計算機アーキテクチャ　99－15
（1993. 3. 12）

サービスインタラクションの解決のための サービス仕様からのプロトコル仕様の合成

浅田 宏幸　　五十嵐 裕孝　　角田 良明　　菊野 亨

大阪大学基礎工学部情報工学科
〒560 豊中市待兼山町 1-1
TEL 06-844-1151 内線 4841

あらまし

本稿では，全ての通信サービスは同時に開発されるのではなく，順次開発されていくことに着目する．そして，サービス仕様の段階でサービスインタラクションを除去するのではなく，プロトコル仕様の段階でサービスインタラクションを解決する方法を提案する．提案する方法では，(1) サービス仕様を STR 手法で記述し，(2) そのサービス仕様からプリミティブ実行系列を生成し，(3) 実行系列からプロトコル合成法を応用してプロトコル仕様を得る．(4) 最後に，サービスインタラクションを解決するための条件分岐機能を挿入する．

和文キーワード　インテリジェントネットワーク　サービス仕様　プロトコル仕様
サービスインタラクション　プロトコル合成　サービスプリミティブ

Synthesis of Protocol Specifications from Service Specifications for Solving Service Interactions

Hiroyuki Asada　Hirotaka Igarashi　Yoshiaki Kakuda　Tohru Kikuno

Dept. of Information and Computer Sciences
Faculty of Engineering Science, Osaka University
Toyonaka-shi, Osaka 560, Japan
E-mail { asada,igarashi,kakuda,kikuno }@ics.osaka-u.ac.jp

Abstract

This paper proposes a method for solving the service interactions in the protocol specifications. The procedures of the proposed method consist of (1) service specifications are described by the STR(State Transition Rules) method, (2) execution sequences of service primitives are generated from the service specifications, (3) protocol specifications are derived from the execution sequences of service primitives using the protocol synthesis method, and (4) Conditional branch functions for solving the service interactions are insterted into the protocol specifications.

英文 key words　Intelligent Network　service specification　protocol specification
service interaction　protocol synthesis　service primitive

1 まえがき

近年，通信サービスへの要求の多様化とともに，インテリジェントネットワークの果たすべき役割が大きくなってきている．ところが，インテリジェントネットワークには解決すべき問題が多く残されている．複数のサービス間で起こる動作競合であるサービスインタラクションもその一つである．サービスインタラクションの検出と解決については様々な研究が行なわれてきている [1] [3] [4] [6]．文献 [1] [6] では，サービスをエレメント (料金レート，呼びだし状況など) の集合で特徴づけ，それぞれに対し用意されたルールを適用することによるサービスインタラクションの解決法を提案している．文献 [3] [4] では，STR(State Transition Rule) 手法を用いてサービス仕様を記述し，サービス仕様の段階でサービスインタラクションを検出除去する方法を提案している．

本稿では，全ての通信サービスが一度に開発されるのではなく，順次追加されていくことに着目する．そして，サービス仕様の段階でサービスインタラクションを検出し除去するのではなく，プロトコル仕様の段階でサービスインタラクションを解決する方法を提案する．提案する方法では，(1) サービス仕様を STR 手法 [3] で記述し，(2) そのサービス仕様からプリミティブ実行系列を生成し，(3) 実行系列からプロトコル合成法 [2] を応用してプロトコル仕様を得る．(4) 最後に，サービスインタラクションを解決するための条件分岐機能を挿入する．

2 サービス仕様とサービスインタラクション

2.1 サービス仕様

通信システムは複数のユーザで構成されているとし，通信システムの状態を各ユーザの状態の積集合で表す．STR 手法 [3] では，通信システムの各ユーザの状態を述語の集合で表し，ユーザが入力するイベントに基づいて複数のユーザの状態遷移を規定する．従って，サービス仕様は「サービスに関与するユーザの現状態」「イベント：」「サービスに関与するユーザの次状態」の 3 つの部分から構成されるサービス規則の集合で表すことができる．図 1 に STR 手法による 2 つのサービス規則の例を示す．

図 1 における述語とイベントについて説明する．述語 path(A,B) はユーザ A からユーザ B への通話路が接続されていることを示す．述語 out-dialtone(C) はユーザ C がダイヤル可能状態であることを示す．述語 m-cw(A) はユーザ A が話中着信サービスに加入していることを示す．述語 out-ringback(C,A) はユーザ C にユーザ A を呼び出す音が出ていることを示す．述語 cw-ringing(A,C) はユーザ C からの話中着信音がユーザ A に出ていることを示す．述語 idle(D) はユーザ D が空き状態であることを示す．述語 m-cfv(A,D) はユーザ A が着信転送サービスに加入していて，転送先にユーザ D を指定していることを示す．述語 out-ringing(D,C) はユーザ D にユーザ C からの呼び出し音が出ていることを示す．dial(C,A) はイベントでありユーザ C がユーザ A にダイヤルすることを示す．

それぞれのサービスについて，サービス適用前後のユーザの状態を表したものを図 2, 3 に示す．同図で○はユーザを表す．また，実線の矢印は通話中であることを，点線の矢印は通話路が接続され，呼出し音が出ていることを表す．

2.2 サービスインタラクション

与えられた通信システムの状態とサービス規則に対し，イベント生起前の「サービスに関与するユーザの現状態」を表す全ての述語が通信システムの状態によって満たされる時，そのイベントは実行可能と言う．複数のサービス規則の中に同一のイベントをもち，それらのイベントがどちらも実行可能となるサービス間の動作競合をサービスインタラクションと定める．サービスインタラクションが起こっている複数のサービス規則の「サービスに関与するユーザの次状態」が互いに論理的に矛盾であれば，そのうちいずれかのイベント一つを選択して実行しなければならない．

今，通信システムの状態が m-cw(A), m-cfv(A,D), path(A,B), path(B,A), out-dialtone(C), idle(D) という述語を満たすとする．図 1 の CW サービスと CFV サービスの規則には同一のイベント dial(C,A) があり，かつ，2 つとも実行可能となるため，これらのサービスはサービスインタラクションを起こしている．その様子を図 4 に示す．CW サービス規則の次状態中の out-ringback(C,A) と CFV サービス規則の次状態中の out-ringback(C,D) は，ユーザ C が 1 つの通話路しか設定しないことに矛盾してしまうので，どちらかのサービスを選択しなければならない．

2.3 プロトコル仕様

一方，サービスプリミティブ (以下，プリミティブ) はユーザの存在する上位層とサービスを実現する下位層のインタフェース (SAP) でやりとりされるものである．代表的なプリミティブの実行系列に request, indication, response, confirmation がある．以降，それぞれを req, ind, resp, conf と略す．これらのプリミティブでは，そのプリミティブが発せられるユーザの状態を伝えることができるとともに，ユーザの状態の変化を与えることもできる．

本稿では，図 5 に示されるように，上位層のユーザに 1 対 1 対応して下位層にプロセスが存在すると仮定する．図 5 において，実線の矢印はプリミティブのやりとり，点線はプロセス間のメッセージのやりとりを表す．プロトコル仕様はこのようなプロセス間のメッセージ送受信遷移とプリミティブ実行系列で規定される．

3 サービスインタラクションの解決指針

本稿では，サービスインタラクションの問題を，次のような問題として考える．

入力: STR 手法で記述されたサービス仕様．

出力: サービスインタラクションの解決機能を含むプロトコル仕様．

サービス仕様には，例えば idle(A), not(idle(A)) といったような矛盾は含まれないとする．

プロトコル仕様はユーザとプロセス間のプリミティブのやりとりおよびプロセス間のメッセージ送受信を状態遷移とする FSM でモデル化されるものとする．ただし，本稿では理解を容易にするためプロトコル仕様をシーケンスチャートで表す．

本稿ではサービスインタラクションの問題を図 6 の手順で解決する．まず，サービス毎にサービス規則からプリミティブ実行系列を生成する．次に，プリミティブ実行系列からサービスインタラクションを含むプロトコル仕様を合成する．これには従来のプロトコル合成法 [2] を応用する．最後に，サービスインタラクションを起こすプロセスに条件分岐機能を挿入してサービスインタラクションを解決する．

サービスインタラクションは，2.2. で述べたように複数のサービス規則のなかの同一のイベントが非決定的に実行可能となることである．各イベントはそれぞれプリミティブ実行系列に対応するので，複数のプリミティブ実行系列が非決定的に実行可能になることと見ることもできる．そこで，どちらのプリミティブ実行系列を実行する

かを判定する条件分岐機能をプロトコル仕様の段階で挿入することによって, サービスインタラクションの問題を解くことができる.

4 プリミティブ実行系列の生成

3. で述べた解決指針のなかで, サービス規則からプリミティブ実行系列を生成する部分が本質的である. 本節ではその生成について述べる.

サービス規則の中のイベントの生起によって複数のユーザの状態が変化させられる. サービスインタラクションが起きれば, 複数のサービス規則の中の同一のイベントが共に実行可能になる. サービスインタラクションを解決するためには, どちらのイベントを選択するかを決定しなければならない. 本稿では, サービス仕様ではなくプロトコル仕様の段階でその決定をする. そのために, 次の 3 つの条件 (1) – (3) を満たすようにプリミティブ実行系列を生成する必要がある.

(1) イベント (プリミティブ実行系列に対応) の選択を 1 つのプロセス (プロセス S と呼ぶ) が行なう.

(2) S はイベントの生起前のサービスに関与するユーザの状態を収集する.

(3) (2) で収集された状態の情報に基づき, S はイベントの選択を行ない, 選択されたイベントの生起後の状態をサービスに関与するユーザに伝達する.

この 3 つの条件を満たすプリミティブ実行系列を生成できれば, それに基づいてプロトコル仕様を合成することが可能となり, サービスインタラクションの問題を正しく解くことができる.

サービスインタラクションを起こす話中着信 (CW) サービスと着信転送 (CFV) サービスに対し, 生成されたプリミティブ実行系列を図 7 に示す.

図 7 の実行系列において, SAP_Aはユーザ A とプロセス A の間の SAP を表す. SAP_Cと SAP_Dも同様である. $\uparrow$はプロセスからユーザへの, $\downarrow$はユーザからプロセスへのプリミティブであることを表す. また, ${S_A}^1, {S_C}^1, {S_D}^1$はそれぞれユーザ A, ユーザ C, ユーザ D のイベント dial(C,A) が生起する前の状態を表す. ${S_A}^1$ = { m-cw(A), m-cfv(A,D), path(A,B)}, ${S_B}^1$ = {path(B,A)}, ${S_C}^1$ = {out-dialtone(C)}, ${S_D}^1$ = {idle(D)} と仮定する. ${S_C}^1$[−out-dialtone(C), +out-ringback(C,A)] はユーザ C の状態 ${S_C}^1$から out-dialtone(C) を削除し, out-ringback(C,A) を付け加えることを表している. 例えば, $dial\_req(SAP_A \downarrow | {S_C}^1)$ は ${S_C}^1$を情報として含み, SAP_Aを通過してユーザからプロセスへのプリミティブ $dial\_req$ を表す.

CW サービスの実行時にやりとりされるプリミティブ実行系列のシーケンスチャートを図 8 に, CFV サービスの実行時にやりとりされるプリミティブ実行系列のシーケンスチャートを図 9 に示す. 図 8, 9 で, プロセス A が●の時点でイベントの選択を行なう. 図 8, 9 の点線部分に対応するプロセス間のメッセージ送受信遷移を生成する手続きについては, 5. で説明する.

5 プロトコル仕様の合成

4. で求まったサービスインタラクションを起こす複数のサービスに対するプリミティブ実行系列に基づいて, プロセス間のメッセージ送受信遷移を生成し, プロトコル仕様を合成する.

合成手順は次の通りである.

(1) プリミティブ実行系列の集合からプリミティブ実行系列グラフを作成する.

(2) プリミティブ実行系列グラフにダミープリミティブを挿入する.

(3) プリミティブ実行系列グラフを各プロセスに射影する.

(4) 各プロセスに射影されたグラフに対し, 文献 [2] の合成ルールを適用してプロセス間のメッセージを生成する.

(5) ダミープリミティブおよびεを除去して, 各プロセスのプロトコル仕様を合成する.

以降, 5.1〜5.5で手順 (1)〜(5) をそれぞれ説明する.

5.1 プリミティブ実行系列グラフの作成

サービスインタラクションを起こすサービス i, jに対するプリミティブ実行系列は, 一般に次のように表すことができる.

$seq_i = ({s_1}^i, {t_1}^i, \cdots, {s_p}^i, {t_p}^i, \cdots, {s_q}^i, {t_q}^i, \cdots, {s_{n_i}}^i)$

$seq_j = ({s_1}^j, {t_1}^j, \cdots, {s_p}^j, {t_p}^j, \cdots, {s_q}^j, {t_q}^j, \cdots, {s_{n_j}}^j)$

ここで, ${t_k}^i (1 \leq k \leq n_i), {t_l}^j (1 \leq l \leq n_j)$ はそれぞれサービス i, j に対するプリミティブを表し, ${s_k}^i (1 \leq k \leq n_i), {s_l}^j (1 \leq l \leq n_j)$ は, それぞれ ${t_k}^i, {t_l}^j$実行前の状態あるいは, ${t_{k-1}}^i, {t_{l-1}}^j$実行後の状態を表す. ${s_p}^i (1 \leq p \leq n_i). {s_q}^j (1 \leq q \leq n_j, p \leq q)$ はそれぞれサービス i, j の規則のイベント実行前の通信システム状態に関する情報を収集できた状態を表す. サービスインタラクションを起こしていれば, 同じイベントを実行するので, seq_i, seq_jの前半部分は等しく, ${s_a}^i = {s_a}^j, (1 \leq a \leq p), {t_b}^i = {t_b}^j (1 \leq b \leq p-1)$ となる.

プリミティブ実行系列グラフは, 節点をすべての SAP の状態, 有向枝をプリミティブによる遷移とする有向グラフである. このグラフは次の 3 つのプリミティブ実行系列で構成される.

$seq_1 = ({s_1}^i, {t_1}^i, \cdots, {s_p}^i, {t_p}^i, \cdots, {s_q}^i, {t_q}^i, \cdots, {s_{n_i}}^i)$

$seq_2 = ({s_1}^i, {t_1}^i, \cdots, {s_p}^i, {t_p}^j, \cdots, {s_q}^j, {t_p}^i, \cdots, {s_q}^i, {t_q}^i, \cdots, {s_{n_i}}^i)$

$seq_3 = ({s_1}^i, {t_1}^i, \cdots, {s_p}^i, {t_p}^j, \cdots, {s_q}^j, {t_q}^j, \cdots, {s_{n_i}}^i)$

seq_1はサービスインタラクションが起こらない場合のサービス i に対するプリミティブ実行系列を表す. seq_2はサービスインタラクションが起こる場合のサービス i に対するプリミティブ実行系列を表す. seq_2には, seq_1に (seq_3の部分系列である)${t_p}^j, \cdots, {s_q}^j$が加わっていることに注意されたい. seq_3は (サービスインタラクションの生起にはかかわらず) サービス jに対するプリミティブ実行系列を表す. サービスインタラクションが起こる場合, ${s_q}^j$がイベントの選択を行なう状態となる.

CW サービスと CFV サービスに対して図 7 のプリミティブ実行系列が得られている. この系列に基づいて作成されたプリミティブ実行系列グラフを図 10 に示す. 同図において, 点線で表した系列が CW サービスに対するプリミティブ実行系列 seq_1, seq_2であり, 一点鎖線で表した系列が CFV サービスに対するプリミティブ実行系列 seq_3 である. ●で表された状態でサービスインタラクションが起こる場合のイベントの選択を行なう. また, 図 10 では $dial\_req(SAP_C \downarrow)$ を $dial_req_atC \downarrow$と記する. 他のプリミティブも同様である. 図 11〜図 14 でも同様の記法を用いる.

5.2 ダミープリミティブの挿入

プリミティブ実行系列グラフを入力し, プロトコル仕様を出力する問題がプロトコル合成問題として知られている. この問題に対する解法は幾つか提案されているが, 本稿では, 一般の $n(n \geq 2)$ 個のプロセスから構成されるプロトコル仕様を合成できる Saleh の合成法 [2] を採用する. しかしながら, 次の 2 つの理由により, この方法をそのまま使用することはできない.

(1) プリミティブ実行系列グラフ上には，1つの有向枝が入って2つ以上の有向枝が出ていく節点を分岐点と呼ぶある i,jに対し，分岐点に入る有向枝はSAP_iを通過するユーザからプロセスへのプリミティブであり，分岐点から出ていく全ての有向枝が$SAP_j(j \neq i)$を通過するプロセスからユーザへのプリミティブであるとする．この時，分岐点から出ていく全てのプリミティブを同時に実行してしまう．

(2) プリミティブ実行系列グラフにおいて，ある $i,j,k(i \neq j, j \neq k, k \neq i)$ に対しユーザ iからプロセス iへのプリミティブを表す有向枝にプロセス kからユーザ kへのプリミティブを表す有向枝が接続されていたとする．このとき，プロセス iからプロセス jを経由してプロセス k に到達するメッセージの送受信遷移を生成することができない．

そこで，上述の問題点を解決するため，ダミープリミティブを挿入する．

(1) 分岐点から出ていく全ての有向枝の前に，SAP_jを通過するユーザからプロセスへのダミープリミティブ dum を挿入する．こうすることによって，分岐点から出ていくプリミティブはいずれか一方のみが実行される．

(2) ユーザ iからプロセス iへのプリミティブとプロセス kからユーザ kへのプリミティブの間にユーザ jからプロセス jへのプリミティブ dum を挿入する．こうすることによってプロセス jを経由させることができる．

図11に図10のプリミティブ実行系列グラフにダミープリミティブを挿入したものを示す．同図で，網目の矢印で表したものがダミープリミティブである．

5.3 各プロセスへの射影

ダミープリミティブを挿入したプリミティブ実行系列グラフは，全ての SAP に対するプリミティブの動的振舞いを表す．これに対し，プロトコル仕様は，プロセス毎にプリミティブあるいはメッセージの動的振舞いを表したものである．このようなプロトコル仕様を得るため，プリミティブ実行系列グラフを各プロセスに射影する．つまり，SAP_iを通過するプリミティブのみを残し，それ以外のプリミティブを εに置きかえたものを作成する．このグラフを SAP_iに対するプリミティブ実行系列グラフと呼ぶ．このグラフは SAP_iのみに関するプリミティブの動的振舞いを表す．

図11のプリミティブ実行系列グラフをプロセス A に対して射影したものを図12 に示す．同様に，プロセス C，プロセス D に対する射影も得られる．

5.4 プロセス間のメッセージの生成

5.3で求まった各プロセスに対する射影，つまり SAP_iに対するプリミティブ実行系列グラフに対し，文献 [2] で示された合成ルールを適用してプリミティブの対からプロセス間で送受信されるメッセージを生成し，各プロセスのプロトコル仕様を合成する．詳細については文献 [2] を参照されたい．なお，メッセージの衝突も考慮した合成法が文献 [5] で示されている．例えば，図12のプロセス A に対する射影から図13のプロセス A のプロトコル仕様が得られる．同様にプロセス C，プロセス D のプロトコル仕様も得られる．図13 において，$?a$ はメッセージ a の受信，$!b$ はメッセージ b の送信，$\mathrm{dum}(SAP_A\downarrow)/!b$ は $\mathrm{dum}(SAP_A\downarrow)$ の実行後に $!b$ を実行することを表す．

5.5 ダミープリミティブとεの除去

最後に．各プロセスのプロトコル仕様からダミープリミティブと ε を短絡除去する．つまり，これらの有向枝の両端の節点を1つにまとめて，有向枝を除去する．図13のダミープリミティブと εが除去され，図14のプロセス A のプロトコル仕様が合成される．同図で，メッセージ c を受信した後の状態(●で表す)でイベントの選択を行なう．同様にプロセス C，プロセス D のプロトコル仕様からもダミープリミティブと εが除去される．

図15にサービスインタラクションが起こった場合の CW サービスと CFV サービスに対して合成されたプロトコル仕様をシーケンスチャートで表す．同図で，黒の太線は CW サービスが選択された場合の実行系列，グレーの太線は CFV サービスが選択された場合の実行系列を表す．

6 あとがき

本稿では，プロトコル仕様の段階でサービスインタラクションを解決する方法を示した．本解決法の本質は，サービス仕様では非決定性の性質を有するイベントを，プロトコル仕様のプリミティブ実行系列に変換したことと，系列の間に条件分岐機能を挿入して決定性に変換したことである．

サービスインタラクションを起こす2つのサービス規則の「サービスに関与するユーザの次状態」に互いに論理的矛盾がなければ，これら2つのサービスを同時に実行すればよい．文献 [4] の TWC サービスと CW サービスがこの例である．この例についても結果を得ているが別途報告する．

今後，様々な通信サービスに適用して本手法の有効性を確認する予定である．

謝辞

本研究は一部平成4年度文部省科学研究費補助金一般研究 (C)(課題番号 04650315)，奨励研究 (A)(課題番号 04750337)，KEC の第13回国際通信研究奨励金，KDD 研究所の委託研究費の補助を受けている．

文献

[1] 酒井，八木，菊田，藤岡，若原："通信サービス競合の明確化と解決法の一考察"，信学技報 SSE91-136 (Jan.1992).

[2] K.Saleh:"Automatic synthesis of protocol specifications from service specifications", Proc. Int'l. Phoenix Conf. on Computers and Communications, pp.615-621 (Mar.1991).

[3] Y.Harada, Y.Hirakawa, T.Takenaka and N.Terashima:"A conflict detection support method for telecommunication service descriptions", IEICE Trans. Commun. E75-B, 10, pp.986-997(Oct.1992).

[4] 井上，中村，高見："通信サービス記述における動作競合の解消支援法"，信学技報 SSE 92-7(May1992).

[5] 五十嵐，角田，菊野："回復機能を含むプロトコル仕様のサービス仕様からの合成"，信学技報 SSE 92-10(May1992).

[6] H.Kikuta, H.Yagi, S.Sakai, M.Fujioka and Y.Wakahara: "Clarification of Service Interaction and its Detection Method", TINA'92, Narita (Jan.1992).

● 話中着信（CW）サービス

path(A,B), out-dialtone(C), m-cw(A)
dial(C,A):
out-ringback(C,A), cw-ringing(A,C), path(A,B), m-cw(A)

● 着信転送（CFV）サービス

out-dialtone(C), idle(D), m-cfv(A,D), path(A,B)
dial(C,A):
out-ringing(D,C), out-ringback(C,D), m-cfv(A,D)

図1：サービス規則の例

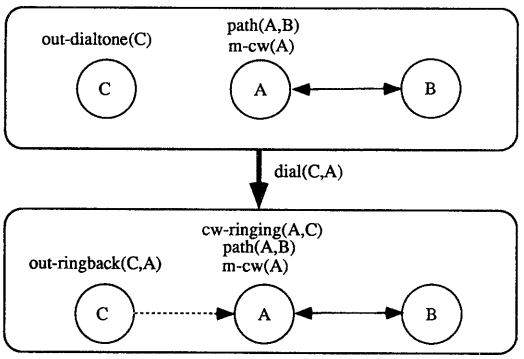

図2：話中着信（CW）サービスの実行

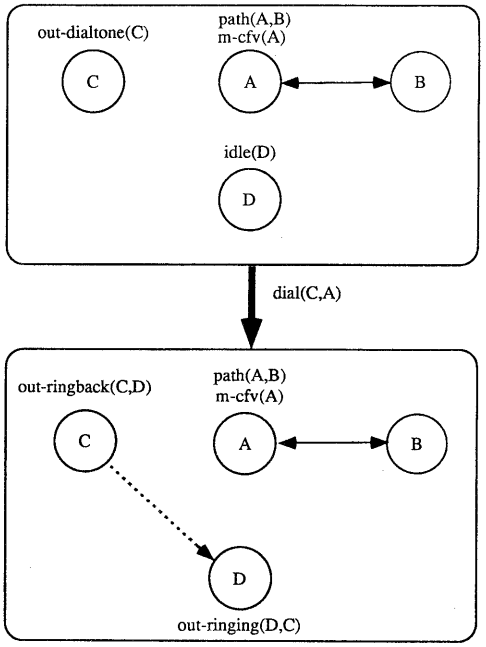

図3：着信転送（CFV）サービスの実行

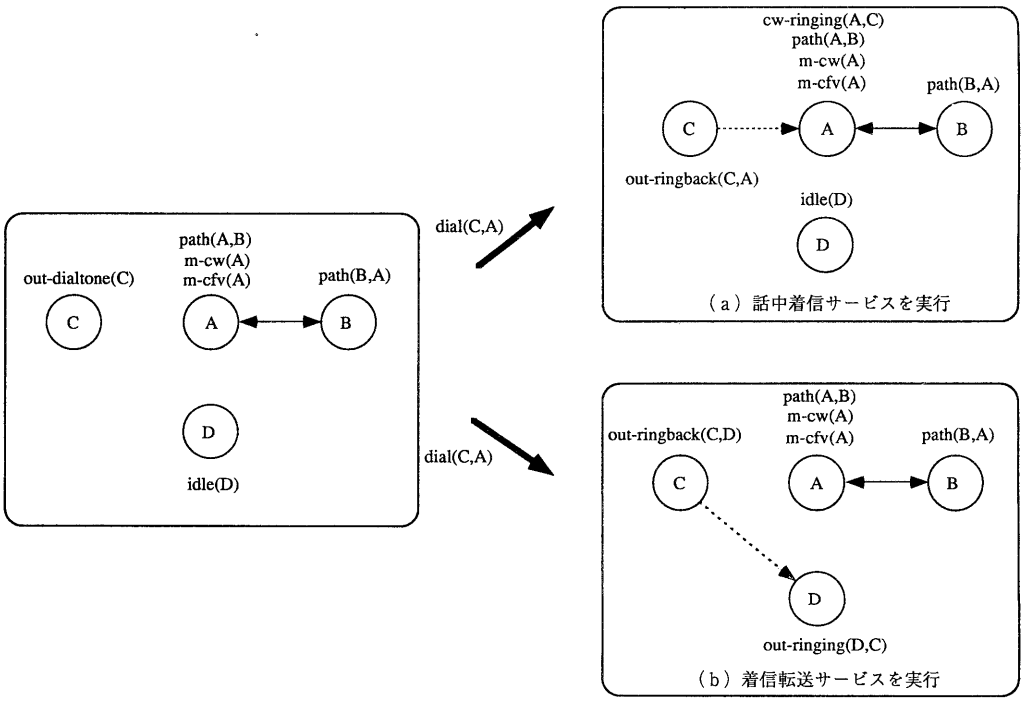

図4：CWサービスとCFVサービスのサービスインタラクション

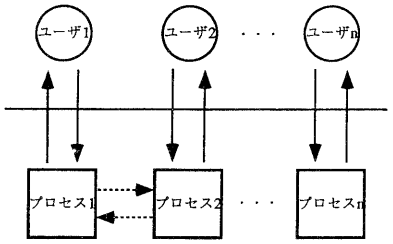

図5：上位層のユーザと下位層のプロセス

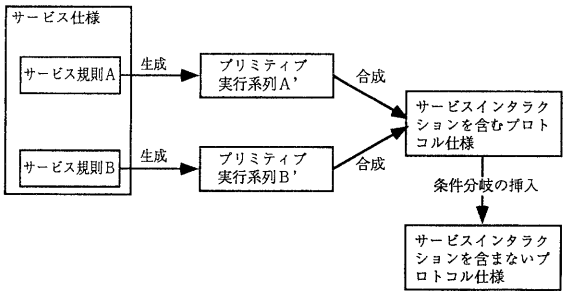

図6：サービスインタラクションの解決

(a) ＣＷサービス規則からの生成

$dial\_req(SAP_C \downarrow ! S_C^1)$
$dial\_ind(SAP_A \uparrow ! S_C^1)$
$dial\_resp(SAP_A \downarrow ! S_A^1)$
$dial\_conf(SAP_C \uparrow ! S_A^1, S_C^1$
[–out-dialtone(C),+out-ringback(C,A)])
$cw\_ind(SAP_A \uparrow ! S_A^1, S_C^1$
[+cw-ringing(A,C)])

イベント生起前のユーザの状態の収集
イベント生起後のユーザの状態の伝達

(b) ＣＦＶサービス規則からの生成

$dial\_req(SAP_C \downarrow ! S_C^1)$
$dial\_ind(SAP_A \uparrow ! S_C^1)$
$dial\_resp(SAP_A \downarrow ! S_A^1, S_C^1)$
$state\_ind(SAP_D \uparrow ! S_A^1, S_C^1)$
$state\_resp(SAP_D \downarrow ! S_A^1, S_C^1, S_D^1)$
$cfv\_ind(SAP_D \uparrow ! S_A^1, S_C^1, S_D^1$
[–idle(D),+out-ringing(D,C)])
$cfv\_resp(SAP_D \downarrow ! S_A^1, S_C^1, S_D^1)$
$dial\_conf(SAP_C \uparrow ! S_A^1, S_D^1, S_C^1$
[–out-dialtone(C),+out-ringback(C,D)])

イベント生起前のユーザの状態の収集
イベント生起後のユーザの状態の伝達

図7：プリミティブ実行系列

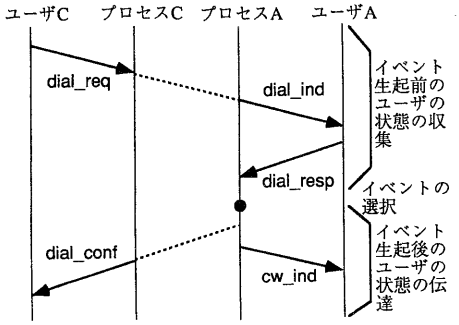

図8：CWサービスのシーケンスチャート

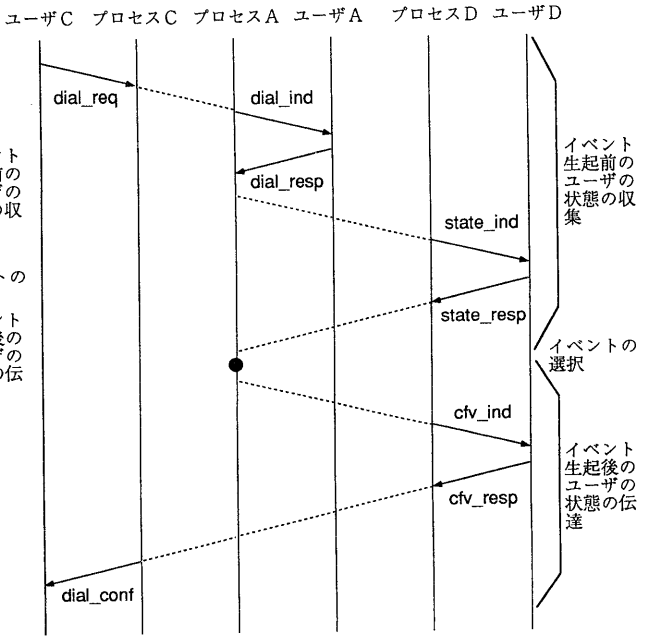

図9：CFVサービスのシーケンスチャート

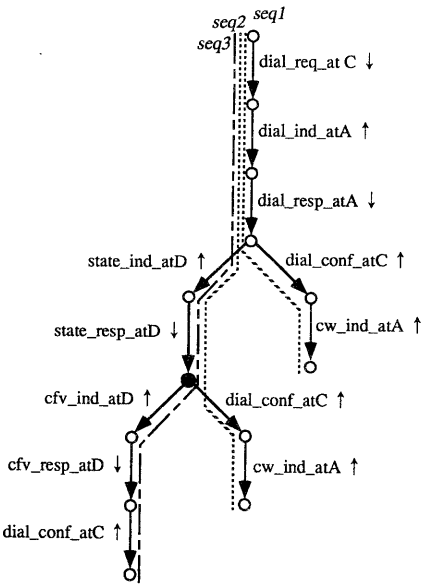

図10 :プリミティブ実行系列

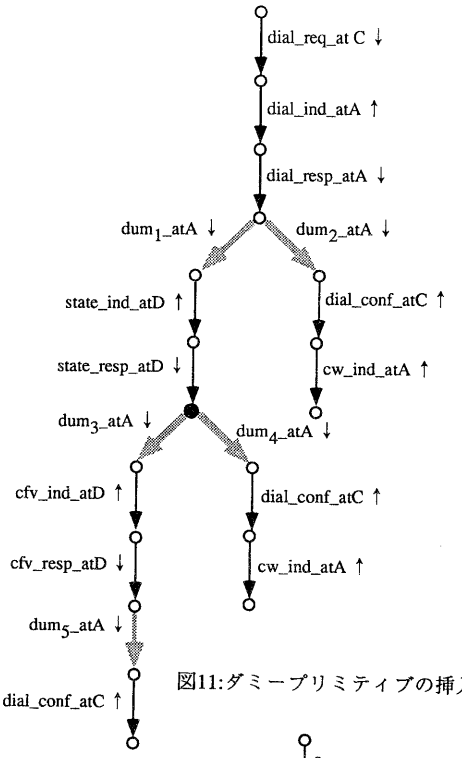

図11:ダミープリミティブの挿入

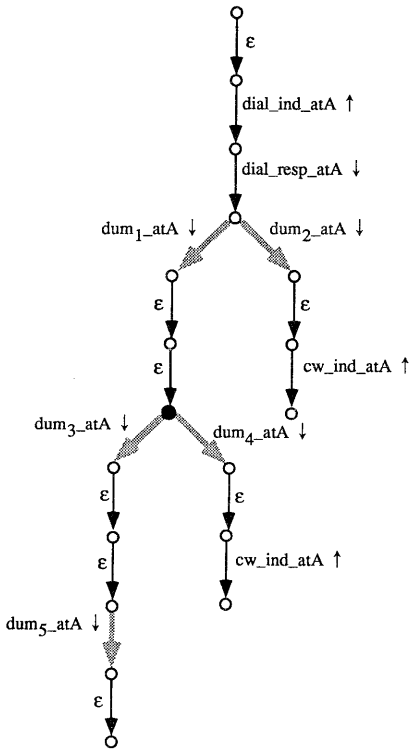

図12 :プロセスAに対する射影

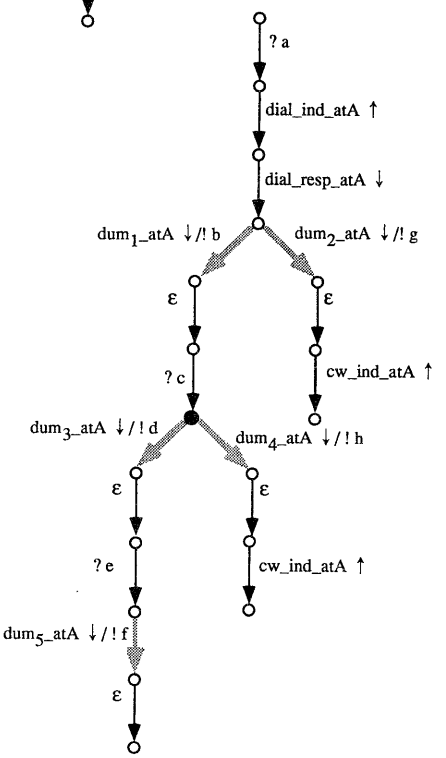

図13 :プロセスAに対する合成ルールの適用

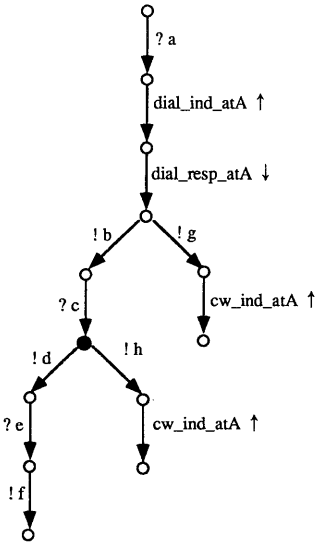

図14 :プロセスAのプロトコル仕様

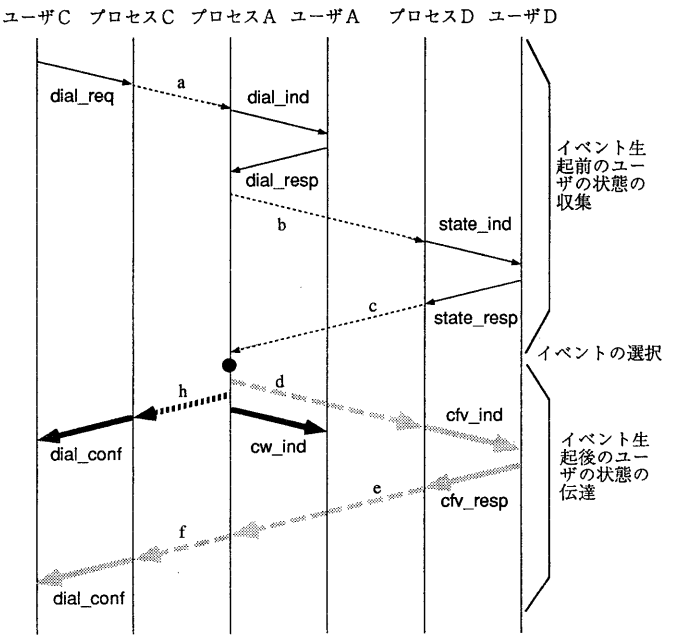

図15 :合成されたプロトコル仕様を表すシーケンスチャート