実世界モデルに基づく言語ＮＡＩＶＥの論理体系（再論）

日野克重

富士通㈱

言語NAIVE は，実体，関係，行為，主体，および場などの基本要素から構成される実世界モデルに基づいて対象を記述する実行可能な仕様記述言語である．

本稿では，その論理体系を提示する．当論理体系は，上に述べた実世界モデルを形式化した体系であり，オブジェクト論理，時区間論理，並行論理，および開放型論理などの側面をあわせもった総合的論理体系になっている．

The Logical System of The Language NAIVE that is based on Real World Model

Katsushige HINO

FUJITSU Ltd.

NAIVE is an executable specification language that is based on a real world model. In this paper, the logical system of the language NAIVE is proposed.

The logical system is the one formalizing the real world model which is composed of concepts such as entity, relation, action, agent, field, and so on.

1．はじめに

言語NAIVE は，実体，関係，行為，主体，および場などの基本要素から構成される実世界モデルに基づいて対象を記述する実行可能な仕様記述言語である．文献6)ではその言語仕様，記述例，ならびに記述・実行実験の結果を中心にNAIVE の全体像をしめした．本稿では，言語NAIVE の論理体系に焦点を当てて議論する．

言語NAIVE の論理体系をあらためて提示することには，次のような意義があると考える．

- 言語NAIVE の実世界モデルは，複数の行為主体による並行協調動作，オブジェクト概念，ならびにデータベースの永続性や更新に関する概念などの重要なソフトウェア概念を統一的に含んでいる．したがって，その形式化は，これら諸概念を一つの枠組の中に収めた総合的なソフトウェア論理の提案になる．

当論理体系は，伝統的論理との連続性ならびに自然語の意味論との親近性を重視して設計されている．その結果，当論理体系は，次のような特徴をあわせもった論理体系になっている．

① モンタギューの内包論理の枠組の中に固有の世界観を注入した論理系である．
② 実在性，種，および属性などの実体概念を形式化した一種のオブジェクト論理系[7),8),10)] である．
③ 行為概念にもとづく時区間論理系[4)] である．
④ 自律的行為主体が場を介して相互干渉する並行論理系である．
⑤ 神および天使などの世界外存在を導入した開放型論理系である．

以降，２章では言語NAIVE の概要を示す．３章では論理体系の骨子を述べ，４章では論理体系の詳細を形式的に提示する．５章では，それを用いた論理計算の例を示す．６章はむすびである．

2．言語NAIVE の概要

言語NAIVE の概要の理解のため，その背後に設定した世界モデル（図－１および図－２参照）ならびにNAIVE による仕様記述例（図－３）を示す．

ここで示した実世界モデルは，個体と関係の概念からなる述語論理的世界モデルの上に実体概念，行為概念，行為主体の概念，および世界外存在の概念などを統合的に付加したものといえる（それぞれの概念を導入する必要性や効果については文献6)で論じた）．これらの概念をすべて統一的に含む形式的体系を提示するのが，以降本論文の主題である．なお，言語NAIVE の言語仕様は，当論理体系の構文規則に糖衣を施したものといえる．

3．論理体系の骨子

当論理体系の骨子を述べる．

(1) 基本的枠組み　－内包論理－

当論理体系はモンタギューの内包論理[2),3)] をその基本的枠組みとしている．内包論理を枠組みとして採るのは，主に次の理由による：　①タイプの概念がある．これは，自然言語の品詞概念に対応するものであり，実世界の構造を表すのに有用である．②可能世界の概念をもつ．この概念は対象世界の動的側面を記述する論理系にとって必須である．③内包論理は，純論理的には，２ソートでタイプ付きのラムダ論理と同一視でき，世界観に関して無色である．このことはNAIVE 固有の世界観を盛り込む上で好都合である．

(2) 実体の概念　－オブジェクト論理－

本論理体系ではタイプの概念と論理定項be(nil) により実体に関する概念（実在性，種，および属性など）を形式的に表現する．すなわち，実体とはタイプe の対象のことであり，タイプe の領域，すなわち集合 D_e の要素と同一視される．したがって本論理体系では，実体の自己同一性（オブジェクト同一性）とは，すなわち，集合 D_e の要素としての自己同一性のことになる．

実体の実在性（非実在性）は論理定項be（およびnil）により表現することができる．なお，この実在性は，述語論理における存在概念とは異なるものであり，むしろ自然言語における「在る（無い）」の概念に近い．

種とは，タイプe のもとに定義されるタイプ e_1,〜, e_n のことである．タイプの間に定義される半順序関係は，種の間の階層（包含）関係を表し，オブジェクト指向で言うところの継承の現象はこれにより説明される．

属性とは，タイプ<ee>，<en>，あるいは<ek>の対象である（ここで，n およびk はそれぞれ自然数と文字列を表すタイプである）．これに対して，関係とはタイプ $<e^n\ t>$の対象のことである．なお，タイプ<ee>は，ある実体の属性が再び実体であることを示しており，これはいわゆるオブジェクト共有の概念に通じるものである．

(3) 行為の概念　－時区間論理，フレーム問題－

基本タイプとして p（行為）を導入している．本論理体系において，行為とは，ある命題（事実）を成立させることである．タイプp の整合式（行為式）の解釈はその行為の主体と，その行為が継続する時区間とが与えられてはじめて定まるものとされる．このように，当論理体系は，行為主体概念をともなう時区間論理系というべきものになっている．

当論理体系はまた，事態（言い換えればデータ）の永続性および更新を扱う．これは，いわゆるフレーム問題を導くことになるが，それに対しては，極小変化モデルの概念によって対応する．フレーム問題の解決に極小変化モデルの概念を用いることは，佐藤ら[9)] によってすでに提案されているが，当論理体系における極小変化モデルの概念はそれが述語論理に拡張され，かつ，多数のエージェントによる並行協調環境にも適用可能になっている点が特徴的である．

(4) 行為主体の概念　－並行論理－

行為主体の概念は，並行動作の単位を表現するものとして，並行論理としての当論理体系において中心的な役割を果たす（４章と５章全体がそれを説明する），また当論理体系では，行為主体は世界（場）を介して相互干渉するというモデルを採っているので，行為主体間の直接通信の概念は導入されていない．なお，当論理体系においては，世界の状態についての情報は定項が担うことになる．

(5) 世界外存在の概念　－開放型論理－

世界外存在の概念として，天使の概念が導入されている．この概念は，世界内の存在ではないが世界を変更する行為主体にはなりえるものの象徴である．これにより開放型システムにおけるシステム外からの外乱やシステムへの操作が多重並行的に行われる現象を形式的に表現することが可能になる．

① 実体と関係 ： 世界の中にはいくつかの実体が存在し，それらの間には種々の関係が成り立っている．実体とは，種が付与された個体のことである．実体は，実在性をもっている．実体はまた，いくつかの属性をもちえる．なお，成立している関係のことを事態とよぶ．世界の状態とは，事態の集まりのことである．

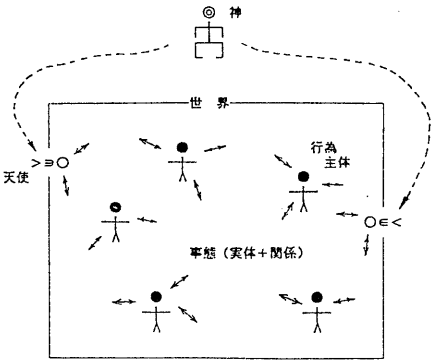
② 行為と変化 ： 世界の状態は時間とともに変化する．そして，その変化は行為によってもたらされる．行為とは，世界の状態を参照しながら新たに事態を成立（非成立）させることである．行為には脈絡がある．

③ 行為主体 ： 行為にはかならずその行為の主体が唯一存在する．行為主体になり得るのは，実体と天使だけである．行為主体たちは，それが実在しているかぎりその行動本性にしたがって，世界をながめながら，おのおの独立に行動する．

④ 神と天使 ： 世界を創成するのは神である．神の意思は天使を介して世界に伝えられる．神は，世界の不変的性質，すなわち，どんな実体および関係が世界に現れ得るか，ならびに，実体および天使がどう行動するものか（それらの行動本性）を知っている．天使は行動するが，実在しない．

⑤ 論理 ： 世界はつねに論理にしたがっている．

図－1　言語NAIVE の実世界モデル
Fig.1 The real world model of the language NAIVE.



図－2　言語NAIVE の実世界モデル（図解）
Fig.2 The figure of the real world model of NAIVE

〔飲酒する哲学者〕
問題：哲学者達が飲酒する．哲学者の傍には何個かのテーブルが置ける．それらのテーブルは複数の哲学者によって共有され得る．テーブルの上には，いろんな種類の酒ビンが何個でも置ける．哲学者達は，一度に何種類かの酒を飲みたくなり，そのときには，それらすべての種類の酒ビンが飲めるようになると飲酒をはじめる（他の哲学者が飲んでいる酒ビンは飲めない）．渇きが癒えたら，ビンは，元のテーブルに戻される．なお，哲学者達は，自分の傍にあるテーブルの上の酒しか飲めない．この饗宴のシミュレーションを行う．
〔言語NAIVE での記述〕

```
☆ (the.哲学者:p) ≡                          * 哲学者とは次のように行動するものである *
   WHILE  the.哲学者:p=be                    * すなわち，それが実在するかぎり          *
     DO                                      * 以下をおこなう                          *
        UNTIL the.哲学者:p=thirsty           * のどが渇くまで                          *
           the. 哲学者:p=tranquil            * 平静であり，のどが渇くと，              *
        WHEN (for all.want(the. 哲学者:p,*).酒種:m  * かれが欲するすべての酒種について  *
                some.(of-the. 酒種:m).(not-drunk).ビン  * 少なくとも一本づつ，だれも飲んでいない *
                  = near-the.哲学者:p )      * ビンが，かれの近くにあるという状態に    *
           DO                                * なるのを待って                          *
             FOR all.want(the.哲学者:p,*).酒種:m  * それらのビンを                      *
                the.哲学者:p=drinking-       * 飲み始める                              *
                   some.(of-the.酒種:m).(not-drunk).  *                                *
                     (near-the. 哲学者:p).ビン  *                                     *
             WHEN  the. 哲学者:p=not-thirsty  * そして渇きが癒えると                  *
                the.哲学者:p=not-drinking-all.ビン  * それらを飲むのをやめる          *
           END                               *                                       *
     END                                     *                                       *
'飲みたい :p :x :y :z' ≡                     * 飲みたいコマンドが投入されると          *
     DO                                      *                                       *
        the.哲学者:p=want-the.酒種:x  and    * その哲学者p は，（のどの渇きを覚え）    *
        the.哲学者:p=want-the.酒種:y  and    * このコマンドで指定された酒種x,y,z を    *
        the.哲学者:p=want-the.酒種:z         * 欲するようになる                        *
     END                                     *                                       *
'飲みたくない :p' ≡                          * 飲みたくないコマンドが投入されると      *
     DO(the.哲学者:p=not-thirsty)            * その哲学者は，のどの渇きが癒える        *
'哲学者生成 :p' ≡                            * 哲学者生成コマンドが投入されると        *
     DO(the.哲学者:p=be)                     * 名前p をもつ哲学者を誕生させる          *
'テーブル生成 :t' ≡                          * テーブル生成コマンドが投入されると      *
     DO(the.テーブル:t=be)                   * 名前t を持つテーブルを生成する          *
'配置 :a :b' ≡                               * 配置コマンドが投入されると              *
     DO(the.物:a=by-the.物:b)                * 指定された物a と物b を隣同士に配置する  *
'酒種 :m' ≡                                  * 酒種定義コマンドが投入されると          *
     DO(the.酒種:m=be)                       * 指定された酒種m を定義する              *
'ビン生成 :m :t' ≡                           * ビン生成コマンドが投入されると          *
     DO                                      *                                       *
       some.ビン:b=be   and                  * 新たなビンを作る                        *
       the. ビン:b=of-the.酒種:m  and        * そのとき，そのビンの酒種をm とし        *
       the. ビン:b=on-the.テーブル:t         * テーブルt の上に置く                    *
     END                                     *                                       *
(the. ビン:b=drunk) ≡                        * ビンが飲まれているということは，だれか  *
     (some. 哲学者=drinking-the.ビン:b)      * がそのビンを飲んでいるということである  *
(the. 哲学者:p=tranquil)≡                    * 哲学者が平静であるということは          *
     (the.哲学者:p=not-thirsty)              * かれののどが渇いていないということ      *
(the. 哲学者:p=thirsty) ≡                    * 哲学者ののどが渇いているということは    *
     (the.哲学者:p=want-some. 酒種)          * なにがしかの酒種を欲しているということ  *
(the. ビン:b=near-the.哲学者:p) ≡            * ビンが哲学者の近くにあるということは    *
     (the.ビン:b=on-some. テーブル:t  where   * そのビンが                            *
        (the. テーブル:t=by-the.哲学者:p  or  * その哲学者の傍にあるテーブルの上にある *
         the. 哲学者:p=by-the.テーブル:t))   * ということ                            *
```

図－3　飲酒する哲学者問題のNAIVE による仕様記述
Fig.3 The specification of the drinking philosophers problem written in NAIVE.

4. 論理体系詳論

本章では論理体系を形式的に示す.

4.1 構文論

YA. タイプ

YA1. タイプの集合

タイプの集合Tyは, 以下を満たす最小集合である.

① $e,g,t,p,n,k \in Ty$

② $e_1, \sim, e_n \in Ty$

③ $u \in Ty$ かつ $v \in Ty$ ならば $\langle uv \rangle \in Ty$

④ $u \in Ty$ ならば $\langle su \rangle \in Ty$

YA2. タイプの階層

タイプ集合上では半順序関係≦が定義されており, 以下の条件を満たしている.

① $e_1, \sim, e_n \leqq e$

② $x \leqq u$ かつ $y \leqq v$ ならば $\langle xy \rangle \leqq \langle uv \rangle$

YB. 整合式

YB1. 定項と変項

① それぞれのタイプu (タイプgを除く) に対して, 可算個の非論理定項 C_u^1, C_u^2, ～ と可算個の変項 x_u^1, x_u^2, ～ がある.
(論理定項は, YB2 の構文規則により導入される)

② すべての非論理定項は, 基礎定項と派生定項とに分類される. (なお, 誤解のない場合, 論理定項beおよびnil をそれぞれ基礎定項および派生定項の一つとして扱うことがある.)

③ 非論理定項のうち, そのタイプがp または<vp>の形のものを行為的定項, それ以外のものを非行為的定項とよぶ. (ここで, <a～<bc>～> のようなタイプも,<uc> の形をしているものとみなす. 以降同様. また, 誤解のない限り, 論理定項beおよびnil をタイプ<et>の非行為的定項として扱うことがある.)

YB2. 構文規則

(通常どおり, 整合式の集合 W_u を, 帰納的に定義できるが, 紙幅の制約のため, ここでは省略し, 後続の4.2 節のMB2 で整合式の意味規則を提示するところで併せて読み取っていただくことにする.)

4.2 意味論

MA. モデル

MA1. フレーム

E, G, N,およびK は次のような集合とする.

E : 空でない可算無限集合.

G : 空でない可算無限集合.

N : 自然数の集合であり, その上では通常の自然数論が成り立つ.

K : 文字列の集合.

なお, E, G, N,およびK は互いに素である.

この上で, フレーム $\{ D_u \}_u$ ($u \in Ty$) を次のように定める.

① $D_e = E \cup \{ u_e \}$ 実体領域

② $D_{ei} \subseteq D_e$ 各種族領域
ただし, D_{ei}は可算無限集合である.
なお, $u \leqq v$ ならば $D_u \subseteq D_v$ である.

③ $D_g = G \cup \{ u_g \}$ 天使領域

④ $D_t = \{T,F\}$ 真偽値領域

⑤ $D_p = \{T,F\}$ 行為値領域

⑥ $D_n = N$ 自然数領域

⑦ $D_k = K$ 文字列領域

⑧ $D_{\langle uv \rangle} = \{ f / f: D_u \rightarrow D_v \}$ D_u から D_v への関数領域

⑨ $D_{\langle su \rangle} = \{ f / f: S \rightarrow D_u \}$ D_u の内包化領域

ここで, $S = \{ \langle a,\langle i,j \rangle \rangle / \ a \in (D_e \cup D_g) ; \ i,j \in N, \ i \leqq j \}$

MA2. 解釈

解釈とは, 順序対 $\langle \{ D_u \}_u , h , m \rangle$ のことである. ここで,

$\{D_u\}_u$: MA1 で述べたフレーム.

h : 行為主体関数. おのおのの i∈N に対して, ($D_e \cup D_g$) の要素を割り当てる関数.

m : 意味関数. 論理定項beおよびnil ならびに各非論理定項 C_u に対して $D_{\langle su \rangle}$の要素を割り当てる関数.

正解釈 (モデル) は, 以降のMBの意味規則を満たす.

MB. 意味規則

MB1. 定項についての意味規則

① 実体定項, 数定項, および文字列定項の意味は, 行為主体や時刻にかかわらず, 不変である:
タイプe , タイプn , およびタイプk のすべての非論理定項について

$$m(C_e)(a,\langle i,j \rangle) = m(C_e)(b,\langle 0,0 \rangle) = m_e(C_e)$$
$$m(C_n)(a,\langle i,j \rangle) = m(C_n)(b,\langle 0,0 \rangle) = m_n(C_n)$$
$$m(C_k)(a,\langle i,j \rangle) = m(C_k)(b,\langle 0,0 \rangle) = m_k(C_k)$$

② 非行為的定項の意味は, 行為主体に依存しない. また時区間にも依存せず, 時点のみで決まる:
pあるいは<vp>の形でないすべてのタイプu の定項について

$$m(C_u)(a,\langle i,j \rangle) = m(C_u)(b,\langle i,k \rangle) = m_u(C_u)(i)$$

③ 派生定項の意味はそれ単独では定まらない. それは基礎定項たちの意味から定まる.

④ 世界の初期状態 :
すべての $X \in D_e$ について
$m(be)(0)(X) = F$
すべての基礎定項 C_t について
$m_t(C_t)(0) = F$
すべての基礎定項 C_p について
$m(C_p)(a,\langle 0,j \rangle) = F$
すべての基礎定項 C_{ue} について
$m_{ue}(C_{ue})(0)(X) = u_e$
すべての基礎定項 C_{un} について
$m_{un}(C_{un})(0)(X) = 0$
すべての基礎定項 C_{uk} について
$m_{uk}(C_{uk})(0)(X) =$ ブランク

⑤ 非実在物の属性と他との関係
$m_t(be)(i)(X) = F$ ならば
be以外のすべての基礎定項C について
$m_{ue}(C_{ue})(i)(*,X,*) = u_e$
$m_{ut}(C_{ut})(i)(*,X,*) = F$
$m_{up}(C_{up})(a,\langle i,j \rangle)(*,X,*) = F$
$m_{un}(C_{un})(i)(*,X,*) = 0$
$m_{uk}(C_{uk})(i)(*,X,*) =$ ブランク
(ここで, (*,X,*) は, $(X_1, \sim, X, \sim, X_m)$ の略記で, m 個のパラメタの中の一つがX で

あることを表している．）

⑥ 無行為天使 u_g は世界を変化させない：
$h(i) = u_g$ ならば
すべての非行為的基礎定項C について，
$m(C)(i) = m(C)(i-1)$
すべての行為的基礎定項D について，
$m(D)(a,\langle i,j\rangle) = F$

⑦ 非実在実体 u_e は永久に実在しない
すべての $i \in N$ について
$m\ (be)\ (i)\ (u_e) = F$

MB2. 整合式についての意味規則

任意の整合式W に対し，解釈M，$\langle a,\langle i,j\rangle\rangle \in S$，および変項への値割り当て関数$\alpha$に関する意味値を与える関数を付値関数Vと呼ぶ．ここで，変項割り当て関数αは，各変数 x_u に D_u の要素を対応させる関数である．付値関数V は，以下を満足する．なお，以降，「$V\ [M,\langle a,\langle i,j\rangle\rangle, \alpha, W]$」を「$Va,i,j,\alpha\ [W]$」と略記する．「$Va,i,j,\alpha\ [W]_u$」と書かれている場合，$_u$ は整合式W のタイプを表していると読み取られたい．

① $Va,i,j,\alpha\ [C_u] = m(C_u)(a,\langle i,j\rangle)$
（ただし，$u \neq p$，$u \neq \langle vp\rangle$のとき）
$Va,i,j,\alpha\ [C_p] = T$ iff
$h(i) = a$ かつ $h(j) = a$ かつ
$m(C_p)(a,\langle i,j\rangle) = T$
なお，C_p が基礎定項ならば，$j=i+1$．
$Va,i,j,\alpha\ [C_{vp}](X) = T$ iff
$h(i) = a$ かつ $h(j) = a$ かつ
$m(C_{vp})(a,\langle i,j\rangle)(X) = T$
なお，C_{vp}が基礎定項ならば，$j=i+1$．

② $Va,i,j,\alpha\ [x_u] = \alpha(x_u)$

③ $Va,i,j,\alpha\ [A_{uv}\ B_u]_v =$
$Va,i,j,\alpha\ [A_{uv}]\ (Va,i,j,\alpha\ [B_u])$

④ $Va,i,j,\alpha\ [\lambda x_u \cdot A_v]_{\langle uv\rangle} =$
$X \in D_u$ における値が$Va,i,j,\alpha'\ [A_v]$ に等しいような D_u から D_v の上への関数．
ここで，$\alpha' = \alpha(x/X)$．
なお，$\alpha' = \alpha(x/X)$ は xに対して Xを割り当てる以外はαと同じ変数割り当て関数のことを表す．以降同じ．

⑤ $Va,i,j,\alpha\ [A_u \equiv B_u]_t = T$ iff
$Va,i,j,\alpha\ [A_u] = Va,i,j,\alpha\ [B_u]$

⑥ $Va,i,j,\alpha\ [\uparrow A_u]_{\langle su\rangle} =$
$\langle a',\langle i',j'\rangle\rangle \in S$ に対する値が
$Va',i',j',\alpha\ [A_u]$ に等しいような，S から D_u への関数．

⑦ $Va,i,j,\alpha\ [\downarrow A_{\langle su\rangle}]_u =$
$Va,i,j,\alpha\ [A_{\langle su\rangle}]\ (a,\langle i,j\rangle)$

⑨ $Va,i,j,\alpha\ [nil\ A_e] = T$ iff
$Va,i,j,\alpha\ [be\ A_e]_t = F$

⑩ $Va,i,j,\alpha\ [\bigcirc A_t]_p = T$ iff
$j = i+1$ かつ
$h(i) = a$ かつ $h(j) = a$ かつ
$Va,j,k,\alpha\ [A_t] = T$

⑪ $Va,i,j,\alpha\ [A_p \Rightarrow B_p]_p = T$ iff
$i < m < n < j$ なるただ一組のm,n があって
$Va,i,m,\alpha\ [A_p] = T$ かつ
$Va,n,j,\alpha\ [B_p] = T$ かつ
$m < x < n$ なるすべてのx について
$h(x) \neq a$

⑫ $\exists!i,j\ (i' < i < j < j')$；
$Va,i,j,\alpha\ [\star x_A]_p = T$ if
$\alpha(x_A) = a$ かつ
$m\ (be)\ (i')\ (a) = F$ かつ
$m\ (be)\ (j')\ (a) = F$ かつ
$i' < k < j'$ なるすべてのk について
$m\ (be)\ (k)\ (a) = T$

⑬ $Va,i,j,\alpha\ [\sim A_t]_t = T$ iff
$Va,i,j,\alpha\ [A_t] = F$

⑭ $Va,i,j,\alpha\ [A_t \wedge B_t]_t = T$ iff
$Va,i,j,\alpha\ [A_t] = T$ かつ
$Va,i,j,\alpha\ [B_t] = T$

⑮ $Va,i,j,\alpha\ [A_p\ and\ B_p]_p = T$ iff
$Va,i,j,\alpha\ [A_p] = T$ かつ
$Va,i,j,\alpha\ [B_p] = T$

⑯ $Va,i,j,\alpha\ [A_t \vee B_t]_t = T$ iff
$Va,i,j,\alpha\ [A_t] = T$ または
$Va,i,j,\alpha\ [B_t] = T$

⑰ $Va,i,j,\alpha\ [A_p\ or\ B_p]_p = T$ iff
$Va,i,j,\alpha\ [A_p] = T$ または
$Va,i,j,\alpha\ [B_p] = T$

⑱ $Va,i,j,\alpha\ [A_t \rightarrow B_t]_t = T$ iff
$Va,i,j,\alpha\ [A_t] = F$ または
$Va,i,j,\alpha\ [B_t] = T$

⑲ $Va,i,j,\alpha\ [IF\ A_t\ B_p\ C_p]_p = T$ iff
$Va,i,j,\alpha\ [A_t] = T$ ならば
$Va,i,j,\alpha\ [B_p] = T$ かつ
$Va,i,j,\alpha\ [A_t] = T$ ならば
$Va,i,j,\alpha\ [C_p] = T$

⑳ $Va,i,j,\alpha\ [\Pi\ x_A\ B_t\ C_t]_t = T$ iff
x_A に対する割り当てを除いてαと同じすべての変項割り当て関数α' について，
$Va,i,j,\alpha'\ [be(x_A) \wedge B_t] = T$ ならば
$Va,i,j,\alpha'\ [C_t] = T$
注）ただし，ここで Aがn(数) のときは，$be(x_A)$ という条件は除く．以下，㉑〜㉕，㉚，および㉛についても同様．

㉑ $Va,i,j,\alpha\ [\underline{\Pi}\ x_A\ B_t\ C_p]_p = T$ iff
x_A に対する割り当てを除いてαと同じすべての変項割り当て関数α' について，
$Va,i,j,\alpha'\ [be(x_A) \wedge B_t] = T$ ならば
$Va,i,j,\alpha'\ [C_p] = T$

㉒ $Va,i,j,\alpha\ [\Sigma\ x_A\ B_t\ C_t]_t = T$ iff
x_A に対する割り当てを除いてαと同じある変項割り当て関数α' について，
$Va,i,j,\alpha'\ [be(x_A) \wedge B_t] = T$ かつ
$Va,i,j,\alpha'\ [C_t] = T$

㉓ $Va,i,j,\alpha\ [\underline{\Sigma}\ x_A\ B_t\ C_t]_p = T$ iff
x_A に対する割り当てを除いてαと同じある変項割り当て関数α' について，
$Va,i,j,\alpha'\ [be(x_A) \wedge B_t] = T$ かつ
$Va,i,j,\alpha'\ [C_p] = T$

㉓' $Va,i,j,\alpha\ [\underline{\Sigma}\ x_A\ true\ \bigcirc be(x_A)] = T$ iff
x_A に対する割り当てを除いてαと同じある変項割り当て関数α' について，
$Va,i,j,\alpha'\ [nil\ (x_A)] = T$ かつ
$Va,i,j,\alpha'\ [\bigcirc be(x_A)] = T$

㉔ $Va,i,j,\alpha\ [\Theta N_n\ x_A\ B_t\ C_t]_t = T$ iff
x_A の割り当て値が異なる他はαと全く同じで，かつ相互に異なる N個の割り当て関数

$\alpha_1 \sim \alpha_N$ があって，$1 \leqq k \leqq N$ なるすべてのk について

$Va,i,j,\alpha_k\ 〔be(x_A) \wedge B_t〕 = T$ かつ

$Va,i,j,\alpha\ 〔C_t〕 = T$

㉕ $Va,i,j,\alpha\ 〔\underline{\Theta}\ N_n\ x_A\ B_t\ C_t〕_p = T$ iff

x_A の割り当て値が異なる他はαと全く同じで，かつ相互に異なる N個の割り当て関数 $\alpha_1 \sim \alpha_N$ があって，$1 \leqq k \leqq N$ なるすべてのk について

$Va,i,j,\alpha_k\ 〔be(x_A) \wedge B_t〕 = T$ かつ

$Va,i,j,\alpha\ 〔C_p〕 = T$

㉖ $Va,i,j,\alpha\ 〔REPEAT\ N_n\ B_p〕_p = T$ iff

$Va,i,j,\alpha\ 〔IF\ (N_n \equiv 0)\ ◎true\ (B_p \Rightarrow (REPEAT\ (N_n - 1)\ B_p))〕 = T$

㉗ $Va,i,j,\alpha\ 〔WHEN\ A_t\ B_p〕_p = T$ iff

$Va,i,j,\alpha\ 〔IF\ A_t\ B_p\ (◎true \Rightarrow (WHEN\ A_t\ B_p))〕 = T$

㉘ $Va,i,j,\alpha\ 〔WHILE\ A_t\ B_p〕_p = T$ iff

$Va,i,j,\alpha\ 〔IF\ A_t\ (B_p \Rightarrow (WHILE\ A_t\ B_p))\ ◎true〕 = T$

㉙ $Va,i,j,\alpha\ 〔(not\ A_{<ut>})\ B_u〕_t = T$ iff

$Va,i,j,\alpha\ 〔A_{<ut>}\ B_u〕 = F$

㉚ $Va,i,j,\alpha\ 〔\#\ x_A\ B_t〕_n =$

$Va,i,j,\alpha'\ 〔be(x_A) \wedge B_t〕 = T$ となるようなα' の個数．ここで，α' は，x_A に対する割り当てを除いてαと同じある変項への値割り当て関数．

㉛ $Va,i,j,\alpha\ 〔sum\ x_A\ B_t\ C_n〕_n =$

$Va,i,j,\alpha'\ 〔be(x_A) \wedge B_t〕 = T$ となるようなすべてのα' についての

$Va,i,j,\alpha'\ 〔C_n〕$ の総和．

ここで，α' は，x_A に対する割り当てを除いてαと同じある変項への値割り当て関数．

（その他＋，×，＝，＜などの通常の算術演算子，数字および真偽定項true,falseは通常の意味をもつ）

4.3 語用論（仕様，操作，実行，極小変化モデル）

P．語用規則

① 仕様 ： システムの仕様とは，次の形のタイプtの整合式のことである．

$(\uparrow A_u^{\ 1} \equiv \uparrow B_u^{\ 1}) \wedge \cdots \wedge (\uparrow A_v^{\ n} \equiv \uparrow B_v^{\ n})$

ここで，整合式 $A_u^{\ 1}$，～，$A_v^{\ n}$ は，たかだか一つの非論理派生定項といくつかの変項のみから構成されているものでなければならない．

② システムの起動 ： 仕様 S_t のシステムを起動するということは，

$\forall \beta\ ; Vu,0,0,\beta\ 〔S_t〕 = T$

なるモデル制約（モデルに対して課せられる制約条件）を与えることである．

③ システムの操作 ： 時点k に操作コマンド A_p を投入するとは，

$\exists !i,j\ (k \leqq i < j)\ ;$

$Va,i,j,\alpha\ 〔A_p〕 = T$ かつ

$\forall x\ (x < i \vee j < x \rightarrow h(x) \neq a)$

なるモデル制約を与えることである．

ここで，a は，u ではない D_g の要素であり，他の操作では使われていないものであること．

また，αは，操作コマンドのパラメタを指定することに相当する．

④ システムの実行 ： システムの実行とは，1つの仕様と複数の操作によって課せられる（上述②および③の意味での）モデル制約を満足するモデルのうちの「極小変化モデル」を構成することである．

⑤ 極小変化モデル ： 極小変化モデルM とは次の条件を満たすモデルのことである．

M は極小変化モデルである≡

M はモデルである ∧

$\forall X$ 【〔X はモデルである ∧ $M_h = X_h$〕 →

$\forall k$ 〔M はX と時点(k-1) まで同一履歴である →

時点k でのM の変化はX の変化と対等か小さい〕】

ここで，

A はB と時点k まで同一履歴である ≡

$(\forall i\ (i \leqq k \rightarrow A_{mt}(i) = B_{mt}(i)) \wedge \forall i\ (i \leqq k \rightarrow A_{mp}(i) = B_{mp}(i)))$

時点k でのA の変化はB の変化と対等か小さい≡

$\sim \{(diff\ (A_{mt}(k), A_{mt}(k-1)) \cup A_{mp}(k)) \supset (diff\ (B_{mt}(k), B_{mt}(k-1)) \cup B_{mp}(k))\}$

A_h ： モデルA の行為主体関数

$A_{mt}(i) = \{ <C,X;Y> / C$：非行為的基礎定項，$m_t(C)(i)(X) = Y$，ただし，m はモデルA の意味関数

$A_{mp}(i) = \{ <C,<a,j>,X> /$ C は行為的基礎定項 $m(C)(a,<i-1,j>)(X) = T$，ただし，m はモデルA の意味関数

$diff\ (A,B) = \{ <C,X> / (<C,X;Y> \in A$ かつ $\sim (<C,X;Y> \in B)$ または $(<C,X;Y> \in B$ かつ $\sim (<C,X;Y> \in A)\}$

5．いくつかの論理計算の例

4章で示した論理体系を適用する論理計算の例をいくつか示す．

〔例1〕単純な連接

仕様 ： $\uparrow C_p \equiv \uparrow (◎ P_t \Rightarrow (◎ Q_t \Rightarrow ◎ R_t))$

操作 ： 時点2 で操作コマンド C_p を投入する

ここで，C,P,Q,およびR はすべて定項とする．このうち，基礎定項は，P,Q，およびR である．

これのモデルについて考える．

---

(1)$\forall \beta$； $Vu,0,0,\beta\ 〔\uparrow C \equiv \uparrow (◎P \Rightarrow (◎Q \Rightarrow ◎R))〕 = T$ （∵ P②）

(2)$\exists !i,j\ (2 \leqq i < j)$；

$Va,i,j,\alpha\ 〔C〕 = T$ かつ

$\forall x\ (x < i \vee j < x \rightarrow h(x) \neq a)$ （∵ P③）

(3)$\forall \beta$； $Vu,0,0,\beta\ 〔\uparrow C〕 =$

$Vu,0,0,\beta\ 〔\uparrow (◎P \Rightarrow (◎Q \Rightarrow ◎R))〕$

（∵(1), MB2 ⑤）

(4)$Va,i,j,\alpha\ 〔C〕$

$= (Vu,0,0,\alpha\ 〔\uparrow C〕)\ (a,<i,j>)$ （∵ MB2⑥）

$= (Vu,0,0,\alpha\ 〔\uparrow (◎P \Rightarrow (◎Q \Rightarrow ◎R))〕)\ (a,<i,j>)$ （∵(3)）

$= Va,i,j,\alpha\ 〔◎P \Rightarrow (◎Q \Rightarrow ◎R)〕$ （∵ MB2⑥）

(5)$\exists !i,j\ (2 \leqq i < j)$；

$Va,i,j,\alpha\ 〔◎P \Rightarrow (◎Q \Rightarrow ◎R)〕 = T$ かつ

$\forall x\ (x < i \vee j < x \rightarrow h(x) \neq a)$

（∵(2), (4)）

(6)∃!i,j (2 ≦i <j);
∃!i1,i2 (i<i1<i2<j);
∀a,i,i1, α〔◎P〕= T かつ
∀a,i2,j, α〔◎Q ⇒◎R〕= T かつ
∀x (i1<x <i2 → h(x)≠a) かつ
∀x (x <i ∨ j<x → h(x)≠a)
(∵(5), MB2 ⑪)

(7)∃!i,i1,i2,i3,i4,j
(2≦i <i1<i2<i3<i4<j);
∀a,i,i1, α〔◎P〕= T かつ
∀a,i2,i3,α〔◎Q〕= T かつ
∀a,i4,j, α〔◎R〕= T かつ
∀x (i1<x <i2→ h(x)≠a) かつ
∀x (i3<x <i4→ h(x)≠a) かつ
∀x (x <i ∨ j<x → h(x)≠a)
(∵(6), MB2 ⑪)

(8)∃!i,i1,i2,i3,i4,j
(2≦i <i1<i2<i3<i4<j);
h(i)=a, h(i1)=a, h(i2)=a, h(i3)=a,
h(i4)=a, h(j)=a , かつ
i1=i+1, i3=i2+1, j=i4+1 かつ
∀a,i1,k1,α〔P〕= T かつ
∀a,i3,k3,α〔Q〕= T かつ
∀a,j,k,α〔R〕= T かつ
∀x (i1<x <i2 → h(x)≠a) かつ
∀x (i3<x <i4 → h(x)≠a) かつ
∀x (x <i ∨ j<x → h(x)≠a)
(∵(7), MB2 ⑩)

(9)∃!i,i1,i2,i3,i4,j
(2≦i <i1<i2<i3<i4<j);
h(i)=a, h(i1)=a, h(i2)=a, h(i3)=a,
h(i4)=a, h(j)=a で, かつ
i1=i+1, i3=i2+1, j=i4+1 かつ
m_t (P)(i1) = T かつ
m_t (Q)(i3) = T かつ
m_t (R)(j)= T かつ
∀x (i1<x <i2 → h(x)≠a) かつ
∀x (i3<x <i4 → h(x)≠a) かつ
∀x (x <i ∨ j<x → h(x)≠a)
(∵(8), MB2①, MB1②)

これより，たとえば，次の解釈は，本例のモデルの一つであることがわかる．

s	0	1	2	3	4	5	6	7	8
h	u	u	a	a	a	a	a	a	u
P	0	0	0	1	1	1	1	1	1
Q	0	0	0	0	0	1	1	1	1
R	0	0	0	0	0	0	0	1	1

（ここでs の行は時点,hの行は行為主体を表す．見やすさのため，真偽(TまたはF)はそれぞれ1 と0 で表す．また，派生定項についての解釈は省略する．以降同様）
なお，このモデルは極小変化モデルでもある．極小変化性についての考察例は例４で示す．

〔例２〕並行動作

仕様 : ↑ C_p ≡↑ (◎ P_t ⇒◎ (～ P_t)) ∧
↑ D_p ≡↑ (◎(Q_t) ⇒ (IF ～ P_t
◎(～ Q_t) ◎true))

操作 : 時点2 で操作コマンド C_p を投入し，時点3 で操作コマンド D_p を投入する．

ここで，C,D,P,およびQ は，すべて定項とする．このうち，基礎定項はP とQ である．

これのモデルについて考える．

(1)∀τ ; ∀u,0,0,τ〔↑C ≡↑ (◎ P⇔
◎ (～ P))〕=T (∵ P②)
(2)∀τ ; ∀u,0,0,τ〔↑D ≡↑ (◎ Q⇔
(IF ～P ◎(～Q) ◎true))〕= T
(∵ P②)
(3)∃!i,j (2 ≦i <j);
∀a,i,j,α〔C〕= T かつ
∀x (x <i ∨ j<x → h(x)≠a) (∵ P③)
(4)∃!v,w (3 ≦v <w);
∀b,v,w,β〔D〕= T かつ
∀x (x <v ∨ w<x → h(x)≠b) (∵ P③)
途中を省略するが，例１と同様にして（ただし新たに MB2 ⑲も使う），
(5)∃!i,i1,i2,j (2 ≦i <i1<i2<j);
h(i)=a, h(i1)=a, h(i2)=a, h(j)=a かつ
i1=i+1, j=i2+1 かつ
m_t (P)(i1) = T かつ
m_t (P)(j)= F かつ
∀x (i1<x <i2 → h(x)≠a) かつ
∀x (x <i ∨ j<x → h(x)≠a)
((3), (1)より)
(6)∃!v,v1,v2,w (3 ≦v <v1<v2<w);
h(v)=b, h(v1)=b, h(v2)=b, h(w)=b かつ
v1=v+1, w=v2+1 かつ
m_t (Q)(v1) = T かつ
m_t (P)(v2) = F ならば m_t (Q)(w)= F
m_t (P)(v2) = T ならば m_t (true)(w) = T
かつ
∀x (v1<x <v2 → h(x)≠b) かつ
∀x (x <v ∨ w<x → h(x)≠b)
((4), (2)より)

(5), (6)およびMB1 より，たとえば次の解釈はモデルの一つであることがわかる（極小変化モデルでもある）．

s	0	1	2	3	4	5	6	7	8	9	10
h	u	u	a	a	b	b	a	a	b	b	u
P	0	0	0	1	1	1	1	0	0	0	0
Q	0	0	0	0	0	1	1	1	1	0	0

〔例３〕属性の継承

ANT,INSECT∈Ty で，ANT ≦INSECT≦ e のとき，
∀a,i,j,α〔 (Π x_{INSECT} true
LEG $_{en}$(x_{INSECT})≡ 6)
→ (Π y_{ANT} true $LEG_{en}(y_{ANT})$ ≡ 6) 〕= T
であることを示す．

ANT ≦INSECT≦e なので，MA1 ②から，
$D_{ANT} \subseteq D_{INSECT}$.
これと MB2⑳およびMB2 ⑱から，本例の命題が成り立つのは明らかである．

本例は直観的には，蟻の属性（足の数）が昆虫のそれを継承することを本論理体系の中で記述したものである

〔例４〕イェール射撃問題[5)]

仕様 : 〔↑ $YALE_p$ ≡↑(◎$LOADED_t$ ⇒ $SHOOT_p$) ∧
↑ $SHOOT_p$ ≡↑((IF $LOADED_t$ ◎$DEAD_t$
◎true) and ◎(～$LOADED_t$))〕

操作 : 時点2 で操作コマンド $YALE_p$ を投入する．

ここで，YALE,LOADED,DEAD, およびSHOOT はすべて定項とする．このうち，基礎定項はLOADEDと DEAD である．これのモデルについて考える．

(1)$\forall \beta$; Vu,0,0,β 〔↑C ≡
↑ (◎LOADED⇒SHOOT) 〕= T　(∵ P②)
(2)$\forall \beta$; Vu,0,0,β 〔↑SHOOT ≡↑((IF LOADED
◎DEAD ◎true) and ◎ (～LOADED)) 〕=T
(∵ P②)
(3)∃!i,j (2 ≦i <j);
Va,i,j,α 〔C 〕= T　かつ
∀x (x <i ∨ j<x → h(x)≠a) (∵ P③)
途中を省略するが，これまでの例と同様にして，
(4)∃!i,i1,i2,j (2 ≦i <i1<i2<j);
h(i)=a, h(i1)=a, h(i2)=a, h(j)=a　かつ
i1=i+1　かつ
m_t (LOADED)(i1) = T　かつ
m_t (LOADED)(i2) = T　ならば
j = i2 + 1　かつ
m_t (DEAD)(j) = T　かつ
m_t (LOADED)(i2) = F　ならば
j = i2 + 1　かつ
m_t (true)(j) = T　かつ
m_t (LOADED)(j) = F　かつ
∀x (i1<x <i2 → h(x)≠a)　かつ
∀x (x <i ∨ j<x → h(x)≠a)

(4)およびMB1 より，たとえば，以下の解釈A はモデルの一つであることが分かる．

<解釈A>

s	0	1	2	3	4	5	6	7	8
h	u	u	a	a	u	u	a	a	u
LOADED	0	0	0	1	1	1	1	0	0
DEAD	0	0	0	0	0	0	0	1	1

ここで，行為主体関数は解釈A と同じで，時点6 でのLOADEDを偽(0) と解釈するもう一つのモデルB を考える．

<解釈B>

s	0	1	2	3	4	5	6	7	8
h	u	u	a	a	u	u	a	a	u
LOADED	0	0	0	1	1	1	0	0	0
DEAD	0	0	0	0	0	0	0	0	0

モデルA とB を比べたとき，両者は時点5 まで同一履歴であるが，時点6 においてB はA より大きな変化をしている（すなわち，B は P⑤の条件を満たしていない）．したがってB は極小変化モデルではない．

一方，詳細な議論は省略するが，P ⑤の条件にてらせば，解釈A は明らかに極小変化モデルの一つである．そしてこの解釈A は，与えられた仕様と操作のもとで常識的に期待されるシステムの実行結果とも確かに一致している．

6．おわりに

言語NAIVE の論理体系について述べた．これにより，オブジェクトの論理，行為の論理，および並行協調の論理などを統合した一つの論理体系を提案した．本論理体系は，論理パラダイムとオブジェクト指向パラダイムの融合というのに近いが，むしろ，実世界の自然な記述ということをより根本的に問い直すところから生まれたものであった．

論理体系構成の技術的詳細の観点からも，本論理体系には次のようないくつかの特徴的な点があった．

① 行為主体の概念．これはプロセスの概念とオブジェクトの概念を統合したものである．これにより，いわゆるオブジェクトの自己同一性だけでなく，プロセスの自己同一性も同時に扱えるようになっている．

② 並行協調環境におけるデータの永続性と更新に関して生じる問題，すなわち並行協調環境におけるフレーム問題の一つの解決法を提示している．その解決法においては，極小変化モデルの概念に加えて，行為主体の概念ならびに基本定項と派生定項の区別が本質的に効いている．

③ 通常の時間論理と異なり，命題式（タイプｔ式）と行為式（タイプｐ式）とを区別し，行為式についての自由な論理的複合を制限している．このことは時間論理としてのNAIVE の記述力を制限することになっているが，そのことが同時に，NAIVE で記述した仕様の実行可能性を保証し，かつ物理法則的（マルコフ的）で人間に理解しやすい仕様記述を促すという利点にもつながっている．

今後は，さらに推敲を加えるとともに，他の論理体系との比較もすすめたい．

参考文献

1) Chandy,K,M. and Misra,J : The Drinking Philosophers Problem, ACM Transactions on Programming Languages and Systems, Vol.6, No.4, pp.632-646, 1984
2) Dowty,D.R., Peter,S. and Wall,R.E. : Introduction to Montague Semantics, D.Reidel, Dordrecht, Germany, 1981
3) Gallin,D : Intensional and Higher-Order Modal Logic, North-Holland, 1975
4) Halpern,J.Y. and Shoham,Y. : A Propositional Modal Logic of Time Intervals, J.ACM, Vol.38, No.4, pp.935-962, 1991
5) Hanks,S., McDermott,D. : Default Reasoning, Nonmonotonic Logics, and the Frame Problem, Proc. of AAAI-86, 1986
6) 日野 : 日常的世界観にもとづく言語NAIVE による実行可能な仕様記述，情報処理学会研究報告，Vol.92, No.10, pp.49-56, 1992
7) Kifer,M. and Lausen,G. : F-Logic: A Higher-Order Language for Reasoning about Objects, Inheritance, and Scheme, 1989
8) 大堀 : オブジェクト指向データベースの形式化，情報処理，Vol.32, No.5, pp.550-558, 1991
9) 佐藤ほか : 最小変化－フレーム問題への１アプローチ，人工知能学会全国大会（第１回）論文集，1987
10) 横田 : 演繹オブジェクト指向データベースについて，コンピュータソフトウェア，Vol.9, No.4, pp.3-18, 1992