

1 T-5

LOTOS による CCR サービス定義の形式記述

データ型部

INTAP 研究開発委員会プロトコル形式記述 WG

岡田康治(電総研)、内山光一(東芝)、奥村薫(日本IBM)、小野昌秀(沖)、五ノ井敏行(富士通)、

高橋薫(東北大)、藤尾光彦(シャープ)、藤田朋生(日本電気)、前田誠(日立)

1 まえがき

我々は、INTAPの形式記述WG(プロトコル記述サブWG)の活動として、OSIのCCRのサービス定義をISOで開発された形式記述言語LOTOSを用いて記述した[1]。全体についての説明は別稿[2]に譲り、本稿では、データ型定義に関する記述について報告する。

LOTOSにおけるデータ型定義の例は、LOTOS国際規格、ガイドライン文書の他に、幾つかの個別のプロトコルの形式記述の試みにおいて見ることができる。今、これらによって、小規模ながら、ある一定の記述の流儀が確立しようとしている。しかし、その記述技術の現状は、まだ未発達であり、さらに多くの者によって記述され、議論され、検討されるべき段階にあると考えられる。この観点から、本稿では、我々のCCR記述のデータ型定義のうち、既存の記述には見られない、新たな記述を採用した点を重点的に取り上げ、説明している。

2 記述の概要

定義されたデータ型は、ライブラリ中のものを除いて、12個あり、全539行中55%にあたる295行を要している。これらの参照関係を図1に示す。これらは、主として、本CCRサービス定義記述で中心となるデータであるCSP、すなわち、CCRサービスプリミティブ(の集合)を定義するものであって、データ型CSPParameterSelectorsがこれにあたる。その上位のCSPQueueは、一方のアクセスポイントからの入力履歴を表現するために設けた、CSPの特殊なキューである。

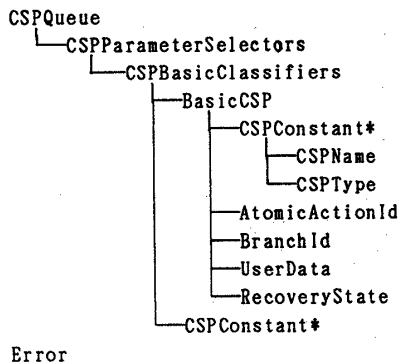


図1 データ型定義間の関係

A Formal Description of CCR service definition in LOTOS - Data Types Part -
 K.Okada(ETL), M.Uchiyama(Toshiba), K.Okumura(IBM Japan),
 M.Ono(Oki), T.Gonoi(Fujitsu), K.Takahashi(Tohoku Univ.),
 M.Fujio(Sharp), T.Fujita(NEC), M.Maeda(Hitachi)

3 記述の説明

3.1 CSP 授受における非決定的挙動の表現

通信においては、情報の伝達の非決定性が常に扱われる。CCRにおいても、一方のアクセスポイントからのCSPの系列において、RollReqが生じたとき、他方に、RollIndが他に先駆けて伝わるか、時間的に先行するCSPがそのまま伝わるか、の非決定性がある。本記述では、CSPQueueというデータ型をCSPの系列として定義し、それに

```

remove : CSP, SPQueue → SPQueue
contain : CSP, SPQueue → Bool
    
```

という二つの関数を定義して、プロセス定義での非決定性の記述を助けている。
 process Relation[sup, sub] の定義の一部を例として示す。

```

(*- receive ROLLinds -*)
[ contain( ...
  i; sup!ROLLind; ...
  ( [ not contain( ...
    []
    [ contain( ...
  )
    
```

3.2 サービスプリミティブの固定部分

サービスプリミティブCSPの固定部分、例えば、BEGreqは、従来、定数演算子として定義されてきている。本記述では、これを機能を表わす要素Begと通信の相を表わす要素Reqとの組み合わせと見て、2項演算子cを構成子とする項で表わすようにした(CSPName, CSPType, CSPConstant)。

```

c : CSPName, CSPType → CSPConstant
c(Beg, Req) = BEGreq
...
    
```

これにより、CSPの固定部分(CSPConstant)の属性を定義する関数の定義が合理的になり、等式の数が飛躍的に減少した。例えば、

```

IsReq(BEGreq) = true,    IndOf(BEGreq) = BEGind
    
```

というような等式は、他の同様の形式をもつ多くの等式と共に、それぞれ、

```

IsReq(c(x, Req)) = true,    IndOf(c(x, Req)) = c(x, Ind)
    
```

で、まとめて表現してしまえることができた。

3.3 サービスプリミティブの表現

CSP は、幾つかのデータを伴うので、SP の全体は、 $BEGreq(a,b,u)$ 、 $ROLLreq(u)$ 等の形で表わされることが多い。しかし、付随するデータは、4 個のパターンに分類できるので、本記述では、それらに対して、CSP を構成する構成子を与えて、CSP の固定部分を一要素 (第 1 引数) として定義した。

```
f1: CSPConstant, AAID, BID, DATA → CSP
f2: CSPConstant, DATA → CSP
f3: CSPConstant, RSTATE, AAID, BID, DATA → CSP
f4: CSPConstant, DATA, AAID, BID, DATA → CSP
f1(BEGreq, a, b, u) = BEGreq(a,b,u)
f3(ROLLreq, u) = ROLLreq(u)
```

これにより、CSP の属性を定義する関数について前小節と同様の改善が得られた。例えば、Udata という関数には、

$$Udata(BEGreq(a,b,u)) = u$$

というような等式が CSP の種類だけ必要だったが、改善後は、

$$Udata(f1(c, a, b, u)) = u$$

というタイプの等式がパターンの数である 4 個で済んだ。

3.4 複数の標準形の積極的採用

3.2, 3.3 の 2 つの小節において導入した表現形 (例、 $c(Beg, Req)$ 、 $f1(BEGreq, a, b, u)$) は、実は、従来型の項 (例、 $BEGreq$ 、 $BEGreq(a,b,u)$) の代替物としてではなく、等式を使って、もうひとつの標準形として与えられている。即ち、言語の意味定義上は、項の商集合の各要素に、明らかなひとつの代表元がない記述を積極的に採用している。この結果、等式数が少なく、関数の定義法がより合理的であり、理解性が高まった記述が得られた。これは、LOTOS 言語の機能をフルに利用することにもなっている。一方、欠点は、項書き換え型の実行系に、このままでは、かからないことである。

3.5 合成型サービスプリミティブの表現

CCR サービスプリミティブの中には、 $BEGreq$ と $COMMreq$ の二つのプリミティブを連結して発行できるものがある。この記述法について、単一型と合成型を平たくひとつのデータ型に押し込める方法と、まづ単一型だけを定義して、それを参照するデータ型として合成型を定義する方法との間で議論があったが、今回の最終の記述は、前者に依っている。

3.6 部分関数の問題

データ型を定義する過程で、議論したことに、関数の (値の) 定義をどの程度厳密に行なうかという問題がある。例えば、 $PRPreq(u)$ という CSP は、データとして、ユーザデータのみをもつのであるが、 BID という関数は、

$$BID: CSP \rightarrow BID$$

とすべての CSP について宣言されているので、 $BID(PRPreq(u))$ という項は BID 型の項集合中に存在する。これに対して、 $Undef$ とかの定義されないことを表わす特殊な値を宣言して等式で結びつけることもできるが、本記述では、これをこのままにした。すなわち、項の商集合中に、これを唯一の元とする集合をもつということであり、意図としては、 BID という関数を、 $PRPreq(u)$ については定義されない、部分関数とするということである。関数の定義されない部分を参照することのないように、上位のデータ型やプロセスで注意することとした。

3.7 等価性判定関数の定義

列挙して宣言された n 個の要素からなるデータ型の各データについて、その等価性判定関数をどう定義するかについて、本記述では、既存の方法にならって、データに自然数値を割り付ける関数 h を定義し、その値の等価性で判定している。

```
h: CSPTyp → Nat
forall d, d1: CSPTyp
d eq d1 = h(d) eq h(d1)
```

この種の、コーディングをデータに対して行なうことは仕様記述としては好ましくない面があるようにも感じられ、より直感的でコンパクトな (等式数が少ない) 記法の開発が待たれる点である。

3.8 任意データを含むプリミティブの定義

普通、サービスプリミティブは、幾つかのデータを伴うが、そのうちのいくつかがオプションと指定されていることがよくある。このとき、プリミティブの形態は、可変となるわけで、これを代数的仕様法で忠実に実現しようとする、ひとつのプリミティブに対して、いくつもの演算子の宣言をすることになり、はなはだ繁雑である。本記述では、すべてのデータを伴う形態のみを定義している。

4 あとがき

比較的小さな記述であったにもかかわらず、興味深く、多くの議論を経て、今回の記述をまとめた。結論として、LOTOS のデータ型定義部分は、まだ記述法が確立しておらず、検討・改善すべき点も多く残っているので、今後の研究や記述実績まだ必要であると感じられた。

謝辞

本研究は、新エネルギー産業技術総合開発機構 (NEDO) の委託を受け、INTAP が行なっている通産省工業技術院大型プロジェクト「電子計算機相互運用データベースシステム」の成果である。

参考文献

- [1] Protocol Formal Description WG of INTAP: "LOTOS description of the service definition for the Commitment, Concurrency and Recovery service element", (Dec. 1990)
- [2] 藤田 他: "LOTOS による CCR サービス定義の形式記述 - 全体報告 -", 情報処理学会第 42 回全国大会 (Mar. 1991)