1F−5

日常的世界観にもとづく実行可能な仕様記述言語 NAIVE — その言語仕様と記述例 —

日野克重
富士通㈱

1．はじめに

さきに，実行可能な仕様記述言語NAIVE の概要と，その背後に設定した世界観について述べた〔2〕．本稿では，その世界観を反映するように設計された言語NAIVE の具体的言語仕様を示す．さらに仕様記述例を示し，当言語のソフトウェア開発上の利点について議論する．

2．言語仕様

言語NAIVE の基本詞を表−1に示す（これだけでは，当言語の言語仕様を正確に述べたことにならないが，その点は，3章の記述例で補うことにする）．

(1) 意味論的観点からの特徴

意味論的観点からは，次の特徴がある．

(a) 当言語の言語仕様は，日野〔2〕で示した日常的世界観を反映するよう設計されている，すなわち，述語論理を包含するとともに，行為主体（エージェント）やそれらの間の並行協調動作などを自然に記述するための基本詞を備えたものになっている．

(b) ポインタ，プロセス，データ入出力，および同期通信などの，計算機やＯＳに関わる概念は排除されている．

(2) 構文論的観点からの特徴

構文規則は，以下のような点に留意して設計されている．

(a) 基本詞は，日常談話においてよく使われる自然語の中の易しい語彙と対応がある．また，構文も自然語の構文に近い．

(b) 自然語と同様の品詞分類がある．そして，使用するコトバの品詞は文脈から判断されるので，明に宣言する必要がない．すなわち，データ宣言のようなことはする必要がない．

(c) 世界観を明に意識するようなメタ用語（例．class，method，precondition，procedureなど）は記述させない．

(d) 日常談話では省略しないようなコトバは，当言語でも明に記述させる．（たとえば，Prologとは異なり，all やsomeなどの限定詞は明に記述させる．）

こうした配慮は，従来のソフトウェア言語では閑却視されてきたことであるが，仕様記述言語の記述性，理解性，習得のしやすさ，および，実世界記述の快適性の点で効果があるものと思われる．

表−1　言語NAIVE の基本詞

品詞分類	基本詞	意　味
定義詞	≡	行為主体の行動本性，ならびに，コマンドや単文の意味を定義するのに使う．自然語における「〜とは〜ということである」という表現に相当する．
叙述詞	＝ ≠	単文の肯定．自然語における「〜である（is）」に相当する． 単文の否定．自然語における「〜でない（is not）」に相当する．
普通名詞	物 数 字 （その他任意に定義可能）	種（クラス）を指定するものである．
実在詞	be nil	実在性．自然語における「有る（there is）」に相当する． 非実在性．自然語における「無い（there is not）」に相当する．
属性詞	（任意）	自然語における「〜の父」や「〜の持物」などの属性名詞に相当する．
関係詞	（任意）	自然語における形容詞や前置詞に相当する．
行為詞	（任意）	自然語における動詞に相当する．
限定詞	all some many the	自然語における「すべての」に相当する． 自然語における「ある」に相当する． 自然語における「いくつの」に相当する． 自然語における「その」に相当する．
論理詞	～ and or → not	文否定． 連言． 選言． 含意． 形容否定
接続詞 （様相詞）	☆ DO ⇨ IF WHEN WHILE UNTIL REPEAT and or WHERE FOR	行動本性．実体の行動本性を表す．（通常省略される．） 行為化．自然語における「〜であるようにする」に相当する． 連接．自然語における「〜して，その次に〜する」に相当する．（通常省略される．） 判断．自然語における「もし〜ならば」に相当する． 待機．自然語における「〜になったとき」に相当する． 継続．自然語における「〜であるあいだ」に相当する． 継続．自然語における「〜になるまで」に相当する． 反復．自然語における「何回〜する」に相当する． 同時実行． 選択実行． 修飾．自然語における「ただし〜である」に相当する． 限定．
算術詞	＋，−，＊，／，＃（集合の要素数），sum(合計）など	
数詞	（任意の数字の列）	
助詞	: . ..	普通名詞と固有名詞あるいは指示詞との連結． 限定詞，関係詞（句），および普通名詞の連結． 属性詞の連結．
固有名詞	（任意）	個体を示す定数．
指示詞	（任意）	変数．
補助記号	括弧や引用符など．	

3．記述例

図−1に，飲酒する哲学者の問題〔1〕とNAIVE による記述例を示す．本問題は，哲学者の食事問題をさらに一般化したものであり，マルチエージェントによる並行協調動作や多少複雑な論理的処理の要素が盛り込まれた問題である．

The Specification Language NAIVE based on Everyday World View - its Language Specification and Example -
Katsushige HINO
FUJITSU Ltd.

〔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ているかもしれない．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，飲んでいたビンは，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.need(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).                * 少なくとも一本づつ，                    *
              (near-the.哲学者:p).ビン=not-drunk  * かれの近くにだれも飲んでいないビンが  *
        DO                                        * あるという状態になるのを待って          *
          FOR all.need(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=need-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=need-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=need-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 *
'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=need-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－1　飲酒する哲学者問題のNAIVE による仕様記述
Fig 1　The specification of the drinking philosophers problem written in NAIVE.

この程度の並行・論理問題は，実世界では日常茶飯に見られるものであるが，それでもこれを，メッセージパシング方式にもとづくオブジェクト指向言語や，Prologのような既存の論理型言語で記述しようとすると，かなり複雑なプログラムになるであろう．

その点NAIVE ではこの問題を図－1に示したように簡潔に記述できる．具体的には次のような利点が見て取れる．

(a) 自然語で実世界を記述するのに近い感じで問題を記述できる．

(b) 問題文に現れる語彙と仕様記述に現れる語彙は，おおむね一致している．仕様記述上には，問題文に現れない状態変数や中間変数は現れないし，計算機やOSに関わる概念も一切現れていない．そのため，与えられた問題と記述した仕様との一致性が見やすくなっている．

(c) all やsomeを含む複雑な条件での待機（同期）処理が接続詞WHENを使って簡潔かつ宣言的に書けている．このことにより，仕様（プログラム）の正当性の検証が容易になる．たとえば，生存性（liveness property ；のどが渇いた哲学者はいずれ酒を飲めること）や安全性（safety property ；同じ酒ビンを複数の哲学者が同時に飲むことはないこと）の要請が満たされていることは，仕様記述上からほぼ自明的に読み取れる．

(d) システム動作中の環境の動的変更（これは動的システムに特有の事象である）に対処するために特別の考慮を払うことなく仕様記述できる．この例題の場合，システム実行中に動的に哲学者やテーブルや酒ビンが新たに追加されてもよいようになっているが，そのことを特に考慮して仕様記述してはいない．

(e) 述語論理や並行処理になじみがないプログラマにとっても読み書きしやすい．

4．おわりに

言語NAIVE の言語仕様と記述例を示した．当言語は，本稿で示したような並行協調問題のみならず，データベース操作を伴う事務処理システムなどの，いわば実世界模写型のソフトウェアシステムを自然かつ簡潔に記述でき，その生産性および品質の向上効果が著しいことが確められている．今後は，当言語を立脚点として，ソフトウェア工学の諸問題を再考してみたい．

謝辞

本研究について日頃御指導戴く東京大学　田中英彦教授ならびに当社技術担当部長　河田汎博士に深く感謝致します．

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) 日野克重 ： 日常的世界観にもとづく実行可能な仕様記述言語NAIVE －その概要と世界観－，本予稿集，1992