取れるかどうか調べます"
 (((OFFER-ACTION         SP2 SP1 ?TPCD10-28   (調べる-1 SP2 ?OBJED10-28))))

D10-29 : "お部屋をお取りできます"
 (((INFORM-ACTION        SP2 SP1 ?TPCD10-29   (取る-1 ?AGEND10-29 ?LOCTD10-29 部屋-1))))

D10-30 : "ではお名前とご住所をお願いします"
 (((ASK-VALUE            SP2 SP1 住所-1       ?PRPD10-30)
   (ASK-VALUE            SP2 SP1 名前-1       ?PRPD10-30)))

D10-31 : "中村一雄です"
 (((INFORM-VALUE         SP1 SP2 ?TPCD10-31   (だ-IDENTICAL ?OBJED10-31 NAME))))

D10-32 : "住所は東京都港区新橋一丁目一番三号です"
 (((INFORM-VALUE         SP1 SP2 住所-1       (だ-IDENTICAL 住所-1 ADDRESS))))

D10-33 : "電話番号もお願いします"
 (((ASK-VALUE            SP2 SP1 電話番号-1   ?PRPD10-33)))

D10-34 : "電話番号は三三一の二五二一です"
 (((INFORM-VALUE         SP1 SP2 電話番号-1   (だ-IDENTICAL 電話番号-1 二五二一-1))))
```

D10-35 : "分かりました"
(((CONFIRMATION SP2 SP1 ?TPCD10-35 ?PRPD10-35))
((ACCEPT-ACTION SP2 SP1 ?TPCD10-35 ?PRPD10-35)))

D10-36 : "京都プリンスホテルに八月四日から八日まで一人部屋をお取りしました"
(((INFORM-STATEMENT SP2 SP1 ?TPCD10-36 (取る -1 SP2 京都プリンスホテル -1 一人部屋 -1)))
((INFORM-VALUE SP2 SP1 ?TPCD10-36 (取る -1 SP2 京都プリンスホテル -1 一人部屋 -1)))
((INFORM-ACTION SP2 SP1 ?TPCD10-36 (取る -1 SP2 京都プリンスホテル -1 一人部屋 -1)))
((INFORM-STATEMENT SP2 SP1 ?TPCD10-36 (取る -1 ?AGEND10-36 京都プリンスホテル -1 一人部屋 -1)))
((INFORM-STATEMENT SP2 SP1 ?TPCD10-36 (取る -1 ?AGEND10-36 京都プリンスホテル -1 部屋 -1)))
((INFORM-VALUE SP2 SP1 ?TPCD10-36 (取る -1 ?AGEND10-36 京都プリンスホテル -1 部屋 -1)))
((INFORM-ACTION SP2 SP1 ?TPCD10-36 (取る -1 ?AGEND10-36 京都プリンスホテル -1 部屋 -1)))
((INFORM-VALUE SP2 SP1 ?TPCD10-36 (取る -1 ?AGEND10-36 京都プリンスホテル -1 一人部屋 -1)))
((INFORM-ACTION SP2 SP1 ?TPCD10-36 (取る -1 ?AGEND10-36 京都プリンスホテル -1 一人部屋 -1))))

D10-37 : "どうもありがとうございました"
(((GREETING-CLOSE SP1 SP2 CLOSING (CLOSE-DIALOGUE)))
((ACCEPT-OFFER SP1 SP2 ?TPCD10-37 ?PRPD10-37)))

D10-38 : "失礼します"
(((GREETING-CLOSE SP1 SP2 CLOSING (CLOSE-DIALOGUE))))