

TM-0497

通信システム用設計仕様における  
サービスの検証方式

柴田健次, 上田佳寛, 長谷川晴朗

May, 1988

©1988, ICOT

**ICOT**

Mita Kokusai Bldg. 21F  
4-28 Mita 1-Chome  
Minato-ku Tokyo 108 Japan

(03) 456-3191~5  
Telex ICOT J32964

---

**Institute for New Generation Computer Technology**

# 通信システム用設計仕様における サービスの検証方式

## A Method of Verifying Services for Design Specification in a Communication System

柴田 健次 上田 佳寛 長谷川 晴朗

Kenji SHIBATA Yoshihiro UEDA Haruo HASEGAWA

沖電気工業株式会社

Oki Electric Industry Co., Ltd.

**Abstract** We are now developing a software support system -EXPRESS (EXPeRt system for ESS).

EXPRESS designs automatically the specification for a communication system from users' requirements. In EXPRESS, each requirement is converted into ISG(Individual Service Graph) and all ISGs are integrated into TSG(Total Service Graph). ISG and TSG are described by using Petri Nets. This paper gives an outline of EXPRESS and describes the feature of Petri Nets in EXPRESS. The Petri Nets which represents ISG is rigidly connected and the Petri Nets which represents TSG is the set of rigidly connected elements. Then this paper describes a kind of verification for the design specification and shows some analyses of the design specification.

### 1. はじめに

近年、通信システムにおいてユーザの要求は多様化、高度化してきている。それに伴い、通信システム用のソフトウェア開発の負担も大きくなってきている。ソフトウェア開発工程のなかで上流工程である要求仕様化段階は、高級技術者にかかる工数が多大であり、後工程に与える影響も大きい。よって、ユーザの要求を正確に反映して通信システム全体として整合のとれた仕様を作ることが重要となる。

我々は、通信システム開発における仕様化段階のサポートシステムであるEXPRESS(EXPeRt system for ESS)を開発中である。EXPRESSは、自然言語で記述されたユーザの要求から全体として整合のとれた設計仕様を自動作成し、検証するシステムである。設計仕様の記述は、ベトリネットを用いており、設計仕様の検証にベトリネットの解析を利用している。

本稿では、はじめにEXPRESSの概要について述べる。次に、本システムで使用している設計仕様の表現について、その性質およびベトリネットとの対応について検討する。設計仕様に対応するベトリネットの性質を明らかにすることにより、設計仕様に要求される検証をベトリネットの解析を利用して行う。

### 2. EXPRESS概要

#### 2.1 システム構成

図1にEXPRESSのシステム構成概念図を示す。本システムは、要求理解サブシステム、仕様統合サブ

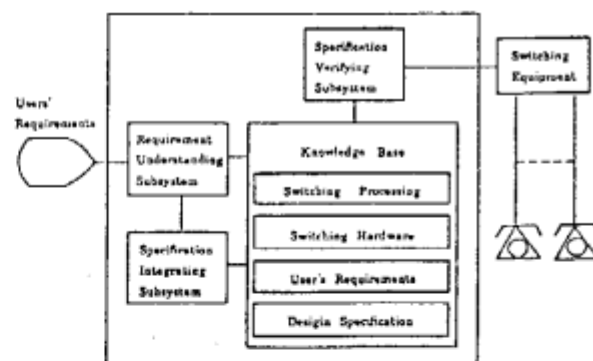


図1. システム構成概念図

システム、仕様検証サブシステムおよび知識ベースから構成される。

#### (1) 要求理解サブシステム

サービス要求定義者が、自然言語により交換サービスに関する曖昧さを含む要求を入力すると、知識ベースの中の知識を用いて要求理解を行う。知識ベースと矛盾するサービスが入力されたときはユーザとインタラクションをとることにより矛盾を解消する。このサブシステムにおいて一つの仕様が作成される。作成された仕様が正しいものかどうかの検証も行う。

#### (2) 仕様統合サブシステム

本サブシステムにおいて、個々の仕様を一つの最終的な設計仕様にする。入力される要求は、一般に

個々のサービス単位で断片的である。従って、相互に矛盾する可能性を有しているが、本サブシステムはこれから誤りのない設計仕様を作成する。

### (3)仕様検証サブシステム

知識ベースに蓄積された設計仕様に従って、交換ハードウェア装置を制御する。ユーザの要求が正確に理解されているか否かの意味検証を行う。

### (4)知識ベース

設計仕様および設計仕様を作成するために必要となる知識が蓄積される。この知識は、交換処理、交換ハードウェア、ユーザ要求に関するものである。

## 2.2 サービス表現

図2はEXPRESSにおいて矛盾、曖昧性のない設計仕様を作成する過程である。専門家がユーザの要求から設計仕様を決定する過程は、ユーザの要求を表すサービス表現から、設計仕様を表すサービス表現への変換として捉えることができる。以下にそれぞれのサービス表現について述べる。

### (1)ure(ユーザ要求要素)

ureは端末に対する操作毎にその前提条件と操作後の状態について表現したものである。ureの動詞オブジェクトにより構成される。動詞オブジェクトは動詞を中心とした格構造として表現されている。図3に表現例を示す。

```
ure([tr(off-hook,[agent(caller)]),
    [st(hear,[goal(caller),object(dial-tone)])])
```

図3. ureの表現例

### (2)sge(サービスグラフ要素)

sgeはサービスグラフの要素に当たるものであり、プレースおよびトランジションを持つ。ureの動作をトランジションとし、リソースを表すトークンに対する動作毎に変化前の状態と変化後の状態をそれぞれ、入力プレースの集合、出力プレースの集合として表される。各プレースは、トークン間の関係を表すリレーションの集合からなる。sgeは、節単位で独立している。つまり、sgeの一節はサービスグラフの要素として捉える

ことができ、sgeにおいては、節単位での追加あるいは削除が容易に行われることを可能とする。図4に表現例を示す。

```
sge(spp([place([rel(token(caller,ext),idle,nil)]),
    place([rel(token(dial-tone,dt),idle,nil)]),
    tr(token(caller,ext,off-hook,nil),
    snp([place([rel(token(caller,ext),receive,
    token(dial-tone,dt)]))])
```

図4. sgeの表現例

### (3)isg(個別サービスグラフ)

isgは複数の断片的なサービスグラフの要素であるsgeを統合し一つの仕様を表すものである。サービスグラフとしてのベトリネットグラフを表現するものであり、単一の状態に一つのプレースを対応させ、かつ一つのプレースに接続しうるトランジションを明確にすることにより、ベトリネットグラフとの対応をより明確にしている。構成要素として、プレース、トランジション、アークがある。図5にisgの表現例を示す。また、図6の(a),(b),(c)にそれぞれ内線相互接続サービスの発呼者先掛、呼び出し中切断、ホットライン呼び出し中切断のベトリネット表現を示す。

```
place(Pt1,gpl([rel(token(Tk1,ext),idle)]),
    spt([Tt2,Tt4,Tt7]),
    snt([Tt1,Tt5]))
transition(Tt1,Arc1,offhook,transit([Arc1,Arc2]),
    spp([Pt1,Pt6]),
    snp([Pt2]))
arc(Arc1,in(Ttk1),out(Ttk3),atr([]))
```

図5. isgの表現例

### (4)tsg(統合サービスグラフ)

tsgは複数のisgを統合して作成する最終の設計仕様である。表現形式はisgと同様である。図6の(d)に(a),(b),(c)を統合したtsgのベトリネット表現を示す。

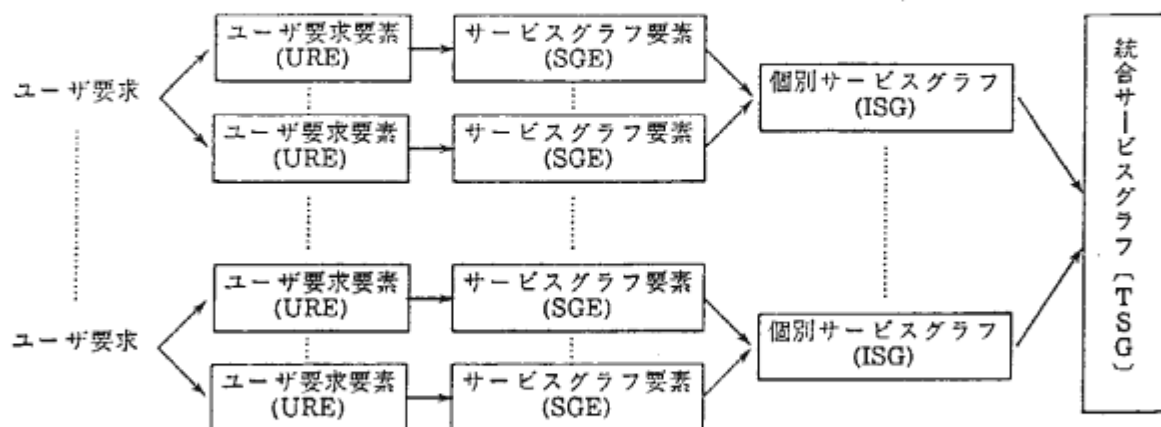


図2. 設計仕様の作成過程

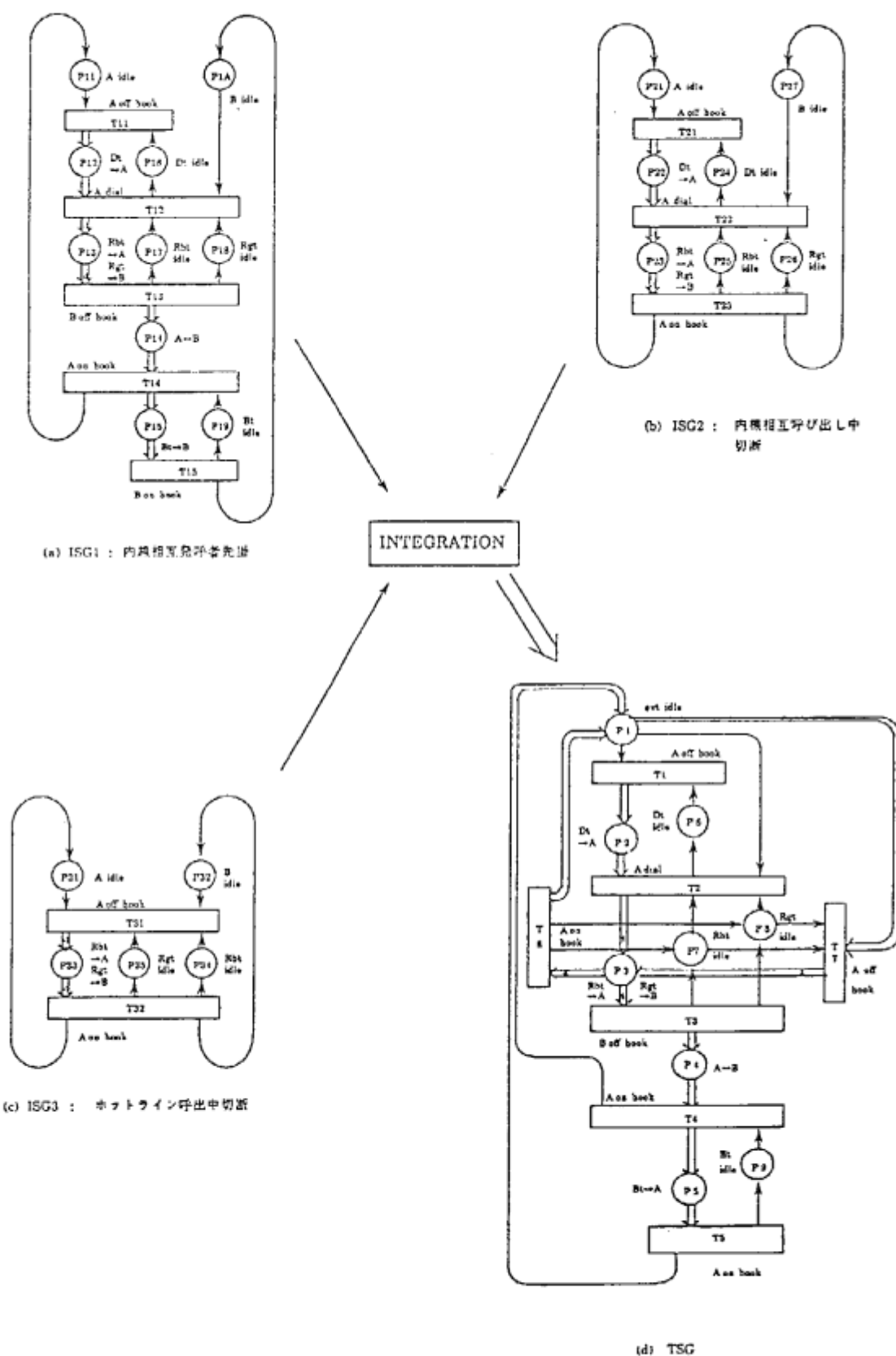


図6. ISG及びTSGの表現例

### 3. ペトリネットの関数行列

本章で、ペトリネットの関数行列について紹介する。トランジションの入力関数および出力関数を表す行列  $D^-, D^+$  を定義する。各行列は、トランジションに対応する  $m$  個の行とプレースに対応する  $n$  個の列からなる。 $D^-$  の  $j$  行  $i$  列の要素はプレース  $P_i$  からトランジション  $T_j$  に入るアークの多重度を、また、 $D^+$  の  $j$  行  $i$  列の要素は  $T_j$  から  $P_i$  に出て行くアークの多重度を示す。各プレースに存在するトークンの数を  $n$  次元ベクトル  $\mu$  (マーキング) で表すと、(1)式が成立するとき  $T_j$  が発火する。

$$\mu \geq e[j] \cdot D^- \quad \dots (1)$$

ただし、 $e[j]$  は第  $j$  成分が1でそれ以外の成分が0の  $m$  次元ベクトルである。また、マーキング  $\mu_0$  において、 $T_j$  が発火すると、新しいマーキングは次のようになる。

$$\mu = \mu_0 + e[j] \cdot D \quad \dots (2)$$

$$D = D^+ - D^-$$

従って、マーキング  $\mu_0$  において発火系列

$$\sigma = T_{j_1}, T_{j_2}, \dots, T_{j_k}$$

が発火すると、新しいマーキングは次のようになる。

$$\mu = \mu_0 + (e[j_1] + \dots + e[j_k]) \cdot D$$

$$= \mu_0 + f(\sigma) \cdot D \quad \dots (3)$$

### 4. 仕様の表現

設計仕様は、ペトリネット表現に対応させている。ペトリネットは、動的なシステムをモデル化し、解析することを目的としたもので、実働性の検証、解析が行いやすいという特徴を有する。また、EXPRESSでは、得られた仕様そのものを知識として利用する。この時、仕様をペトリネットで表現しておくことにより、追加、変更柔軟に対応できるので、知識表現に有効であると考えられる。

ここでは、本システムで用いられている仕様の表現についてその性質を検討する。これらの表現形式が、ペトリネットのある部分クラスに対応することを明らかにする。このことにより、仕様の検証はそのクラスのペトリネットの解析に対応させることができる。

#### 4.1 設計仕様の性質

次にあげることが交換ソフトウェアにおける設計仕様に要求されることである。

- (a) 仕様に記述された動作が可能であり、仕様化された全ての遷移は論理的に起り得る。
- (b) リソースは必ず初期状態に戻る。
- (c) 少なくとも一つのサービスが存在する。

また、設計仕様は、ユーザの要求した複数のサービスの集合と見做すことができる。ここで、使用しているサービスは、次のように定義される。

#### <サービスの定義>

$f(\sigma)$  が次の項目を満たすとき、発火可能ベクトルと定義する。

- (a)  $f(\sigma)$  は  $t$ -invariant である。つまり、 $f(\sigma) \cdot D = 0$  を満たす。
- (b)  $f(\sigma)$  の示すトランジションの発火系列において、少なくともひとつは(1)式を満たす。

このとき、 $f(\sigma)$  は、マーキング  $\mu$  が  $\mu$  自身から可達となるときの解になる。 $\mu$  において、idle状態を示すプレースのみにトークンがあれば、 $f(\sigma)$  がひとつのサービスになる。つまり、空き状態から空き状態にいたるループがサービスである。また、発火可能ベクトルで表されるサービスは、他のどのようなサービスの和でも表されない独立したサービスの和で構成される場合が多い。このような独立したサービスをここでは素サービスと呼ぶことにする。

#### 4.2 仕様表現とペトリネットとの対応

2章で説明したように本システムにおいては、sge, isg, tsgと段階を経て仕様が作成される。isg, tsgについてその性質を調べておくことは仕様を解析するのに重要となる。

設計仕様の性質から、ペトリネットとして満たすべき性質は次のようになる。

- (1) 動作が可能になるということは、可達性が成り立つということであり、すべての遷移が起り得るということは活性を満たすということである。
- (2) リソースは必ず初期状態に戻るということから、リソースを表すトークンの遷移がなくなっていくといけない。また、トークンの消滅がないということから、入力プレースの集合と出力プレースの集合は同じとなる。もし同じでないとすると、入力トランジションしか持たないようなシンクプレースあるいは出力トランジションしか持たないようなソースプレースが存在することになり、リソースが初期状態に戻るができなくなる。なお、サービスとしては、ある状態から、動作を起こすことによって必ず別のある状態に移ることを前提とする。
- (3) 少なくとも一つのサービスが存在するには、必ず一個以上の  $t$ -invariant が存在することが必要である。つまり、 $\text{rank}(D) \leq m-1$  である。ここで、 $m$  は行列  $D$  の行数である。

以上により、ペトリネットとしては、 $t$ -invariant が存在することが必要となり、さらに、トランジションは動作を表すため、その要素は非負整数に限られる。また、すべてのサービスが過不足なしに求められることが必要であり、そのためには、 $t$ -invariant が和に分解不可能でなければならない。

次に、設計仕様についてさらに詳しく性質を調べると共に、それらの表現するベトリネットがある部分クラスに対応することを示す。

### ③ isg

isgは、一つのサービスに対応しているもので、どの一つのトランジションを取り去っても、サービスとしては成立しなくなる。即ち、isgはt-invariantが存在する可能性のある最小のネット構成である堅結合の形を取っていると言える。

また、図7において、入出力トランジションの発火回数 $g(t1), g(t2)$ が $g(t1)\beta = g(t2)\alpha$ を満たす、つまり、トークンが均衡されていれば、t-invariantが存在する<sup>[11]</sup>が、図6(a)のようにisgは強保存であり、各プレースに対してプレースの入出力トランジションの発火回数は等しく1であり、よって、必ずt-invariantが存在する。

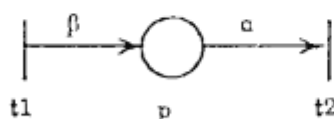


図7.

さらに、isgにおいて、t-invariantは要素が負のものは考えないので、それに対応するトランジションで生成したネットはconsistent成分となる。また、上述のようにisgは堅結合であるので、そのネットは初等的consistentネットとなる。初等的なconsistentネットの独立なt-invariantは唯一である<sup>[11]</sup>ので、isgにおけるプレース・トランジション接続行列のランクは $m-1$ であり、1個の独立なt-invariantが存在する。これが、素サービスを表すベクトルとなる。

### ④ tsg

tsgはisgを統合したものであり、構造としてはt-完全であり、isgを包含している極小なt-完全成分の集合となっている。

## 5. 仕様の検証について

### 5.1 どのような検証が必要か

交換システムにおいては、要求の新規作成、変更、追加ということが頻繁に起こり得る。この時、新しい要求そのものに誤りが含まれていたり、追加することによって誤りや矛盾が生じることも多々あると考えられるが、専門家はサービス追加時に発生するエラーを設計仕様にまで持ち込まないようにしている。

設計仕様を作成する過程において、どのような検証が要求されるかを考えてみる。2章で説明したように、本システムでは一つのサービスを作成した後、統合されたサービスを作成する。大きく分けると、一つのサービスに対する検証および統合されたサービスに対する検証の二つが必要となる。

以下にそれらを列挙する。

- (1) 個々の断片的なサービスそのものが正しいということ
- (2) 個々のサービスを統合した後で、要求したサービスがサービスとして含まれているかどうか

- (3) 統合した後のサービスに対して要求以上のサービスが存在し得るかどうかの検出
- (4) 部分的なサービスの記述により追加したサービスがサービスとして満足できるか否かの検証
- (5) サービスとしては異なった要求となっているが、それらが詳細化の度合いが異なるだけで実は同じことを意味しているものだという指摘上にあげた、要求される検証についての対処法を概説する。

- (1) 個々の断片的なサービスが正しいか否かを検証するということは、isgが初等的consistentネットであるか否かを調べればよい。
- (2) 要求したサービスが満足できるものか否かは、tsgの素サービスの中にそのサービスが存在するか否かを調べることによって確認することができる。
- (3) 要求以上のサービスが存在するか否かは、tsgからすべての素サービスを求めることによって確認することができる。
- (4) 部分的なサービスの記述によって、それが一つのサービスの一部として充たされているか否かは、追加したトランジションを含む極小なt-完全成分が存在するか否かを調べることによって、確認することができる。
- (5) 詳細化の度合いが異なる場合は、発火するトランジションに二つ同じものがあり、その間の発火トランジションの系列が異なるような素サービスを表すベクトルを指摘する。

### 5.2 仕様の検証例

前節に上げた要求のうち、(1),(2)は既に取り上げている<sup>[14]</sup>ので、ここでは(3)の要求以上のサービスが存在し得るかどうかの検出、(4)の部分的なサービスの記述により追加したサービスの検出を取り上げる。

#### ① 要求以上のサービスの検出

要求以上のサービスが存在し得るかどうかは、得られたtsgの全素サービスを求めることで検出できる。全素サービスを求めることは、文献[2]で言うところのt-baseを求めることと同義である。実際に、図6(d)のtsgに対してt-baseの構築を適用することにより、p, q, r, sの素サービスを表す初等的t-invariantが得られる。

$$\begin{aligned} p &= (1111100) \\ q &= (1100010) \\ r &= (0000011) \\ s &= (0011101) \end{aligned}$$

また、図6(d)のtsgにおける全てのトランジションに対し、極小なt-完全成分の抽出を行えば、全素サービスを求めることが可能である。

#### ② 部分的サービスの検証

4章で述べたように、t-完全であるtsgは極小なt-完全成分の集合となっており、この極小なt-完全成分そのもののトランジションの集合が素サービスを表す初等的t-invariantの非零の要素となる。即ち、ある部分サービスを追加したとき、それによって加えられ

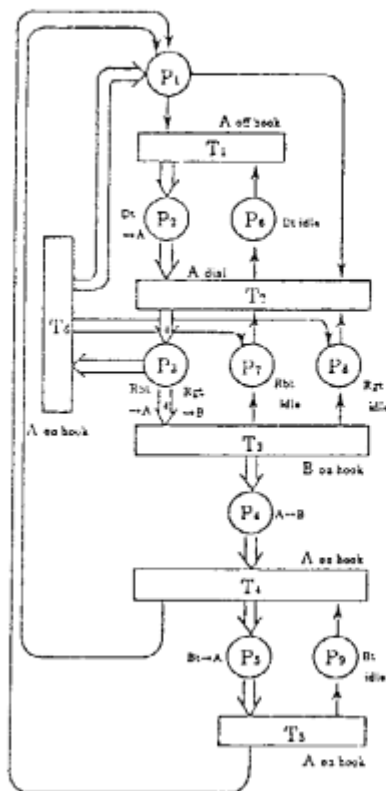


図8. TSG

たトランジションを含む極小な $t$ -完全成分が存在するか否かを調べればよい。

あるトランジション $t_0$ を含む極小な $t$ -完全成分は次のようにして抽出する。 $T=\{t_0\}$ と置く。 $T$ における入力プレース $T$ , 出力プレース $T'$ を求め、 $T'$ に属し、 $T$ に属さないプレース $p$ を入力として持つトランジションあるいは $T$ に属し、 $T'$ に属さないプレース $p$ を出力として持つトランジションのうち $T$ に属さないトランジションの一つを $T$ に追加し、更新された $T$ における $T, T'$ を求める。 $T=T'$ となったときの $T$ および $T'$ が極小な $t$ -完全成分となる。ただし、この過程で、 $t_0$ を含まないトランジションの部分集合で入力プレース、出力プレースの集合が等しくなる場合は、選択したトランジションを $T$ から除去し、他のトランジションを選択して行く。

例として、発呼者先掛サービスに呼び出し中からidleへ戻るサービスの部分を追加する場合を考える。図8のtsgにおいて、トランジション $t_6$ が追加した部分サービスに含まれている。トランジション $t_6$ を含む極小な $t$ -完全成分を抽出すると、 $T$ および、 $T'$ はそれぞれ $T=\{t_1, t_2, t_6\}$ ,  $T'=\{p_1, p_2, p_3, p_6, p_7, p_8\}$ となり、呼び出し中切断サービスの一部として正しいと見なすことができる。

## 6. おわりに

本稿では、通信システムの開発における仕様化段階のサポートシステムであるEXPRESSの概要および

ベトリネットによって表した設計仕様について述べた。

対象とするモデルの性質を明確にすることにより、それに対応するベトリネットが特化され、その解析も容易になり得る。勿論、モデル化能力を高めると、それに対する解析能力は減少するが、EXPRESSにおける設計仕様の場合、モデルの性質により、対象となるベトリネットは堅結合の集合となり、その解析は、文献[1]等の成果を参照できる。

対象としている設計仕様において、isgは必ず $t$ -invariantが唯一存在し、それが求まることによって素サービスを求めることが可能となる。さらに、tsgは構造が $t$ -完全であるので、極小な $t$ -完全成分を抽出することによって素サービスを求めることが可能となる。素サービスを求めることがこのように容易になることにより、サービスの様々な検証、解析を行うことが可能となった。

しかしながら、これらの検証はあくまでも形の上でのことであり、交換の意味的なものは入ってはきていない。意味的なことの確認のために本システムにおいては実際に交換ハードウェアを用い、それを動作させることによりユーザが自身で検証するようになっている。また、これと同等ではあると思うが、ベトリネットをグラフィカルに表現し、画面上でシミュレートすることも考えているところである。さらに、今後は対象とするサービスを増やし、設計仕様に対する検証の評価を行う予定である。

なお、本研究は第5世代コンピュータプロジェクトの一環として行っているものである。

## 参考文献

- [1] 翁長, 葛: "ベトリネットのT-invarianceの構造的解析", 電子情報通信学会論文誌 '87/2 Vol. J70-A No. 2
- [2] 葛, 谷田, 翁長: "ベトリネットにおける $t$ -baseの構築と周期発火系列の設計", Proceedings of the 8th Mathematical Programming Symposium, Japan (November, 1987)
- [3] J.L. Peterson: "Petri Net Theory and The Modeling of Systems", Prentice-Hall, (1981)
- [4] 上田, 柴田, 田中, 長谷川: "T-invariantによる通信システム用サービスの解析", 第三回ネット理論研究会, pp. 82-89 (1987)
- [5] 柴田, 上田, 湯山, 田中, 長谷川: "通信システム用設計仕様の論理検証方式", 電子情報通信学会情報ネットワーク研究会, IN87-75 (1987)
- [6] H. Hasegawa, W. Tanaka, K. Shibata: "Analysis of Design Specification in a Communication System by means of Petri Nets", Proceeding of 11th Computer Software and Application Conference, pp. 701-706, (1987)
- [7] J. Sifakis: "Use for Timed Petrinets for Performance Evaluation", 3rd Int. Symp. Measuring, Modeling and Evaluating Comput. Syst., Beilner and Gelenbe, pp. 75-95 (1977)