サービス競合における異常な状態への遷移の検出法

高見一正† 井上泰彰†† 太田 理††

新しい通信サービスを既存サービスに追加する場合に，新サービスと既存サービス間の相互作用によって引き起こされるサービス競合は通信サービスを迅速に開発するうえでの大きな障害となっており，サービス競合検出法は重要な課題となっている．一方，最近の IN (Intelligent Network) の研究においては，通信システムに関する専門的な知識を必要とする交換機制御ソフトウェアの開発者に加えて，専門的な知識が無くても新しいサービスを提案できるスキルを持ったサービス開発者（非専門家）が通信サービスソフトウェアを設計できる支援手法が要求されている．本論文では，非専門家が形式的言語により記述したサービス仕様間の競合を検出することを目的とする．サービス競合を状態遷移の意味的矛盾として形式的に定義し，その定義に基づく自動検出法を提案する．本論文では，サービス仕様は端末状態とイベントに基づく状態遷移モデルで定義する．また，その状態遷移関係は if-then 型ルールの集合で規定し，1つのルールが1つの状態遷移を表す．次に，意味的矛盾として2つのサービス仕様を同時に提供した場合に生成される異常な状態（単独サービス仕様には規定されていない状態）への遷移の定義と機械的な検出手法を提案する．更に，本手法に基づくサービス仕様競合の机上検出実験結果についても述べる．

A Detection Method of Transition to Abnormal State in Service Interaction

Kazumasa Takami,† Yasuaki Inoue†† and Tadashi Ohta††

When a new service is added to a telecommunication system, designers must resolve service interactions between the new service and existing services. These service interactions become a serious obstacle to rapid development of new services. Therefore, the service interaction detection method is an important issue. In contrast, as a characteristic trend of service development in IN (intelligent network), a new service design support method that facilitates new service software design by service designers (so-called non-experts who have no professional knowledge of communication systems), is studied. In this paper, service specifications are described with formal description language by non-experts. The service interaction is formally defined as a semantically inconsistency of state transition. An automatic detection method based on the definition is proposed. In this paper, a state transition model based on terminal states and its events is adopted as a specification description model. Also, the state transitions are described by a set of if-then type rules whose rule represents one state transition. A transition to abnormal state, that is not prescribed in individual service specifications and that is generated when executing two services simultaneously, is defined as a semantically inconsistent. An automatic detection method of the transition to abnormal state is proposed. Moreover, experimental results of the proposed method are described.

1. はじめに

情報社会の進展に伴い，多種多様な通信サービスが要求されており，サービスの迅速な開発と提供が求められている．通信サービスの開発は要求仕様を明確化し，その仕様を実現したソフトウェアにより提供される．新サービスの追加にあたっては，新しい通信サービスの仕様を規定し，その新サービスを通信システムに追加した場合の正常動作を検証する必要がある．この検証は開発工数を考慮すると，なるべく上流工程で行うことが望ましい．サービス数の増加に伴い新しいサービスと既存サービス間におけるサービス競合を人手によって検出し，解消することは益々困難となり，コンピュータの支援による機械的な手法を確立する必要がある[1),2)]．一方，最近の IN (Intelligent Network) の研究においては，通信システムに関する専門的な知

† NTTネットワークサービスシステム研究所高機能処理研究部ネ式G
Intelligent Network Systems Laboratory, NTT Communication Switching Laboratories
†† ATR 通信システム研究所通信ソフトウェア研究室
Communication Software Department, ATR Communication Systems Research Laboratories

識を必要とする交換機制御ソフトウェアの開発者に加えて，専門的な知識が無くても新しいサービスを提案できるスキルを持ったサービス開発者（非専門家）が通信サービスソフトウェアを設計できる支援手法が要求されている[3]．非専門家でもサービス仕様が定義できるためには，通信システムをブラックボックスとして利用者の視点から記述できる必要がある．また，サービス競合検出法には，既存サービス仕様の詳細を知識がなくてもサービス競合動作を検出できる支援法が非専門家には要求される．

これまでに，サービス仕様レベルでの競合検出法として文献 4)〜6) が提案されている．文献 4) は，サービスを実現する通信システムの内部処理，プロトコルおよび既存サービス仕様に関する詳細な知識を必要とする方法であり，専門家を対象とした検出支援法である．文献 5) では通信システムをブラックボックスとして利用者の視点からサービス仕様を LOTOS で記述する．LOTOS のインタプリタを用いてサービスを設計者がシミュレーションしながら競合を検出する手法であり，機械的には検出できない．文献 6) は，サービス仕様を if-then 型のルール集合で記述し，ルール競合として状態遷移の非決定性が機械的に検出できる手法である．また，知識処理分野においてルール集合間の矛盾回避，無矛盾保持のためのシステムとして TMS, ATMS が提案されている[7),8)]．更に，ルール集合間の制約をチェックする手法として述語論理等による手法がデータベースの分野で提案されている[8)]．しかし，これらはルール集合間の論理的一貫性保持および論理的矛盾検出のための手法であり，サービス仕様間の意味的矛盾による競合検出法としてはまだ提案されていない．一方，文献 9) において，意味的矛盾を検出する手法が提案されている．しかし，本手法は検出するための意味的矛盾を定義した知識を事前に獲得し，設定しておく必要がある．

本論文では，サービス仕様は状態遷移モデルで定義され，既存サービス仕様に新しいサービス仕様を追加した場合に発生する意味的矛盾に着目したサービス競合検出支援法を提案する．具体的には，意味的矛盾として，既存サービス仕様と新規サービス仕様には定義されていない状態への遷移（以下，「異常な状態への遷移」と呼ぶ）を定義し，その検出法を提案する．2章では，異常な状態への遷移について定義する．3章では，異常な状態への遷移を機械的に検出するための仕様記述言語への要求条件について示す．また，この要求条件を満たす言語として状態遷移を if-then 型のルール集合として記述できる言語について示す．4章では，異常な状態への遷移の検出アルゴリズムを提案する．5章で具体的なサービス仕様記述例に基づき，提案手法を説明する．6章では，提案した異常な状態への遷移の検出に基づくサービス競合検出の机上実験結果と考察を示す．本稿のまとめを7章で述べる．

2. 異常な状態への遷移の定義

本章では，異常な状態への遷移を定義するための前提条件について説明し，定義を与える．

2.1 サービス仕様競合検証支援と前提条件

異常な状態への遷移の検出法の概要を**図1**に示す．新規と既存の2つのサービス仕様を機械的に合成し，2つの被合成サービスと合成サービスのそれぞれの状態遷移を比較・判定しながら仕様競合を検出する．なお，本稿で用いる「合成」とは，2つのサービス仕様を機械的に組み合わせる過程までのことを意味する．この過程では，生成されたサービス仕様の矛盾は，まだ解消されていない．次に，図1に示すような検出法を実現するためのサービス仕様の前提条件を以下に示す．

（1） サービス仕様は状態遷移モデルに基づく仕様記述法で記述されている．

（2） サービスを合成する前の各サービス仕様は共に正常な状態遷移をすることが設計者により確認されている．例えば，個々のサービス仕様において状態遷移

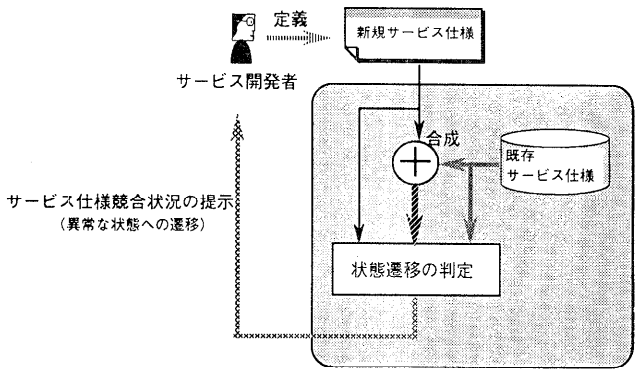


図1 サービス仕様競合検出の概要
Fig. 1 An overview of service specification interactions detection system.

の非決定性，デッドロック，ループ等の論理的矛盾は解消されている．

2.2 サービス仕様

異常な状態への遷移を検出するための前提となるサービス仕様定義について述べる．

定義 1：サービス仕様

サービス仕様は，通信システムをブラックボックスとし，端末の状態と端末から入力されるイベントに基づく状態遷移モデルとして以下のように定義する．

$SPEC=\langle T, S, E, \Phi, s_0, f_0\rangle$

ここで，

a）T は端末の集合である．要素を t で表す．

b）S はサービスの状態の集合であり，要素を s で表す．s は端末の状態の集合として定義される．端末 t の状態を $\tau(t)$ で表し，サービスの状態 s に含まれる端末数を n とすると，$s=\{\tau(t_1), \tau(t_2), \cdots, \tau(t_n)\}$ となる．また，端末状態 $\tau(t)$ は，より細分化された状態である状態記述要素 p の集合として定義される．状態記述要素 p は以下の 2 種類の表記を持つ．

- $p(t)$：端末 t 自身の状態を表記する時．
- $p(t, \gamma t)$：端末 t の状態が，T に属する t 以外の端末と関係のある状態を表記する時．ここで，γt は端末 t と関連がある任意の端末を示す．

c）E は端末から入力されるイベントの集合であり，要素を e で表す．イベント e は以下の 2 種類の表記がある．

- $e(t)$：端末から入力され，端末 t の状態を変化させるイベントを示す．
- $e(t, \gamma t)$：端末 t から入力され，端末 t と関連のある端末 γt の状態も変化させるイベントであることを示す．

d）Φ はサービス状態の遷移関数である．

$s_j=\Phi(s_i, e_i)$

$(e_i \in E$ かつ $s_i, s_j \in S)$

これは，サービスの現状態 s_i において，イベント e_i が入力されると次状態 s_j に遷移する，ことを示している．

e）s_0 は初期状態である．$s_0 \in S$

f）f_0 は最終状態である．$f_0 \in S$　□

電話交換における着信転送サービスを例にとり，定義 1 を説明する．着信転送サービスは，例えば別の相手と通話中で着信を受け付けられない場合に，事前に登録した転送先に着呼を転送するサービスである．**図 2** は着信転送サービスの一部を端末状態の遷移により示したものである．本サービスは 4 端末で構成され，$T=\{t_1, t_2, t_3, t_4\}$ である．状態 s_i は 4 端末で構成され，端末 t_2 と t_3 が通話中で，端末 t_2 から t_4 に転送登録され，かつ端末 t_1 が t_2 にダイヤルするために受話器を上げた状態を示している．すなわち，$s_i=\{\tau(t_1), \tau(t_2), \tau(t_3), \tau(t_4)\}$ となる．更に，各端末の状態は状態記述要素を用いて，以下のように記述される．

$\tau(t_1)=$ {ダイヤル可(t_1)}

$\tau(t_2)=$ {通話中(t_2, t_1)，着信転送設定(t_2, t_4)}

$\tau(t_3)=$ {通話中(t_3, t_2)}

$\tau(t_4)=$ {空(t_4)}

ダイヤル可(t_1)，空(t_4) が 1 つの端末の状態を表記する状態記述要素の例である．また，通話中(t_2, t_1)，着信転送設定(t_2, t_4) 等が他の端末と関係のある状態を表記する状態記述要素の例である．関連する端末間は図 2 では線または矢印で結び示している．

次に，イベントの例について説明する．1 つの端末だけの状態を変化されるイベントは，*offhook*($t1$) で

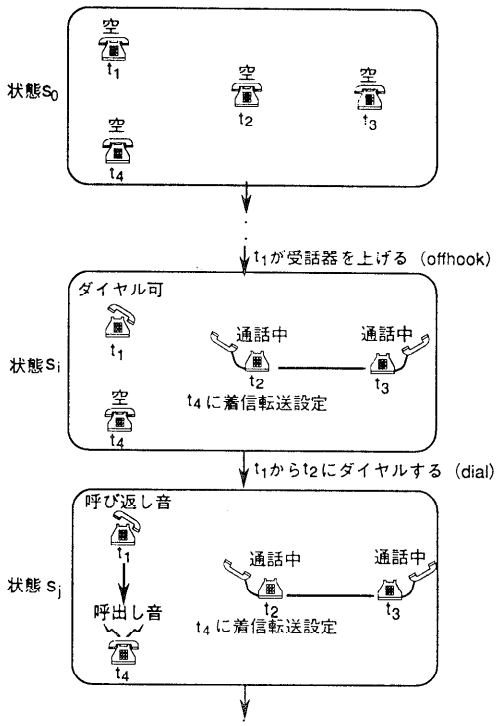


図 2 端末の状態遷移とイベントに基づくサービス仕様記述例（着信転送サービス仕様の例）

Fig. 2 An example of service specification description with terminal state transitions and events (A call-forwarding service example).

あり，関連する他の端末の状態も変化させるイベントは $dial(t1, t2)$ である．図2に示すように $dial(t1, t2)$ イベントによる状態 s_i から s_j (呼びが端末 t_4 に転送され，端末 t_1 から呼び出されている状態) への遷移が遷移関数 Φ により定義される．状態 s_j では，端末 t_1 と t_4 の状態が以下のように変化している．

$\tau(t_1)=$ {呼び返し音(t_1, t_4)}

$\tau(t_4)=$ {呼出し音(t_4, t_1)}

2.3 異常な状態への遷移

本節では，「異常な状態への遷移」を定義し，具体例により説明する．

2つの単独サービス仕様，Xサービス仕様とYサービス仕様から X と Y サービスの合成サービス仕様を設計する場合を考える．X サービスと Y サービスをそれぞれの仕様に基づき実行させ，**図3**に示すような X と Y サービスが重なり合うような状態 s_i に遷移したと仮定する．このような状態において，XとYの何方か一方のサービス，例えばXサービスを実行させたとすると，新たに重なりあった次状態 s_j が生成される．この次状態における Y サービスに関連した状態 s_{yj} は，Y サービス仕様に規定されていない可能性がある．この Y サービスに関連した状態 s_{yj} が Y サービス仕様に規定されていない場合，次状態において Y サービスを実行できない．このように，2つのサービス仕様を合成して新たなサービス仕様を設計する場合，一方のサービス仕様の実行により，他方のサービスの実行が妨害されるような状態遷移を「異常な状態への遷移」とし，以下のように定義する．

定義 2：異常な状態への遷移

X と Y サービスの仕様 $SPEC_X$ と $SPEC_Y$ を以下のように定義する．

$SPEC_X=\langle T_X, S_X, E_X, \Phi_X, s_{X0}, f_{X0}\rangle$

$SPEC_Y=\langle T_Y, S_Y, E_Y, \Phi_Y, s_{Y0}, f_{Y0}\rangle$

X と Y サービスの状態 s_{xi} と s_{yi} において，XとYサービスが重なりあった状態 $s_i=\{s_{xi} \cup s_{yi}\}$ が生成されたとする．s_{xi} は n 個の端末により，s_{yi} は m 個の端末によりそれぞれ定義され，$\{s_{xi} \cap s_{yi}\}$ の状態に k ($\leqq \min(n, m)$) 個の端末が含まれているとする．

この状態 s_i において，X サービスの仕様が実行され，X サービスの次状態 s_{xj} が生成されたとする．Xサービスの状態遷移によって，X と Y サービスが重なりあった状態に含まれる k 個の端末のうち q ($\leqq k$) 個の端末の状態が変化したとする．変化しなかった $k-q$ 個の端末の状態を ε_i とする．新たに重なり合った状態 s_j は以下のようになる．

$s_{xj}=\Phi_x(s_{xi}, e_{xi})$

$s_j=\{s_{xj} \cup \{s_{yi}-\{s_{xi} \cap s_{yi}\}\} \cup \varepsilon_i\}$

状態 s_j において，Y サービスに関する端末の状態を s_{yj} とすると，s_{yj} は以下のようになる．

$s_{yj}=\{\{s_{yi}-\{s_{xi} \cap s_{yi}\}\} \cup \kappa_j\}$

ここで，κ_j は状態 s_j において，X と Y サービスが重なりあっている状態に含まれる k 個の端末の状態を表す．この s_{yj} が，$s_{yj} \subseteq S_Y$ を満足しない時，異常な状態への遷移と定義する． □

次に異常な状態への遷移の具体例を示す．着信転送サービス仕様と着信拒否サービス仕様を同時に提供したときに発生する異常な状態への遷移を示す．以下の説明で用いる着信転送サービスと着信拒否サービスの仕様の一部を**表1**に示す．

表1の着信拒否サービス仕様を定義1に基づく状態遷移モデルにより記述した例を**図4**に示す．着信転送サービスについては，図2の状態 s_i と s_j に相当する．図4の状態 s_i は端末 t_4 と t_1 から構成されており，端末 t_4 に端末 t_1 からの着信拒否登録が設定されている状態を示している．状態 s_i は端末の状態を状態記述要素で表し，以下のように記述できる．

$s_i=\{\tau(t_1), \tau(t_4)\}$

$\tau(t_1)=$ {ダイヤル可(t_1)}

$\tau(t_4)=$ {空(t_4), 着信拒否登録(t_4, t_1)}

また，同図の次状態 s_j は端末 t_1 が端末 t_4 にダイヤルしたため，着信拒否サービスが起動され，端末 t_1 が

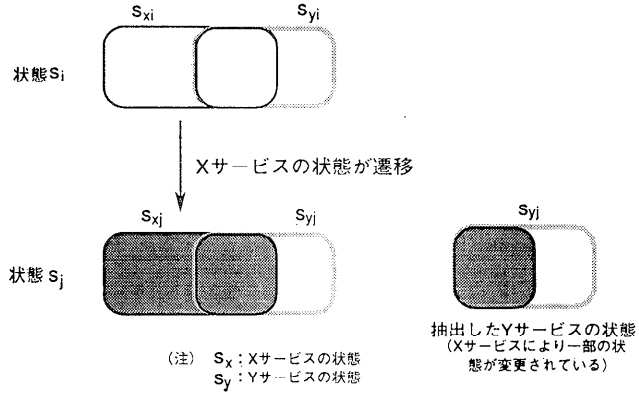


図 3 合成サービスの状態遷移

Fig. 3 A state transition in a composite service.

着信拒否を示すビジー音を受信している状態に変化したことを示している．

$s_j = \{\tau(t_1), \tau(t_4)\}$

$\tau(t_1) = \{$着信拒否のビジー音$(t_1, t_4)\}$

$\tau(t_4) = \{$空(t_4)，着信拒否登録$(t_4, t_1)\}$

次に，**図5**に，着信転送サービス仕様（X の添字で表す）と着信拒否サービス仕様（Y の添字で表す）を同時に提供した時に生成される2つのサービスの重なり合った状態とその後の遷移を示す．図5の前状態 s_i は，表1に記述されている2つの現状態の和集合をとった状態 $\{s_{xi} \cup s_{yi}\}$ を示している．この状態 s_i は以下に示す端末の状態を表している．

- 端末 t_1 においてダイヤルトーンが鳴っている．
- 端末 t_2 と t_3 が通話している．
- 端末 t_2 が端末 t_4 に転送を登録している．
- 端末 t_1 から t_4 への着信を禁止している．
- 端末 t_4 が空き状態となっている．

また，状態 s_i を形式的に表現すると以下のようになる．

$s_i = \{\tau(t_1), \tau(t_2), \tau(t_3), \tau(t_4)\}$

$s_{xi} = \{\tau(t_1), \tau(t_2), \tau(t_3), \tau(t_4)\}$

$s_{yi} = \{\tau(t_1), \tau(t_4)\}$

$\tau(t_1) = \{$ダイヤル可$(t_1)\}$

$\tau(t_2) = \{$通話中(t_2, t_1)，着信転送設定$(t_2, t_4)\}$

$\tau(t_3) = \{$通話中$(t_3, t_2)\}$

$\tau(t_4) = \{$空(t_4)，着信拒否登録$(t_4, t_1)\}$

s_i を構成する着信転送サービスと着信拒否サービスの端末数はそれぞれ n，m とすると，$n=4$，$m=2$ である．状態 $\{s_{xi} \cap s_{yi}\}$ に含まれる端末は t_1 と t_4 の2端末であり，$k=2$ となる．

図5の次状態 s_j は，表1に示した着信転送サービス仕様を規定している状態遷移に従って，その状態遷移の現状態を次状態に遷移させたときに出現する状態を示している．端末 t_1 と t_4 の状態が共に変化しており，$q=2$ である．

$s_j = \{\tau(t_1), \tau(t_2), \tau(t_3), \tau(t_4)\}$

$s_{xj} = \{\tau(t_1), \tau(t_2), \tau(t_3), \tau(t_4)\}$

$s_{yj} = \{\{\phi\} \cup \{\tau(t_1), \tau(t_4)\}\}$

$= \{\tau(t_1), \tau(t_4)\}$

$\tau(t_1) = \{$呼び返し音$(t_1, t_4)\}$

$\tau(t_2) = \{$通話中(t_2, t_1)，着信転送設定$(t_2, t_4)\}$

$\tau(t_3) = \{$通話中$(t_3, t_2)\}$

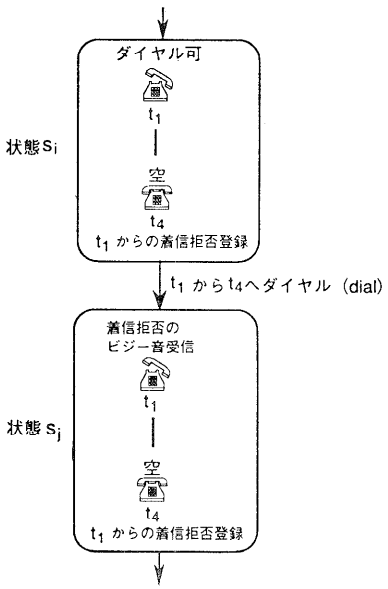


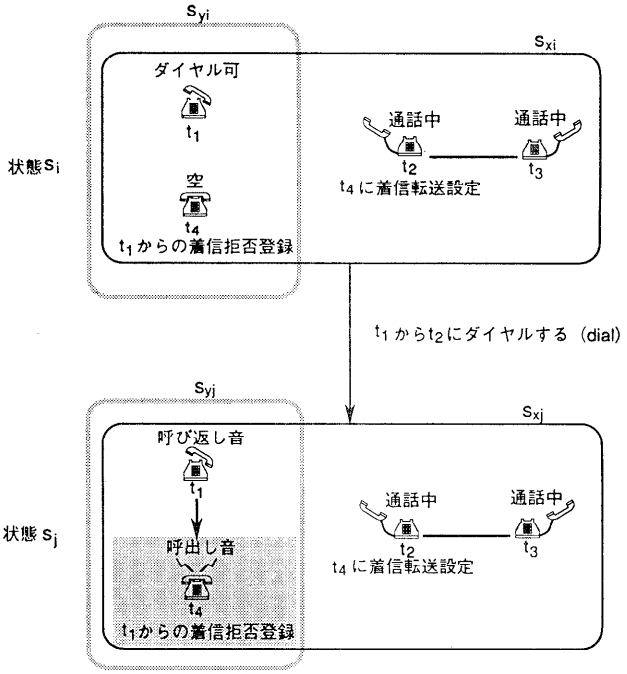
図 4　着信拒否サービス仕様の状態遷移例
Fig. 4　A state transition specified in terminating-call-screening service.

表 1　着信転送サービスと着信拒否サービスを規定している状態遷移
Table 1　State transitions specified in call-forwarding and terminating-call-screening service specifications.

	着信転送サービスの状態遷移	着信拒否サービスの状態遷移
イベント発生前の状態	t1端末においてダイヤルトーンがなっており、t2端末がt3端末と通話しており、かつt4端末に転送を登録しており、t4端末が空き状態となっている状態	t1端末においてダイヤルトーンがなっており、かつt1端末からt4端末への着信を禁止する機能が起動されており、t4端末が空き状態となっている状態
イベント	t1端末がt2端末に発呼すること	t1端末がt4端末に発呼すること
イベント発生後の状態	t1端末がt4端末を呼び出しており、t2端末がt3端末と通話しており、かつt4端末に転送を登録している状態	t1端末において発信ができなかったことを示す音がなっており、かつt1端末からt4端末への着信を禁止する機能が起動されており、t4端末が空き状態となっている状態

$\tau(t_4)$= {呼出し音(t_4, t_1),
着信拒否登録(t_4, t_1)}

この状態遷移において，図5の次状態 s_j における着信拒否サービスの状態 s_{yj} に着目する．この状態 s_{yj} は，端末 t_4 が端末 t_1 から呼び出されており，かつ端末 t_4 において端末 t_1 からの着信を禁止する機能が起動されている状態である．この端末 t_4 の状態 {呼出し音(t_4, t_1), 着信拒否登録(t_4, t_1)} は，着信拒否サービス仕様には規定されていない状態（このサービスの本来の意味として禁止されねばならない状態）である．従って，図5の前状態 s_i に示された状態から次状態 s_j に示された状態への遷移は，異常な状態への遷移である．



（注）S_{yj} は着信拒否サービスとしては矛盾した状態である。

図5 着信転送サービスと着信拒否サービスを合成した場合の異常な状態への遷移の例
Fig. 5 An example of transition to abnormal state when compounding call-forwarding and terminating-call-screening service specifications.

3. 仕様記述言語

本章では，通信サービス仕様から異常な状態への遷移を機械的に検出するために要求される仕様記述言語の機能について述べる．

複数の単独サービス仕様を同時に提供したときに発生する異常な状態への遷移を機械的に検出するためには，通信サービス仕様を記述する仕様記述言語は，少なくとも以下に示す2つの機能を有する必要がある．

(1)形式的に記述できること．

(2)単独サービス仕様の機械的な足し合わせ（和集合をとる）ができること．

上記(1)と(2)の機能は，定義1で示したサービス仕様の遷移関数 Φ を単一の状態遷移を規定したルールの集合とそれらのルールの適用関数を規定することにより実現できる．このルールは現状態，イベント，次状態から構成する．現状態と次状態は，定義1で示した端末の状態を細分化した状態記述要素 p の集合として表現される．ルールの集合と適用関数により状態遷移を規定することにより，2つのサービス仕様の足し合わせは，単独サービス仕様を規定しているルールの集合の和集合として実現できる．このようなルールに基づく状態遷移記述を以下に定義する．

定義 3：ルールによる状態遷移記述

サービス仕様 SPEC の遷移関数 Φ は以下のように定義することができる．

$\Phi=\langle R, \Sigma\rangle$

ここで，

a) R はルールの集合を示す．要素を r で表す．r は if-then 型のルールであり，条件部と動作部をコロン(：)で区切り，以下のように記述する．

r=cs e : ns.

cs：現状態を表し，端末の状態記述要素 p を用いて記述する．

ns：次状態を表し，端末の状態記述要素 p を用いて記述する．

e：遷移の起因となるイベントを示す．$e\in E$

b) Σ はルールの適用関数であり，以下の2つの関数から構成される．ここで，s_i と s_j も状態記述要素 p を用いて表現される．

$r=M(s_i, e)$：

関数 M はルールのマッピング関数であり，イベント e が選択された時にサービス状態 s_i に対して，ルール r の条件部 cs が $s_i\supseteq$cs を満足するルールを抽出する．なお，サービス仕様が確定した時点においては，ルール競合は解消され，唯一

のルールが決定される．

$s_j = W(r, s_i)$：

関数 W はルール r が適用された結果，s_i をそのルールの動作部に記述されている内容に従って書き換え，次状態 s_j を生成する関数である．□

図2の着信転送サービスの状態 s_i から s_j への遷移をルール r_1 として記述した例を以下に示す．

$r_1 = cs_1 \quad e_1 : ns_1.$

$cs_1 = \{$ダイヤル可(t_1), 通話中(t_2, t_1),
着信転送設定(t_2, t_4), 通話中(t_3, t_2),
空$(t_4)\}$

$e_1 = \mathrm{dial}(t_1, t_2)$

$ns_1 = \{$呼び返し音(t_1, t_4), 通話中(t_2, t_1),
着信転送設定(t_2, t_4), 通話中(t_3, t_2),
呼出し音$(t_4, t_1)\}$

図2の例では，サービス状態 s_i は r_1 の現状態 cs_1 と等しいので，ルールの適用条件の1つである $s_i \supseteq cs_1$ は満足している．また，イベント $\mathrm{dial}(t_1, t_4)$ が選択されているため，ルール r_1 が適用でき $(r_1 - M(s_i, \mathrm{dial}(t_1, t_2))$，$s_i$ をルールの次状態記述 ns_1 で書き換え，次状態 s_j が生成される $(s_j = W(r_1, s_i))$．

定義3において，Σ はルール適用を規定した関数であり，サービスに依存して変わることはない．従ってサービスの仕様は R により規定される．定義3に基づくサービス仕様記述言語として if-then 型のルール形式で記述する STR (State Transition Rules) 言語[10]を開発した．STR 言語を用いて既存電話サービス仕様の記述実験[10),11)]を行い，上記(1)と(2)の要求条件への適合性を確認した．

4. 異常な状態への遷移検出法

本章では，前章で示した2つの機能を有する形式的な仕様記述言語（定義3）を用いて記述したサービス仕様を対象として，異常な状態への遷移を機械的に検出する手法を示す．

4.1 異常な状態への遷移検出アルゴリズム

異常な状態への遷移を検出するアルゴリズムに用いる4つの関数について定義する．

(1) $\delta T_i = \mathrm{terminal}(s_i)$：

あるサービスの状態 s_i において，この s_i を構成する端末の集合 δT_i を求める関数である．

(2) $\Gamma_i = \mathrm{relation}(s_i)$：

あるサービスの状態 s_i において，この s_i を構成する端末のそれらの間の関係の集合 Γ_i を求める関数である．Γ_i は関連する2つの端末の組み合わせを1つの要素として構成される．2つの端末 t_i と t_j の組み合わせを示す要素は $\langle t_i, t_j \rangle$ と表す．

(3) $s(\delta T_i) = \text{terminal-state}(s_i, \delta T_i)$：

あるサービスの状態 s_i において，この s_i に含まれる端末集合 δT_i の端末自体の状態の集合 $s(\delta T_i)$（他の端末とは関係のない端末の状態）だけを求める関数である．求めた端末の状態は状態記述要素 p で表現される．

(4) $s(\Gamma_i) = \text{relation-state}(s_i, \Gamma_i)$：

あるサービスの状態 s_i において，この s_i に含まれる端末集合 δT_i の他の端末と関係している端末の状態 $s(\Gamma_i)$ だけを求める関数である．求めた端末の状態は状態記述要素 p で表現される．

検出アルゴリズム

入力：

2つの単独サービス仕様として，X サービス仕様と Y サービス仕様を規定したルールの集合である R_x と R_y が入力される．

出力：

異常な状態への遷移を検出した X と Y サービス仕様を合成した現状態と次状態，およびその遷移に関するルール r_x と r_y を出力する．

ステップ：

(1) 合成した状態の生成

R_x と R_y から1つずつルールを選択した後，それらのルールの現状態の和集合を取り，合成サービスの前状態 s_i を生成する．

$s_i = cs_{rx} \subset cs_{ry}$

(2) s_i が実際に出現するかどうかの判定

初期状態から，新たな状態が出現しなくなるまで，$R_x \cup R_y$ に属するルールを順次適用し，その過程において出現した状態の中に状態 s_i が存在する場合，状態 s_i を実際に出現する状態であると判定する（文献12）参照）．

(3) ルールの実行後の合成サービスの状態の導出

状態 s_i において，r_x を適用し，遷移後の状態 s_j を導出する．

$s_j = W(r_x, s_i)$

(4) 単独サービスの状態の抽出

状態 s_j における Y サービスを構成する状態 s_{yj} を抽出する．抽出式を以下に示す．

$s_{yj} = \text{terminal-state}(s_j, \mathrm{terminal}(s_{yi}))$

$\cup$relation-state(s_j, relation(s_{yi}))

（5） 異常な状態への遷移の判定

ステップ（4）で抽出された状態 s_{yi} が，Y サービスの状態として出現するかを判定する．ステップ（2）の手法を適用して Y サービス仕様を規定したルール集合 R_y を初期状態から順次適用し，すべてのルールを適用し終わっても状態 s_{yj} が生成されない時に異常な状態への遷移であると判定する．この結果，合成サービス状態 s_i，s_j および各サービスのルール r_x と r_y を出力する．

ステップ（1）からステップ（5）の処理を，X と Y サービスのすべてのルールの組み合わせに対して行うことにより，合成した状態において X サービス仕様を規定したルールを実行するときに発生する異常な状態への遷移を検出する．同様の手法を用いて，合成した状態において Y サービス仕様を規定したルールを実行するときに発生する異常な状態への遷移を検出する．

5. 異常な状態への遷移の検出例

3章の定義3で示した if-then 型のルール形式に基づく形式的仕様記述言語として，STR 言語[10]を採用し，その検出例を説明する．**表2**に，表1に示された2つの状態遷移を STR 言語を用いて記述した例を示す．本章では，この STR 言語を用いて記述したサービス仕様を例に取り，図5で示した異常な状態遷移の検出例について以下に示す．なお，以下の説明では，図5の状態 s の添字は $i=1$，$j=2$ とする．

入力：

着信転送サービス仕様と着信拒否サービス仕様の一部を規定しているルールとして，表2に示した規則（r_{tcs} と r_{cfv}）を用いる．

ステップ．

（1） 合成した状態の生成

r_{tcs} と r_{cfv} の現状態記述から図5で示す状態 s_1 を生成する．

$s_1 = cs_{rtcs} \cup cs_{rcfv}$

$= \{$dial-tone(t_1), idle(t_4), m-tcs(t_4, t_1)$\}$

$\cup \{$dial-tone(t_1), path(t_2, t_3),

path(t_3, t_2), m-cfv(t_2, t_4), idle(t_4)$\}$

$= \{$dial-tone(t_1), path(t_2, t_3), m-cfv(t_2, t_4),

path(t_3, t_2), idle(t_4), m-tcs(t_4, t_1)$\}$

（2） s_1 が実際に出現するかどうかの判定

この判定法（文献 12)）に関しては既に提案されているため，ここではその説明を省略する．

（3） ルールの実行後の合成サービスの状態の導出

状態 s_1 において r_{cfv} を実行する．これは，r_{cfv} の現状態に記述されているすべての状態記述要素を次状態に記述されているすべての状態記述要素に置き換えることによって実現することができる．以下に，状態 s_1 の遷移後の状態 s_2 を以下に示す．

$s_2 = \{$ringback(t_1, t_4), path(t_2, t_3),

m-cfv(t_2, t_4), path(t_3, t_2), ringing(t_4, t_1),

m-tcs(t_4, t_1)$\}$

（4） 着信拒否サービスの状態の抽出

状態 s_2 における着信拒否サービスの状態 s_{tcs2} を抽出する．以下に抽出式を示す．

s_{tcs2} = terminal-state(s_2, terminal(s_{tcs1}

))$\cup$relation-state(s_2,

relation(s_{tcs1}))

表2 STR 言語によるサービス記述例
Table 2 Examples of service description in STR language.

	着信転送サービス	着信拒否サービス
規則	dial-tone(t1), path(t2, t3), path(t3, t2), m-cfv(t2, t4), idle(t4) dial(t1, t2) : ringback(t1, t4), ringing(t4, t1), path(t2, t3), path(t3, t2), m-cfv(t2, t4).	dial-tone(t1), m-tcs(t4, t1), idle(t4) dial(t1, t4) : busy-dial(t1), m-tcs(t4, t1), idle(t4).
意味	t1端末においてダイヤルトーンがなっており(dial-tone(t1))、t2端末がt3端末と通話しており(path(t2, t3), path(t3, t2))、かつt4端末に転送を登録しており(m-cfv(t2, t4))、t4端末が空き状態 (idle(t4))となっている状態において、t1端末がt2端末に発呼すると(dial(t1, t2))、t1端末がt4端末を呼び出している状態(ringback(t1, t4), ringing(t4, t1))に遷移する。	t1端末においてダイヤルトーンがなっており(dial-tone(t1))、かつt1端末からt4端末への着信を禁止する機能が起動されており(m-tcs(t4, t1)、t4端末が空き状態となっている(idle(t4))状態において、t1端末がt4端末に発呼すると(dial(t1, t4))、t1端末において発信ができなかったことを示す音がなる状態(busy-dial(t1))に遷移する。

$=$terminal-state(s_2, terminal(
{dial-tone(t_1), idle(t_4), m-tcs(t_4, t_1)
}))$\cup$relation-state(s_2, relation
({dial-tone(t_1), idle(t_4),
m-tcs(t_4, t_1)}))
$=$terminal-state(s_2, $\{t_1, t_4\}$)$\cup$
relation-state(s_2, $\{\langle t_1, t_4\rangle\}$)
$=\{\phi\}\cup\{$ringback(t_1, t_4),
ringing(t_4, t_1), m-tcs(t_4, t_1)$\}$
$=\{$ringback(t_1, t_4), ringing(t_4, t_1),
m-tcs(t_4, t_1)$\}$

（5） 異常な状態遷移の判定

着信拒否サービス仕様を規定したルールにより生成される状態にはステップ（4）において抽出された状態 s_{tcs2}（着信拒否サービスでは禁止している状態）は出現しない．従って，状態 s_{tcs2} は着信拒否サービス仕様には規定されていない状態であると判断し，図5の状態 s_1 から状態 s_2 への遷移は異常な状態への遷移である．

出力：

異常な状態への遷移を検出した着信拒否サービスと着信転送サービスを合成した現状態 s_1 と次状態 s_2，およびその遷移に関するルール r_{tcs} と r_{cfv} が出力される．

6. 評価および考察

着信転移（CFV），着信拒否（TCS），発信拒否（OCS）の3つのサービス仕様を STR 言語を用いて記述し，提案した異常な状態への遷移検出法の有効性を机上実験により評価した．なお，検出アルゴリズムの正当性は，定義2より明らかである．

6.1 検出実験

実験で用いた STR 言語で記述した CFV と TCS の基本仕様の例を付録1に示す．これらの個々のサービスの記述は端末の状態遷移の様子をアニメーション的に表示できる STR 言語のインタプリタ[11]を用いて確認している．実験では，CFV と TCS および CFV と OCS の2つの組み合わせを4端末で行った．それぞれのルールの和集合を構成して，インタプリタにより逐次実行させながら提案したアルゴリズムに従って異常な状態への遷移の机上検出実験を行った．検出結果を**表3**に示す．表3の第1行の略号の意味は，以下のとおりである．

T：ルール形式で記述したサービス仕様として規定されている実行可能な状態遷移の数．

T_{ab}：検出された異常な状態への遷移の数．

T_d：検出されるべきサービス競合の数．

T_{ab}/T：絞り込み率．

6.2 検出精度

ルール形式で記述したサービス仕様を合成した場合に提案手法を用いて検出できる異常な状態への遷移の検出精度は，その有効性を評価するうえで重要となる．何故ならば，もしすべて検出されなければ，残りの検出されなかった状態遷移を見つけ出すために，合成した仕様に含まれる実行可能なすべての状態遷移を設計者が再度すべて確認する必要があり，機械的に検出した効果が無くなるからである．記述実験した3つのサービスの CFV と TCS，CFV と OCS をそれぞれ組み合わせた場合の検出実験を行った．この結果，検出された異常な状態への遷移の中に，検出されるべきサービス競合はすべて含まれていたため，異常な状態への遷移の検出により，状態遷移の意味的矛盾に起因するサービス競合の検出支援として有効であることを確認できた．

6.3 人手による検出削減効果

ルール形式で記述した2つのサービス仕様を組み合わせて，それぞれのサービス動作を満足する合成サービス仕様の設計において，サービス仕様を組み合わせることによって生成される状態が，合成サービス仕様として含むべき状態であるか否かを設計者が判断しながら，設計を進める必要がある．提案手法により，このような設計者が検討しなければならない状態を機械的に検出できるため，設計者が2つのサービス仕様を組み合わせた場合にすべて実行可能な状態遷移を検討する必要が無くなり，検出された状態だけに絞り込めるため，設計の効率化が図れる．

CFV と TCS，CFV と OCS をそれぞれ組み合わせた場合の検出実験の結果（表3）から，絞り込み率はおよそ 0.1 となり，本手法により設計者が確認しなければならない状態が 10 分の1に削減されたことを

表3 サービス仕様競合検出の実験結果
Table 3 An experimental result of service specification interaction detection.

サービスの組み合わせ	T	T_{ab}	T_d	T_{ab} / T
CFV+TCS	112	7	1	0.06
CFV+OCS	168	23	1	0.13

示している．

6.4 実現上の課題

提案手法を実現する場合の検出精度に関する課題として，4.1節で述べた検出アルゴリズムのステップ(2)における合成サービスの状態 s_i が実際に存在することを判定する手法の実現が重要となる．本判定法は，ルールの集合として記述したサービス仕様において，すべての状態への可到達性の判定であり，この可到達性判定法が提案検出アルゴリズムのカバー率に影響する．ルールで記述したサービス仕様では，端末数に比例してサービスの状態が増大するため，端末数に依存してすべてのサービス状態が生成されるか，されないかが変わってくる．例えば，端末数 n の場合に生成される状態には，サービス状態 s_i は存在しないが，$n+1$ の場合には s_i が存在する場合が考えられる．これは，端末数を n として検出アルゴリズムを実行した場合には異常な状態は検出されないが，端末数を $n+1$ として検出アルゴリズムを実行した場合には，異常な状態が検出される可能性があることを示している．検出精度を100%とするためには，端末数を無限とする必要があり，実現が困難となる．しかし，ルールで記述したサービス仕様における可到達状態の数は，端末数の増加に対してある端末数以上では増加しなくなることが証明されている[13]．従って，検出される異常な状態が飽和する端末数が存在し，その端末数を与えることによりすべての異常な状態が検出できる．実際の通信サービスの検証支援システムでは，端末数の任意な設定を可能とし，端末数をパラメータとして振らして繰り返し実行することにより実現できる．ただし，このような端末数の存在に関する定量的な証明は，今後の課題である．

6.5 ルール集合の管理検証に関する考察

本稿で提案した手法は既存のルール集合に新しいルール集合を追加した場合のルール集合間の矛盾に関する問題である．このような問題は知識情報処理の分野でも研究されており，非専門家に対する利点と矛盾検出法の観点から比較・考察する．

(1) 非専門家に対する利点

既存ルール集合に新しいルール集合を追加した場合のルール集合間の矛盾回避，無矛盾保持等の方式として，知識情報処理ではTMS, ATMSが研究されている[7),8)]．これらの方式では，ルール集合間の矛盾の機械的な検出と回避が可能であるため，非専門家に対しては，以下のような利点がある．

a) 非専門家自身が人手でルール集合間の矛盾を検出しなくてもよい．なお，もし検出しようとした場合には，既存のすべてのルールとそれらのルール集合から推論される結論を非専門家が理解する必要がある．

b) また，非専門家が解消方法を考えなくても，システムより提示が得られる．

本稿で提案した手法は，上記a)と同じ利点を狙っている．すなわち，非専門家が既存のルール集合により定義されたサービス仕様，具体的には単一の状態遷移を定義した個々のルールの意味とそれらのルールを適用することによって生成されるすべての状態遷移を知らなくても異常な状態への遷移として矛盾が機械的に検出できる．また，上記b)については，検出した異常な状態への遷移を解消できる仕様がルールとして定義できることは確認している[14)]．しかし，非専門家に対しては，解消ルールの機械的な獲得支援が必要と考えており，具体的な手法については，今後の課題である．

(2) 矛盾検出法

ルール集合間の制約チェックとしてデータベース等で行われている矛盾検出法がある[8)]．この手法は，制約とデータを命題論理や述語論理の論理式の集合として捉え，論理的な矛盾として検出する方法である．この手法では，状態の論理的矛盾，例えば状態Aとその否定状態 ¬A という矛盾は検出できるが，意味的な矛盾は検出できない．本稿で提案する手法は，if-thenのルール集合で規定された状態遷移の意味的な矛盾の検出を狙っている．すなわち，意味的な矛盾を異常な状態への遷移として形式的に定義し，その定義に基づく矛盾の検出アルゴリズムを提案している．

7. あとがき

通信サービスを迅速に開発するための，重要な課題であるサービス競合検出を支援するための方法について提案した．具体的には，状態遷移モデルに基づいて記述された通信サービス仕様を対象として，2つの単独サービス仕様を合成した時に発生する意味的矛盾の機械的な検出法について示した．意味的矛盾として，「異常な状態への遷移」の定義を示した．また，異常な状態への遷移を機械的に検出するために必要となる仕様記述言語への要求条件を示した．その記述言語に基づき記述された仕様を合成した場合の異常な状態への遷移を検出するアルゴリズムについて提案した．更に，実際のサービス仕様をSTR言語により記述して

実験を行い，サービス競合として設計者が検証しなければならない状態遷移の数が約1割に絞り込むことができることを示した．

今後の課題として，以下の項目を検討する．

(1)提案アルゴリズムを試作により実現し，有効性を実証する．

(2)LSSGR[15] で規定されているより多くの通信サービス仕様を用いて実験を行い，提案した手法の洗練化を進める．

(3)異常な状態への遷移の解消仕様の獲得支援法の検討を行う．

謝辞 本研究を進める上で，ご指導と励ましをいただいた国際電気通信基礎技術研究所葉原副社長，ATR通信システム研究所寺島社長に深く感謝いたします．また，有益な議論を頂いた ATR 通信システム研究所の皆様に感謝します．

参 考 文 献

1) Cameron, E. J., Griffeth, N., Lin, Y., Nilson, M. E., Schnure, W. K. and Velthuijsen, H.: A Feature-Interaction Benchmark for IN and Beyond, *IEEE Communication Magazine*, Vol. 31, No. 3, pp. 64-69 (1993).
2) Cameron, E. J. and Velthuijsen, H.: Feature Interactions in Telecommunications Systems, *IEEE Communication Magagine*, Vol. 31, No. 8 (1993).
3) Suzuki, S.: IN Rollout in Japan, *IEEE Communication Magagine*, Vol. 31, No. 3 (1993).
4) Wakahra, Y., Fujioka, M., Kikuta, H., Yagi, H. and Sakai, S.: A Method for Detecting Service Interactions, *IEEE Communication Magagine*, Vol. 31, No. 8 (1993).
5) Boumezbeur, R. and Logrippo, L.: Specifying Telephone Systems in LOTOS, *IEEE Communication Magagine*, Vol. 31, No. 8 (1993).
6) Harada, Y., Hirakawa, Y. and Takenaka, T.: A Design Support Method for Telecommunication Service Interactions, *GLOBECOM '91*, Phoenix, Arizona (Dec. 1991).
7) 松本裕治，佐藤 健：非単調論理と常識推論，情報処理，Vol. 30, No. 6 (1989).
8) 人工知能学会：人工知能ハンドブック，オーム社 (1990).
9) 原田良雄，高見一正，太田 理，寺島信義：通信サービス相互作用に含まれる意味的矛盾動作検出方式，情報処理学会論文誌，Vol. 35, No. 8, pp. 1602-1613 (1994).
10) Hirakawa, Y. and Takenaka, T.: Telecommunication Service Description Using State Transition Rules, *Sixth Int. Workshop on Software Specification and Design*, Como, Italy (Oct. 1991).
11) Takami, K., Harada, Y., Ohta, T. and Terashima, N.: A Visual Design Support System for Telecommunications Services, *IEEE Phoenix Conference on Computers and Communications* (Mar. 1993).
12) Shibata, K., Hirakawa, Y. and Takenaka, T.: Reachability Analysis for a Behavior Description Independent of the Number of Processes, *5th Joint Workshop on Computer Communication* (July 1990).
13) 佐藤正和：プロダクションルールを用いた通信ソフトウェア仕様の可到達解析に関する考察，信学技法 KBSE 93-45, Vol. 93, No. 408 (Jan. 1994).
14) 井上泰彰，高見一正，太田 理：通信サービス仕様における異常な状態への遷移の解消支援法，信学春季全大，B-620 (1994).
15) Bellcore: LSSGR Features Common to Residence and Business Customers 3, Issue 2 (1989).

付録 STR 言語によるサービス仕様の記述例

着信転送サービス(CFV)と着信拒否サービス(TCS)の基本仕様に関する記述例を以下に示す．実験では，このほかに2者間の通話サービス（基本電話サービス）仕様を記述した9個のルールを加えて行った．

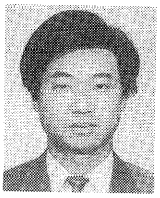
(1) 着信転送サービス (CFV)

1) dial-tone(A) cfv(A): cfv-tone(A).
2) dial-tone(A), m-cfv(A, B) cfv(A): cfv-tone(A).
3) cfv-tone(A), idle(B) dial(A, B):
confirm-tone(A), m-cfv(A, B), idle(B).
4) cfv-tone(A), not[idle(B)] dial(A, B):
confirm-tone(A), m-cfv(A, B).
5) cfv-tone(A) dial(A, A): busy-dial(A).
6) dial-tone(A), cond :idle(B), idle(C),
cond :m-cfv(B, C) dial(A, B):
ringback(A, C), ringing(C, A).
7) dial-tone(A), not[idle(B)], idle(C),
cond :m-cfv(B, C) dial(A, B):
ringback(A, C), ringing(C, A).
8) dial-tone(A), cond :idle(B), cond :m-cfv(B, C),
not[idle(C)] dial(A, B): busy-dial(A).
9) dial-tone(A), not[idle(B)], cond :m-cfv(B, C),
not[idle(C)] dial(A, B): busy-dial(A).
10) cfv-tone(A) onhook(A): idle(A).
11) confirm-tone(A) onhook(A): idle(A).

（2） 着信拒否サービス（TCS）

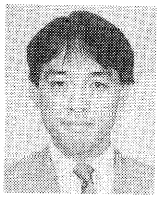
1) dial-tone(A) tcs(A) : tcs-tone(A).
2) dial-tone(A), m-tcs(A, B) tcs(A) : tcs-tone(A).
3) tcs-tone(A), idle(B) dial(A, B) :
   confirm-tone(A), m-tcs(A, B), idle(B).
4) tcs-tone(A), not[idle(B)] dial(A, B) :
   confirm-tone(A), m-tcs(A, B).
5) tcs-tone(A) dial(A, A) : busy-dial(A).
6) dial-tone(A), cond :m-tcs(B, A), idle(B)
   dial(A, B) : busy-dial(A), idle(B).
7) dial-tone(A), cond :m-tcs(B, A), not[idle(B)]
   dial(A, B) : busy-dial(A), idle(B).
8) confirm-tone(A) onhook(A) : idle(A).
9) tcs-tone(A) onhook(A) : idle(A).

（平成6年5月9日受付）
（平成6年10月13日採録）



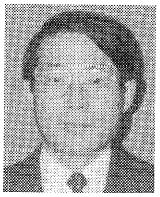
高見 一正（正会員）

1977年静岡大学工学部電気卒業．1979年同大学院修士課程修了．同年日本電信電話公社（現NTT）入社．以来，電話・パケット網間接続サービスの研究実用化，マルチメディアパケット通信方式，ATM呼制御方式と通信サービスの仕様記述法の研究に従事．現在，同社ネットワークシステム研究所主任研究員．電子情報通信学会，IEEE各会員．



井上 泰彰（正会員）

昭和38年生．昭和61年広島大学電気系電気工学科卒業．同年三洋電機株式会社入社．平成2年より4年間ATR通信システム研究所に出向．通信サービス仕様の記述法や検証法などに興味を持つ．



太田 理（正会員）

1945年生．1968年九州大学工学部電子卒業．1970年同大学院修士課程修了．同年電電公社（現NTT）入社．電子交換機のソフトウェア研究開発に従事．1987年からNTTにおける情報・通信機器向けの共通OSの開発に従事．1992年2月にATR通信システム研究所に出向，現在に至る．通信ソフトウェア研究室長．工学博士．1994年3月電気通信普及財団賞を受賞．電子情報通信学会，IEEE各会員．