LOTOSによる交換サービス仕様の記述

1R−8

山野 敬一郎*　大友 弥生*　更科 克幸*　安藤 津芳*　太田 正孝*　高橋 薫**
*(株)高度通信システム研究所　**東北大学

1.はじめに

近年，通信プロトコルの厳密な仕様化を行うために，各種の形式的な記述技法(FDT)が提案されている．

このFDTの1つとして開発されたLOTOSは，数学的モデルをベースとしており記述能力が高く，言語自体に検証能力を持つため，その活用が期待されている．しかし，OSIなどへの適用は十分な検討がなされているが，一般的な交換サービスなどについては適用例が少ない．

従って本報告では，従来自然言語やSDLによってい記述されていた交換サービスをとりあげ，LOTOSによる形式的仕様記述の評価・検討を行う．これによって，その記述の妥当性を調べるとともに，LOTOSの記述上の特徴をどれだけ交換サービスに生かせるか，さらに検証への適用などを考えていくことを目的とする．

2.交換サービス仕様

まず，記述の対象となる交換サービスの仕様を示し，どのような観点でLOTOSによる記述を行うかを述べる．

図1は，回線交換基本サービスの，一般的な正常呼設定を示した基本的なシーケンスである．

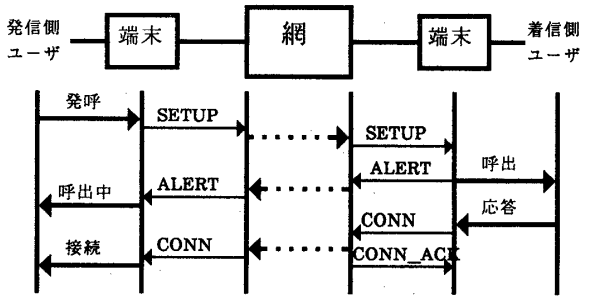

図1. 回線交換サービス

呼設定のプロセスは，発信側ユーザが発呼の操作を端末に対して行い，網にSETUPメッセージを送出することにより，起動される．そして，呼出の後，着信側ユーザが応答すると，CONNメッセージが送られ，回線が接続されて通信が行えるようになる．

この仕様記述を行うにあたって，多くの記述モデルが考えられる．例えば，仕様記述の対象としては，ユーザ,端末,網,およびこれらの組合せが考えられる．さらに，単純な要求仕様レベルのものから，実装までを考えた設計仕様レベルのものまで，記述上のレベルもさまざまである．

ここでは，ユーザ要求として，利用者から見た端末のふるまい，実装を考えた端末および網のふるまいの仕様記述を行うこととする．

3.LOTOSによる仕様記述

図1のシーケンスに対して，図2のような記述モデル(プロセス)を考え，仕様記述を行う．

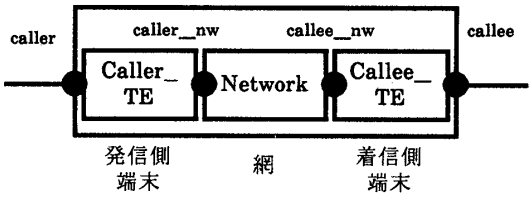

図2. 交換システムのモデル

交換システムとして，まず発信側・着信側端末を含んだプロセスを考える．環境は，発信者および着信者となるユーザであり，それぞれcaller, calleeゲートを通じて通信を行う．この考えによって記述したサービス仕様が，図3に示す記述例である．

ここでは，プロセスの内部はブラック・ボックスとみなされる．従って，図1のSETUPなど，端末と網との間でやり取りされる信号は含まれない．つまり，ユーザ仕様となっている．

次に，システムの内部動作を記述し，ユーザ仕様から実装を考えた設計仕様にする方法を示す．

システムの内部としては，発信側端末・着信側端末および網を考え，ここではそれぞれCaller_TE, Callee_TE, Networkプロセスに対応させ，内部ゲートcaller_nw, callee_nwによって通信する．これらは，内部イベントとして環境からは隠されており，環境から見たプロセスの仕様は，ユーザ仕様と同様である．図4に，設計仕様レベルに詳細化した交換サービスの仕様を示す．

4.考察

LOTOSの記述スタイルとしては，さまざまなものがあるが，交換サービスの仕様記述に適用する上で，次のような点に留意する必要がある．

・情報のやりとりが明確にわかる．

Specification of Switching System Services in LOTOS

Keiichirou YAMANO*, Yayoi OHTOMO*, Katsuyuki SARASHINA*, Tsuyoshi ANDO*,
Masataka OHTA*, Kaoru TAKAHASHI**
*Advanced Intelligent Communication System Lab.　**Tohoku University

```
specification Switching_System_Service[caller,callee]:noexit
type Action is
    sorts   action
    opns    offhook     :   →action (* 発呼、応答 *)
            onhook      :   →action (* 終話      *)
            ringing     :   →action (* 呼出      *)
            ringingback :   →action (* 呼出中    *)
            connect     :   →action (* 接続      *)
            disind      :   →action (* 切断通知  *)
endtype
type State is
    sorts   state
    opns    free        :   →state
            busy        :   →state
            conversation:   →state
endtype
behaviour
    Service[caller,callee](free)
where
process Service[caller,callee](s:state):noexit:=
    [s=free]→
        caller?x:action[x=offhook];
        (calleelringing; exit ||| callerlringingback; exit)
        >> Service[caller,callee](busy)
    [][s=busy]→
        callee?x:action[x=offhook];
        callerlconnect;
        Service[caller,callee](conversation)
    [][s=conversation]
        ..... (* conversation process *)
endproc (* Service *)
endspec (* Switching_System_Service *)
```

図3. LOTOS仕様記述例1(ユーザ要求の仕様記述)

・要求仕様と設計仕様が関連づけられる.

・インプリメントしやすい.

本報告ではこれらを考慮し，リソース指向・状態指向のスタイルでのプロセス記述を行った．システムのリソース，および状態を明確に表現することによって，実際の交換ソフトウェアのインプリメントに即した記述が行える．

また，**hide**などを用いた階層的な記述によってモジュール化が行え，内部仕様の変更がほかに影響を与えず，ユーザ仕様から設計仕様へと記述を詳細化していくことができる．

5.まとめ

本報告では，交換サービスの基本的なモデルについて，**LOTOS**による仕様記述を行い，その評価・検討を行った．

現在は，実際の回線交換サービスの仕様についての記述，および検討を行っている．

今後の課題としては，実際のサービスを記述するため，網の内部処理，信号のパラメータをどのように表現するか，新サービスの付加をどのように記述していくかを検討し，**LOTOS**による交換サービスの仕様記述，および検証を行う．

[参考文献]

[1]Chris A.Vissers, et al. : "Architecture and Specification Style in Formal Descriptions of Distributed Systems," Protocol Specification, Testing, and Verification Ⅷ, North-Holland (1988)

[2]M.Faci, et al.:"Formal Specification of Telephone Systems in LOTOS," Protocol Specification, Testing, and Verification Ⅸ, North-Holland (1990)

```
specification Switching_System_Service[caller,callee]:noexit
type Action is
    sorts   action
    opns    offhook     :   →action (* 発呼、応答 *)
            onhook      :   →action (* 終話      *)
            ringing     :   →action (* 呼出      *)
            ringingback :   →action (* 呼出中    *)
            connect     :   →action (* 接続      *)
            disind      :   →action (* 切断通知  *)
endtype
type Signal is
    sorts   signal
    opns    SETUP       :   →signal
            ALERT       :   →signal
            CONN        :   →signal
            CONN_ACK    :   →signal
            DISC        :   →signal
            REL         :   →signal
            REL_COMP    :   →signal
endtype
type State is
    sorts   state
    opns    free        :   →state
            busy        :   →state
            conversation:   →state
endtype
behaviour
    Service[caller,callee]
where
process Service[caller,callee]:noexit:=
hide caller_nw, callee_nw in
    (Caller_TE[caller,caller_nw](free) |||
                Callee_TE[callee,callee_nw](free))
    |[caller_nw,callee_nw]|
    Network[caller_nw,callee_nw]
    where
    process Caller_TE[caller,caller_nw](s:state):noexit:=
        [s=free]→
            caller?x:action[x=offhook];
            caller_nw!SETUP;
            Caller_TE[caller,caller_nw](busy)
        [][s=busy]→
            caller_nw?y:signal[y=ALERT];
            callerlringingback;
            caller_nw?y:signal[y=CONN];
            callerlconnect;
            Caller_TE[caller,caller_nw](conversation)
        [][s=conversation]→
            ..... (* conversation process *)
    endproc (* Callee_TE *)
    process Callee_TE[callee,callee_nw](s:state):noexit:=
        [s=free]→
            callee_nw?x:signal[x=SETUP];
            (calleelringing; exit ||| callee_nw!ALERT; exit)
            >> Callee_TE[callee,callee_nw](busy)
        [][s=busy]→
            callee?x:action[x=offhook];
            callee_nw!CONN;
            callee_nw?y:signal[y=CONN_ACK];
            Callee_TE[callee,callee_nw](conversation)
        [][s=conversation]→
            ..... (* conversation process *)
    endproc (* Callee_TE *)
    process Network[caller_nw,callee_nw]:noexit:=
        caller_nw?x:signal[x=SETUP];
        callee_nw!SETUP;
        Network[caller_nw,callee_nw]
        []
        callee_nw?y:signal[y=ALERT];
        caller_nw!ALERT;
        Network[caller_nw,callee_nw]
        []
        callee_nw?y:signal[y=CONN];
        (caller_nw!CONN; exit ||| callee_nw!CONN_ACK; exit)
        >> Network[caller_nw,callee_nw]
    endproc (* Network *)
endproc (* Service *)
endspec (* Switching_System_Service *)
```

図4. LOTOS仕様記述例2(交換サービス仕様)