

状況意味論を用いた質問応答システム

幡野浩司

小林豊

新美康永

(京都工芸繊維大学 工芸学部)

1 はじめに

同一の文であってもそれが発話された状況の違いによって発話者の意図が異なることがある。また、文が状況によっては反語的な使われ方をすることがしばしばある。ユーザの発する質問文からそのユーザがどのような情報を要求しているかを読み取らなければならない質問応答システムにおいては以上のような状況の違いによる意図の違いの考察が重要である。J.Barwise と J.Perry によって発表された状況意味論 (Situation Semantics) [1]においては、発話の表わす状況(発話内容)と発話が起きた状況(発話状況)との関係において発話の意味をとらえようとしている。本稿で報告する質問応答システムは、この状況意味論の考え方に基づいて発話状況を「発話の履歴」、「ユーザモデル」、「ワールドモデル」の3個の知識ベースを用いて表現し、質問文を構文解析することによって得られる発話内容とこれらの発話状況を用いてユーザの意図を解析している。現在、京都観光案内をタスクとし、小規模な実験システムで検討を進めている。この方法により、次のように様々な状況の変化に対応した意味解析を行なうことができる。Qがユーザの問、Aがシステムの応答である。

Q1: 寺を見たい。

A1: 洛中にある寺でいいか。

Q2: よくない。

A2: 東山にある寺でいいか。

Q3: いい。

(承諾の意味に解釈する。)

A3: 東山には清水寺がある。

Q4: 清水寺はおもしろいか。

(情報の要求をしていると解釈する。)

A4: 首ふり地蔵がある。

Q5: それがおもしろいか。

(不満を抱いていると解釈する。)

A5: ぬれ手観音がある。

Q6: おもしろくない。

A6: 高台寺の情報がほしいか。

Q7: いい。

(遠慮の意味に解釈する。)

A7: 銀閣寺の情報がほしいか。

:

2 発話内容・発話状況・発話者の意図の表現法

この章ではユーザの発話が起きた時点の状況、ユーザの発話から抽出されるユーザの意図などを表現するための枠組を説明する。[2]

2.1 プリミティブ (Primitive)

次の4種類のものを、状況を表現するうえでの基本要素とする。

a) 個体 (Individuals)

机、本などの具象物の集合や、祭、情報などの抽象物の集合およびそれらの集合の要素を個体と呼び、アトムで表わす。例えば「desk」、「book」などは、机、本の集合を表わす個体である。

b) 関係 (Relations)

個体と個体との関係、状況と状況との関係を「関係」と呼び、関係の名前を示すアトムで表わす。例えば「eating」、「seeing」は、それぞれある個体がある個体を食べているという関係、ある個体がある個体を見ているという関係である。1個の個体あるいは状況を修飾するものも関係と呼ぶ。例えば、「red」は、ある個体が赤いことを表わす関係である。

c) 時空領域 (Space-time Location)

ある事象が起きたときの時間と空間とをまとめたものを時空領域と呼び、 l で表わす。ある時ある所で事象Aが起きたという状況は、「時空領域 l において事象Aが起きた。」と表現する。実際には種々の時

空領域を識別できるように、 l の後に識別子を付加した形で表現する。例えば「 l_{1020} 」、「 l_2 」は時空領域を表わす。また、全時空(Universal location)を考え、 l_u で表わす。これは「すべての場所ですべての時間で」を表わす。

d) 不定要素 (Indeterminates)

上述のa)、b)、c)を変数で表わしたものを不定要素と呼ぶ。不定要素は変数名を表わすアトムの前に「\$」付加して表わす。例えば「\$ l 」、「\$man」は不定要素である。

2.2 イベント型 (Event Type)

事象を表現するための枠組をイベント型という。イベント型は上で述べたプリミティブを組み合わせて表現する。例えば「時空領域 l において個体 a が個体 b を食べる。」という事象を意味するイベント型は、

$E := \text{at } l : \text{eating}, a, b ; \text{yes}$

と表現される。「yes」は、 a 、 b 間に「eating」という関係が成立していることが真であることを表わす。関係が成立しないときは、「no」を用いる。一般に、イベント型を

<Event Name>:=

at { <Space-time Location>
<Indeterminates> }
:
<Relation> ,
{ <Space-time Location>
<Individuals>
<Event Type>
<Indeterminates> } * ; { yes
no }

という形式で表現する。但し、「*」は0回以上の繰り返しを示し、{ } は、その中の要素のいずれか1つを選択することを表わす。

事象の集合を用いて事象の流れを表現することができる。例えば、イベント型

$E := \text{at } l_1 : \text{hungry}, a ; \text{yes}$
 $\text{at } l_1 : \text{eating}, a, b ; \text{yes}$
 $\text{at } l_2 : \text{hungry}, a ; \text{no}$
 $\text{at } l_u : <, l_1, l_2 ; \text{yes}$

は、「 a が空腹で、 b を食べると空腹でなくなった。」という事象の流れを表わしている。但し「hungry, a 」は「 a が空腹である」という関係を、「<, l_1, l_2 」は「 l_1 が l_2 に時間的に先行している」ことを表わす。また、 a 、 b 、 l_1 、 l_2 を不定要素に置換すると、事実ではなく定理が表現できる。

$E := \text{at } \$l_1 : \text{hungry}, \$a ; \text{yes}$
 $\text{at } \$l_1 : \text{eating}, \$a, \$b ; \text{yes}$
 $\text{at } \$l_2 : \text{hungry}, \$a ; \text{no}$
 $\text{at } l_u : <, \$l_1, \$l_2 ; \text{yes}$

これは、「ある個体 $\$a$ が時空領域 $\$l_1$ において空腹であり、個体 $\$b$ を食べると、時空領域 $\$l_2$ には空腹ではない。しかも、 $\$l_1$ は $\$l_2$ に時間的に先行している。」という定理を表わしている。

2.3 ロール (Role)

イベント型 e とその中で用いられる不定要素 $\$x$ とを組み合わせたもの「< $\$x$, e >」をロールと呼ぶ。これは、「 e というイベント型の中の $\$x$ 」を表現する。例えば、

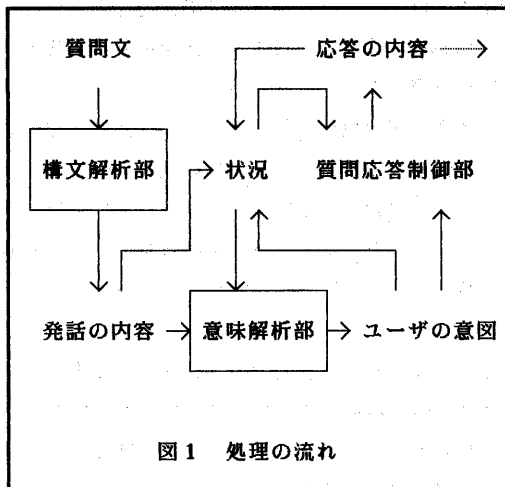
$e := \text{at } l_1 : \text{red}, \$a ; \text{yes}$

というイベント型は、「 $\$a$ が赤い」ということを表現している。このイベント型と不定要素 $\$a$ との組「< $\$a$, e >」は、「赤い $\$a$ 」を表現している。

3 質問応答システムの構成

本システムの処理の流れを図1に示す。

自然言語で表現された質問文は構文解析されて発話の内容を表わすイベント型に変換される。発話の内



容は質問応答における「状況」を用いて意味解析され、ユーザの意図が抽出される。ユーザの意図をもとに質問応答の制御および情報検索が行なわれ応答の内容を表わすイベント型を作成する。この章では図の各部について説明する。現在のところ、入力文の形態素解析とシステムの応答を自然言語で出力する部分は未完成である。

3.1 構文解析部

構文解析部では、単語ごとに分かち書きされたユーザからの入力文（形態素解析の済んでいる）を構文解析し、質問文の内容を表わすイベント型が作成される。

3.1.1 単語辞書

単語辞書には各単語の品詞名とともに次のような情報を格納しておく。

a) 名詞

普通名詞に対してはそれを表わすロールを格納する。固有名詞に対してはそれを表わすアトムを格納する。代名詞に対しては不定要素を格納しておく。

例 「寺」: <\$x, E (\$x)>

E (\$x) := at l: isa, \$X, 寺 ;yes

「銀閣寺」: 銀閣寺

「それ」: \$x

b) 用言

用言に相当する関係を含むイベント型を格納する。そのとき、イベント型中の各プリミティブは不定要素で記述しておき、それらの不定要素と用言に係る格との対応を格納しておく。

例 「食べる」: at l: eating, \$x, \$y ;yes
case(主格, \$x)
case(目的格, \$y)

c) 助詞

それが表現する格およびmodalを示すマーカを格

入力文「金閣寺 を 見る たい か」

単語	単語辞書中に格納されている情報
金閣寺	金閣寺
を	case (目的格)
見る	atl: seeing, \$x, \$y ;yes
たい	modal(希望)
か	modal(疑問)

「金閣寺」: [金閣寺]

「を」: [金閣寺 case (目的格)]

「見る」: [atl: seeing, \$x, 金閣寺; yes
case (主格, \$x)
case (目的格, 金閣寺)]

「たい」: [atl: seeing, \$x, 金閣寺; yes
case (主格, \$x)
case (目的格, 金閣寺)
modal(希望)]

「か」: [atl: seeing, \$x, 金閣寺; yes
case (主格, \$x)
case (目的格, 金閣寺)
modal(希望)
modal(疑問)]

図2 文「金閣寺を見たいか」の構文解析過程

納する。

例 「が」 : case(主格)

「か」 : modal(疑問)

d)副詞

用言を表現するイベント型中の時空領域を修飾する関係あるいは文全体を修飾する関係を含むイベント型の枠組を格納しておく。

例 「非常に」 : at f: very, \$e ;yes
mod(\$e)

3.1.2 構文解析

入力文の構文解析には、電子技術総合研究所の松本らによって開発されたボトムアップパーザ [3]を用いる。入力文をパースするとともに単語辞書中に格納されている用言の不定要素にプリミティブをバインドしていき、発話内容を表わすイベント型表現を得る。その際、modal 情報は後の処理のためにイベント型表現と別に集めておく。文「金閣寺を見たいか」の解析過程を図2に示す。

3.1.3 modal 処理

ボトムアップパーザにより作成されたイベント型表現はmodal 情報を用いて更に変形され、発話の内容を表現するイベント型表現になる。変形のための

変形ルール: modal(希望)

$e \rightarrow \text{at } f:\text{want}, \text{user}, e;\text{yes}$

例 at f:seeing, user, 金閣寺 ;yes



at f:want, user,
at f:seeing, user, 金閣寺 ;yes
;yes

図3 modal変形規則と適用の例

知識はプロダクションルールの形式で記述されている。modal 「希望」に対する変形規則とその規則の適用例を図3に示す。

このような変形規則を各modal ごとに用意しておき、構文解析結果であるイベント型に次々に適用することにより、発話の内容を表わすイベント型表現を得る。ルールの適用順序は予め定められており、受身、希望等が先に、否定、疑問等が後に処理される。

3.1.4 イベント型中の共起関係の検査

modal の処理後、構文解析における曖昧性を解消するために、イベント型の中の個体と関係との共起関係を検査する。共起関係に不適切なものが発見された場合は、構文解析結果の別の可能性を調べるためにボトムアップパーザによるパーズングをやりなおす。共起関係の検査のための規則は、個々の関係について定められている。規則には、関係r に対して

$r, a_1, a_2, \dots, a_n$

という共起関係が成立するためには個体 $a_1, a_2, \dots, a_n$ がどのような条件をみたさなければならないかが示されている。共起関係の検査を構文解析と並行して行なうと、modal 「受身」によって主語と述語とが入れ替わる場合に処理が複雑になるので、modal 処理後に行なっている。関係「eating」に対する共起関係の検査のための規則を図4に示す。

eating, \$x, \$y

isa, \$x, 動物

isa, \$y, 食物

図4 関係「eating」と個体との共起関係の検査のための規則

このような拘束条件を各関係ごとに記述すると、同一の拘束条件を関係ごとに別々に記述するという無駄が生じる。そこで、本システムでは関係同士の階

層を考えて拘束条件を共有することにした。例えば、関係「making (作る)」と「constructing (建てる)」の共起関係はそれぞれ

making , \$x , \$y

isa , \$x , 人間

isa , \$y , 物

constructing , \$x , \$y

isa , \$x , 人間

isa , \$y , 建物

であるが、「constructing」が「making」より下位のレベルの関係であることを関係同士の概念的上下関係 act-isa を用いて

act-isa , constructing , making

と表現することにより、共起関係に現われる条件

isa , \$x , 人間

を「making」のみに付けておくことができる。

3. 2 意味解析部

意味解析部では、構文解析結果である質問文のイベント型表現と質問文が発せられた状況とを用いてユーザの意図が抽出される。

3. 2. 1 発話状況の表現のための知識ベース

同一の質問文であってもそれが発話されたときの状況によって発話者の意図が異なる場合がある。本システムでは、状況の違いによる発話者の意図の違いに柔軟に対処するために「発話の履歴」、「ユーザモデル」、「ワールドモデル」という3個の知識ベースを用意し、質問応答の経過に伴いこれらを変化させて状況の変化を追跡する方法をとっている。質問文が発せられた時点でこの3個の知識ベースの状態で質問文が発せられた状況を表わしていると考え

る。知識ベースには「状況」がイベント型表現で格納される。構文解析部の出力である質問文の内容と3個の知識ベースで表現される状況との関係をもとにして発話者の意図が抽出される。

a) 発話の履歴

「いいです」という文は、「承諾」と「遠慮」の2通りの意味に解釈される。どちらの意味に解釈されるべきであるかを決定するためには、「いいです」がどのような質問文に対する回答であるかを考慮する必要がある。また、ユーザの質問文中に現われる代名詞による照応に備えるためにユーザの質問の内容とシステムの応答の内容を記憶しておく必要がある。「発話の履歴」は以上の2点の必要性に応えるためのもので、ユーザの質問文の構文解析結果であるイベント型表現と情報検索の結果作成されるシステムの応答のイベント型表現の両者が質問応答が交わされるごとに「発話の履歴」に格納されていく。知識ベースにはブッシュグウン形式で知識が次々に格納され、参照は後に格納されたものから行なわれる。次に述べる「ユーザモデル」も「発話の履歴」と同じ形式で知識の格納、参照が行なわれる。

b) ユーザモデル

「おもしろいか」という文は通常「情報の要求」という意図を表わしているが、「おもしろくない」という意見を含んだ反語的な使われ方をすることが少なくない。本システムでは、ユーザがどのような情報をもっているか、どのような情報を要求しているかを手掛かりにしてこのような意図の違いを認識しようとしている。上の例では、ある情報に対して「おもしろいか」が発話された時点において発話者が既にその情報の詳細を知っている場合には反語的に、知らない場合には詳細の情報の要求であるとシステムは解釈する。ユーザがどのような情報を既に知っているか、どのような情報を要求しているかをユーザの発話やシステムの応答からシステムが推論し、「ユーザモデル」に格納する。「ユーザモデル」は、応答の生成においてユーザが既に知っている情

報を削除するためにも用いられる。

の抽出のための規則はプロダクションルールで記述されており、次の形をしている。

c) ワールドモデル

「金閣寺に大仏がある。」というユーザの発話からは、「金閣寺には大仏はない。」というシステム内の知識を使うことによって「発話者は『金閣寺に大仏がある』と信じているだけである。」という情報を得ることができ、この情報を「ユーザモデル」に格納して以後の質問応答の処理に利用することができる。「ワールドモデル」には、「金閣寺には〇〇がある。」、「清水寺は△△にある。」等、外界の状況が格納される。「発話の履歴」、「ユーザモデル」が質問応答の過程に従って動的に変化するのに対し、「ワールドモデル」は静的な知識ベースである。「ワールドモデル」は応答作成における情報検索のためのデータベースとしても用いられる。

in NU, $e(1), e(2), \dots, e(n_n)$
iff
in TA, $e(T,1), e(T,2), \dots, e(T, n_T)$
in WM, $e(W,1), e(W,2), \dots, e(W, n_W)$
in UM, $e(U,1), e(U,2), \dots, e(U, n_U)$
in LC, $e(L,1), e(L,2), \dots, e(L, n_L)$

TA、WM、UM はそれぞれ「発話の履歴」、「ワールドモデル」、「ユーザモデル」であり、LC は発話の内容、NU はユーザの意図である。 $e(i)$ 、 $e(i,j)$ はイベント型を表わしている。ルールは、「TA、WM、UM を用いてそれぞれ $e(T,1) \sim e(T, n_T)$ 、 $e(W,1) \sim e(W, n_W)$ 、 $e(U,1) \sim e(U, n_U)$ が証明できるような状況において、LC を用いて $e(L,1) \sim e(L, n_L)$ が証明できるような発話 LC がなされた」とすると、発話者の意図は $e(1) \sim e(n_n)$ である。」ということを表わしている。ルール中の証明の段階で不定要素のアンカリングがおきる。すなわち、意味解析と同時に発話文で省略されていた情報が補われる。

3.2.2 発話内容からの発話者の意図の抽出

前節で説明した「発話の履歴」、「ユーザモデル」、「ワールドモデル」の3個の知識ベースと質問文の構文解析結果であるイベント型を用いてユーザの意図が抽出される。ユーザの意図とは、例えば、「ユーザは清水寺の情報を欲しがっている。」、「ユーザはシステムの問に対して同意している。」等の事象をイベント型で表現したものである。意図

3.3 質問応答の制御部

前節までに説明した方法により抽出されたユーザ

```

(質問応答制御規則1
  (ユーザの意図 「ユーザが *X を見たがっている」)
  (システムの応答 *ANS))

(*Xが存在する場所 *P2 をワールドモデルから検索する)
(NOT (「*P2にあるものは見たくない」がユーザモデルを用いて証明できる))

(
  (質問応答制御規則2
    (ユーザの意図 *NU)
    (システムの応答 「*P2にあるものでよいか」)) を呼び出す)
  IF (「ユーザが承諾している」が *NU を用いて証明できる)
  THEN (AND (*P2にある *X をワールドモデルから検索する)
          (「*P2には *X がある」を *ANS に代人する))
  ELSE (IF (「ユーザが反対している」が *NU を用いて証明できる)
          THEN (AND (「ユーザは *P2 をつまらなく思っている」をユーザモデルに格納する)
                      (
                        (質問応答制御規則1 「ユーザが *X を見たがっている」)
                        (システムの応答 *ANS)) を呼び出す)))
  ELSE (*NUに関する話題に変える))

```

図5 質問応答制御規則の例

(QUESTION-ANSWERING-HISTORY (

(EVENT ASK YES \$L16_2831 I (EVENT GOOD YES \$L16_2831 \$XDX_2831))
(EVENT ISA YES LU \$XDX_2831 TERA)
(EVENT EXIST YES L_0020 \$XDX_2831 *PLC_2831 *TIM_2831)
(EVENT WHERE YES LU L_0020 RAKUCHOU)
(EVENT WHEN YES LU \$L_0805 \$T_2191)
(EVENT CP-BEGIN YES LU \$T_2191)
(TIME (YEAR 1985) (MONTH 3) (DATE 8) (HOUR 15) (MINUTE 4)))
(EVENT WANT YES \$L_0805 USER
(EVENT SEE YES \$L_8621 USER \$X_8596 *CMP_8621 *PLC_8621 *TIM_8621))
(EVENT ISA YES LU \$X_8596 TERA)

; \$XDX_2831 でよいかシステムが尋ねた
; \$XDX_2831 は寺である
; \$XDX_2831 は L_0020 にある
; 時空領域 L_0020 は洛中である
; 時空領域 \$L_0805 は時間 \$T_2191
である
; 時間 \$T_2191 は 1985 年 3 月 8 日 15 時
4 分から始まっている
; ユーザは \$X_8621 を見たい
; \$X_8621 は寺である
))

(USER-MODEL (

(EVENT WANT YES \$L_0805 USER
(EVENT SEE YES \$L_8621 USER \$X_8596 *CMP_8621 *PLC_8621 *TIM_8621))

; ユーザは \$X_8621 を見たい
))

-- "KIYOMIZUDERA WA OMOSHIROI KA"

; 入力文「清水寺はおもしろいか」

(PARSING-RESULT

(WORD (MODAL GIMON)
(MODAL SENTENCE)
(CASE AGT (WORD (EXPR Kiyomizudera)))
(EXPR (EVENT INTERESTING YES \$L_6817 (WORD (EXPR Kiyomizudera))))))

(CONTENTS-OF-THE-INPUT-SENTENCE (

(EVENT WHEN YES LU \$L_8009
(TIME (YEAR 1985) (MONTH 3) (DATE 8) (HOUR 15) (MINUTE 6)))
(EVENT ASK YES \$L_8009 USER
(EVENT INTERESTING YES \$L_6817 Kiyomizudera))

; 清水寺はおもしろいかユーザが尋ねた
))

(QUESTION-ANSWERING-HISTORY (

C

A

))

(MEANING-OF-THE-SENTENCE (

(EVENT WANT YES \$L_8657 USER (EVENT KNOW YES \$L1_8657 USER \$Y_8657))

; ユーザは \$Y_8657 を知りがっている (\$Y_8657 は清水寺に関する情報である)
))

(USER-MODEL (

D

B

))

(ANSWER (

(EVENT SAY YES \$L_0597 I (EVENT EXIST YES L_0004 KUBIFURIJIZOU))

; システムは「首ふり地蔵がある」という応答をした
))

(QUESTION-ANSWERING-HISTORY (

E

F

(EVENT WHEN YES LU \$L_0597
(TIME (YEAR 1985) (MONTH 3) (DATE 8) (HOUR 15) (MINUTE 7)))
(EVENT ISA YES LU \$Y_8657 INFORMATION)
(EVENT ZOKUSURU YES LU \$Y_8657 Kiyomizudera)

; \$Y_8657 は情報である
; \$Y_8657 は清水寺に関するものである
))

(MEANING-OF-THE-SENTENCE (

(EVENT KNOW YES \$L_1274 USER
(EVENT EXIST YES L_0004 KUBIFURIJIZOU))
(EVENT WANT NO \$L_1274 USER
(EVENT KNOW YES \$L1_1274 USER
(EVENT EXIST YES L_0004 KUBIFURIJIZOU)))

; ユーザは「清水寺に首ふり地蔵がある」ということを知っている
; ユーザは「清水寺に首ふり地蔵がある」という情報をもはや欲していない
))

(USER-MODEL (

G

D

B

))

(注) 重複している部分は点線で囲んで示した

図 6 処理の例

の意図をもとにして、ユーザの要求する情報が検索され、応答が作成されるが、検索するために必要な情報が前節までに説明した処理によって得られない場合は、ユーザに問い返すことによって必要な情報を得なければならない。本システムでは、質問1個に回答1個が対応する「質問回答の対」の実行方法を1個の手続き（質問回答制御規則）として記述し、それらが互いに呼び出し合ったり情報を交換し合ったりすることにより質問回答の入れ子をシミュレートする。質問回答制御規則は、他の制御規則を呼び出すための手続き、他の制御規則を呼び出すことによって得られた情報を用いて情報検索を行ない応答を作成する手続き等から成っている。制御規則には、ユーザの質問に対してシステムが応答するための規則とシステムがユーザに質問して応答を求めるための規則の2種類がある。図5にユーザの質問に対してシステムが応答するための規則を示す。この規則の中では他の制御規則および自分自身が呼び出されている。

4 処理例

次の質問回答 Q1 ~ A2 において、Q2 が発話されたときの処理の様子を図6に示す。

- Q1: 寺を見たい。
 A1: 洛中にある寺でよいか。
 Q2: 清水寺はおもしろいか。
 A2: 清水寺には首ふり地蔵がある。

図では、質問文の構文解析結果(modal 処理前「PARSING-RESULT」、modal 処理後「CONTENTS-OF-THE-INPUT-SENTENCE」、システムの応答の内容「ANSWER」、抽出されたユーザの意図「MEANING-OF-THE-SENTENCE」、および処理が進むにつれて発話の履歴「QUESTION-ANSWERING-HISTORY」、ユーザモデル「USER-MODEL」がどう変化するかを示した。なお2章で述べたイベント型表現

は、システム内では

(EVENT r s l a₁ a₂ ... a_n)

と表現される。

5 結言

状況の違いによる意味の違いを扱うことのできる自然言語による質問回答システムを開発した。現在、システムは処理のための枠組を作成した段階で、今後各種の規則を増やし、広範な質問回答に応えることができるようシステムを強化していく予定である。特に、規則の作成においてどの程度の規則の一般化が可能であるかなどの検討が必要である。更に、質問回答の状況と質問の内容との関係の規則の他に、発話に含まれる単語の現われる頻度をもとにユーザの好みを判断する規則、獲得したユーザの好みの知識の一般化の規則などを盛り込むことにより、便利なシステムが実現できるのではないかと考えている。なお、本システムは京都大学大型計算機FACOM 上にプログラミング言語PROLOG/KR [4] を用いて記述されている。

参考文献

- [1] J.Barwise , J.Perry : "Situation and Attitudes", MIT Press,(1983)
- [2] 鈴木浩之 : "日本語文の意味の状況意味論的な記述", 自然言語処理研究会, 42-3, 1984年 3月
- [3] 松本裕治, 田中穂積 : "Prolog に埋め込まれた bottom-up parser", 自然言語処理研究会, 34-6, 1982年12月
- [4] H.Nakashima: "Prolog/KR User's Manual", Department of Mathematical Engineering and Instrumentation Physics Faculty of Engineering ,University of Tokyo ,(1982)

at l : r , a₁ , a₂ , ... , a_n ; s